2013

Analysis of multivariable controller designs for diesel engine air system control

Daniel Albert Humke
Iowa State University

Analysis of multivariable controller designs for diesel engine air system control

by

Daniel A. Humke

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Major: Electrical Engineering

Program of Study Committee:

Ratnesh Kumar, Major Professor

Nicola Elia

Atul Kelkar

Iowa State University

Ames, Iowa

2013

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I would like to take this opportunity to express my thanks to those who helped me with the various aspects of writing this thesis.

First and foremost, I would like to thank Dr. Ratnesh Kumar for his patience and support throughout this research and the writing of this thesis, as well as his insightful and helpful feedback on the multiple revisions I sent his way.

I would also like to thank my colleagues, it their support of my pursuit of this degree. I would especially like to thank those who have also reviewed drafts of this document and provided feedback and input to help create a complete, well written thesis.

Last but not least, I would like to thank my family for their unwavering support and encouragement as I have gone through this process. Their thoughts and words helped provide me with the motivation needed to see this through to completion.

ABSTRACT

As diesel engine emissions standards become increasingly stringent, one of the most commonly employed method of emissions reduction by engine manufacturers is active control of inducted air and recirculated exhaust gas (EGR). Most often actuators such as an EGR valve and a variable geometry turbocharger (VGT) are used to manipulate the air flow through a diesel engine to provide the reduced engine out emissions. This paper evaluates four different multivariable controller designs for control of a diesel engine air path: three built up from Single Input, Single Output (SISO) transfer functions and one that is a fully multivariable design. Various performance metrics are analyzed to gauge the relative difference in performance capability while maintaining a simple controller architecture. As part of the analysis, the controller designs are simulated with a realistic non-linear engine model against a modified standard certification test cycle to give a sense of how the controllers would perform in a real-world application.

CHAPTER 1. INTRODUCTION

Over the past several years, government mandated regulations for diesel engine out emissions - specifically oxides of nitrogen (NOx) and particulate matter (PM) - for both on and off road vehicles have become increasingly stringent. This has resulted in many diesel engine manufacturers adding more complexity to the engine systems: more sensors to better measure and/or model what the engine conditions and emissions are at any given time as a function of operating conditions, and more actuators to allow for simultaneous control of additional variables and an increasing number of performance objectives. Specifically, it has been shown that increasing the amount of EGR in the cylinder decreases NOx production while providing an over-abundance of air in the cylinder (i.e. increasing the air/fuel ratio) allows for the fuel to burn cleaner and provide a reduction in PM.

1.1 Motivation

Commonly, EGR (Exhaust Gas Recirculation) and VGTs (Variable Geometry Turbochargers) are employed in order to manipulate the flow of gases through a diesel engine to achieve the desired reduction in NOx and PM emissions, as mandated by the US government (shown in Figure 1.1[1]) and other governments around the world. The drastic reduction in the areas of the boxes certainly show the challenges associated with emissions control, and along with Figure 1.2, begin to illustrate the increased need for higher levels of technology, and therefore higher levels of control. However, the movement of either one of these actuators has a direct impact on the percentage of EGR in the intake manifold as well as the amount of air that can be inducted into the cylinder, resulting in a complex, interactive, multivariable system. A diagram of a common air path layout is shown in Figure 1.3, where one can see the system

[1]Downloaded from John Deere Machine Finder Blog

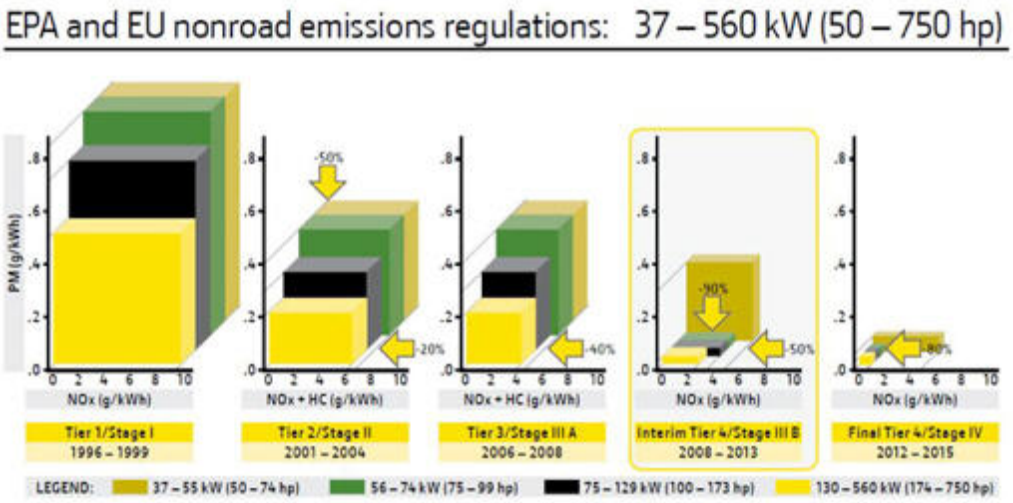

Figure 1.1 EPA Emissions Tiers

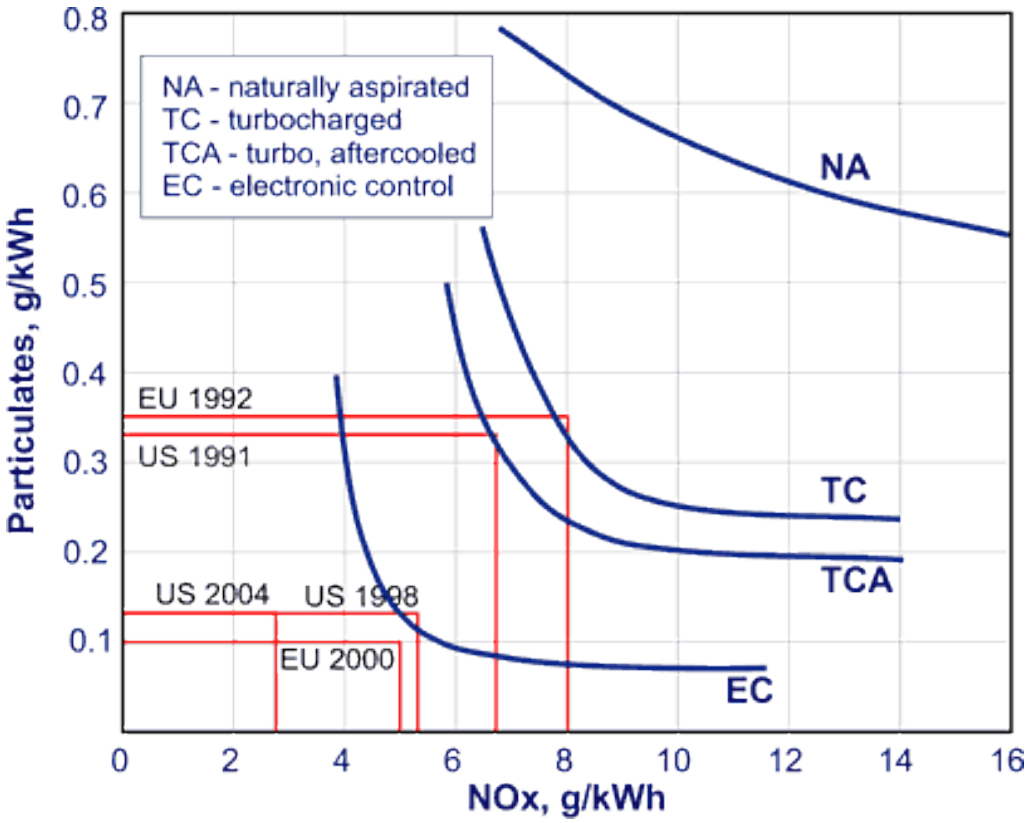

Figure 1.2 NOx - PM Trade-Off Curve

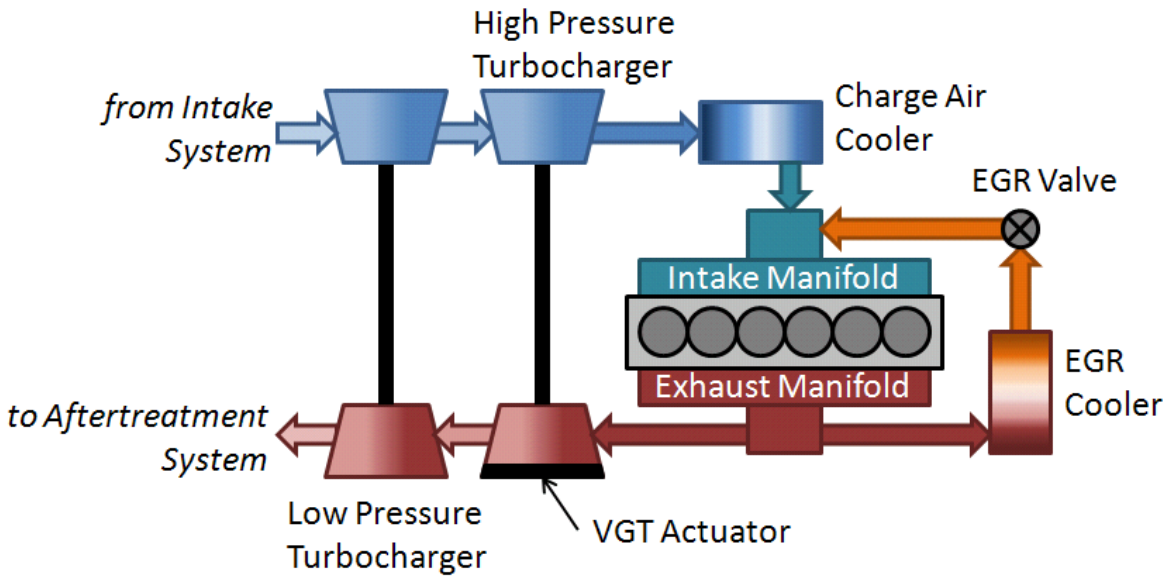

Figure 1.3 Engine Air System Diagram

interactions and internal feedback caused by the EGR loop and turbochargers. In addition to the internal feedback nature of the diesel engine air path, the first principles equations that govern the behavior of the engine are highly non-linear. Heywood (1988) discusses many of these principles of engine behavior, including a whole chapter on the difficulties of modeling engine behavior. This serves to compound the internal feedback nature of the engine, making the application of standard linear control theory more than just a trivial exercise.

Another factor that increases the difficulty of engine air path control is the need to balance customer requirements of responsive engine performance (in terms of speed and power generation), the desire for more generated power, and improvements in fuel economy against the mandates of government agencies to reduce emissions. The majority of the common emissions reduction strategies involving air path management often have negative impacts on the response of key customer performance variables (i.e. speed, power generation, and fuel economy to name a few). For example, the introduction of EGR into the air system has a negative impact on the overall fuel economy of the engine, and so the trade-off of how much EGR to use versus how much fuel economy to trade off becomes a design challenge related to customer needs. In addition to this trade-off, there exists a trade-off between the amount of NOx and PM reduction.

As was mentioned above, increasing the EGR rate decreases NOx production, but it does so at the expense of increasing PM emissions (see Figure 1.2[2] showing various NOx/PM trade-off curves for various levels of engine emissions control technologies). Adding to these design trade-off difficulties is that the most recent versions the emissions regulations contain not just static test points to measure emissions compliance - there is a transient component to the test as well. This transient test component aligns very much with what the customer desires are for responsive performance, and combined with the fact that the engine is a non-linear, highly coupled multivariable control system lays the framework of the control challenges associated with the diesel engine air system.

One intuitive next step idea to help address the design challenges mentioned above is to continue to use more and more complex controller architectures (e.g. Model Predictive Control (MPC), Neural Networks, Sliding Mode Control, etc.) to meet the necessary control requirements. While this is a possibility explored by many - and will be discussed some later in this document - there are some practical implementation reasons why many of these complex controller architectures may not implemented outside of a research setting. In many cases the end user of the controller may not be the actual controller designer. For example, a controls engineer designs the controller and implements it in software, but an engine calibrator may then be responsible for tuning the controller's performance. The addition of more sensors and actuators quickly takes one into a multivariable control domain, where the use of complex control architectures can often meet the tight control specifications often required. However, these control architectures are often designed and implemented for a specific system (and therefore are not generic enough to apply a given controller architecture to a similar system without a completely separate design), thus making the designer and end user's job more difficult. This is especially true if the end user/calibrator is unfamiliar with the complex control strategy.

Regardless of what style of controller architecture is selected (simple versus complex), for a manufactured engine the control strategy must be implemented in real time on an embedded processor in an Electronic Control Unit (ECU). Due to keeping costs down for manufacturers, these processors are generally of lower computing power, and may not have large amounts of

[2]Downloaded from DieselNet Technology Guide

throughput or onboard memory, especially when compared to high-power workstation computers or modern desktop PCs. When looking at the large amount of calculations, or many matrix equations that need to be solved using linear algebra, or even static mapping of various engine responses in large numbers of tables, as well as the hard requirement of executing the control algorithms in real time, the ECU processor speed and memory issues become limiting constraints on the use of highly complex controller architectures, and again driving back to a desire for simpler control architectures/strategies.

1.2 Objectives

The intent of this work then is to explore a proposed control architecture that is capable of providing improved performance in terms of the response of intermediate control variables, not the direct regulated engine emissions. These intermediate variables are used instead, because real time measurement of the regulated emissions is extremely difficult to do on engine. Indeed, development and certification centers use test cells and emissions gathering equipment that fill additional rooms in order to determine how well an engine is meeting emissions. Therefore, variables that are derived from actual sensors on the engine that correlate well to the higher level regulated emissions are used. A focus will be kept on a simple architecture, in order to address some of the implementation concerns mentioned above, but there will also be a focus on decoupling the individual control loops' performance. Skogestad and Postlethwaite (2005) discuss more on decentralized, decoupled, and simplified control elements further (and more of this will be reviewed later in this paper), but a short summary of some of the main reasons for these simplified - or decoupled - control structures are listed here.

- Easy for operators/end users to understand, in terms of controller structure, implementation, and calibration procedure(s)
- Reduced computational load
- Analysis of individual control loops can become easier, because the impacts of the individual control loops are separated

- The overall closed loop system is more fault tolerant, in that one control loop can be taken off-line without significantly impacting the performance of the other loop(s)
- Tuning/calibration of the individual controllers' performance can be done more easily on-line, as well as sequentially (i.e. one at a time), due to the fault tolerant nature mentioned in the previous bullet

While most of the items in the list directly address the motivations discussed, the fault tolerant nature of the decoupled architecture addresses other implementation concerns that are not covered in this paper, but are essential to consider when implementing a controller design on an actual engine.

1.3 Contributions

This work provides an exploration into a controller architecture that will be referred to in this study as the Cross Interaction Filter (CIF) design. The remainder of the document will perform an analysis of a linearized version of the engine air system plant model to better understand the system and the associated control difficulties. In addition, analysis will be performed to determine the recommended pairing selection of the available inputs/outputs (i.e. sensors and actuators). Following that, a review will be made of existing engine air system control architectures, including examples of simple and more advanced architectures, as well as a brief review of decoupling controllers outside of engine air system control.

Closed loop performance analysis will be performed on a baseline diagonal PI[3] controller architecture, as well as a common, intuitive feed-forward decoupling architecture. Along with those, a full multivariable controller architecture will be analyzed for closed loop performance as well. Once a baseline has been established and other competing architectures analyzed, the concept of the CIF design will be developed, analyzed in form, and explored. As the CIF is meant to be a decoupling architecture, this will also include a comparison to the feed-forward decoupling architecture and the implications of the inherent differences between the

[3]The Proportional - Integral controller, common in many control theory textbooks. For the remainder of the document, a PI controller could be exchanged with PID (Proportional - Integral - Derivative) controller, and the two terms in many cases are essentially being used interchangeably.

two. For all control architectures explored, simulations will be performed on the linearized plant model so that standard step response metrics can be directly compared. The analysis of each controller architecture will be concluded with a closed-loop simulation of a high fidelity John Deere 9.0L interim Tier 4 diesel engine model (with an air system architecture like that shown in Figure 1.3) so that comparisons can be made of the relative changes of real world performance of each controller design.

CHAPTER 2. ENGINE AIR SYSTEM ANALYSIS

This chapter will analyze the architecture of the engine air system, including detailing the linearization of the engine model, analysis of the controllability, and final recommendations for closed loop control bandwidth (i.e. closed loop system response characteristics) and input/output pairing.

2.1 Engine Linearization and System Identification

Much of the literature today on engine air system controls explore the use of dynamic models of varying complexity of the engine, often derived from first principles equations as a basis for the controller synthesis and control law calculation. Skogestad and Postlethwaite (2005) express - in addition to much of the literature today - that creating these types of models can at times be a fairly laborious and time consuming process. While few would argue that it is certainly a worthwhile endeavor to identify the states of an engine air system and derive the non-linear differential equations with which to produce a non-linear state-space model of the system so that a highly capable control system could be designed, at times in the manufacturing process this path is not chosen due to schedule pressures, availability of resources (including personnel), and limitations of measurement and data collection techniques. The calibration and verification of a non-linear state-space engine model is can require a significant amount of test cell time that most manufacturing companies often would rather not sacrifice in order to continue to pursue standard development procedures of their products. As a result, control architectures are kept simple to keep development and calibration time down, and because of an assembly line philosophy that is prevalent in industry, the development tasks are often handled in various stages by different teams. This staged, sequential development approach lends support to the

objective of keeping a simple controller architecture that is easy to understand, calibrate, and analyze when problems occur.

Instead of developing a first principles non-linear state-space model of the engine air system (for the reasons mentioned above), for this study empirical data has been collected from the engine air system[4], and used to tune a linear fourth order state-space model using the commercially available System Identification Toolbox software from The Mathworks™. The linearization point selected is the Mode 1 engine certification speed/load operating condition, and this point was selected because it is the point that defines the engine's power rating. A fourth order state-space form was selected, because the dominant states in an engine's air system typically correspond to the intake and exhaust manifold conditions, and the speeds of the turbochargers (for this system, two manifolds plus two turbochargers is four dominant states).

The manipulated inputs are the VGT Vane position and EGR Valve position, which as mentioned earlier are common actuators used on diesel engines for air system management and emissions reduction. The output (or controlled) variables are proprietary modeled in-cylinder conditions, but they are variables that correlate well to the previously mentioned in-cylinder air/fuel ratio (AFR) and intake manifold EGR percentage, and therefore will be refered to by these names for the remainder of this study. In addition, these variables have been normalized to further protect their proprietary nature and to make direct comparisons in response magnitudes. Since empirical input/output response data was used to tune the model, the matrices are therefore all full (i.e. all non-zero elements, which is usually not the case for first principles based models) and the states have no direct physical meaning (which is possible as state-space representations are not unique). The results of this tuning are shown below in Equations 2.1 and 2.2 (see Appendix A for more details on the system identification procedure and results).

$$\begin{bmatrix} \%EGR \\ InCylAFR \end{bmatrix} = G \begin{bmatrix} VGT_{vane} \\ EGR_{vlv} \end{bmatrix} \tag{2.1}$$

[4]The empirical response data collected for this work was collected from a high-fidelity engine model rather from an actual engine. It should be noted though that this data could easily come from an actual engine in a test cell with minimal interruptions to its test schedule, or as Skogestad and Postlethwaite (2005) mention about simple architectures, is that the controllers could even be tuned online, without the need for this step.

$$G = \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] = \left[\begin{array}{cccc|cc} -8.8487 & -0.0399 & -5.5500 & 3.5846 & 0.0564 & 0.0319 \\ -4.5740 & 2.5010 & -4.3662 & -1.1183 & 0.0165 & -0.0001 \\ 3.7698 & 16.1212 & -18.2103 & 4.4936 & 4.4939 & 1.5985 \\ -8.5645 & 8.3742 & -4.4331 & -7.7181 & -1.4269 & -0.2730 \\ \hline -3.2988 & -2.1932 & 0.0370 & -0.0109 & 0 & 0 \\ 0.2922 & -2.1506 & -0.0104 & 0.0163 & 0 & 0 \end{array}\right] \tag{2.2}$$

From the linear state-space model, an array of transfer functions can be created, which has been reduced to first order transfer functions with a time delay by using the methods in Skogestad (2003), and are shown in Equation 2.3. This method captures the dominant dynamics of the engine air system (as seen in the open loop step responses in Figure 2.1), keeps the system representation simple, will keep controller tuning simple (as will be seen later), and will eventually show the benefits of decoupling control in that reduced modeling resources and effort are needed (see Appendix C for more details on the Skogestad Internal Model Control reduction technique applied here).

$$\begin{bmatrix} \%EGR \\ InCylAFR \end{bmatrix} = \begin{bmatrix} \frac{0.4867}{0.0987s+1}e^{-0.0340s} & \frac{0.2155}{0.0987s+1}e^{-0.0643s} \\ \frac{-0.1344}{0.4426s+1}e^{-0.5960s} & \frac{0.07611}{0.2165s+1}e^{-0.0997s} \end{bmatrix} \begin{bmatrix} VGT_{vane} \\ EGR_{vlv} \end{bmatrix} \tag{2.3}$$

One immediate thing to notice in Equation 2.3 and Figure 2.1 is the response in-cylinder AFR to the VGT Vane position input. First off, it has a negative gain, while the other three responses do not. This serves to compound the internal feedback nature of the engine air system, and will negatively impact response performance of the feed-forward decoupling design (discussed later in Chapter 3). In addition, the full system model step response of the in-cylinder AFR to a step change in the VGT Vane position (lower left in Figure 2.1) shows a response that has a Right Half Plane (RHP) zero effect. This is also represented in the reduced models with a very large time delay - roughly an order of magnitude larger than the others - due to the SIMC model reduction technique. One other aspect that is prevalent in Equation 2.3 and Figure 2.1 is that the response of the Percent EGR is noticeably faster than that of the in-cylinder AFR (more than double). This is because the response of opening or closing the EGR Valve position

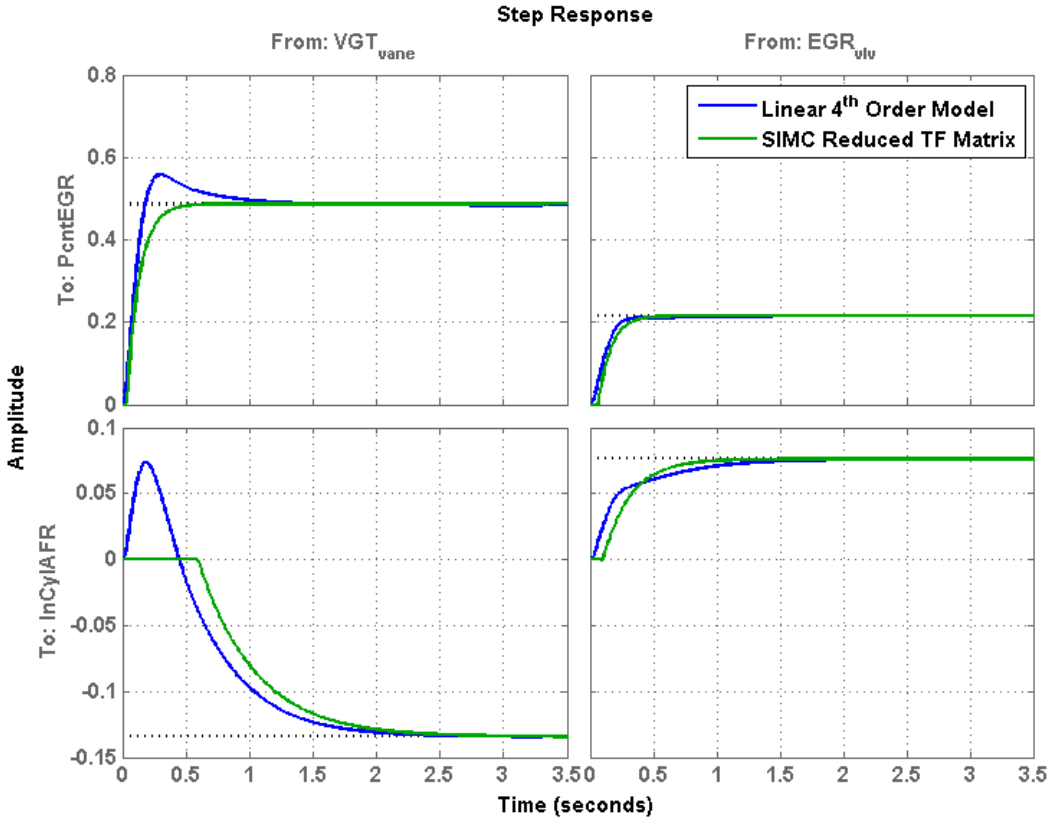

Figure 2.1 Open Loop Step Response

has a much more direct impact on the percent of EGR in the intake manifold, as the EGR loop essentially directly connects the exhaust and intake manifolds. This is contrasted by the loop formed with the VGT: adjusting the VGT Vane position affects the amount of power the turbine can reclaim from the exhaust stream, which then needs to be transmitted through the turbocharger shaft to the compressor wheel, which then introduces some lag (related to the upstream and downstream volumes) to compress the intake charge air before it becomes inducted into the engine cylinders.

2.2 Open Loop Linear Model Analysis

Before beginning a controller design, one must understand the system that is being controlled. Even though our plant model is not derived from first principles equations, the in-

put/output response data does capture the dominant dynamics well enough that the following analysis of our linear model will provide insight into the expected dynamic behavior and initial conclusions on the ease (or difficulty) of control. Since a decoupling controller is being designed, at the end of the section a recommended input/output pairing will be made as well as a closed-loop bandwidth recommendation. The decoupling goal of our controller design will also help with the tuning of the individual PI controllers, because the 2×2 MIMO (Multiple Input, Multiple Output) system will be decomposed into a collection of SISO (Single Input, Single Output) systems that can each be analyzed independently.

2.2.1 Pole-Zero Analysis

The first analysis performed is to look at the poles and zeros of the engine air system. The poles and zeros are shown in Equation 2.4 as well as in Figure 2.2. The location of the poles shows that with all poles in the Left Half Plane (LHP), the engine air system is stable but will exhibit some aspect of oscillatory behavior due to the complex pole pair. There is also a LHP and a RHP transmission zero. The LHP zero serves to increase the response speed, and could potentially drive overshoots in the response, but the bigger impact comes from the RHP zero, which first off creates an initial inverse response (as seen in the VGT Vane to in-cylinder AFR response, Figure 2.1), but also has a negative impact on the overall achievable response. The implication for a closed loop controller is that the initial response for the controller will be to overdrive the corresponding actuator, since the control variable is moving in the wrong direction. This has the potential to cause increases in settling time, minimum overshoot, oscillations, etc., as well as minimum requirements on achievable closed-loop bandwidth (see Skogestad and Postlethwaite (2005) for more details and equations on the implications of RHP zeros and poles).

$$\begin{aligned} p &= [-2.4410,\ -5.3148,\ -12.2601 \pm 8.9809j]^T \\ z &= [-55.5586,\ 1056.9]^T \end{aligned} \tag{2.4}$$

In looking at the RHP zero present in this air system model, the question arises as whether or not the RHP zero is to be expected. In their work on engine air system modeling, Wahlström

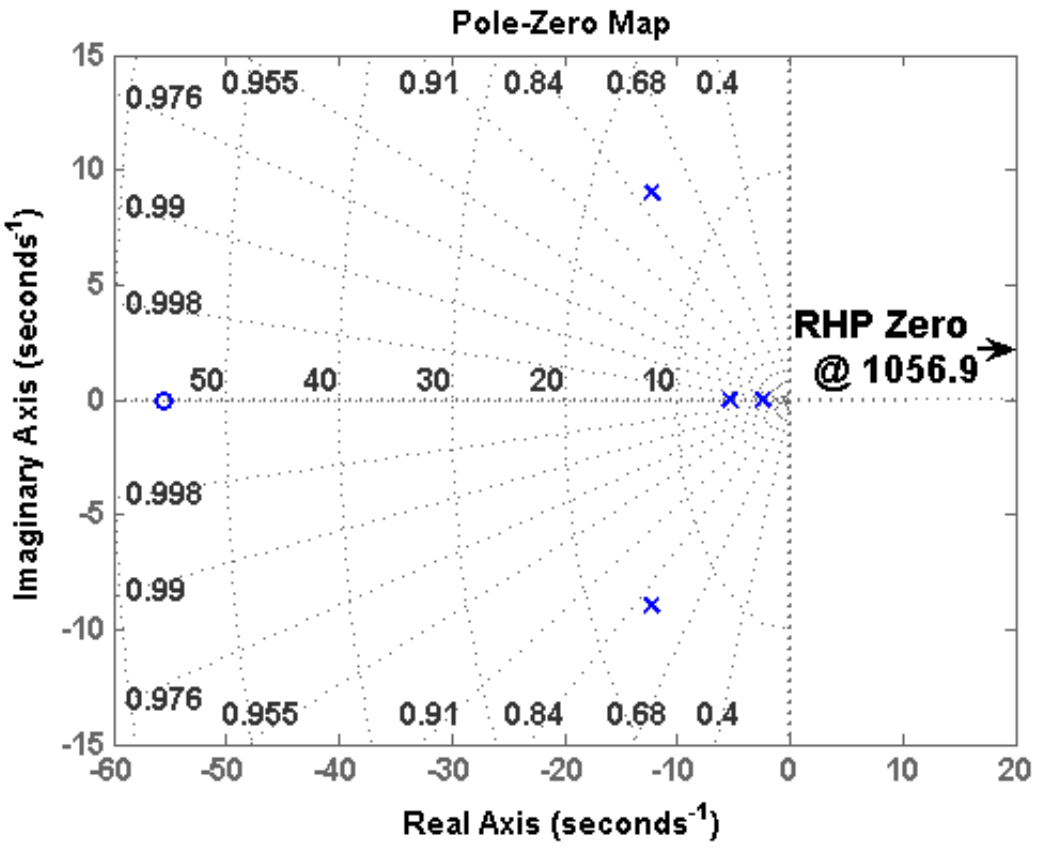

Figure 2.2 Linear Air System Model Pole-Zero Plot

and Eriksson (2010), Wahlström et al. (2009) show examples of RHP zeros in their models, and have determined that they are physically real. In a fuller analysis of the system dynamic equations, they conclude the RHP zeros are primarily due to the response lag of the turbochargers when changing operating conditions (also echoing the discussion above on the physical nature of the larger response time constant due to adjusting the VGT Vane position).

2.2.2 Singular Values

Skogestad and Postlethwaite (2005) mention that just using the eigenvalues of a system is a poor measure of the system gains for multivariable systems. Their reasons for this include that an eigenvalue analysis is applicable only for square systems, and that there are directional interaction effects present within the inputs and outputs that will impact on how well a given

control trajectory can be applied. Instead, their recommendation for multivariable systems is to use the Singular Value Decomposition (SVD), and perform an analysis on the system's singular values. They cite the ability of the SVD to apply to non-square systems, the ability to obtain directions for the inputs (which indicate the strong inputs, i.e. the inputs with the most amplification) and the directions for the outputs (on which the inputs have the most effect). Therefore, the singular values are used in the following system analysis in evaluating the system magnitude. Thus, the frequency dependent Singular Value decomposition provides a richer context of information for a MIMO system than do eigenvalues or a Bode plot.

As seen in Figure 2.3 (and also in Figure 2.4 for a comparison to classical control system evaluation), the frequency response of the engine air system has a flat shape at low frequencies and a -20 dB/decade roll-off at high frequencies; this is indicative of a relative first order system which shouldn't have any complications for control, and legitimizes the reduction of the fourth order system to a first order with time delay. However, there is an intermediate bandwidth region where the roll-off is initially -40 dB/decade before returning to the -20 dB/decade. While the higher initial roll-off rate would be beneficial to close-loop noise attenuation tuning goals, it also pushes the phase of the system towards the $-180°$ phase lag stability line. But based on what is seen in Figure 2.4, there is infinite phase margin as the higher order and non-minimum phase dynamics push the actual input/output pairings' phase to positive values. The gain margins seen in Figure 2.4 also show that the linearized engine air system model is stable.

One of the benefits of the singular values mentioned above was that of providing input and output directions. This becomes important when one studies the definition of the frequency dependent system gain. For SISO systems (as shown in Equation 2.5), the common measure of magnitude is used, and the definition should be familiar to many who study control theory. But in looking at Equation 2.6, the measure of magnitude has been generalized to use the 2-norm for the MIMO case, which can be thought of as the measure of the magnitude of a vector, so it is very similar to the measure of magnitude from the SISO case (in fact, the SISO definition is a special case of the MIMO definition). One can see though, that in the MIMO case, not only is the magnitude of each individual input signal important, but so is the combination of the magnitudes, from which the concept of input and output directions are obtained. This becomes

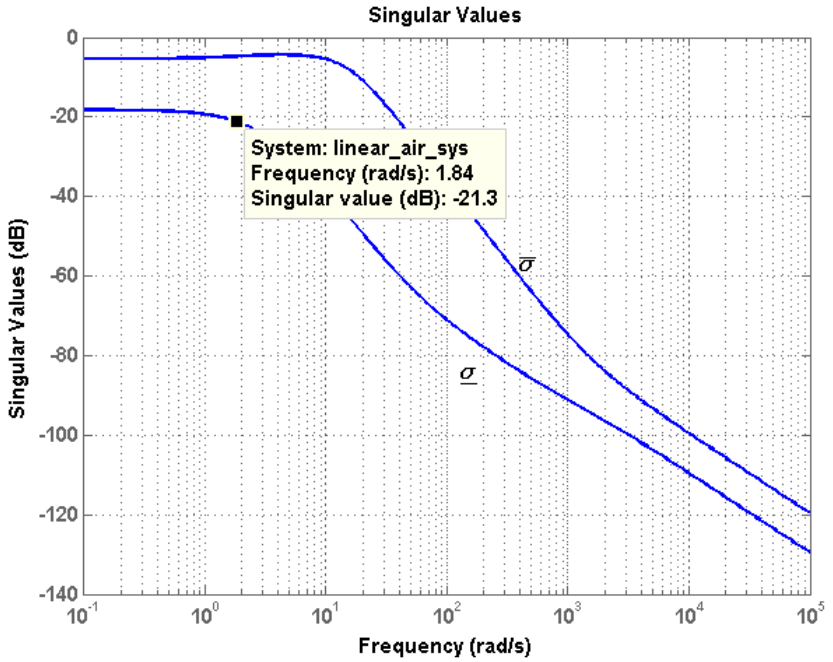

Figure 2.3 Open Loop Singular Values

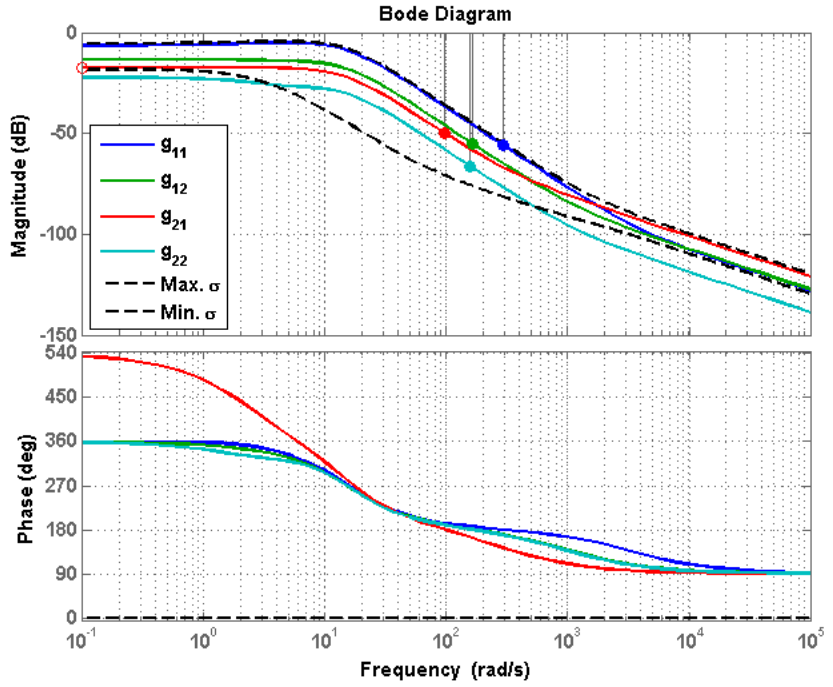

Figure 2.4 Singular Value - Bode Plot Comparison

important then because it suggests that the magnitude of each individual input/output SISO pairing may not give the full story when it comes to the system gain, and in fact this is seen in Figure 2.4. What the singular values then provide is a maximum and minimum bounds on the actual achievable MIMO system gain; in the middle frequency range, the MIMO system can achieve a much lower gain than the individual Bode plots indicate, and at the extreme high and low frequencies, the lowest system gain indicated by the individual Bode plots is not actually achievable due to the combination of individual inputs, and so the actual system gain is higher than the Bode plots would originally indicate (refer to Skogestad and Postlethwaite (2005) for more information on the benefits of the frequency dependent singular value decomposition).

$$\frac{|y(\omega)|}{|u(\omega)|} = \frac{|G(j\omega)\,u(\omega)|}{|u(\omega)|} = |G(j\omega)| \tag{2.5}$$

$$\frac{\|y(\omega)\|_2}{\|u(\omega)\|_2} = \frac{\|G(j\omega)\,u(\omega)\|_2}{\|u(\omega)\|_2} = \frac{\sqrt{y_1^2 + y_2^2 + \cdots}}{\sqrt{u_1^2 + u_2^2 + \cdots}} \tag{2.6}$$

In addition to the above discussion on achievable MIMO system gain, the singular values can also provide slightly better recommendations regarding the closed loop bandwidth. From Figures 2.3 and 2.4, one can see that the open loop bandwidth (i.e. the -3 dB reduction point) occurs at a much lower frequency on the $\underline{\sigma}$ curve than on the individual Bode plots of Figure 2.4. If the $\underline{\sigma}$ curve is treated as the worst-case achievable system gain, then this becomes a more appropriate measure to provide a recommendation for the closed-loop bandwidth than the individual Bode plots.

2.2.3 Condition Number

In analyzing the system further, a metric based on the singular values called the condition number is used. The condition number, defined as $\gamma = \overline{\sigma}/\underline{\sigma}$, is plotted in Figure 2.5 for the engine air system for various frequencies. One benefit of using the condition number as a method of analyzing a system, is that for multivariable systems it is independent of the controller input/output pairing, unlike the Relative Gain Array discussed later. Guidelines from Skogestad and Postlethwaite (2005) suggest that large condition numbers (where large is

arbitrarily chosen to be ≥ 10 and is in reference to a system that has been normalized according to the expected ranges of the signals) may indicate control problems - specifically sensitivity to input uncertainty - though this is not guaranteed. However, the opposite of that statement tends to hold up better, in that smaller condition numbers do imply insensitivity to input uncertainty. The very slight frequency peak seen in $\overline{\sigma}$ coupled with the steeper initial roll-off certainly do produce a noticeable peak in the condition number (Figure 2.5).

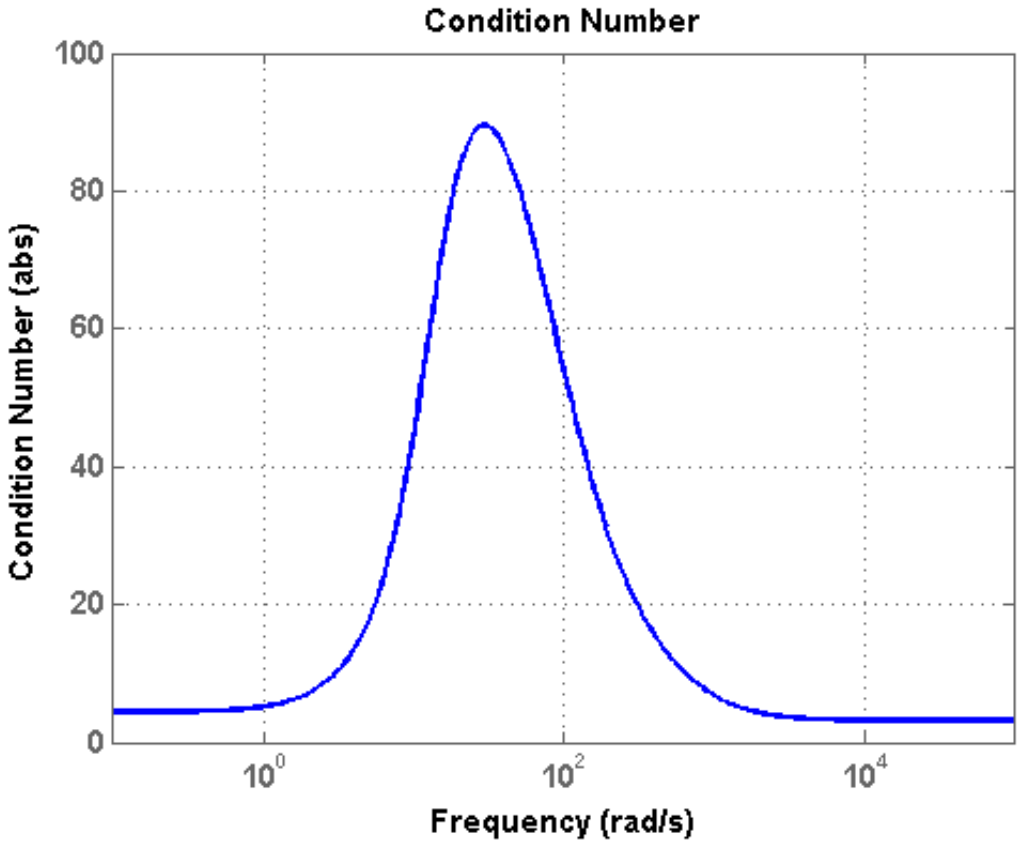

Figure 2.5 Open Loop Condition Number

If having the smallest condition number suggests more insensitivity to input uncertainty, it stands to reason one would potentially want to control the system at a bandwidth that minimizes the open loop condition number. The condition number for this system becomes ≥ 10 at approximately 4 rad/sec and remains so until approximately 600 rad/sec (this corresponds to the intermediate range of frequencies where the singular value plot had -40 dB/decade roll-off).

However, since closed-loop control serves to slow down the response from the initial open-loop system, this drives the bandwidth recommendation to the lower frequency range.

One thing to keep in mind about analyzing the system's condition number that Skogestad and Postlethwaite (2005) mention is that the condition number is very sensitive to the scaling of the inputs and outputs, therefore a initial and crucial step in multivariable plant analysis should be to scale the inputs and output appropriately. However, once a scaling has been established for a system the condition number is a useful system analysis tool.

2.2.4 Frequency Dependent Relative Gain Array

The next analysis tool applied to the engine air system is the Relative Gain Array (RGA), which is often used as a measure of interaction and is defined as $\Lambda(G) = G \times (G^{-1})^T$, where G is the system evaluated at a given frequency and $\times$ denotes element by element multiplication. The RGA has a benefit over the condition number in that it is independent of system scaling. However, it must be analyzed with regards to a specific input/output pairing, and would need to be repeated for each possible input/output pairing. While this is not a significant effort for a 2×2 system (as there are only two possible pairings), it does become much more difficult to analyze for larger numbers of possible input/output pairings. For this engine air system, the main diagonal pairing has been arbitrarily defined as controlling the percent EGR variable with the VGT Vane position, and the in-cylinder AFR variable with the EGR Valve position. Conversely, the off diagonal pairing is to control the percent EGR variable with the EGR Valve position and the in-cylinder AFR variable with the VGT Vane position. The frequency dependent RGA magnitude values for the engine system are plotted in Figure 2.6.

Because the frequency dependent RGA values are dependent on the input/output control pairing, Skogestad and Postlethwaite (2005) and Tham (1999) both present similar guidelines for selecting the most appropriate pairings based on the RGA magnitudes. To summarize, these pairing rules as defined in (Skogestad and Postlethwaite, 2005, p. 449) are as follows:

1. Prefer pairings such that the rearranged system, with the selected pairings along the diagonal, has an RGA matrix close to identity at frequencies around the closed-loop bandwidth.

2. For a stable plant, avoid pairings that correspond to negative steady-state RGA elements.

3. Prefer a pairing ij where g_{ij} puts minimal restrictions on the achievable bandwidth. Specifically, the effective delay θ_{ij} in $g_{ij}(s)$ should be small.

Equation 2.7 shows the steady-state RGA values for the linear air system model on the left, and it is clear all of the values are positive therefore satisfying pairing rule 2 with either of the input/output pairing options. In addition, looking at which point in Figure 2.6 becomes close to I for either pairing, it can be seen that both pairings become close to 1 near 4 rad/sec. This helps to support the bandwidth arguments from earlier, but could also be indicating that either pairing could be preferable, although at this point the matrix term magnitudes would not be near the identity matrix I, but instead would be a matrix full of 1s. Therefore, by reducing the bandwidth requirement a bit to 2 rad/sec, the main diagonal pairings $(\lambda_{11}, \lambda_{22})$ of the RGA matrix (Equation 2.7, on the right) have a magnitude that is closer to I when compared to the off diagonal values. Therefore by pairing rule 1, this would suggest that the main diagonal pairing is preferable to the off-diagonal pairing, but not by much.

$$\Lambda(0) = \begin{bmatrix} 0.5597 & 0.4403 \\ 0.4403 & 0.5597 \end{bmatrix}, \quad \Lambda(2) = \begin{bmatrix} 0.5517 + 0.4056j & 0.4483 - 0.4056j \\ 0.4483 - 0.4056j & 0.5517 + 0.4056j \end{bmatrix} \tag{2.7}$$

While pairing rule 3 doesn't necessarily fall under the RGA analysis, one can use the transfer function array from Equation 2.3 to evaluate this pairing rule. Looking at the delays in transfer function matrix, the minimum delay is present in the main diagonal pairing. Similarly the largest delay is present in the off-diagonal pairing, suggesting that the main diagonal pairing is preferred and the off-diagonal pairing should be avoided.

Returning more to the RGA analysis, Skogestad and Postlethwaite (2005) mention that plotting just the magnitudes of the frequency dependent RGA (as was done in Figure 2.6) can be somewhat misleading, because an RGA element of -1 (which is an undesireable pairing) has a magnitude of 1. As a remedy, they introduce the RGA Number, which they have defined as the sum norm of the RGA at the various frequencies (see Figure 2.7). Extrapolating from pairing rule 1, they provide a simple rule that recommends pairing around small RGA numbers. As can be seen at all frequencies in Figure 2.7, the main diagonal pairing is smaller than the

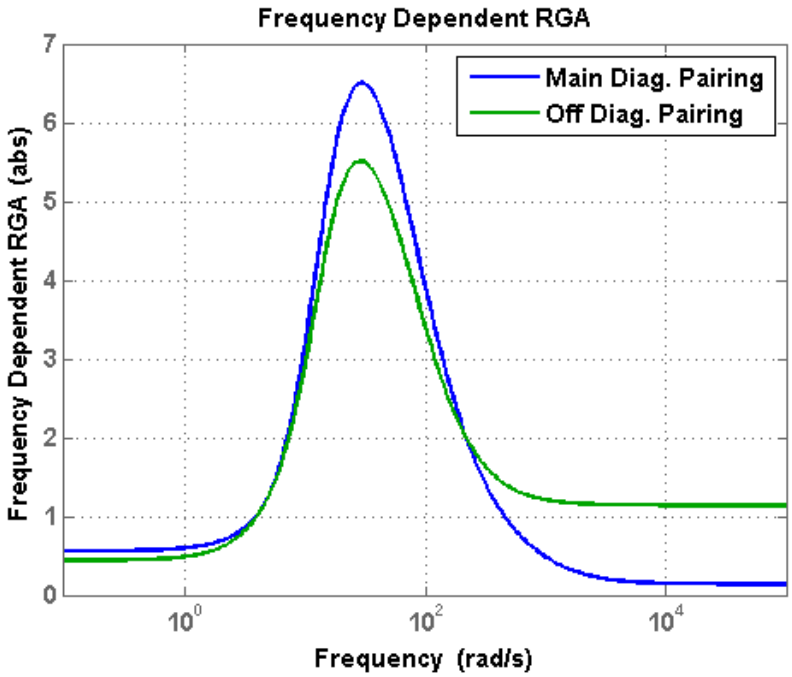

Figure 2.6 Open Loop RGA Magnitudes

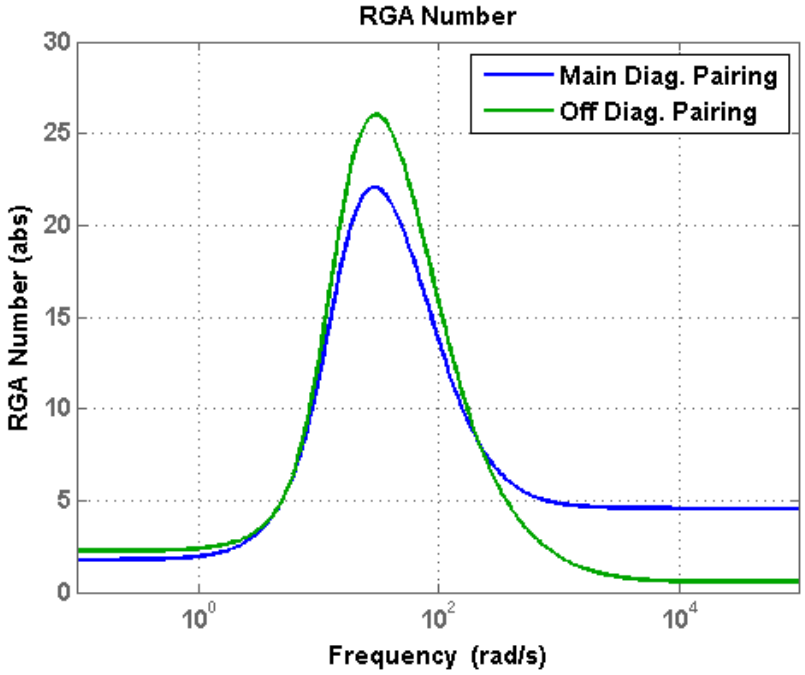

Figure 2.7 Open Loop RGA Number

off-diagonal pairing, yet again confirming the arbitrarily made pairing selection for this study. The main downside with the RGA number mentioned by Skogestad and Postlethwaite (2005) is the same as with the RGA itself - each individual pairing option must be evaluated separately. An interesting note from comparing Figures 2.6 and 2.7 is that they have the same shape and relative magnitudes, but the actual magnitude values differ and which curve corresponds to which input/output pairing is swapped; this is to be expected given the definition of the RGA Number.

2.2.5 Diagonal Dominance

In addition to the frequency dependent RGA, Skogestad and Postlethwaite (2005) discuss another measure of interaction which can also provide some guidance into input/output pairing selections. This measurement is known as "diagonal dominance", and is calculated as shown in Equations 2.8 and 2.9. The matrix $\tilde{G}$ is defined as having the only desired input/output pairings of G along the diagonal, with all other elements 0. By evaluating the "magnitude" of the matrix E (and subsequently the matrix E_S, which is derived from a factorization involving the sensitivity function S), a "measure of interaction" is obtained that can be used to evaluate how interactive the pairings are, which can also be interpretted as how difficult are they to decouple. Skogestad and Postlethwaite (2005) state that the use of the structured singular value of these matrices ($\mu(E)$ and $\mu(E_S)$) are the best (i.e. least conservative) measure of interaction, and that "generalized diagonal dominance" is defined as $\mu(E) < 1$.

$$\tilde{G} = diag\{g_{ii}(s)\} = \begin{bmatrix} g_{11}(s) & & & \\ & g_{22}(s) & & \\ & & \ddots & \\ & & & g_{mm}(s) \end{bmatrix} \tag{2.8}$$

$$E = \left(G - \tilde{G}\right)\tilde{G}^{-1},\ E_S = \left(G - \tilde{G}\right)G^{-1} \tag{2.9}$$

By evaluating $\mu(E)$ and $\mu(E_S)$ at various frequencies (shown in Figures 2.8 and 2.9), the first item that is clear is that none of the curves completely satisfy the "generalized diagonal

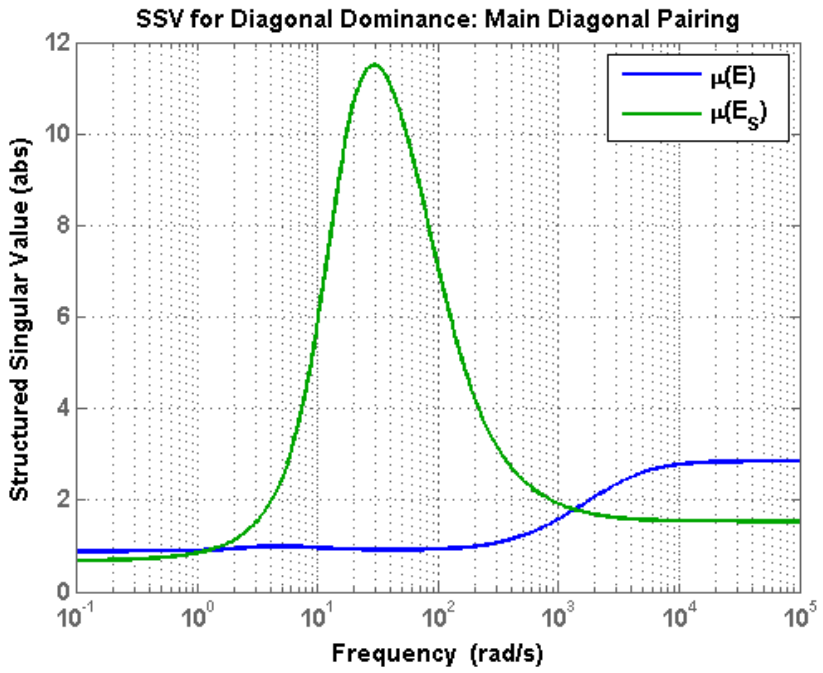

Figure 2.8 Diagonal Dominance - Main Diagonal Pairing

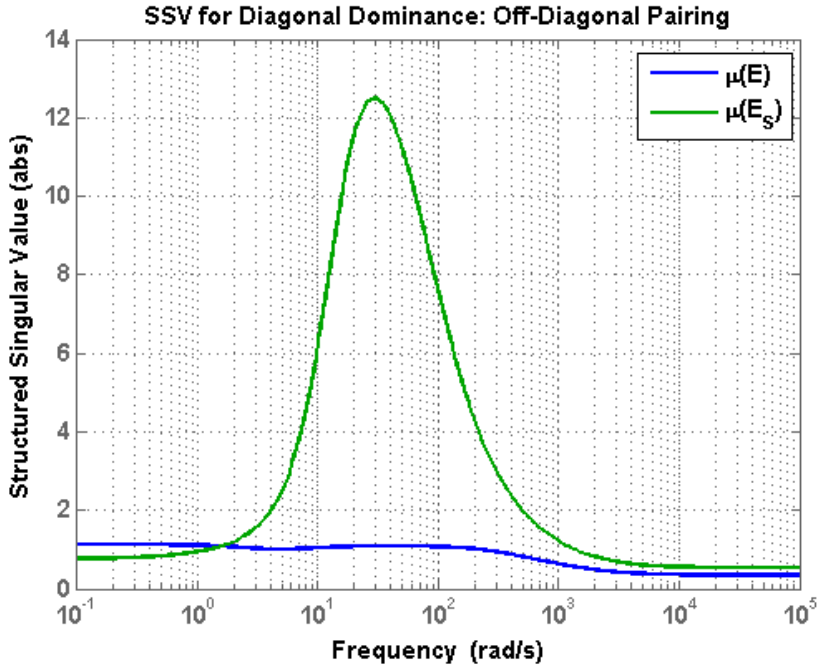

Figure 2.9 Diagonal Dominance - Off-Diagonal Pairing

dominance" criteria mentioned above for all frequencies. However, this leads into the notion that there may be input/output pairing and bandwidth guidelines that can be obtained from looking at the plots. In fact, Skogestad and Postlethwaite (2005) do provde two additional pairing rules that are listed below, and support the three pairing rules mentioned in Section 2.2.4.

1. Prefer pairings with $\mu(E) < 1$ ("diagonal dominance") at frequencies within the closed-loop bandwidth

2. Prefer pairings with $\mu(E_S) < 1$ ("diagonal dominance") at crossover frequencies

Applying these pairing rules to the linear engine air system model, it shows that the off-diagonal pairing should not be used, as $\mu(E)$ is greater than 1 until approximately 200 rad/sec, thus violating pairing rule 1. For the main diagonal pairing, it does actually stay below 1 until about 200 rad/sec, though it does come close around 5 to 6 rad/sec. Applying pairing rule 2 then to the main diagonal pairing supports earlier arguments for selecting a closed loop bandwidth of less than approximately 2 rad/sec, so that $\mu(E_S)$ is kept less than 1 around the crossover frequencies.

2.3 Linear Model Analysis Summary

This chapter described the system identification procedure followed to create a fourth order linear engine air system model from a high fidelity non-linear engine mode. Initial analysis into the open loop system model revealed that the system is stable, but may present some control challenges due to a complex pole pair (introduces slight oscillations), a RHP zero, and other non-minimum phase higher order dynamics. The analysis of the singular values and Bode plots showed regions where the minimum gain of the individual Bode responses was not the actual achievable minimum system gain and provided initial input into a closed-loop bandwidth recommendation of approximately 2 rad/sec.

The next analysis (condition number) again provided input to the closed-loop bandwidth recommendation in terms of keeping the system insensitive to input uncertainty. However, this recommendation was higher than what the initial recommendation was from the singular values (at 4 rad/sec compared to 2 rad/sec from the singular value analysis), suggesting that

closed-loop control at the initial bandwidth will provide some robustness to the closed-loop design. Using the bandwidth recommendation of 2 rad/sec, the next set of analyses are used to provide guidance on a prefered input/output pairing.

The first of these analyses (frequency dependent RGA) used three pairing rule recommendations (along with the reduced transfer function matrix from Equation 2.3) to suggest that the initially (and arbitrarily) defined main diagonal pairing of controlling the percent EGR with the VGT Vane position and the in-cylinder AFR with the EGR Valve position should be the pairing that the controllers are designed around. The final analysis on the open loop linear engine air system model confirmed this pairing, as the application of the diagonal dominance pairing rules clearly showed the main diagonal pairing to be prefered.

In summary, the following conclusions about the engine air system model are being made.

- While the engine air system model is stable, there are aspects about it that suggest some potential control challenges, namely a RHP zero (causes an inverse initial response) and a complex pole pair (signifies oscillation in the response)
- Because the plant would become ill-conditioned and sensitive to input uncertainty (as seen in the condition number plot), the recommendation for closed loop control is to keep the bandwidth of the system below 2 rad/sec
- Even though an intuitive first thought would be to control the percent EGR variable with the EGR Valve position (which is a very direct relationship), the pairing rules from the RGA and diagonal dominance analyses suggest that creating the input/output pairing of controlling the percent EGR with the VGT Vane position (and thus the in-cylinder AFR with the EGR Valve position) is a preferable pairing in order to achieve a decoupling controller design

The last item to mention before diving in further to the controller analysis is that with the recommended closed-loop bandwidth, the SIMC tuning rules (see Skogestad (2003) and Skogestad (2006) will be applied to the baseline PI controllers, and will be tuned to target a desired first order closed-loop response. The 2 rad/sec bandwidth recommendation corresponds to a 0.5 sec response time constant, and the desired closed-loop step response for the main

diagonal elements is shown in Figure 2.10 (both will be tuned to the same response). Applying the decoupling requirement from earlier implies that the off-diagonal elements will be zero, and are therefore not shown here. This desired response is also applied later in this study to filter the setpoint signals for the engine case study simulations, and the Integral Square Error metrics are calculated as being the error from the actual response back to the desired response, not the setpoint.

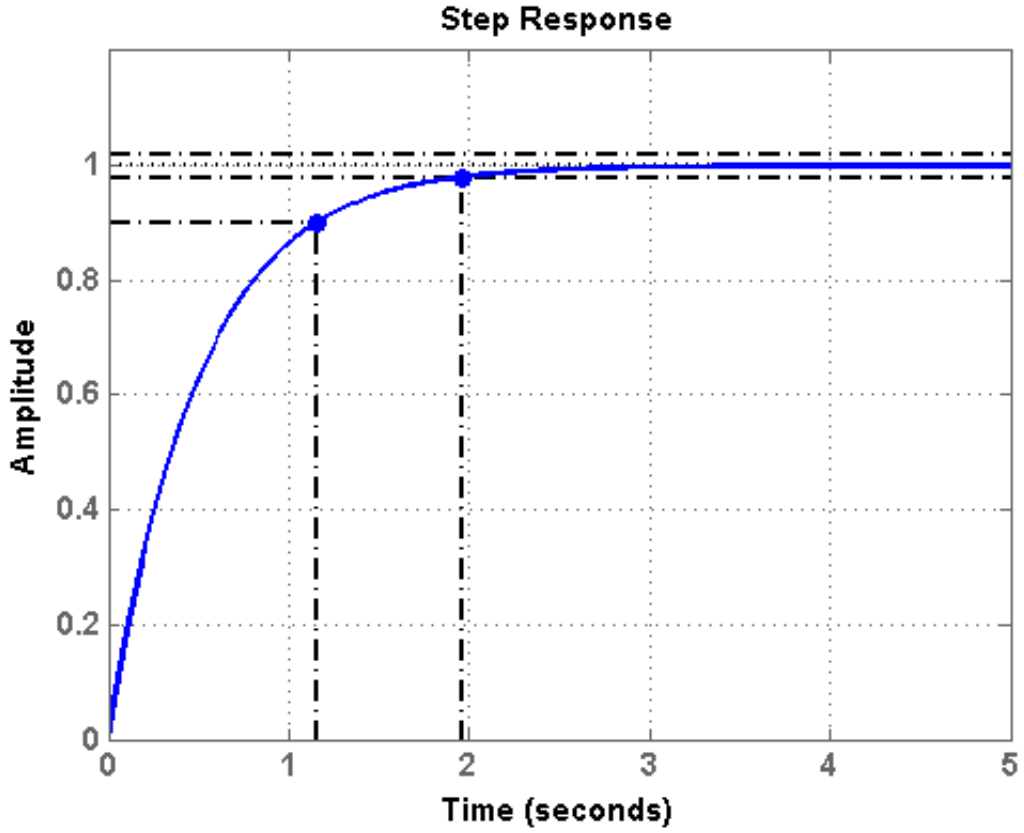

Figure 2.10 Closed Loop Reference Response

Tuning the closed-loop response to be first order also implies that the steady-state value will match the input setpoint (i.e. zero steady-state error), as well as a few other closed-loop control specifications.

1. The targeted overshoot for the response is 0%, as first order systems have no overshoot.

A driving factor to impose this requirement is that "spikes" (i.e. overshoots) in the controlled variables correlate to spikes in the higher level regulated emissions, which in turn leads to a poorer certification score and possible emissions non-compliance.

2. The standard first order response equation $y(t) = 1 - e^{-t/\tau}$ can be solved for a 90% response and rearranged as a function of the time constant as $t = -\tau ln\,(0.1)$. Using the desired first order response time constant of 0.5 sec (derived from the recommended bandwidth of 2 rad/sec), the rearranged response equation can be solved to give the desired 0% to 90% rise time of 1.1513 seconds.

3. Since the first order response has no oscillation, the 2% settling time is achieved when the response is first within 2% of the final value. The first order response equation can similarly be rearranged as $t = -\tau ln\,(0.02)$ and solved for the 2% settling time using the desired response time constant, resulting in a settling time target of 1.9560 seconds.

These values are shown later on in Table 4.1 in order to compare the various controller designs back to this closed-loop control specification list.

CHAPTER 3. EXISTING CONTROLLER ARCHITECTURE ANALYSIS

Various existing multivariable control architectures are analyzed here and compared to a simple baseline of diagonal PI controllers. Prior to analyzing the various architectures' performance, a review of existing literature on diesel engine control as well as decoupling control is performed. Aspects of the various controller architectures that will be analyzed are performance response and decoupling capability of the linear model to a standard step response, as well as a case study simulation against a high fidelity, mean-value, non-linear engine model around the Mode 1 linearization point.

3.1 Literature Review

The addition of sensors and actuators to modern diesel engines to regulate emissions levels opens up a multitude of control architecture possibilities, and as a result there is a great deal of variation in the literature when it comes to implementing closed-loop diesel engine air system control. The existing control strategies reviewed have been divided into two main groups: architectures that appear to maintain simplistic architecture and/or strategy (i.e. using one to a few PID controllers, either one at a time or simultaneously), and architectures that explore more complex controller designs.

In Friedrich et al. (2009), two different arrangements of a single PI controller with table based feed-forward control are considered, and the trade-offs between performance and emissions are studied. The architecture is kept relatively simple, with the controller determining an overall effective flow area, which is arbitrated into specific EGR valve and intake air throttle commands[5]. In most cases thought, the additional actuators and sensors result in more control loop possibilities that are often closed with multiple simple controllers (i.e. this is the

[5]Intake air throttles are another commonly used air system actuator for emissions control

decentralized control architecture that Skogestad and Postlethwaite (2005) discuss) such as a PID. The work of Chauvin et al. (2007) and Wahlström and Eriksson (2009), Wahlström and Eriksson (2010) all use parallel (i.e. simultaneously operating) PI controllers and some aspect of a simplified physical engine model to achieve control. In Chauvin et al. (2007), an observer based on a simplified engine model is used to estimate the EGR mass flow, which is then used as a control variable for one of the PI controllers. In Wahlström and Eriksson (2009), Wahlström and Eriksson (2010) two parallel PI controllers are used to determine desired flow rates at various points on the engine. Then a static, first principles based, non-linear input transformation is used to obtain the desired actuator position. The inherent simplicity of the PID controller - including their capability to apply a consistent architecture across multiple systems - makes their use prevalent in many control situations, including engine control; however, just using simplistic architectures (such as a PID) still introduce additional challenges in that the tuning of the controller has an impact on the performance and emissions of the engine, as seen in Wahlström et al. (2010). The optimization routine proposed also illustrates common trade-offs that are encountered which cause challenges in tuning the controller, as was done in the work of Friedrich et al. (2009).

In the works reviewed above, the use of the PID controller reflects on a desire of the designers to keep the overall system simple. However the use of a PID controller can lead to undesired response lag in the system. This response lag can lead to non-optimal control results, especially with multiple control loops active, as seen in the results of Shirakawa et al. (2001). As was noted in their work, a significant reduction in smoke generation was achieved by the use of a co-operative model-based control strategy. The use of more co-operative, more complex, or more advanced control strategies results from balancing the increasingly stringent steady-state and transient control requirements against the customers' performance expectations. Therefore, the use of these advanced control strategies has been an increased topic of research in the area of engine emissions control. The work of Herceg et al. (2006) studies the use of a non-linear model predictive controller to achieve emissions control while handling the non-linear aspects of the engine behavior as well as actuator constraints. The non-linear behaviors of engine air system control in Pfeiffer and Hedrick (1999) are handled by transforming the non-linear control

problem into a linear regulation problem by use a sliding mode controller[6]. Their coordinated control method provides tighter control of the desired air-fuel ratio while tracking the desired engine speed trajectory. In Omran et al. (2009), a simplified engine model is used to perform complex, non-linear optimizations of the engine's control variables to maximize power and minimize emissions. However the authors state this optimization routine cannot be run on an ECU because of a limitation of computational throughput. The authors then mention using the results from the optimization to create a series of lookup tables for more conventional control methods; however, it is suggested then that the optimization routine would create a large number of tables and available memory in the ECU becomes an issue. In order to implement the results of the optimization routine, the selected approach of the authors was to train a neural network controller that can be implemented on an ECU and showed that this approach realized a noticeable reduction in smoke generation. One potential downside of this approach, however, is that a very large database of system responses will need to be generated to ensure that sufficient, if not optimal, control is maintained under all possible operating conditions and use cases of the engine.

As part of trying to maintain complex control objectives as well as keep the controller design simple, methods of decoupling control are sometimes employed. In the work of Jung and Nam (1999), the use of decoupling control is used for electrical induction motors at high speeds where the coupling voltage becomes significant and has an impact on the ability to have tight, fast torque control of the motor. Various types of decoupling architecture are explored, including feed-forward decoupling designs from measured variables, feed-forward decoupling from the PI controller inputs (tracking variable error similar to P-canonical form from Tham (1999)), and cross coupling control before settling on a style of decoupling control containing types of cascade elements. In Tham (1999), the architecture studied here for decoupling control is shown in what is called a V-canonical form, but only the feed-forward style of decoupling control (the P-canonical form) is studied in depth. Closed form solutions are determined for the P-canonical form of decoupling, including some discussion on implementation issues, and

[6]Even though this work is for a spark ignited engine, the control challenges of emissions reduction and air path management are still quite similar to compression ignition (i.e. diesel) engines

will be evaluated in this paper for engine air system control.

3.2 Existing Controller Architectures: Design, Simulation, and Analysis

The following sections discuss a few common controller architectures that may be used to control multivariable systems. First, a high level design procedure is discussed, before presenting and discussing the step results of the controller and the linear air system model. The step response is used in order to capture common performance metrics and provide a more "apples-to-apples" comparison of the different architectures. Each controller analysis is then followed up by a simulation of a high fidelity diesel engine model with commands generated as one would see from a Non Road Transient Cycle (NRTC) certification test. The NRTC commands have been modified for this study though, first off because the engine air system was linearized at a specific speed/load operating point to design the controllers. Therefore analysis of these controllers on the engine model are really only valid within a region around this point. Second, there are regions within the operating envelop where the EGR Valve is commonly closed, resulting in mode switching of the controllers. Since this was not part of the original analysis scope of this study, the speed and load commands of the NRTC have been limited to the top 25% and 50% ranges, respectively, for the case study simulation. This also serves to keep the operation of the engine closer to the linearization point. Even with these modifications to the engine simulation setup, it still provides ample excitation of the engine and controllers to obtain a realistic comparison of how one could expect these controllers to perform in the real world.

3.2.1 Diagonal PI Controller Architecture

The first architecture design analyzed will be based on a PI controller, and will establish the performance response against which the other architectures are compared. The individual PI controller used throughout this paper is defined in Equation 3.1 and is common throughout industry and known as the "ideal" or parallel design.

$$u = K_c \left(1 + \frac{1}{\tau s}\right) e \tag{3.1}$$

Skogestad and Postlethwaite (2005) define a decentralized diagonal control architecture generically[7] as shown in Equation 3.2 (where m is the number of plant inputs and outputs, and is equal for a square system), and shown specifically in Figure 3.1 for a 2×2 system. This is the baseline diagonal PI controller architecture used as our initial performance analysis because it follows similar trends in the literature reviewed above, and provides the foundation for the remaining architectures in this study.

$$\mathbf{K}(s) = diag\{k_{ii}(s)\} = \begin{bmatrix} k_{11}(s) & & & \\ & k_{22}(s) & & \\ & & \ddots & \\ & & & k_{mm}(s) \end{bmatrix} \tag{3.2}$$

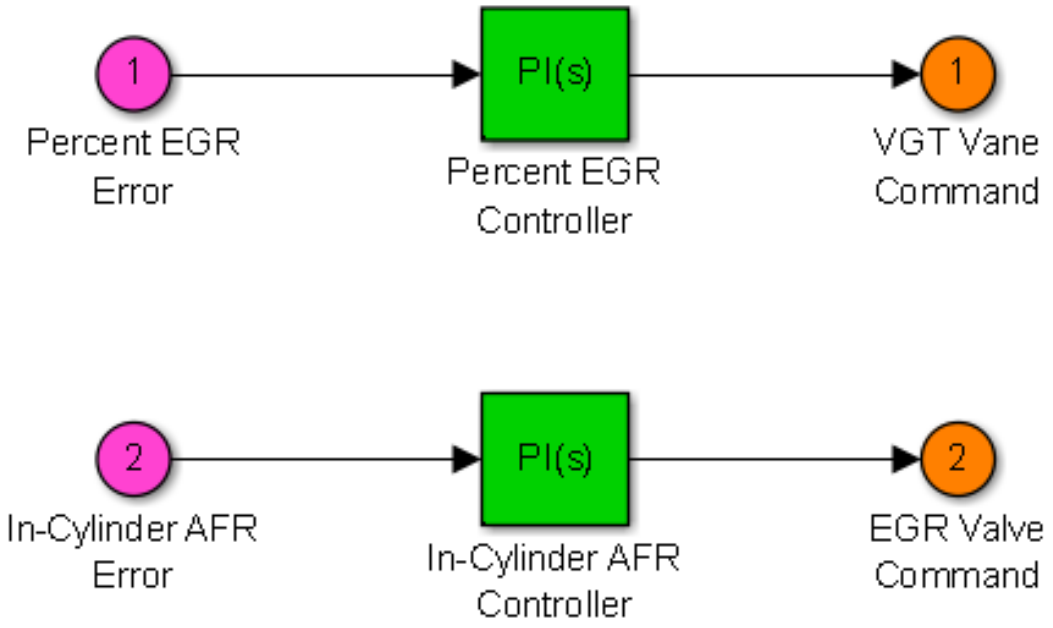

Figure 3.1 Block Diagram of Diagonal PI Controllers - Baseline Architecture

Using the controller pairing and bandwidth response guidelines from 2, the individual PI

[7]The equation from Skogestad and Postlethwaite (2005) has been slightly modified to show two subscript indices, so that the main diagonal controller elements are represented in a similar fashion to the off-diagonal decoupling elements later on.

controllers - one to control the percent EGR variable with the VGT Vane position and the other to control the in-cylinder AFR variable with the EGR Valve position - are tuned using the methods described in Skogestad (2003) and Skogestad (2006). To further validate the SIMC model reduction rules of Skogestad (2003) for the diesel engine air system, the individual closed loop responses are isolated from each other and simulated individually, considering no other interactions. The results are shown for the percent EGR and in-cylinder AFR variables in Figures 3.2 and 3.3, respectively. In both plots, the closed loop response of the fourth order linear plant model is shown in red, while the closed loop response of the SIMC reduced first order with delay model is shown in green. Both sets of responses are also compared to the desired reference response shown in blue (repeated here from Figure 2.10).

Both diagrams show that the individual closed loop results - and therefore the initial PI tunings - are quite satisfactory in terms of matching the desired response. However, various works in system architecture design all indicate that the performance of a system is defined by more than just the functionality of the components, the interactions between components are also of significant importance. In that context then, the individual PI controllers and control loops are just components within the larger system of controlling the diesel engine air path. Therefore, we should expect that interactions between these loops (as supported by the previous discussions with regards to the air system's non-linearity, internal feedback nature, and control implications due to non-minimum phase elements) to potentially have an impact on the overall system performance. This is indeed the case, as shown in the closed-loop step response in Figure 3.4 of the 2×2 linear air system model with the baseline diagonal PI controller architecture.

As can be seen from the step response in Figure 3.4, each of the main diagonal response have deviated from the desired reference response, and also when compared to the isolated, individual results. This proves out the idea of the system performance being the aggregation of not just the component performance but also interactions between the components, and also supports the claim for more complex, fully multivariable, and/or coordinated designs. But since the main diagonal responses still settle out within a reasonable time after only a moderate overshoot, one could still count these responses as being acceptable. However, the effect on the

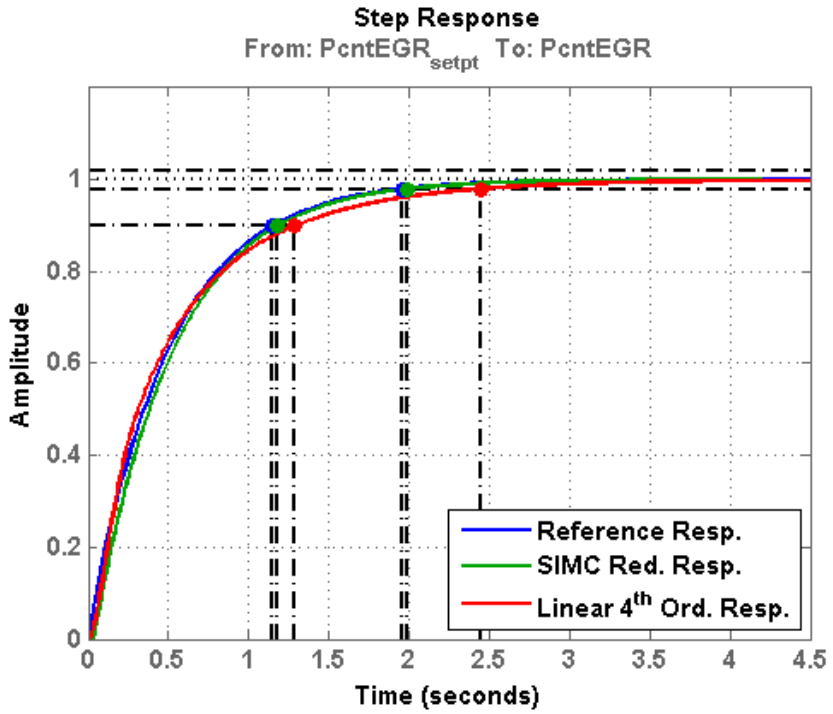

Figure 3.2 Individual Closed Loop Percent EGR Response

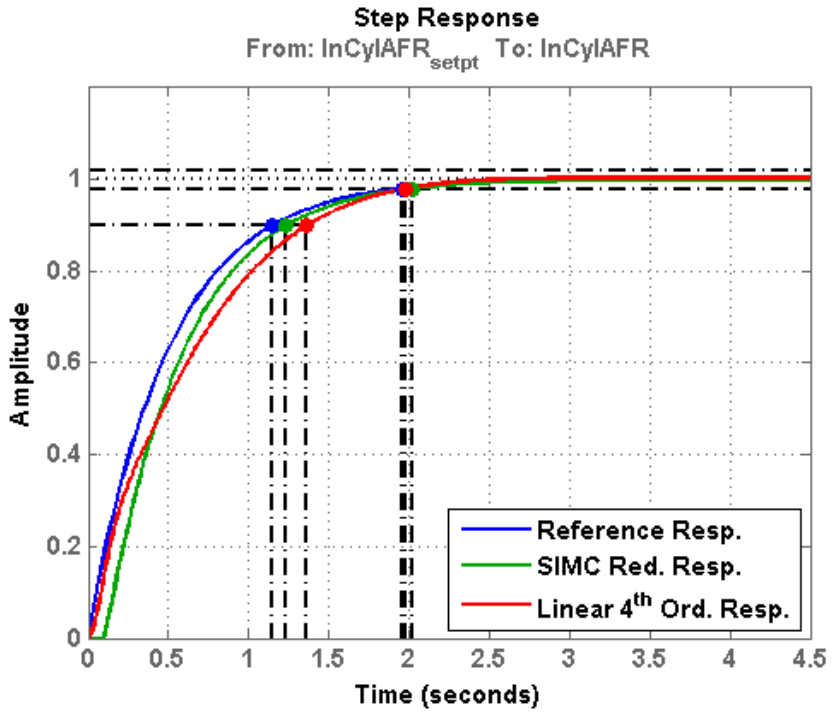

Figure 3.3 Individual Closed Loop In-Cylinder AFR Response

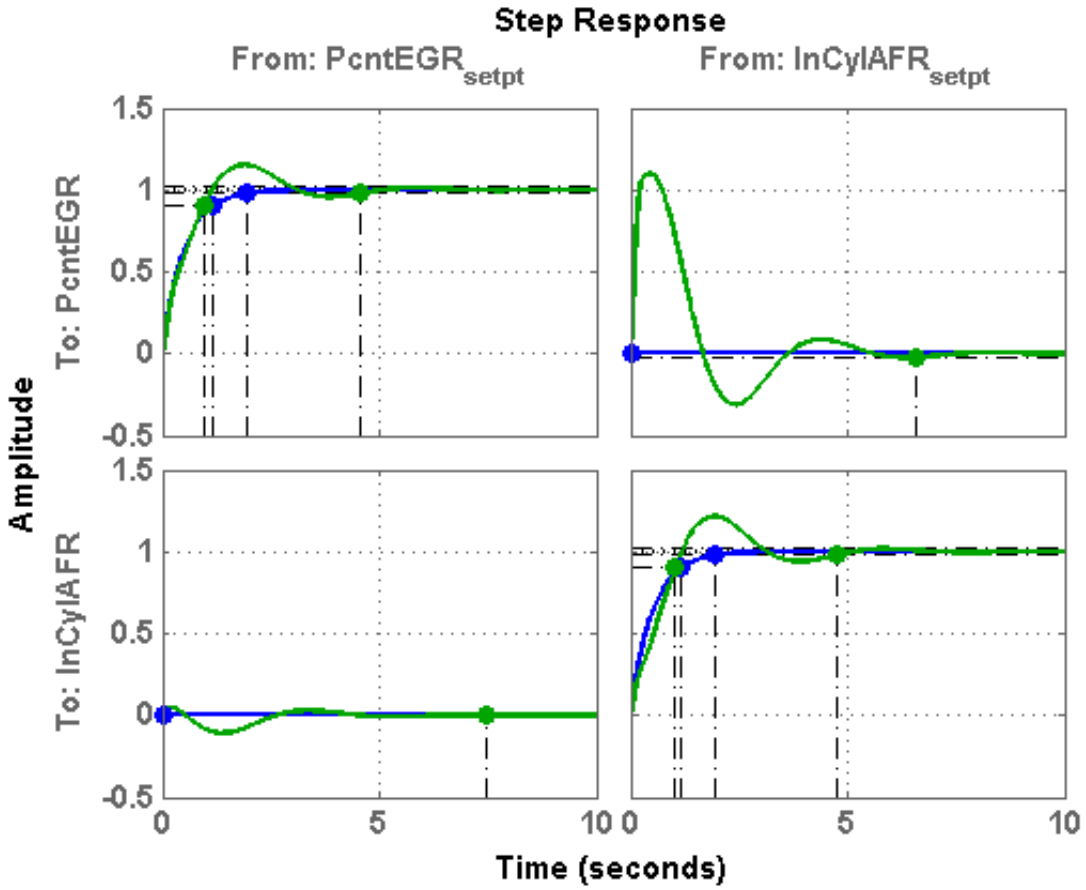

Figure 3.4 Diagonal PI Controller Closed Loop Step Response

percent EGR variable due to the in-cylinder AFR setpoint change is much different from the desired zero response of a decoupled system. This response clearly shows the highly interactive nature of the air system and supports the desire for decoupling so that the system can be analyzed easier and internal disturbances can be reduced or eliminated.

Looking at the responses of the controlled variable for this case study engine simulation (Figures 3.5 and 3.6), it is clear that the closed loop system is indeed stable and the setpoint is tracked fairly effectively. There are overshoots and undershoots that are larger than what the linear step response analysis from above would suggest should occur, but this highlights either the non-linear nature of the diesel engine air system model or the impacts of operating too far from the linearization point. From the linear step response results and the interactive nature of the engine air system, one should not be too surprised at these over- and undershoots. There

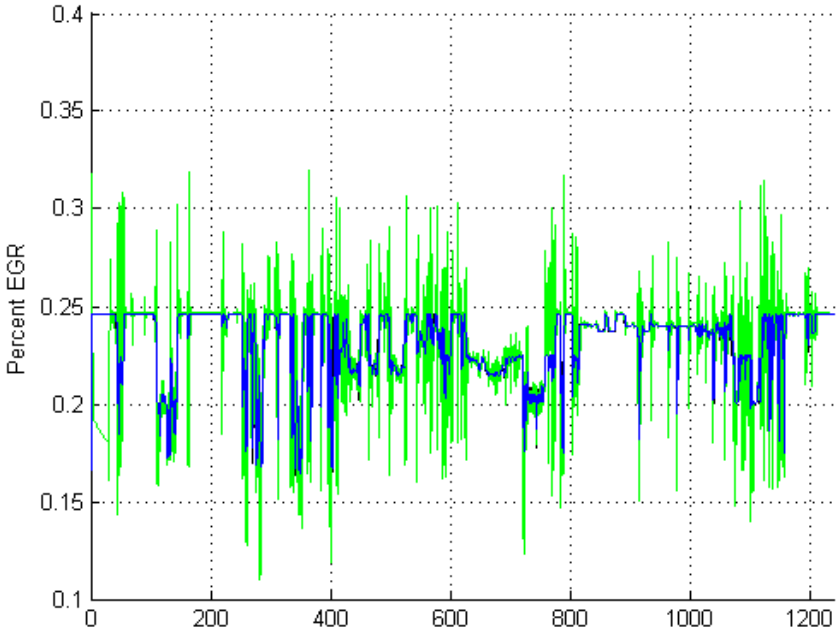

Figure 3.5 Diagonal PI Controller, Percent EGR Response

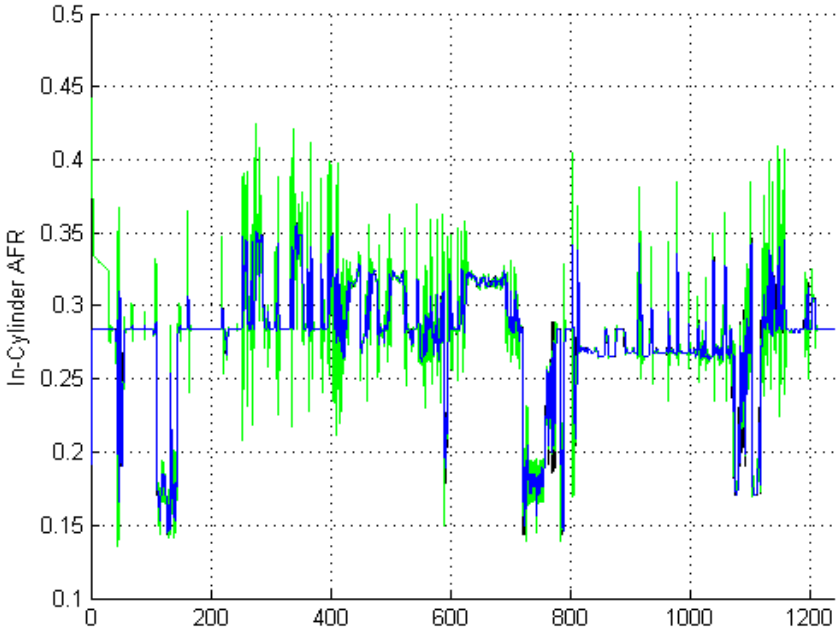

Figure 3.6 Diagonal PI Controller, In-Cylinder AFR Response

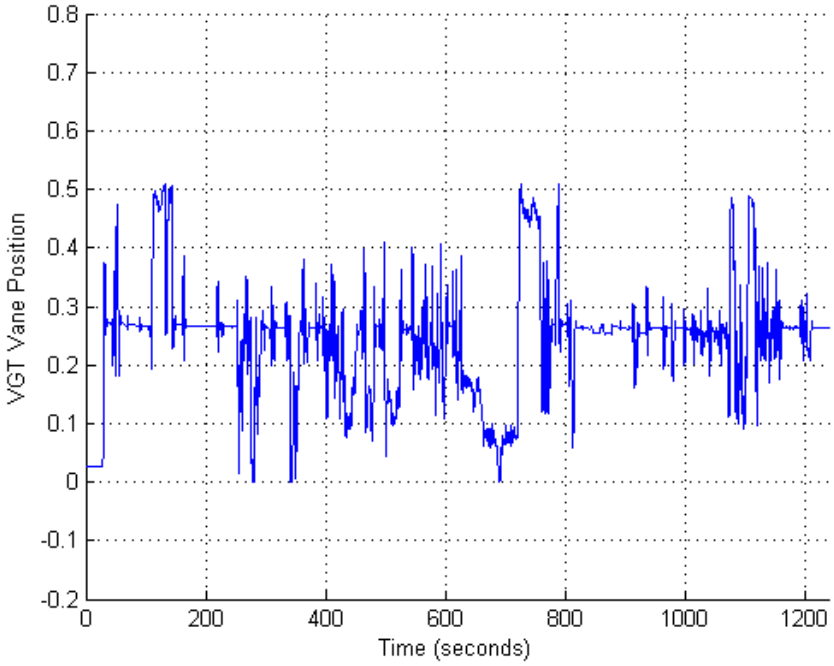

Figure 3.7 Diagonal PI Controller, VGT Vane Command

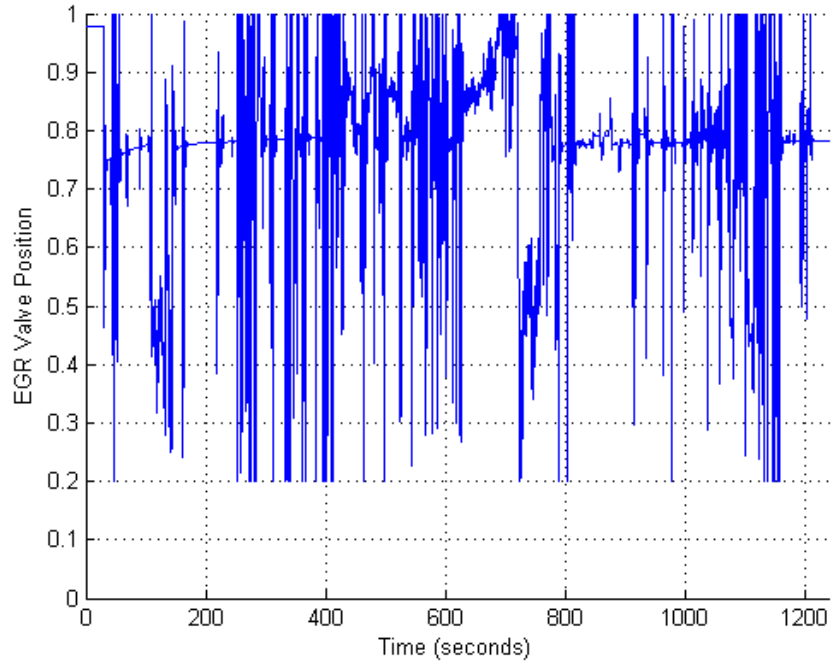

Figure 3.8 Diagonal PI Controller, EGR Valve Command

is additional motivation to decoupling the system in removing these internal disturbances, and that is that the controlled variables here correlate well to the high level regulated emissions, therefore reducing the internal disturbances should help reduce the maximum peaks (i.e. maximum overshoots), which will also reduce spikes in the regulated emissions during a certification test.

The plots of the VGT Vane and EGR Valve commands (Figures 3.7 and 3.8, respectively) show the activity of the two actuators. While there appears to be a relatively constant position held by the two (approximately 0.275 for the VGT Vanes and 0.775 for the EGR Valve), one can see the deviations the controllers impose. Though the frequent full range swings of the EGR Valve may be undesirable for an actual production engine, it'll be left as is in order to provide a baseline for comparisons. In contrast, the VGT Vanes appear to be much more stable in terms of their activity.

Reviewing the discussion and plots above, the following conclusions are made about the baseline Diagonal PI architecture.

- The benefit of the Diagonal PI design is that it maintains a simple control structure, and there is plenty of literature providing all manner of guidelines for tuning a PID controller (e.g. Zeigler-Nichols, SIMC, etc).

- It is possible to achieve decoupled control with this, provided that the system is not terribly interactive. One would need to perform sufficient analysis on a system to determine if this is the case; as it turned out the engine air system proved to be a very interactive system.

- For highly interactive systems, the simplicity of the Diagonal PI design hides a trap that is easy for control designers to fall into, and that is tuning the individual PI controller elements in an isolated manner and assuming that the final MIMO system will function as desired. This certainly turned out to be the case for the engine air system.

- The Diagonal PI controller also appears to not have sufficient ability to decouple a system effectively in the case of highly interacting plants. This introduces the need for more complex control designs, which in turn begs the question of how much added complexity.

3.2.2 Feed-forward Decoupling Design

As a result of the performance results from the baseline diagonal PI controller and continuing the argument to design a decoupling controller, the next controller architecture studied is an attempt to do just that. The design used here is a common, "intuative" next step when deciding upon a decoupling controller architecture defined in Tham (1999) as the P-canonical form, and is shown in Figure 3.9. To support the idea of this being a common first-step decoupling architecture, Tham (1999) mentions a different decoupling architecture (termed the V-canonical form), but only gives discussion to the P-canonical form. In Jung and Nam (1999), the first few decoupling controllers designed had the decoupling elements arranged in a feed-forward manner, however they found these feed-forward architectures were sensitive to plant model parameteric uncertainty.

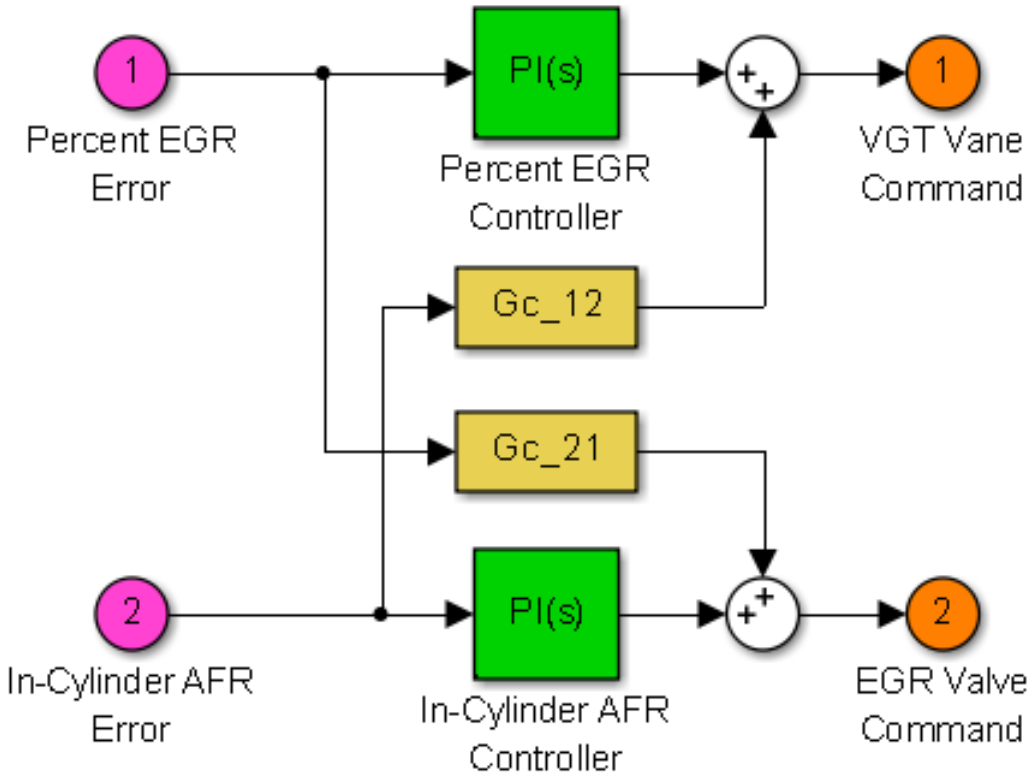

Figure 3.9 Block Diagram of Feed-forward Decoupling Design

The design procedure for this controller architecture begins with the diagonal PI architecture (and tuning) from the previous section and add controller elements in the off-diagonal locations of $\mathbf{K}$ in Equation 3.2 and shown as blocks $G_{c,12}$ and $G_{c,21}$ in Figure 3.9. With the same set of starting conditions and desired decoupling approach, Tham (1999) derives solutions for these feed-forward decoupling elements (shown in Equation 3.3) that are based on the existing PI tunings and the known plant dynamics, and are designed to attempt to cancel out the undesired responses.

$$G_{c,12} = -\frac{g_{12}PI_{22}}{g_{11}},\ G_{c,21} = -\frac{g_{21}PI_{11}}{g_{22}} \tag{3.3}$$

Looking at the block diagram of the feed-forward controller design and the solutions of the decoupling elements, one potential disadvantage is that the sum of two integrators are used to hold each actuator command at a steady-state level (resulting from the PI controllers and the fact that the PI controllers show up in the numerator of the feed-forward decoupling element solutions). Because of this, there is no unique solution for a linear system at any given steady-state point. For example, one case may have the integrator from PI_{11} at 0.4 and the integrator from $G_{c,12}$ at 0.2 resulting in a VGT Vane command of 0.6, but another case may have the integrator from PI_{11} at 0.1 and the integrator from $G_{c,12}$ at 0.5 again resulting in a VGT Vane command of 0.6. This can lead to the various controller elements (e.g. PIs and $G_{c,ij}$s providing conflicting commands to the sum blocks, and potentially fighting each other, especially with a highly interactive plant or for a non-deterministic reference input.

In the first glance of the closed-loop step response in Figure 3.10, it certainly appears that the feed-forward decoupling design has improved the closed-loop response performance. The response peak in the percent EGR loop due to the in-cylinder AFR variable setpoint step is reduced by well over half, and the overshoot in the percent EGR control loop has essentially been eliminated, while at the same time improving the overall response of the in-cylinder AFR (slightly smaller peak, redcued oscillations and quicker settling time). However, the percent EGR response has been noticeably increased, and while in most cases a faster responding system is more desireable, there are situations where responding too fast could over-excite a system and resulting in overall performance degradation. Therefore it is more preferred that

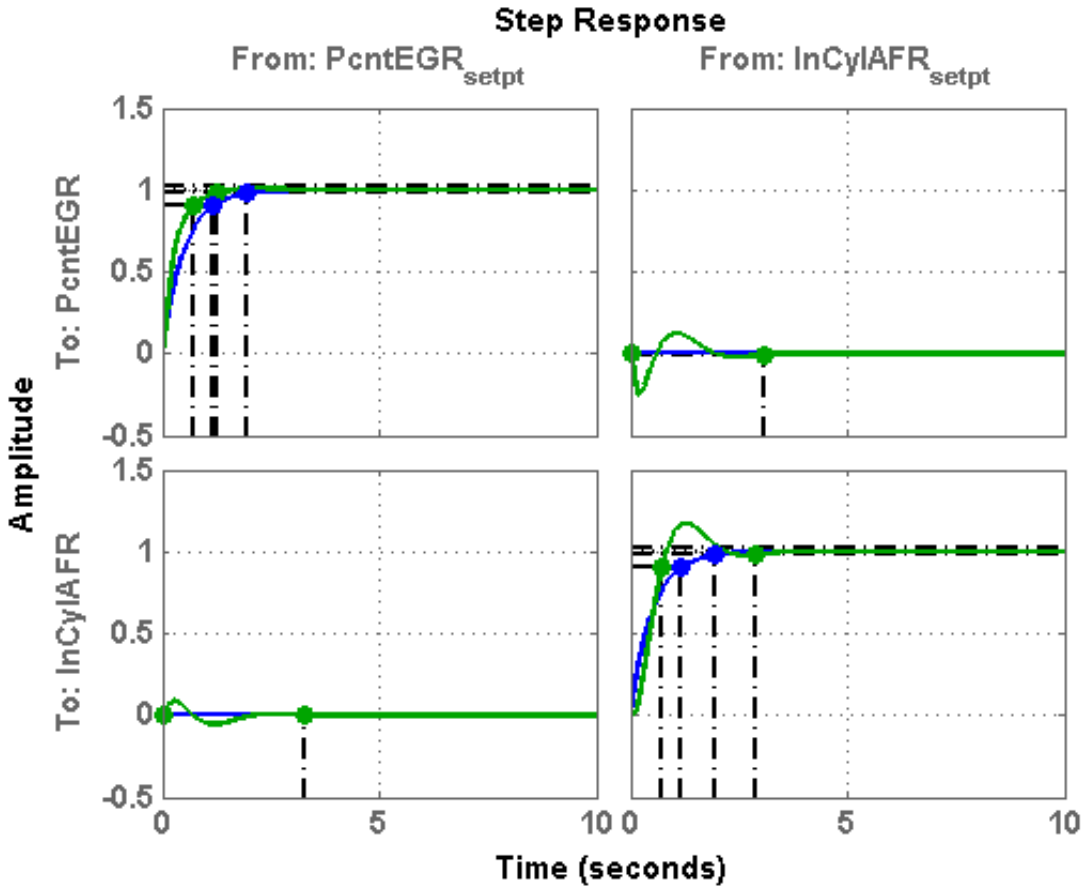

Figure 3.10 Feed-forward Decoupling Design Closed Loop Step Response

the response match a desired response, which is the position that this study is taking. The controller syntheses of the designs (and the corresponding metrics) are designed to target the desired decoupling reference response.

While the linear analysis of this feed-forward decoupling design indicates that some improvement should be acieved in the case study engine simulation, the responses shown in Figures 3.11 and 3.12 show this does not happen. Both the percent EGR and in-cylinder AFR responses have peaks well above the top of the charts (ranges of the charts have been kept consistent to provide for good visual comparisons) and increased activity in the responses overall when compared to the baseline design.

The noticable increase in the actuator commands (Figures 3.13 and 3.14) also echoe this increeased activity response. The frequency of the full-range swings of the EGR Valve have

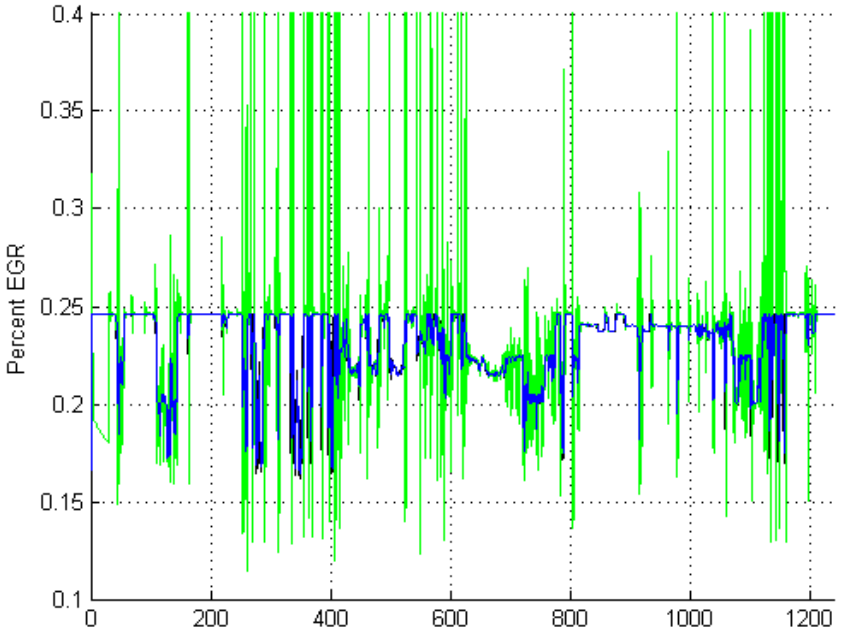

Figure 3.11 Feed-forward Decoupling Design, Percent EGR Response

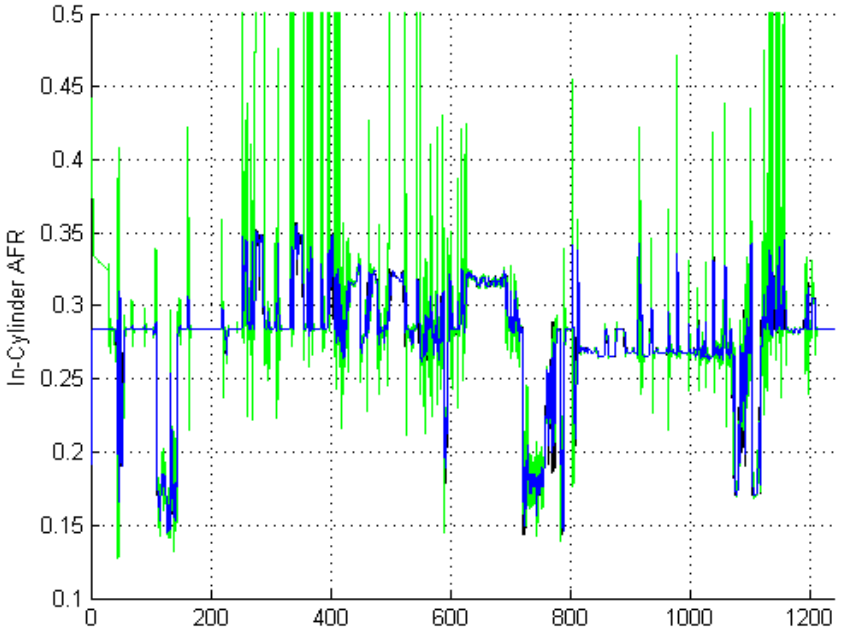

Figure 3.12 Feed-forward Decoupling Design, In-Cylinder AFR Response

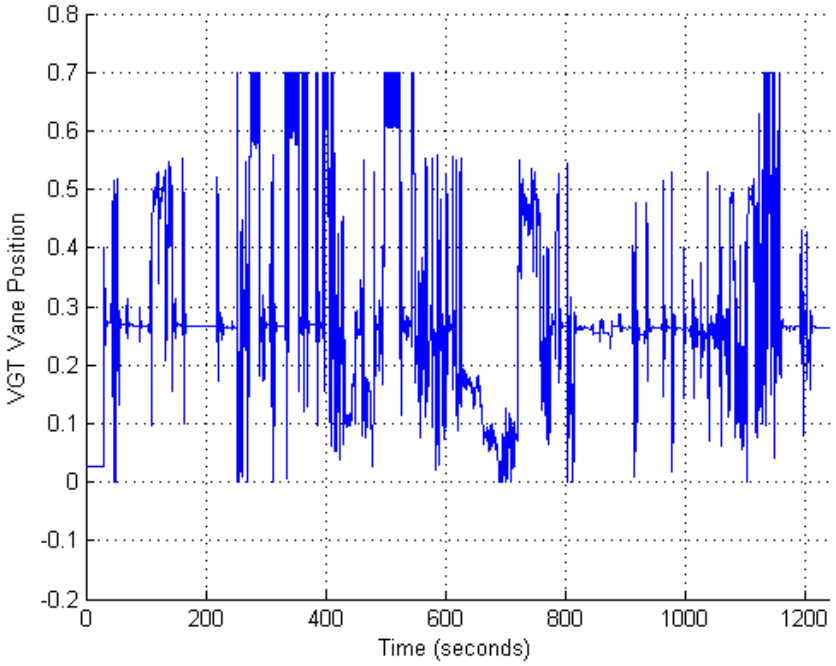

Figure 3.13 Feed-forward Decoupling Design, VGT Vane Command

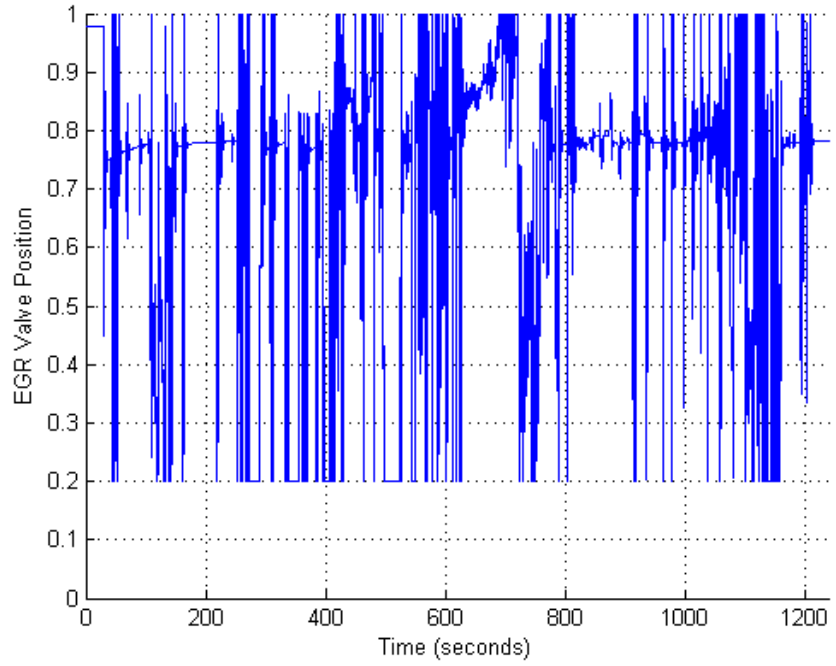

Figure 3.14 Feed-forward Decoupling Design, EGR Valve Command

increased, in addition the VGT Vane command is also saturated at times at a maximum position limit. Possible reasons to not see similar improvement in performance compared to the linear analysis follow along arguments of the integrators fighting, as well as the non-linear engine system and operation away from the linearization point.

One of the biggest takeaways from this analysis is that while this architecture was designed and calibrated to be a decoupling controller, it has fallen far away from its designed performance requirements. It is especially concerning because the linear analysis of the system indicated that substantial improvement in the response over the baseline design should be achieved, belying the actual behavior of the closed-loop engine air system.

The first attempt at a decoupling controller design has the following key points that summarize the results in the plots and discussion above.

- The linear response of the feed-forward decoupling design showed promising results in reducing the peak values and settling time of the off-diagonal elements, i.e. the disturbances from a setpoint change in the the other control loop.

- Though the main diagonal responses were closer to the desired reference as well, there were still aspects of the response that one would desire to improve upon in order to better meet the control specifications from the previous chapter. Some of the error in the response though could be attributed to portions of the simplifications used to create a controller design that still maintained a simplistic architecture.

- A major weakness of the feed-forward decoupling design of sensitivity to plant parametric uncertainty is demonstrated in the engine case study simulation (also shown in the work of Jung and Nam (1999) in high speed induction motors). Since operation of a non-linear system away from the linearization point the controller was designed at can be thought of as a type of parametric uncertainty, the results of the engine simulation clearly show the feed-forward decoupling design's weakness in this instance.

3.2.3 H_∞ 2 Degree of Freedom Loop Shaping Observer Based Controller

Since the feed-forward decoupling controller design - and many complex controller architectures in literature - result in a full matrix controller, a fully multivariable design is explored for comparison purposes to see how well the designs built up from SISO transfer functions compare to a fully coordinated design. This is also done to help address the issue of the iterative calibration procedure from the previous section, in that one could approach the design and calibration of the decentralized controller elements in a fully coordinated approach, rather than sequentially. Skogestad and Postlethwaite (2005) mention this approach, but state that this is not common practice in industry, because one begins to lose the benefits that a sequential design process of decentralized controller elements provides. At this point, Skogestad and Postlethwaite (2005) mention that if tuning multiple SISO controllers in a decentralized architecture simultaneously, one could just as easily design a full multivariable controller.

The multivariable controller design that is analyzed is the 2 Degree of Freedom (DoF) Loop Shaping (LS) Observer-based H_∞ design discussed in Skogestad and Postlethwaite (2005); this design is based on the work of McFarlane and Glover (1992), Hoyle et al. (1991), Limebeer et al. (1993), Walker (1996), and is shown in a general form in the block diagram of Figure 3.15. This controller is called a 2 DoF controller because it has separate dynamics in K_1 that act on the reference signal independently of the dynamics of K_2 that act only on the control variables; this is in contrast to the other controller designs in this study, as they are only 1 DoF designs (i.e. the dynamics of the controller act only on the error signal $e = r - y$). The thicker signal lines in Figures 3.15 and 3.16 indicate a vector signal; for the 2×2 system being controlled in this paper it represents a vector containing two elements; this is in contrast to the other block diagrams, where the thin signal lines represent a scalar signal.

For this controller synthesis procedure, the plant is first shaped using a shaping filter (in this instance the baseline PI controller architecture is used, i.e. $W_s = diag\{PI_{11}, PI_{22}\}$) resulting in a shaped plant $G_s = GW_sW_a$ and represented with state-space matrices A_s, B_s, C_s, D_s. Skogestad and Postlethwaite (2005) also mention an optional constant gain matrix W_a, which is used to align the singular values at a given frequency (i.e. the closed-loop bandwidth) and

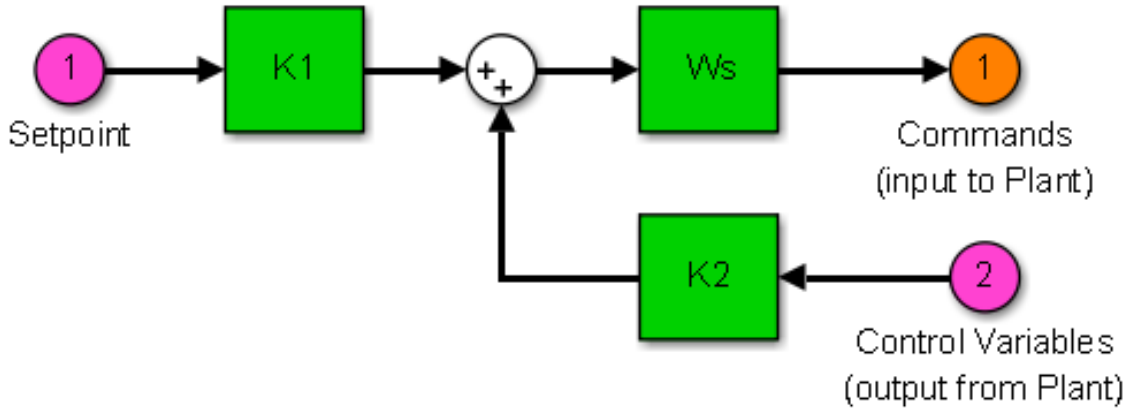

Figure 3.15 General 2 Degree of Freedom Controller Block Diagram

is included in this design. Often W_s is chosen to have integral action in order to result in zero steady state error, which is why the baseline diagonal PI architecture is used. Given this shaped plant and the desired reference response filter $T_{ref} = diag\{1/(\tau_c s + 1)\}$ with corresponding state-space matrices A_r, B_r, C_r, D_r, a signal based H_∞ control problem is set up, and the resulting controller(s) $K = [K_1\ K_2]$ are produced. As mentioned by Skogestad and Postlethwaite (2005), the work of Walker (1996) has shown that a controller solution can be derived by solving algebraic Riccati equations (detailed in Equation 3.4 and shown in implementation form in Figure 3.16). This controller design gets its full name by the fact that the structure of the controller mimics that of an observer (hence the Observer-based), the base plant is augmented with a shaping filter to give a desired open loop frequency response (an activity commonly referred to as Loop Shaping), the solution satisfies the H_∞ robustness criteria, and the 2 DoF aspect mentioned above.

$$\begin{aligned} \dot{\hat{x}}_s &= A_s\hat{x}_s + H_s(C_s\hat{x}_s - y) + B_s u_s \\ \dot{x}_r &= A_r x_r + B_r W_i r \\ u_s &= -F_s\hat{x}_s - F_r x_r \end{aligned} \tag{3.4}$$

A few things to mention about the H_∞ design procedure is that - at a high level - it amounts to a gain reduction technique, but one that systematically provides for the proven

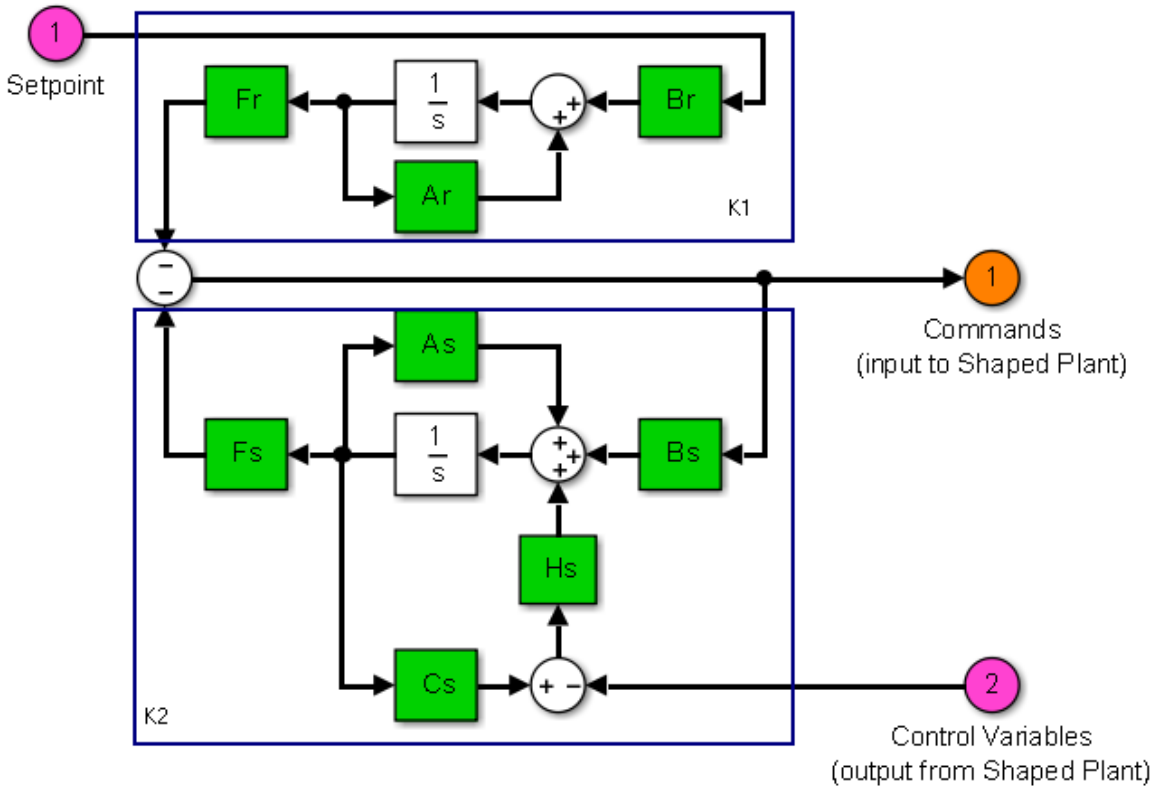

Figure 3.16 H_∞ 2 DoF Loop Shaping Observer-based Controller Block Diagram

reduction of the closed-loop system gain at all frequencies to obtain a desired level of robustness. Therefore, one thing to anticipate in the closed-loop responses would be that there should be some decrease in the response speed (i.e. increase in response time). This change is consistent with general design philosophies that in order to achieve robustness in general, one should reduce the aggressiveness (i.e. gain) of the controller being designed. In addition, this design employs linear algebra to calculate the control law (as can be seen in Equation 3.4), and while this is not a strict prohibit against implementation in an embedded processor, larger system matrices and reduced processor capabilities (for cost reduction purposes) may make a practical, real-time implementation of this architecture very difficult if not impossible.

It should be noted here that this is just an example multivariable controller design, and in some ways it may not the best multivariable controller design selection as the full potential of the H_∞ design is not yet realized. Because the H_∞ design comes from the field of robust

control theory, it deals with representing and managing uncertainty in control of real systems. Though the robustness of the various controller designs are not studied in this paper, this H_∞ architecture is still analyzed because of the manufactured nature of the diesel engine (e.g. part-to-part variability, modeling inaccuracies, etc.), and provides a starting point for future exploration into the robustness of the designs presented in this paper. Another reason this controller choice may not be the most suitable for this work is because it provides more internal degrees of freedom than the other designs in this paper. This is due to the design procedure, which results in the H_∞ controller being a full order controller for the shaped plant (8 states for the H_∞ design compared to the 2 for the diagonal PI and 4 for the feed-forward decoupling design and cross interaction filter design - discussed later). In general, more internal degrees of freedom (i.e. more states) in a controller results in improved closed loop performance. Therefore, a reduced order controller was also designed by reducing G_s appropriately so that the final controller has 6 total states in order to make the comparisons with the other designs (except the baseline diagonal PI architecture) more equitable. The full order H_∞ controller is still included in the analysis so that the validity of the reduced order controller design can be shown, and one can determine how much - if any - performance degradation is introduced due to the model/controller order reduction.

The response of the linear engine air system in Figure 3.17 shows the gain reduction nature of an H_∞ design: there is a general increase in the response time, not terribly significantly in the percent EGR response but especially so in the in-cylinder AFR response. One the same note, the H_∞ nature is shown in the complete elimination of any overshoots, and by a "smoothing out" of the off-diagonal responses. Also seen in Figure 3.17 is the response of the reduced order H_∞ design (shown in red). The gain reduction aspects of the H_∞ design are still seen, and one nice aspect to note is the recovery of some of the response time in the in-cylinder AFR variable. This would be due to the model order reduction removing higher order lag elements, thereby allowing the overall system design to respond a touch faster. What is potentially more interesting is the nearly complete decoupling of the system due to this design. This behavior though is slightly misleading in that - because of the two integrators in the shpaing filter W_s, the reduction techniques keep only those, resulting in a reduced system of two integrators with

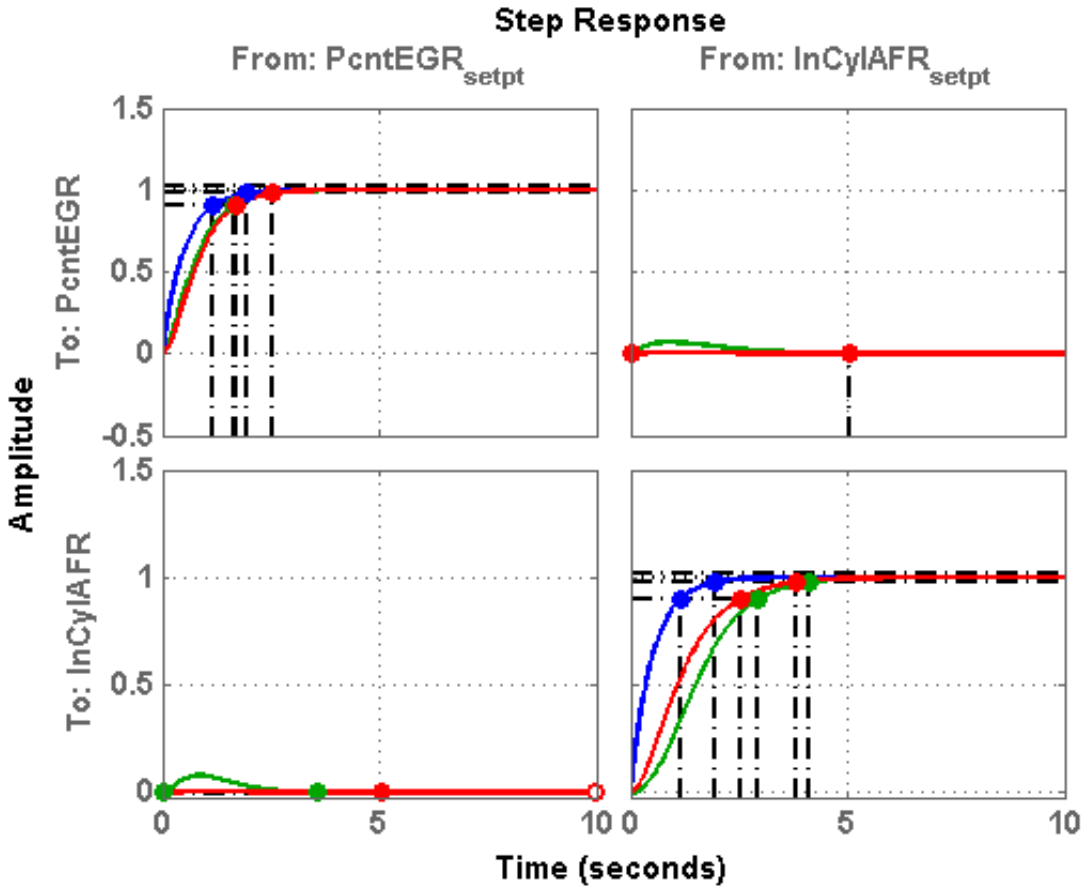

Figure 3.17 H_∞ Controller Closed Loop Step Response

a nearly decoupled gain matrix.

If the general extrapolation of the linear air system simulations to the case study can be believed (though as was seen in the Feed-forward Decoupling Design this can be very misleading), it would be expected as well to see some improvement in overall performance of the H_∞ design. However, a quick compare to the baseline plots, the responses in Figures 3.18 and 3.19 show behavior that appears to be essentially equivalent to the baseline in the percent EGR variable, with a slight degradation in the in-cylinder AFR variable. This is not entirely surprising from the linear analysis results showing the noticeable increase in response time to the in-cylinder AFR variable.

Where there is noticeable improvement in the H_∞ controller's response can be seen in the actuator commands (Figures 3.20 and 3.21) which show considerably less noisy responses. In

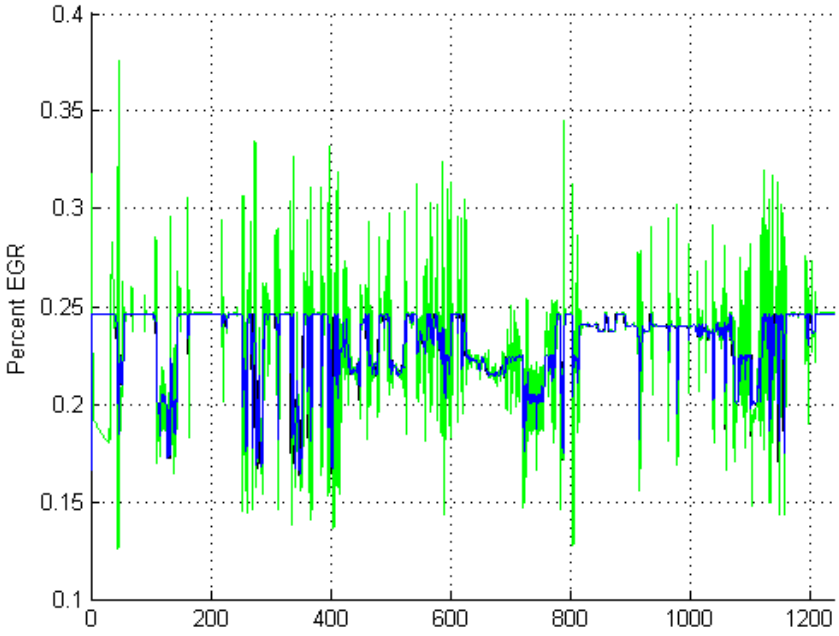

Figure 3.18 H_∞ Controller, Percent EGR Response

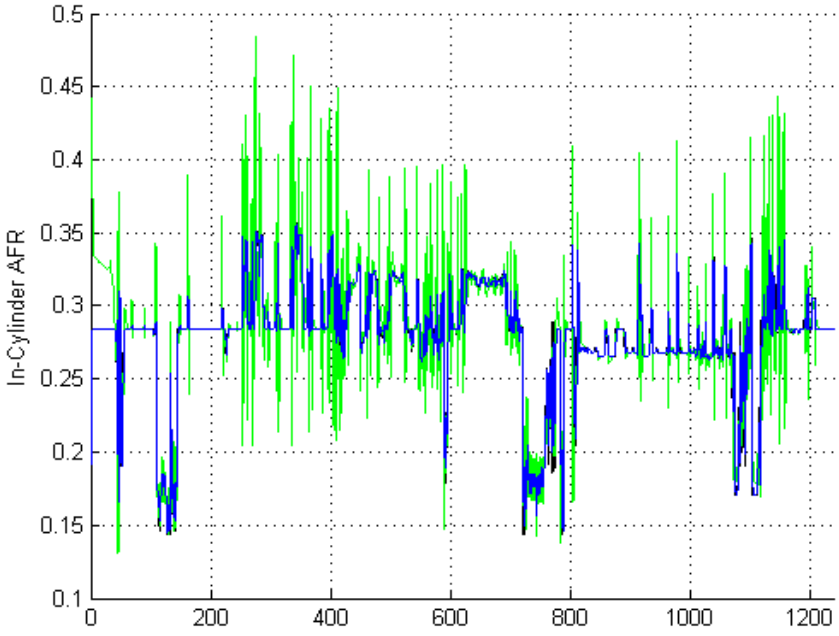

Figure 3.19 H_∞ Controller, In-Cylinder AFR Response

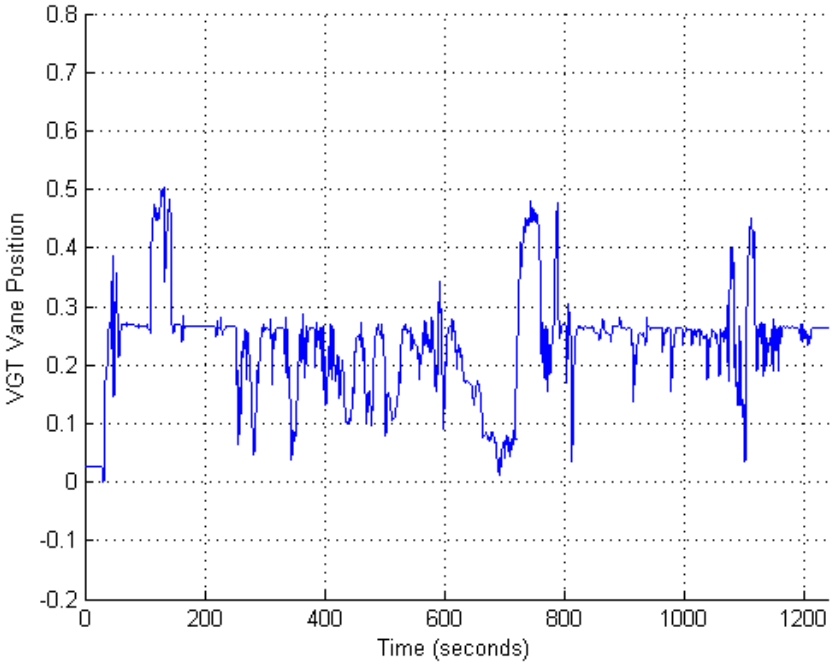

Figure 3.20 H_∞ Controller, VGT Vane Command

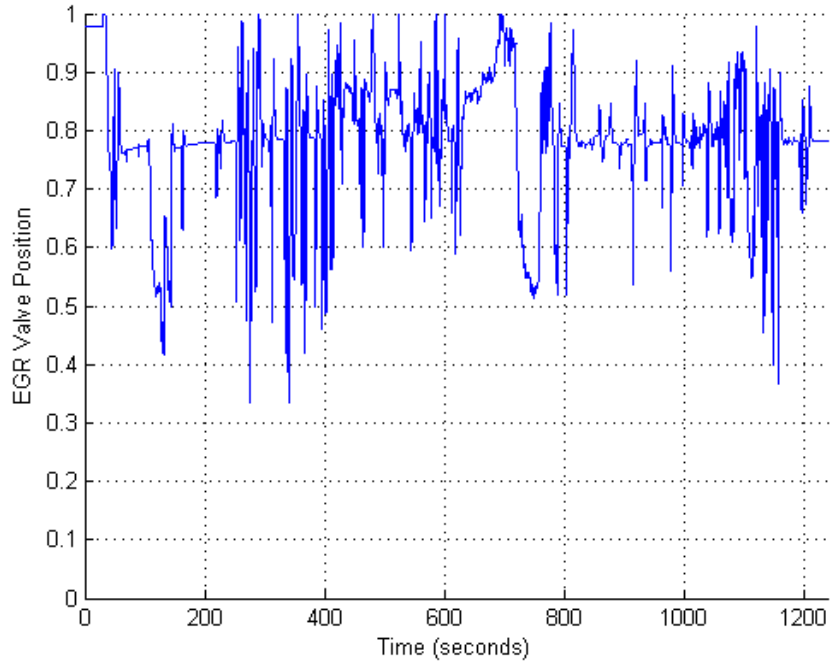

Figure 3.21 H_∞ Controller, EGR Valve Command

addition, the H_∞ design does not saturate the EGR Valve at the minimum position command, which is something none of the other designs have accomplished in this study. This shows one other benefit of the H_∞ design, in that - as a gain reduction technique - it typically does not drive the actuators as hard, which in terms of a physical system component can save on part wear and potentially extend the life of the component. The engine simulation response for the reduced order H_∞ design was also collected, however the plots do not appear significantly different at a high level (as one would expect based on the linear simulation results), but the cumulative ISE and TV traces have been included as part of the comparison later on, as well as the various metrics collected for analysis purposes.

The main points that can be taken away from the H_∞ design and the corresponding plots can be summarized as follows.

- The 2 DoF aspect of this controller provides for smooth control, both in terms of the output responses (which doesn't accelerate as fast as the other designs, or even the reference response) as well as actuator usage (as seen in Figures 3.20 and 3.21).
- The robust control design satisfying the H_∞ criteria enables "better" control away from the linearization point (though better would need to be defined within the context of the design trade-offs).
- Following the procedure outlined above results in a full order controller, i.e. a controller that has as many states as the plant itself. Therefore, for very large order plants, large order controllers are developed without some aspect of model reduction. This is not the case with the PI-based designs in this study, and can even have implementation concerns.

3.3 Controller Architecture Summary

This chapter has analyzed different controller architectures and shown their respective control capabilities with regards to a linear system step response as well as a realistic engine simulation case study. First, the baseline architecture of a Diagonal PI controller was defined, tuned, and simulated. As a result of this, it was shown how an easy trap to fall into is that assuming successful tunings of the individual PI controllers is sufficient and that the entire

MIMO controller will behave as desired for a highly interactive system. Following that - and building on the desire of a decoupled closed-loop system - a common decoupling architecture was explored, where the decoupling elements were added to create a full matrix controller in a feed-forward manner. The non-intuitive nature of this design was clearly demonstrated, in where the linear analysis indicated an overall improvement in performance, the case study engine simulation proved otherwise. Finally, a full order multivariable controller was designed, and while it seemed to show at least minimal change in performance, there are aspects in the increased response time and potential implementation concerns that make this architecture one that stretches to meet the desired controller requirements defined at the onset of this study.

At this point a different style of a decoupling controller will be proposed and analyzed. Similar decoupling solutions will be explored, and the resulting MIMO controller will be compared to the designs analyzed here, in both structure and performance capability.

CHAPTER 4. CROSS INTERACTION FILTER DESIGN AND ANALYSIS

At this point the idea of the cross interaction filter (CIF) design will be fully developed, explored, and analyzed. Comparisons will be made to the controller designs analyzed in Chapter 3, as well as a deeper analysis and comparison to the feed-forward decoupling controller design.

4.1 Cross Interaction Filter Concept

The idea for the CIF design came about as a potential method to reduce or remove the effects of the PI fighting as seen in the feed-forward decoupling controller design. The idea was, that instead of decoupling by looking at the error input to the controller and adjusting the actuator commands, that instead one could look at the actuator commands that are being provided to the system, and - using knowledge of the plant dynamics from the transfer function matrix - *feed back* a correction term that would modify the error coming into the controller with the goal of effectively decoupling the system. This is the same structure that Tham (1999) termed the V-canonical form, yet gave no discussion to, and is pictured in Figure 4.1. The goal of this section is to derive solutions for the cross interaction filters (blocks W_{12} and W_{21} in Figure 4.1) for a generic 2×2 plant, similar to how Tham (1999) obtained solutions for the P-canonical form (feed-forward decoupling design), analyze the resulting controller design, and performing the same simulation studies as were done to the designs from the previous chapter.

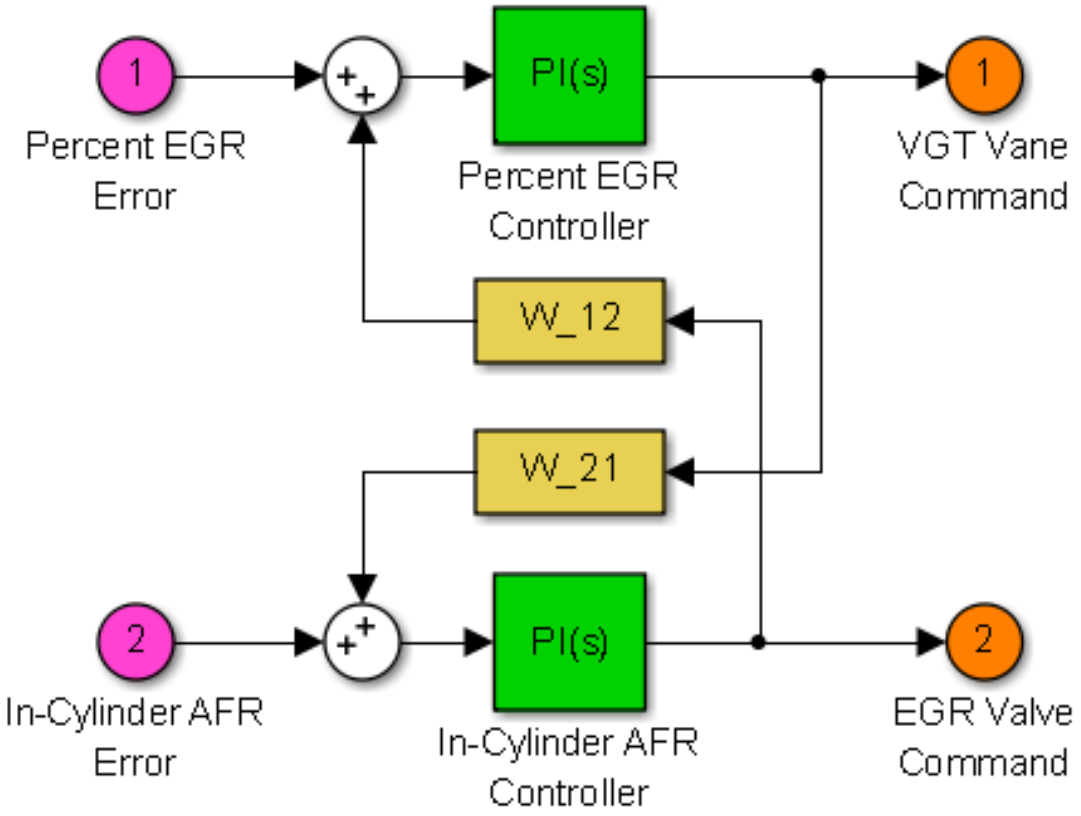

Figure 4.1 Cross Interaction Filter Block Diagram

4.1.1 Derivation of CIF Solution

Starting with the baseline diagonal PI controller - as all the designs from the previous chapter did - and augmenting with the cross interaction filters as shown in Figure 4.1, the result is the following set of equations.

$$\begin{aligned} u_1 &= PI_{11}(e_1 + W_{12}u_2) \\ u_2 &= PI_{22}(e_2 + W_{21}u_1) \end{aligned} \tag{4.1}$$

By simultaneously substituting the first equation into the second and vice versa, and by using a little algebra, the resulting controller equations are shown in Equation 4.2, which has only one unique control output term (actuator command u_i) in each equation. Each equation can then be solved individually for the control output terms as a function of the error inputs and having parameters of PI_{ii} and W_{ij}; the result is represented in the transfer function

matrix in Equation 4.3. It is acknowledged here that this algebraic manipulation is possible only by decomposing the 2×2 MIMO system into SISO elements and considering all of them individually.

$$\begin{aligned} u_1 &= PI_{11}(e_1 + W_{12}PI_{22}(e_2 + W_{21}u_1)) \\ u_2 &= PI_{22}(e_2 + W_{21}PI_{11}(e_1 + W_{12}u_2)) \end{aligned} \tag{4.2}$$

$$\begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \frac{1}{1 - PI_{11}PI_{22}W_{12}W_{21}} \begin{bmatrix} PI_{11} & PI_{11}PI_{22}W_{12} \\ PI_{11}PI_{22}W_{21} & PI_{22} \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} \tag{4.3}$$

Using the basic closed loop identity $T = (I + L)^{-1}L$ the requirement of a decoupled closed loop system is applied. This means that the off-diagonal elements of T must be zero. One can see then that whenever an element of T is zero, the same element of $L = GK$ must zero as well. This must be the case because if a matrix is diagonal, then its inverse is also diagonal, and the product of any two diagonal matrices is also diagonal. Therefore, the closed-loop decoupling requirement is transferred to the open-loop transfer function matrix, resulting in a set of four equations for this 2×2 system (Equation 4.4). Because the initial assumption is that the PI controllers are already tuned and that the plant transfer function matrix components are known, one can see that only two unknowns (the cross interaction filters W_{12} and W_{21}) remain and that an over-specified system of equations exists.

$$\begin{aligned} l_{11} &= g_{11}k_{11} + g_{12}k_{21} &&= \tfrac{g_{11}PI_{11} + g_{12}PI_{11}PI_{22}W_{21}}{1 - PI_{11}PI_{22}W_{12}W_{21}} \\ l_{12} &= g_{11}k_{12} + g_{12}k_{22} &&= \tfrac{g_{11}PI_{11}PI_{22}W_{12} + g_{12}PI_{22}}{1 - PI_{11}PI_{22}W_{12}W_{21}} &&= 0 \\ l_{21} &= g_{21}k_{11} + g_{22}k_{21} &&= \tfrac{g_{21}PI_{11} + g_{22}PI_{11}PI_{22}W_{21}}{1 - PI_{11}PI_{22}W_{12}W_{21}} &&= 0 \\ l_{22} &= g_{21}k_{12} + g_{22}k_{22} &&= \tfrac{g_{21}PI_{11}PI_{22}W_{12} + g_{22}PI_{22}}{1 - PI_{11}PI_{22}W_{12}W_{21}} \end{aligned} \tag{4.4}$$

By using only the off-diagonal elements of L (which equal zero by definition of the decoupling requirement), the over-specified system of equations above is reduced to a system of two equations and two unknowns. Thus this reduced system of equations can be solved to produce the solutions for the cross interaction filters that satisfy the requirement of decoupling the closed loop system (Equation 4.5).

$$W_{12} = -\frac{g_{12}}{g_{11}PI_{11}}, \; W_{21} = -\frac{g_{21}}{g_{22}PI_{22}} \tag{4.5}$$

Because the cross interaction filter design is equivalent to the V-canonical form in Tham (1999), Equation 4.3 can be represented in a condensed, multivariable manner as shown in Equation 4.6, with **C** and **W** defined in the line following.

$$\mathbf{u} = [\mathbf{I} - \mathbf{CW}]^{-1}\,\mathbf{Ce}$$
$$\mathbf{C} = \begin{bmatrix} PI_{11} & 0 \\ 0 & PI_{22} \end{bmatrix},\ \mathbf{W} = \begin{bmatrix} 0 & W_{12} \\ W_{21} & 0 \end{bmatrix} \tag{4.6}$$

One thing to notice about the CIF solutions is that the inversion of the PI controller results in the overall CIF solution having the form of an approximate derivative transfer function, which is - on the surface - easily implementable. It also has the benefit that at steady-state the effect of the CIF goes to zero, which eliminates the potential PID "fighting" issue from the feed-forward controller design. However, there are a few drawbacks to this solution that need to be addressed, and are covered in the next section.

4.1.2 Realizability of CIF Solution

Looking at the results for the cross interaction filters in Equation 4.5, as well as the solutions for the off-diagonal controller terms from Equation 3.3, it can be seen that both solutions for the decoupling controller elements involve the ratios of g_{12}/g_{11} and g_{21}/g_{22}. This can cause potential issues in the controller since any RHP zeros or time delays in g_{11} or g_{22} will cause the resulting controller to be unstable or non-causal, neither of which are desired or effectively implementable controller design practices. In addition, if the elements of the plant transfer function matrix are not reduced prior to calculating the ratios, differing orders of the transfer function elements can cause the transfer function ratio to be either strictly proper, semi-proper, or possibly improper which is not implementable.

Tham (1999) discusses some of these realizability ideas, and shows that for a first order transfer function with a time delay, the resulting ratios of transfer functions can be represented as shown in Equation 4.7. To get to a ratio of first order system with a time delay, one could use the SIMC model reduction rules from Skogestad (2003).

$$g_{12} = \frac{k_{12}}{\tau_{12}s+1}e^{-s\theta_{12}},\ g_{11} = \frac{k_{11}}{\tau_{11}s+1}e^{-s\theta_{11}}$$
$$\frac{g_{12}}{g_{11}} = \frac{k_{12}(\tau_{11}s+1)}{k_{11}(\tau_{12}s+1)}e^{-s(\theta_{12}-\theta_{11})} \tag{4.7}$$

In looking at this simplification of the ratio of the first order transfer functions with a time delay, the resulting transfer function will take the form of a lead or lag filter (lead filter if $\tau_{11} > \tau_{12}$, and lag filter if $\tau_{12} > \tau_{11}$), with a time delay that can result in the filter being causal or non-causal. If not reducing the initial transfer functions and the order of g_{11} is greater than g_{12}, the resulting filter is not proper and therefore not directly implementable, which gives good reason to first reduce the system as described by Skogestad (2003). If g_{11} and g_{12} are the same order the system is semi-proper (as is the case with the first order transfer functions with a time delay), and if the order of g_{11} is greater than g_{12}, the system is strictly proper and no implementation issues exist (with regards to transfer function order and lag).

With regards to the resulting time delay, there is a danger in creating a non-causal controller because of the inversion of the time delay from the transfer function in the denominator. However, the CIF solution from Equation 4.7 only becomes non-causal if $\theta_{11} > \theta_{12}$, which means that the SIMC reduction rules may not be appropriate, and any model reduction of the transfer functions will need to stop at higher orders until a workable solution can be found. In addition, if both θ_{11} and θ_{12} are roughly the same value, the subtraction will essentially zero out the net time delay. This delay could even be neglected without significant loss of performance if very near to zero and/or is significantly smaller than the either of the time lags from the transfer functions for either a causal or non-causal result. Having dealt with a possibility of a non-causal controller, any causal net time delay could be accounted for and implemented by using a Padé approximation or a Smith predictor. The preceding discussions apply as well to the ratio of g_{21}/g_{22}.

Because the SIMC reduction rules may not always be applicable to the CIF solutions, yet another concern exists for implementing the CIF solutions from Equation 4.5, and that is a RHP zero may exist in the resulting denominator transfer function that produces a RHP pole in the CIF implementation. While certainly implementable, it is definitely ***not*** desired to produce a controller that is unstable, therefore a method needs to be provided to accommodate such scenarios. A method then for dealing with this potential is that if the dominant closed loop response of the denominator can be represented sufficiently by a first order response, then the open loop Bode magnitude shape (i.e. $g_{11}PI_{11}$ and g_2PI_{22}) can be represented as that of an

integrator and time constant, and the solutions can be represented in a more simplified format (Equation 4.8). With a little further manipulation an even simpler implementation format can be derived (right section of Equation 4.8) that will be termed the Simplified Cross Interaction Filter design.

$$\begin{array}{rcccl} W_{12} & = & -\frac{g_{12}}{\tau_{c,11}s} & = & \left(-\frac{k_{12}\tau_{c,11}}{\tau_{12}}\right)\left(1-\frac{1}{\tau_{12}s+1}\right)e^{-s\theta_{12}} \\ W_{21} & = & -\frac{g_{21}}{\tau_{c,22}s} & = & \left(-\frac{k_{21}\tau_{c,22}}{\tau_{21}}\right)\left(1-\frac{1}{\tau_{21}s+1}\right)e^{-s\theta_{21}} \end{array} \tag{4.8}$$

Another benefit of the Simplified Cross Interaction Filter (SCIF) design is that once a reduced plant transfer function matrix is known, diagonal PI controllers can be tuned (using either the SIMC tuning rules with one tuning parameter per PI controller, or whatever other favorite tuning method the reader prefers), and then the SCIFs can be calibrated using the plant transfer function matrix data and the simplified tuning result from the main diagonal control loops. This produces a simplified, sequential design procedure which fits well into the assembly line philosophy of manufacturing companies, and helps to simplify the overall tuning problem and results in an easy to implement and understand controller architecture. Due to the amount of simplification that goes into producing the SCIF implementation results, the overall closed loop system response would need to be evaluated afterwards to confirm the performance and robustness of the design (as one should do with any controller design).

4.1.3 Comparison to Feed-forward Decoupling Controller Design

While both the feed-forward decoupling and the cross interaction filter designs result in a 2×2 controller array for this 2×2 system, an obvious difference between the two architectures is the direction of the decoupling elements in the block diagrams of Figures 3.9 and 4.1. To investigate additional similarities and/or differences between the two designs, the solutions for the feed-forward decoupling design (Equation 3.3) are substituted into the full matrix controller resulting in the transfer function matrix in Equation 4.9.

$$K^{FFD} = \begin{bmatrix} PI_{11} & -PI_{22}\frac{g_{12}}{g_{11}} \\ -PI_{11}\frac{g_{21}}{g_{22}} & PI_{22} \end{bmatrix} \tag{4.9}$$

Similarly, the cross interaction filter solutions from Equation 4.5 are substituted into Equation 4.6 resulting in the controller transfer function matrix in Equation 4.10.

$$K^{CIF} = \frac{1}{1 - g_{12}g_{21}/g_{11}g_{22}} \begin{bmatrix} PI_{11} & -PI_{22}\frac{g_{12}}{g_{11}} \\ -PI_{11}\frac{g_{21}}{g_{22}} & PI_{22} \end{bmatrix} \tag{4.10}$$

It is interesting to note that based on the resulting controller equations above, $K^{CIF} = FK^{FFD}$ where F is based entirely on the information from the plant transfer function matrix. In addition, for this 2×2 example F has the same form as the main diagonal elements of the frequency dependent RGA (Skogestad and Postlethwaite (2005), Tham (1999)). However, as these elements are transfer functions and not the actual RGA value at a given frequency, the conversion filter F does not have the same frequency dependent shape as the main diagonal elements of the frequency dependent RGA. If the elements of the transfer function matrix have been reduced to first order systems (for example, by using the SIMC model reduction rules), F would certainly have the form of a SISO lead-lag transfer function.

4.1.4 Baseline Architecture and Decoupling Network Plant Analysis

As another comparison of the decoupling nature between the feed-forward decoupling and CIF designs (see Equations 4.9 and 4.10, respectively), the two different full matrix controllers will be factored into the baseline diagonal PI controller design (the $\mathbf{C}$ matrix from Equation 4.6), and use the remaining controller transfer function matrix factor with the plant model to generate an augmented plant. Tham (1999) presented a similar idea as a decoupling network with a diagonal PI controller. This will allow a comparison as to how the decoupling elements effectively become a decoupling network and "modify" the plant model and what implications that has on the controller performance and the type of system the baseline Diagonal PI architecture would "see". In order to keep the analysis somewhat simple, the bulk of this refactoring analysis will be done on the open loop transfer function $L = GK$.

Starting with the feed-forward decoupling design:

$$\begin{aligned} L^{FFD} = GK^{FFD} = GK^{FFD}I = GK^{FFD}(\mathbf{C}^{-1}\mathbf{C}) = G(K^{FFD}\mathbf{C}^{-1})\mathbf{C} = \tilde{G}^{FFD}\mathbf{C} \\ \hookrightarrow \tilde{G}^{FFD} = GK^{FFD}\mathbf{C}^{-1} = \begin{bmatrix} g_{11} - g_{12}\frac{g_{21}}{g_{22}} & 0 \\ 0 & g_{22} - g_{21}\frac{g_{12}}{g_{11}} \end{bmatrix} \end{aligned} \tag{4.11}$$

It can be seen from Equation 4.11 that because the off-diagonal elements of $\tilde{G}^{FFD}$ and $\mathbf{C}$ are all zero, this means that the off-diagonal elements of L^{FFD} will also be zero, and the resulting closed-loop off-diagonal elements will also be zero, thus satisfying the decoupling criteria used to derive the closed form solutions for the feed-forward decoupling design and the requirement for this study. However, it is striking to note that the resulting main diagonal components of the augmented plant $\tilde{G}^{FFD}$ have had their original dynamics modified by a combination of the other elements of the original plant transfer function matrix G. This could potentially cause differences in the actual individual responses compared to the desired, if the individual PI_{11} and PI_{22} controllers were initially tuned from the g_{11} and g_{22} transfer functions, respectively. This doesn't mean that a suitable tuning solution for the PI controllers can't be found, but it does suggest that a final PI tuning solution will may only be determined iteratively. These results - plus the non-unique steady-state value issue mentioned in Chapter 3 - suggests that the feed-forward decoupling controller design may not an ideal design.

Performing the same factorization analysis on the cross interaction filter solutions:

$$\begin{aligned} L^{CIF} = GK^{CIF} = G\left[I - \mathbf{CW}\right]^{-1}\mathbf{C} = \left(G\left[I - \mathbf{CW}\right]^{-1}\right)\mathbf{C} \\ \hookrightarrow \tilde{G}^{CIF} = G\left[I - \mathbf{CW}\right]^{-1} = \begin{bmatrix} g_{11} & 0 \\ 0 & g_{22} \end{bmatrix} \end{aligned} \tag{4.12}$$

This result is interesting in that it is a diagonal system and shows the impact the F shaping function has on the overall decoupling capability of the CIF. As with the feed-forward factorization, the off-diagonal elements are also zero, therefore the decoupling requirement is still satisfied. Unlike the feed-forward design, the main diagonal elements remain unaffected compared to the initial transfer function matrix G, which means the initial tunings of the PI controllers will still provide the desired response that the controllers were tuned for without any iterative solutions. This - coupled with the fact that at steady-state the effects of the CIF tend to zero (because of their derivative-like structure) - means that the CIF architecture tends to a truly decoupled and diagonal system at steady-state, and would appear to be a more desirable design than the feed-forward decoupling architecture.

At this point the question arises, that if both of these types of architectures are decoupling architectures, why does one form impact the main diagonal elements and the other does not?

Referring back to the argument that was used to derive the decoupling solutions (both for the feed-forward and CIF designs), recall that only two of the equations (i.e. the equations that equaled zero) were selected from our over-specified system equations, and that the only solution that was solved for only guaranteed that the off-diagonal elements would be zero, which is true for both forms. But because the other two equations were "discarded", nothing can be said about the resulting main diagonal elements. As a result, they are left to be whatever the result of the decoupling constraint on the off-diagonal elements produces. It just so happened that with the CIF design, the main diagonal elements had no impact on them and that the resulting closed-loop system results in a truly decoupled system. It may be possible to specify additional constraints (in the form of desired open loop shapes for the main diagonal components l_{11}, l_{22}) on the original system of four equations to result in a solution for a decoupling controller. However, since these specifications would relate to the desired closed-loop performance of the system, the set of these specifications would be immense and could vary tremendously; therefore the exploration of this idea will be left for future research.

4.2 Simulation Analysis

The CIF controller architecture is now applied to the same fourth order linear air system model (Equation 2.2) and the high-fidelity mean value engine model, as was done with the controllers from the previous chapter to generate and obtain the similar results and metrics for this controller architecture.

Looking at the CIF closed-loop step response in Figure 4.2 (and comparing it to the feed-forward decoupling design from Figure 3.10) the truer decoupling effects of the CIF solution are seen. The peak response in the percent EGR response is essentially eliminated, and the response tracks significantly close to the desired response. In addition, the in-cylinder AFR response tracks noticeably better than any of the other designs analyzed. On top of all that, the off-diagonal responses are slightly closer to a zero response than any of the other designs. In short, the linear air system step response shows that the CIF design appears to meet the decoupling requirement and desired response performance better than any of the other designs studied.

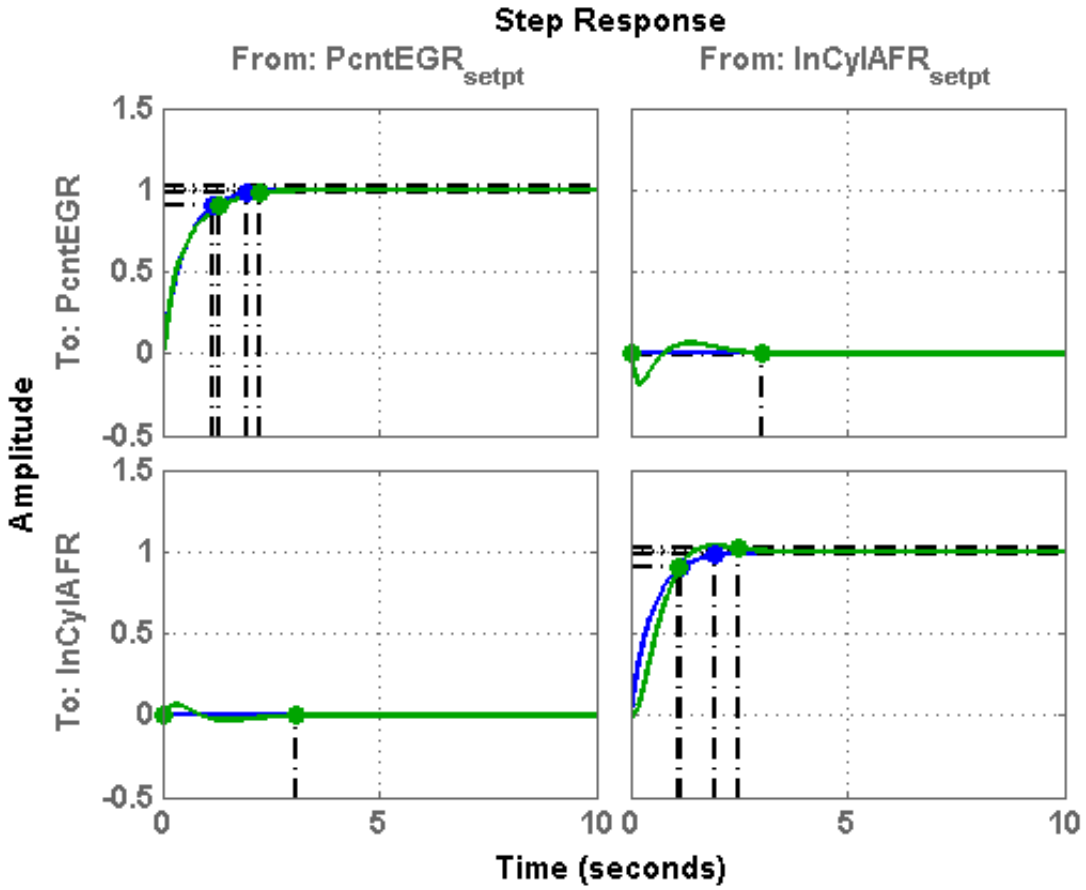

Figure 4.2 Cross Interaction Filters Closed Loop Step Response

As a result of the linear air system model's performance, one would expect similar improvements in the engine simulation case study. For the most part, this appears to have played out in the percent EGR response (Figure 4.3). However, it seems to appear that this improvement is delivered by increasing the actuation activity of the VGT Vane, as the "noise level" in Figure 4.5 is slightly higher than the baseline. This might be explained by the pseudo-derivative form of the CIF solutions; while they have a benefit of approaching zero at steady-state, derivatives also have the potential of exciting noise. In turn, this has triggered a noisier response in the in-cylinder AFR response (Figure 4.4), but this appears to be balanced out by the EGR Valve command having about the same level of activity as the baseline (Figure 4.6). In addition - and like all the other controller designs - the case study results may also be impacted by operation away from the linearization point.

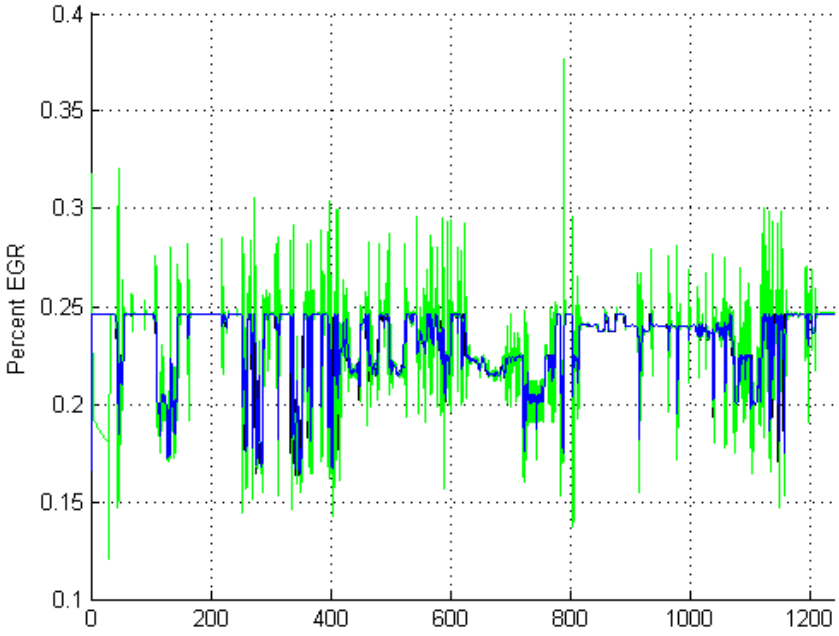

Figure 4.3 CIF Design, Percent EGR Response

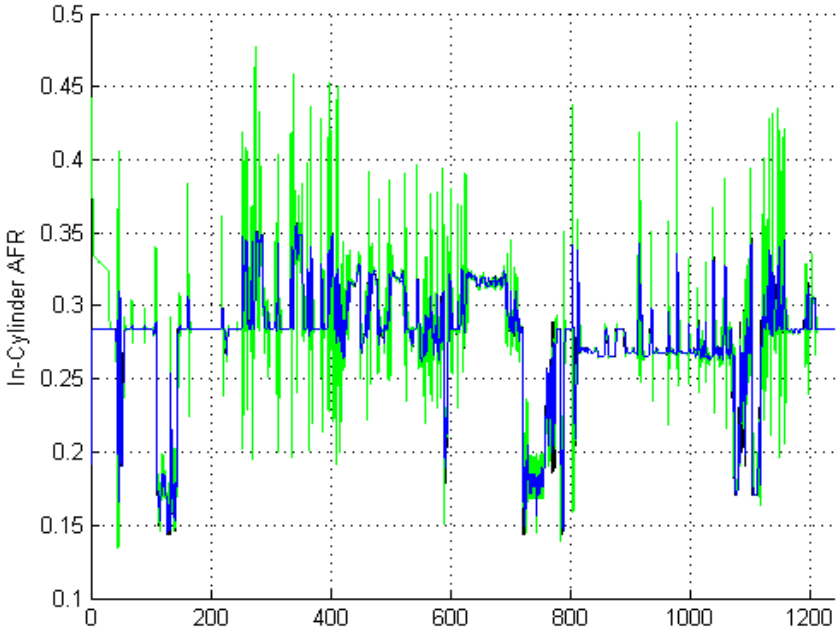

Figure 4.4 CIF Design, In-Cylinder AFR Response

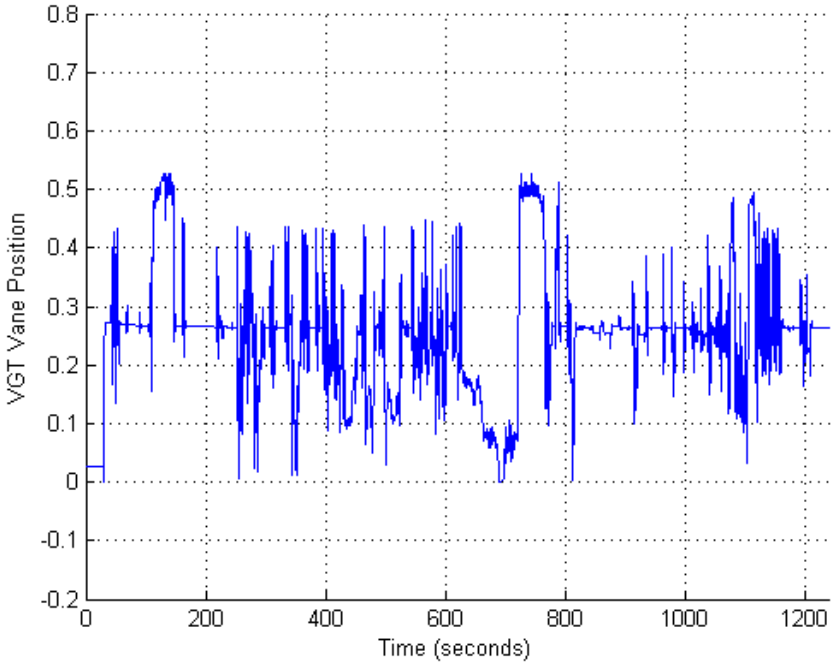

Figure 4.5 CIF Design, VGT Vane Command

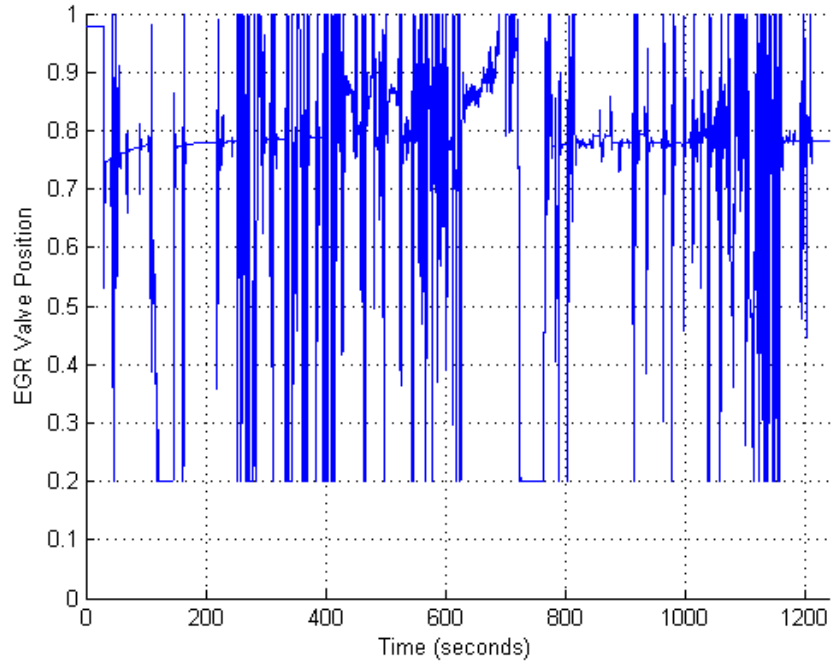

Figure 4.6 CIF Design, EGR Valve Command

4.3 CIF Design Summary and Controller Comparison

As has been seen in the analysis of this chapter (plus the analysis of the previous chapter), the architecture provided here proves to be a plausible - and potentially preferable - design for a decoupling controller. Reviewing the requirements stated at the start, this design maintains a simple architecture, as it is based off of a diagonal PI design and counteracts the off-diagonal interactive plant dynamics in a simple and easy to understand manner - especially if the SCIF calculations are implemented. The factorization analysis from this chapter proved out that the decoupling behavior of this design is a truer decoupling than the intuitive feed-forward design, and that because of this the simulation results are easier to understand, which provides for ease of implementation, use/calibration/tuning, troubleshooting if the system isn't behaving as expected, and greater fault tolerance should one of the actuators fail during operation.

In short, the review of the CIF design from above generates the following main ideas.

- The "intuitive" decoupling design of feed-forward decoupling elements has some significant drawbacks that are not readily apparent; the CIF design reduces or eliminates these shortcomings (i.e. a truer decoupling capability, better decoupling performance, elimination of integrator fighting to name a few).

- In many aspects the CIF design - while kept simple and easy to understand - performed on par with the fully multivariable design studied, and with fewer internal degrees of freedom (i.e. controller states).

- While the CIF decoupling solutions (Equation 4.5) have the form of a derivative, which means their effect goes to zero at steady-state, means they also have a potential drawback in that they can amplify high-frequency noise.

4.4 Comparison of Performance and Decoupling Metrics of Analyzed Controllers

At this point, the comparisons made thus far between the designs have all been very qualitative, and it is desired at this point to make a more quantitative comparison. It would be

tempting at this point to plot all the percent EGR, in-cylinder AFR, etc. responses and actuator commands from the different controllers, and while this is feasible for the simple step response (Figure 4.7), the long test simulation times and fast sampling rate of the NRTC engine case study simulations makes these types of plot comparisons very difficult to read and glean useable information from. Because the ISE and TV were metrics used to evaluate and compare the performance of the various controllers, the cumulative values of these metrics have been plotted for each of the controlled variables and actuator commands. This makes a visual quantitative analysis much easier to accomplish. In addition, Tables 4.1 through 4.3 provide numerical results of the metrics (and percent changes) of the controlled variables and actuator commands for a more direct quantitative comparison. One last item to mention about the TV metrics is the potential of them being skewed lower than what they actually may be due to saturation of an actuator for occasional periods of time (the difference calculation of the TV becomes zero). However, it terms of relative performance - and because all controller designs are subject to the same actuator constraints - it is still a valid metric for controller design comparison.

In Table 4.1, the peak value, 2% settling time, and 0% to 90% rise time have been captured for the reference response as well as the closed-loop linear air system model step responses as these are the control specifications stated at the start of this study. The percentage difference for each controller in these three columns has been calculated back to the reference response. As was discussed earlier, the Diagonal PI controller design does have a higher peak value, which all the controller designs do reduce, with the H_∞ and CIF designs eliminating the overshoot for the percent EGR response, but only the H_∞ designs eliminating it for the in-cylinder AFR. The CIF design does reduce the in-cylinder AFR overshoot to be more in line with the H_∞ design, thus showing that the CIF design performs on par with a full multivariable controller in this instance.

The ISE and TV metrics from Table 4.1 are not calculated for the reference response, because for the various controller designs, the "error" in ISE is in reference back to the desired reference response, and because the TV is calculated for the actuator commands which do not exist for the reference response. In addition, the ISE and TV percent changes are not

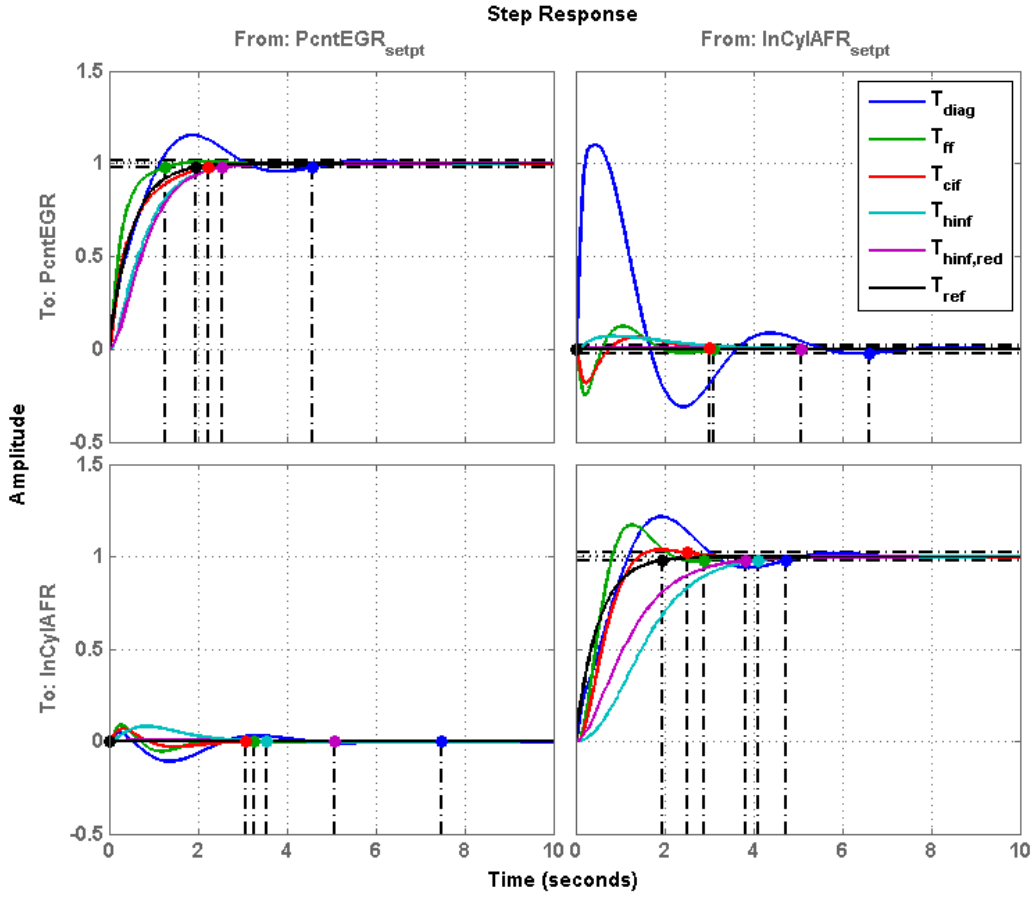

Figure 4.7 Closed Loop Step Response Comparison

calculated for the Diagonal PI design, as this is the baseline that the remaining architectures' percentage changes are calculated back to. The gain reduction aspects of the H_∞ designs are again apparent here as they are the only designs that increases the ISE compared to the Diagonal PI controller. However, the CIF design is again proved out to be the best design in terms of being able to converge the most to the reference response. An interesting aspect to note about the TV metrics though, is that even though the case study analysis plots for the H_∞ and CIF designs show that the H_∞ designs are much smoother on the actuators, for the linear analysis the CIF design is again the best in that it uses the least "activity" to achieve the desired decoupling response.

Table 4.1 Linear Air System Model Performance Metrics

Percent EGR Control Loop					
	Max. Peak	Settling Time	Rise Time	ISE	TV
Reference Response	1.00	1.96	1.15	–	–
	(–)	(–)	(–)		
Diagonal PI (Baseline)	1.15	4.57	0.942	0.0373	2326.4
	(15%)	(133%)	(-18%)	(–)	(–)
Feed-forward Design	1.01	1.26	0.695	0.0250	1511.8
	(1%)	(-36%)	(-40%)	(-33%)	(-35%)
H_∞ 2DoF LS Obs.	1.00	2.56	1.64	0.0557	1157.9
	(0%)	(31%)	(43%)	(49%)	(-50%)
H_∞ 2DoF LS Obs. (Red.)	1.00	2.56	1.69	0.0772	1345.5
	(0%)	(31%)	(47%)	(107%)	(-42%)
Cross Interaction Filter	1.00	2.25	1.32	0.0011	981.8
	(0%)	(15%)	(15%)	(-97%)	(-58%)
In-Cylinder AFR Control Loop					
	Max. Peak	Settling Time	Rise Time	ISE	TV
Reference Response	1.00	1.96	1.15	–	–
	(–)	(–)	(–)		
Diagonal PI (Baseline)	1.22	4.74	1.03	0.073	15860.0
	(22%)	(142%)	(-10%)	(–)	(–)
Feed-forward Design	1.17	2.89	0.727	0.0652	11291.0
	(17%)	(47%)	(-37%)	(-11%)	(-29%)
H_∞ 2DoF LS Obs.	1.00	4.12	2.94	0.4915	7354.3
	(0%)	(110%)	(156%)	(573%)	(-54%)
H_∞ 2DoF LS Obs. (Red.)	1.00	3.84	2.52	0.2402	8866.1
	(0%)	(96%)	(119%)	(229%)	(-44%)
Cross Interaction Filter	1.04	2.51	1.12	0.0417	6191.1
	(4%)	(28%)	(-3%)	(-43%)	(-61%)

Turning now to the decoupling metrics in Table 4.2, similar data is again shown. The reference response is absent from this table, as the ideal decoupled response produces a zero response in the off-diagonal elements and as such would generate a zero target metric in each category. Thus the goal of all these controllers is to minimize these values as much as possible. In addition, the rise time is not present as the intent again is to not have any response in these channels, and since rise time is calculated with the expectation of changing the steady-state value of the response, it has no meaning for the decoupling analysis as the desire is to not change from the steady-state value. The 2% settling time is still applicable though, in that it represents

how long it takes a step disturbance from the other control loop to die out. As a result, all the percent change metrics from this table are calculated back to the baseline Diagonal PI design. Because this analysis pertains to the off diagaonal responses, the interpretation of what the data means can be a bit tricky at firsst. As an example, for the percent EGR loop, the maximum peak value is due to a setpoint step change in the in-cylinder AFR variable. The TV variable then is how much VGT Vane movement was needed to drive the disturbance effect to zero, and the remaining variables should retain much of the same meaning as from Table 4.1.

Table 4.2 Linear Air System Model Decoupling Metrics

Percent EGR Control Loop				
	Max. Abs. Peak	Settling Time	ISE	TV
Diagonal PI (Baseline)	1.10	6.59	1.1018	6423.1
	(–)	(–)	(–)	(–)
Feed-forward Design	-0.249	3.1	0.0248	3774.1
	(-123%)	(-53%)	(-98%)	(-41%)
H_∞ 2DoF LS Obs.	0.0694	5.07	0.0075	3276.0
	(-94%)	(-23%)	(-99%)	(-49%)
H_∞ 2DoF LS Obs. (Red.)	0.00767	5.07	0.00008	3890.1
	(-99%)	(-23%)	(-100%)	(-39%)
Cross Interaction Filter	-0.185	3.03	0.0139	2200.4
	(-117%)	(-54%)	(-99%)	(-66%)
In-Cylinder AFR Control Loop				
	Max. Abs. Peak	Settling Time	ISE	TV
Diagonal PI (Baseline)	-0.108	7.47	0.0127	4690.8
	(–)	(–)	(–)	(–)
Feed-forward Design	0.888	3.27	0.0042	2447.4
	(-922%)	(-56%)	(-67%)	(-48%)
H_∞ 2DoF LS Obs.	0.0818	3.55	0.0073	2043.7
	(-176%)	(-52%)	(-43%)	(-56%)
H_∞ 2DoF LS Obs. (Red.)	0.00767	5.07	0.00008	2056.3
	(-107%)	(-32%)	(-99%)	(-56%)
Cross Interaction Filter	0.0681	3.07	0.0025	1813.3
	(-163%)	(-59%)	(-80%)	(-61%)

As was seen from the data in Table 4.1, the CIF design still performs on par with the full multivariable H_∞ designs, as each one has the smallest magnitude peak for a disturbance response and total ISE, the full H_∞ for the percent EGR variable and CIF for the in-cylinder AFR. It is interesting to note - as it was in the analysis of Chapter 2 that the feed-forward

design here promises decent decoupling capability as its metrics show improvement over the baseline Diagonal PI controller, but in some cases are still an order of magnitude worse (or at best on par with) the H_∞ and CIF designs. Again - as was seen with the Linear Performance Metrics - the CIF design proved to use the least amount of "activity" to effectively decouple the system.

Part of the reason for using the linear step analysis is that for the case study simulations, it is difficult to be able to separate desired closed-loop response of a variable with the disturbance effect of simultaneous responses in the other control loop. As a result, only the ISE and TV metrics are shown in Table 4.2 (the table shows the final values of the metrics). Figures 4.8 through 4.11 show comparisons of the cumulative ISE and TV values for the NRTC engine simulation case study, as these types of plots are easier to compare visually than the actual NRTC responses plotted on top of each other.

The PID integrator fighting issue - compounded also by the non-linear engine system - of the feed-forward decoupling design is immediately apparent in the tables and plots, as both sets of metrics for the feed-forward decoupling design are multiple orders of magnitude above all the other metrics, and the ISE traces for the feed-forward decoupling design are allowed to go off the top of Figures 4.8 and 4.9 so that the other traces can be compared. As Table 4.2 indicates and Figures 4.8 shows for the percent EGR variable, the CIF design does provide substantial improvement over the baseline Diagonal PI controller. Interestingly enough the H_∞ designs perform slightly worse than the baseline, possibly an artifact of the increase response time of this design and the effects of operating away from the linearization point. For the in-cylinder AFR response, the baseline Diagonal PI controller proves to be the best, with both the H_∞ and the CIF designs performing essentially the same in terms of difference to the desired response.

Table 4.2 and Figures 4.10 and 4.11 clearly show the decreased actuator usage of the H_∞ designs. While this wasn't the case with the linear responses, this begins to hint at the superior robustness properties of this style of controller, as operation on a non-linear system away from the operating point constitutes a type of uncertainty (i.e. model uncertainty represented by a complex full matrix disturbance). Conversely, the feed-forward decoupling design also proves to be the architecture needing the most actuator "activity" to accomplish the control task, which

Table 4.3 Engine Case Study Performance Metrics

Percent EGR Control Loop		
	ISE	TV
Diagonal PI (Baseline)	0.3275	4041.1
	(–)	(–)
Feed-forward Design	139.1461	30986.0
	(42388%)	(667%)
H_∞ 2DoF LS Obs.	0.3616	1847.5
	(10%)	(-54%)
H_∞ 2DoF LS Obs. (Red.)	0.3712	1824.7
	(13%)	(-55%)
Cross Interaction Filter	0.1693	8709.5
	(-48%)	(116%)
In-Cylinder AFR Control Loop		
	ISE	TV
Diagonal PI (Baseline)	0.3926	24332.0
	(–)	(–)
Feed-forward Design	75.3588	25650.0
	(19095%)	(5%)
H_∞ 2DoF LS Obs.	0.6494	4781.6
	(65%)	(-80%)
H_∞ 2DoF LS Obs. (Red.)	0.6630	4828.8
	(69%)	(-80%)
Cross Interaction Filter	0.6623	20486.0
	(69%)	(-16%)

was evident in the plots for this design from Chapter 2. In this case study analysis the VGT Vanes were moved around significantly more than the other designs, while the EGR Valve was only moved slightly more than the Diagonal PI design. Balancing the two plots together for the CIF design, one could conclude that it is about a wash when compared to the baseline design; the CIF design doesn't move the EGR Valve around as much, but has a similar magnitude scaling on the cumulative VGT Vane TV.

The final comparison results of the controller designs studied in this paper are summarized in the following list for the various simulations performed and metrics studied.

- In terms of reducing the maximum peak response (in both the main diagonal input/output pairings and the off-diagonal pairings), the H_∞ designs performed the best with complete

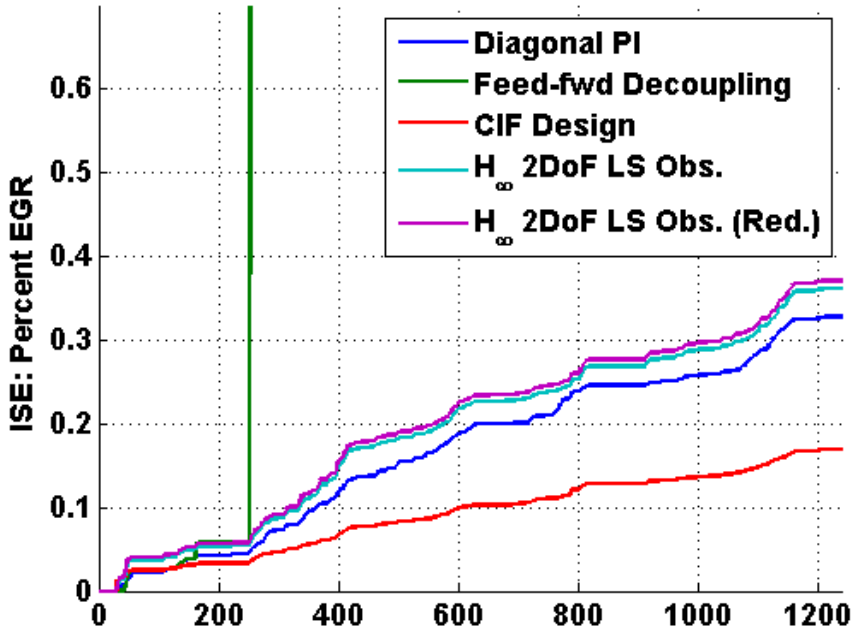

Figure 4.8 Cumulative Integral Square Error, Percent EGR

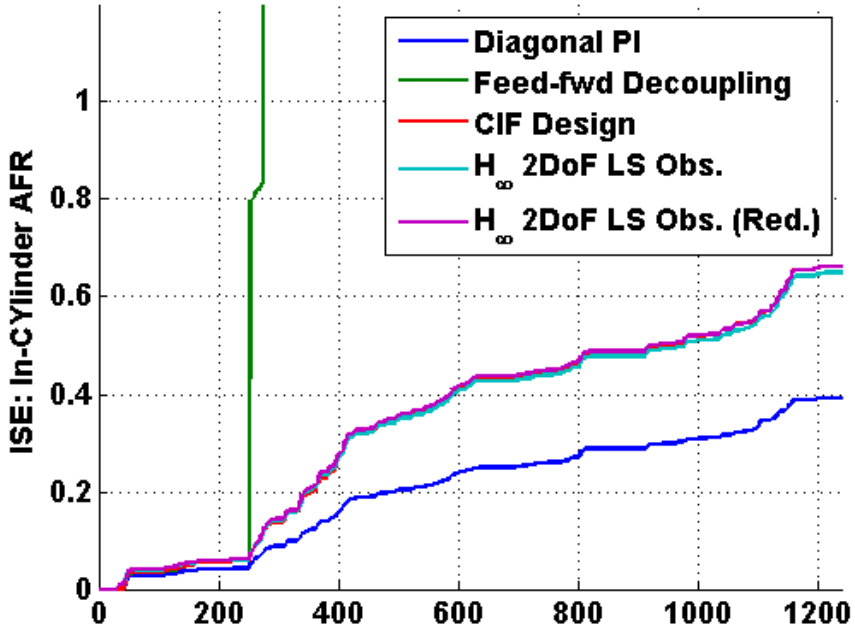

Figure 4.9 Cumulative Integral Square Error, In-Cylinder AFR

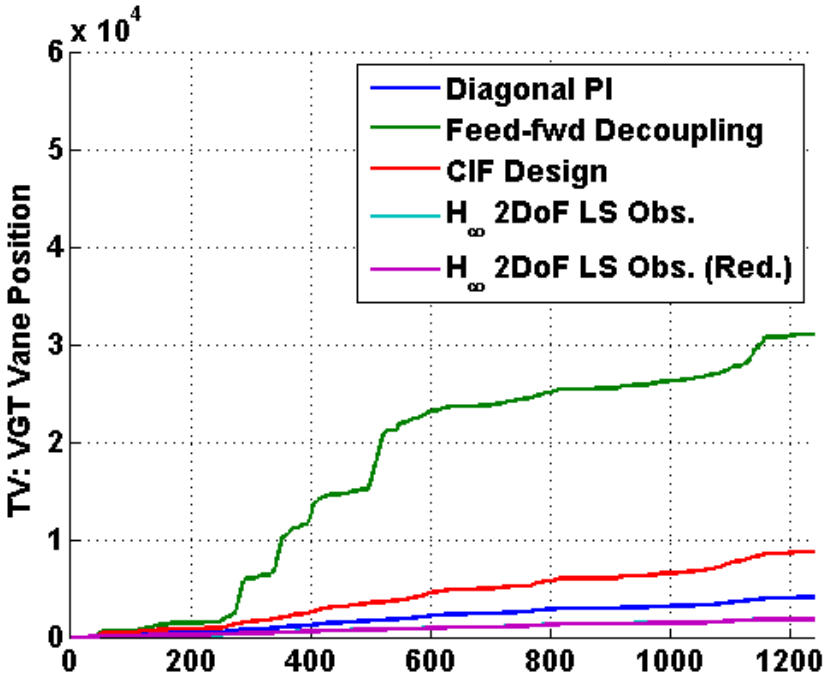

Figure 4.10 Cumulative Total Variation, VGT Vane Position

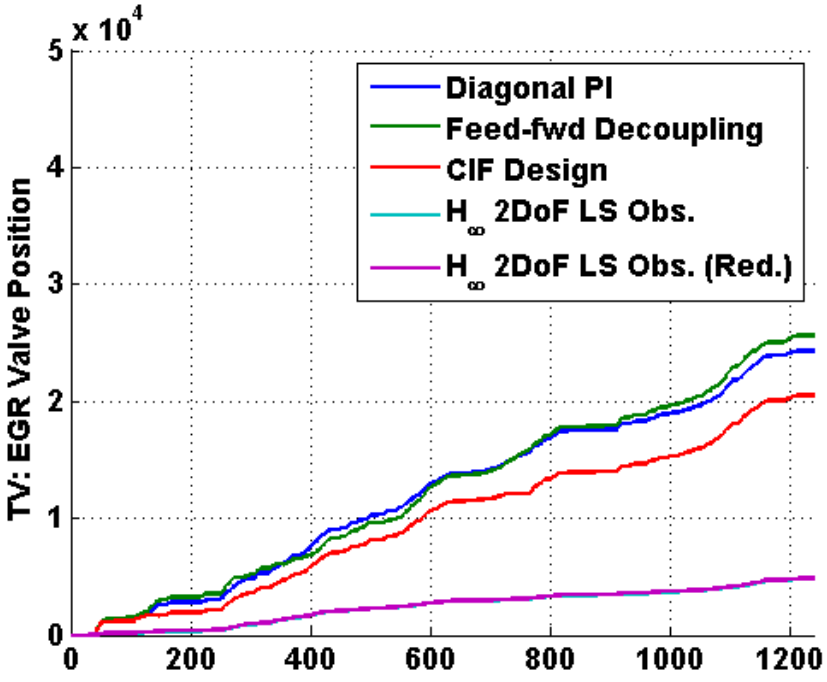

Figure 4.11 Cumulative Total Variation, EGR Valve Position

elimination of the overshoot peaks in the main diagonal and the nearly the best reduction in the off-diagonal peaks. However, the CIF design was not far behind and showed that - for a very simplistic architecture - it could satisfy the control requirements very well.

- The CIF design proved to be the best in terms of the settling time metric, in that it not only tracked the reference response the closest (smallest magnitude percentage change for both settling time and tracking time differences) but also reduced the disturbance response of the other loops setpoint change the fastest (largest negative percentage change).

- The CIF design was again the top performer in the ISE category with the smallest values for the main diagonal pairing and nearly the smallesst in the off-diagonal pairing (the H_∞ designs performed slightly better in this category for the off-diagonal pairings). In addition, the CIF design showed that it needed the least amount of actuator "activity" to achieve the control objectives of tracking the reference response and decoupling the system, as indicated by the CIF design having the smallest TV for the linear responses.

- In general, it would appear (based on the summary of the linear responses int the points above) that the CIF design out-performed all the others studied in this paper. As a result, we would expect the CIF design to perform the best in the engine simulation analysis. Though the CIF performed well for the engine simulation, it essentially split the ISE categories with the baseline Diagonal PI controller in terms of which design was the best, though the H_∞ designs did perform well on the percent EGR ISE metric and significantly out-performed all others in the amount of actuator "activity" needed (i.e. TV metric).

- One last item to note, is that while the "intuitive" feed-forward decoupling design showed some performance improvement on the surface, there are some underlying aspects of this design that clearly make it not a suitable choice for a controller design in this instance.

The last statement to make about these comparisons is that - like any controller design - the actual results are highly dependent on how the controllers are tuned. While some strategies were presented in this paper on how to tune the controller designs studied, they are by no means an extensive list, and different tuning approaches may impact these results to favor one

design over another. Though this activity will be left for future work, it does highlight the notion that controller designers need to also think critically about what the motivations for the controller really are, and what implications does that have on methods and strategies for controller design, tuning, and implementation.

CHAPTER 5. CONCLUSION

This paper analyzed four different controller architectures to evaluate their relative performance benefits or shortcomings in controlling a fourth order linear system as well as a non-linear model of a diesel engine air system. As part of the analysis, the general desire to have a controller and a system that is simple to understand, implement, analyze, and calibrate was balanced with a need to meet tight controller performance objectives and overall system robustness. The performance metrics used show that the simple diagonal PI controller architecture doesn't necessarily meet all the desired objectives; the fully multivariable design performs quite well. Interestingly enough, for the engine air system model, the cross interaction filter design to be quite an effective design in terms of balancing simplicity of design with decoupling performance and ability to meet tight control objectives. Even with the noted similarities between the cross interaction filter design and the feed-forward decoupling controller, the performance and decoupling capability of the feed-forward design performed worse than the cross interaction filters or the diagonal PI controllers (in certain instances), especially in the realistic engine model simulation. The simplicity and capability of the cross interaction filter design - along with the truer decoupling nature, something that is not readily apparent in the "intuitive" feed-forward decoupling design - then appears to be a promising method for controller design, and one that can be easily broken into design phases: the first phase to design the diagonal PI controllers, and the second to use the solutions presented in this paper to calibrate the cross interaction filters themselves.

Because this rasearch was more of a knowledge building exercise and somewhat open ended in its goals, future directions to extend this work could head a few different directions. Looking first at the area of engine control, the first direction to extend this work would be to evaluate and analyze the engine air system at more than just a single operating point. One could then

look into designing and evaluating controller designs with some aspect of gains scheduling to see what performance improvements could be attained. In addition, there are more factors that influence the regulated emissions than just the variables studied here, and the reduction of regulated emissions really is the high level goal of the controller design. Therefore, additional sensors and actuators can be added into the controller designs.

So far the future directions mentioned above haven't mentioned anything about operation off of nominal ambient conditions. Once a controller design is synthesized for across the operating envelop under nominal conditions, the next exploration step would be to evaluate it in off-nominal cases (i.e. cold temperatures, high altitude, etc). Coupled with an analysis of part to part variablity and a characterization of the air system model error, one could begin to realize the fuller potential of the H_∞ designs in creating a controller that is robust against all expected variations in environment and the engine system.

Since one of the control specifications of this study was a decoupled system and a different decoupling architecture proposed, yet another direction to take this work would be to continue to explore the domain of decentralized and decoupling control. The decoupling soultions explored here were developed only for a 2×2 system, therefore a first next step could be to extend this work into larger systems (also tying in the point from the engine control future direction of adding additional sensors and actuators). Along these lines, the solutions for the decoupling controllers were also derived from a system decomposition into independent SISO elements; the ideas explored here could be generalized into a MIMO block diagonal system and evaluated for potential benefit on much larger systems. Two other last direction points that were mentioned in previous chapters is that the current decoupling solutions derived rely on ratios of transfer functions. While a few potential methods were explored here to help address potential inversion issues, more research can be done into better or more effective ways of managing the inversion of plant dynamics. Secondly, from the comparison of the decoupling controller factorizations into the baseline diagonal PI design and a decoupling network to augment the plant (Section 4.1.4), it was shown that some of the assumptions made in deriving the decoupling solutions had impacts that are not readily apparent. Future work could then explore more into the assumptions that were made at the start of those solutions to see if they

can be removed and replaced with other aspects of a design procedure to improve synthesis of a decoupling controller.

In conclusion, the study was started with the intent of exploring controller designs that would maintain a simple, easy to understand architecture, would decouple the closed loop system so that troubleshooting becomes easier and fault tolerance to failed actuators is increased. In addition, these requirements were balanced against the stringent control objectives that imposed upon diesel engine air system control and derived from emissions regulations mandated by government agencies. The final results of the cross interaction filter design meet these stated goals, as far as the analysis in this study has exercised this architecture, and does so in a manner that shows it to be a very promising architecture while trying to balance the conflicting requirements that are part of modern diesel engine control.

APPENDIX A. ENGINE SYSTEM ANALYSIS MATERIAL

This section details the system identification analysis procedure and presents the results of the linear fourth order model tuning. In addition, MATLAB code from the open loop analysis, linear analysis, and case study analysis - including the metrics calculations - will be presented.

System Identification Procedure and Results

To collect the system identification data, the engine plant model was run in Simulink at the Mode 1 engine speed/load operating point. The command inputs for the EGR Valve and VGT Vane positions were obtained from a pre-developed steady-state calibration, using the Mode 1 operating point as the input condition for the lookup table. The steady-state map outputs were each offset with a Poisson random signal (generated in Simulink's Signal Builder block) with a 0.2 Hz average rate and a 0.2 amplitude for the EGR Valve command offset, and a 0.1 amplitude for the VGT Vane command. The Poisson random signal was used to excite random step deviations in the EGR Valve and VGT Vane position commands, and to have a mix of rises, falls, offset high and offset low conditions. The Poisson random signal was also active only during the last half of a 180 second simulation run, so that the engine model would have sufficient time to reach equilibrium. Results of the system identification are shown in Figure A.1, with the output signals having a Normalized Root Mean Square Error fit of 88.13% and 86.11%, respectively. Once the source data had been collected, the first half of the inputs and outputs were truncated and the detrended, so that both the inputs and the outputs all started at a magnitude of zero. The output signals were then rescaled to a point where the changes in magnitude were all approximately the same size.

The MATLAB commands used to process the raw data and generate the linear fourth order

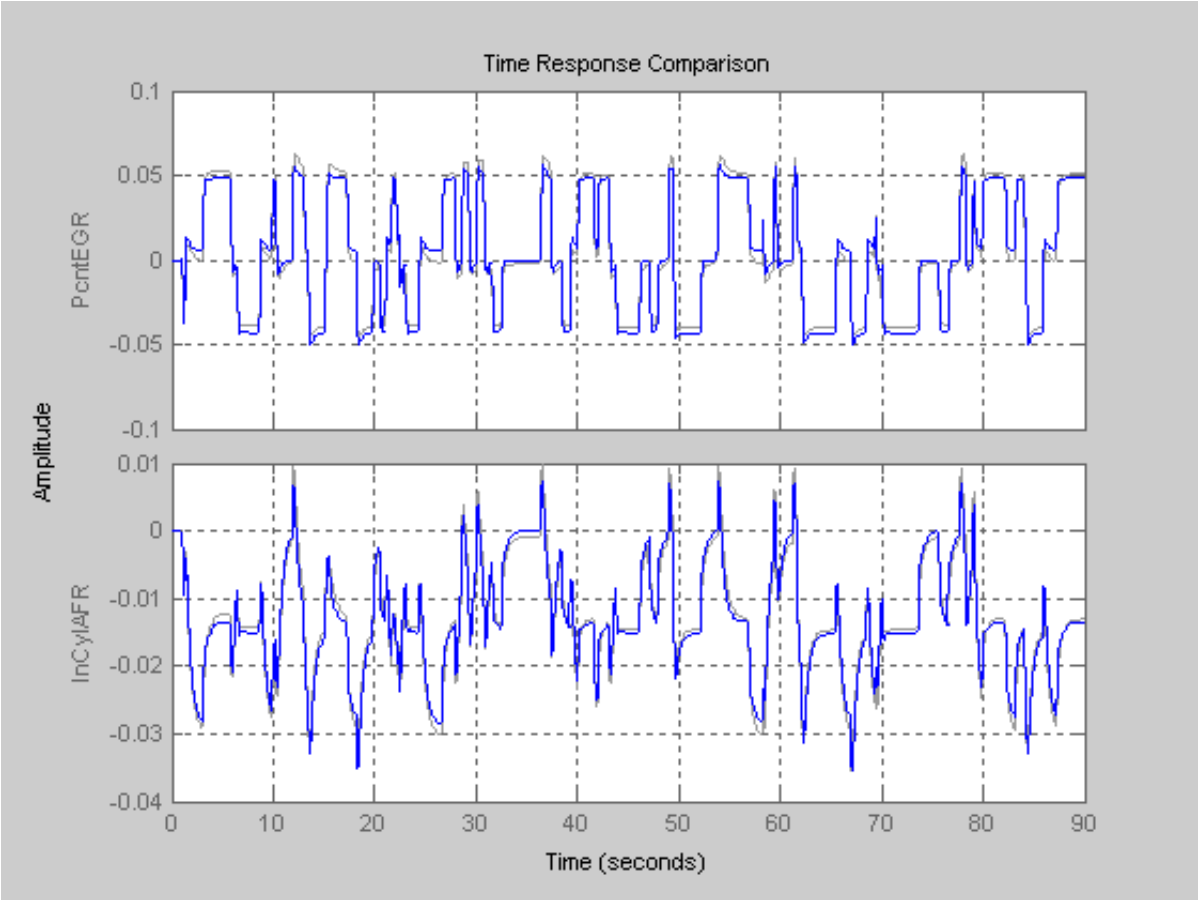

Figure A.1 System Identification Results

model are shown below.

```
%%  Load raw system identification data
load('20130321_thesis_workspace_sysid_results',...
    'air_sys_data_raw');

%%  Truncate first half of data (engine warming up)
air_sys_data = air_sys_data_raw(90001:end,:,:);
air_sys_data.Tstart = 0;

%%  Detrend data (inputs/output all begin at 0)
PcntEGR0 = air_sys_data_raw.OutputData(90000,1);
```

```
InCylAFR0 = air_sys_data_raw.OutputData(90000,2);
air_sys_data.OutputData = air_sys_data.OutputData - ...
    [PcntEGR0*ones(90001,1) InCylAFR0*ones(90001,1)];

%%  Create state-space model & comparison plot
air_sys_data.OutputName = {'PcntEGR','InCylAFR'};
air_sys_data.InputName = {'VGT_{vane}','EGR_{vlv}'};
air_sys_model = ssest(air_sys_data,4,'Ts',0,...
    'InitialState','backcast','Focus','prediction','MaxIter',40);
linear_air_sys = ss(air_sys_model(:,'measured'));
figure; compare(air_sys_data,air_sys_model);
```

Open Loop Analysis Code

The following code was used to perform the analysis on the open loop engine air system model.

```
%%  Load data
load('20130321_thesis_workspace_sysid_results','linear_air_sys');

%%  Pole/Zero Analysis
p = pole(linear_air_sys);
z = tzero(linear_air_sys);
figure; pzmap(linear_air_sys);

%%  Singular Values
freqs = logspace(-1,5,300)';
figure; sigmaplot(linear_air_sys,freqs);
max_sigma = arrayfun(@(f) max(svd(freqresp(linear_air_sys,f))),freqs);
min_sigma = arrayfun(@(f) min(svd(freqresp(linear_air_sys,f))),freqs);

%%  Bode plots w/ stability margins & Singual Value bounding
%   need to plot each I/O pairing on one plot, otherwise unable
%   to use Matlab charateristics to highlight margins
```

```
figure; bodeplot(linear_air_sys(1,1),linear_air_sys(1,2),...
    linear_air_sys(2,1),linear_air_sys(2,2),...
    frd(max_sigma,freqs),'k--',frd(min_sigma,freqs),'k--');

%%  Condition Number
gamma = max_sigma./min_sigma;
figure; h = sigmaplot(frd(gamma,freqs));
title_opts = getoptions(h,'Title'); title_opts.String = 'Condition Number';
label_opts = getoptions(h,'YLabel'); label_opts.String = 'Condition Number';
setoptions(h,'MagUnits','abs','Grid','on',...
    'Title',title_opts,'YLabel',label_opts);

%%  Frequency Dependent RGA, RGA Number
for k=1:numel(freqs)
    Gf = freqresp(linear_air_sys,freqs(k));
    RGAw(:,:,k) = Gf.*pinv(Gf).';
    RGAno_Main(k) = sum(sum(abs(RGAw(:,:,k) - eye(2))));
    RGAno_Off(k) = sum(sum(abs(RGAw(:,:,k) - [0 1; 1 0])));
end
RGA = frd(RGAw,freqs);
figure; h = bodeplot(RGA(1,1),RGA(1,2));
title_opts = getoptions(h,'Title');
title_opts.String = 'Frequency Dependent RGA';
label_opts = getoptions(h,'YLabel');
label_opts.String{1} = 'Frequency Dependent RGA';
setoptions(h,'MagUnits','abs','PhaseVisible','off',...
    'Title',title_opts,'YLabel',label_opts);
figure; h = bodeplot(frd(RGAno_Main,freqs),frd(RGAno_Off,freqs));
title_opts = getoptions(h,'Title');
title_opts.String = 'RGA Number';
label_opts = getoptions(h,'YLabel');
label_opts.String{1} = 'RGA Number';
setoptions(h,'MagUnits','abs','PhaseVisible','off',...
    'Title',title_opts,'YLabel',label_opts);
%   calculate individual frequency point RGAs
```

```
RGA_ss = dcgain(linear_air_sys).*pinv(dcgain(linear_air_sys)).';
RGA_bw = freqresp(linear_air_sys,2).*pinv(freqresp(linear_air_sys,2)).';

%%  Interaction Measure (Diagonal Dominance)
%   main diagonal pairing
G = tf(linear_air_sys);
Gtilde = tf([G(1,1) 0; 0 G(2,2)]);
E = minreal((G - Gtilde)/Gtilde);
Es = minreal((G - Gtilde)/G);
muE = ssv(E,freqs)';
muEs = ssv(Es,freqs)';
figure; h = bodeplot(frd(muE,freqs),frd(muEs,freqs),freqs);
title_opts = getoptions(h,'Title');
title_opts.String = 'SSV for Diagonal Dominance: Main Diagonal Pairing';
label_opts = getoptions(h,'YLabel');
label_opts.String{1} = 'Structured Singular Value';
setoptions(h,'MagUnits','abs','PhaseVisible','off',...
    'Grid','on','Title',title_opts,'YLabel',label_opts);
%   off diagonal pairing
G = tf(linear_air_sys([2 1],:));
Gtilde = tf([G(1,1) 0; 0 G(2,2)]);
E = minreal((G - Gtilde)/Gtilde);
Es = minreal((G - Gtilde)/G);
muE = ssv(E,freqs)';
muEs = ssv(Es,freqs)';
figure; h = bodeplot(frd(muE,freqs),frd(muEs,freqs),freqs);
title_opts = getoptions(h,'Title');
title_opts.String = 'SSV for Diagonal Dominance: Off-Diagonal Pairing';
label_opts = getoptions(h,'YLabel');
label_opts.String{1} = 'Structured Singular Value';
setoptions(h,'MagUnits','abs','PhaseVisible','off',...
    'Grid','on','Title',title_opts,'YLabel',label_opts);
```

Linear System Analysis Code

The following code was used to create the controllers and perform the linear step response analysis of the controller designs.

Baseline tuning:

```
%%  Perform diagonal PI tunings
s = tf('s');
[~,Kc,tauI] = simc_tuning(g11,tau_c);
PI_11 = Kc*(1 + 1/(tauI*s));
PI_11.InputName = 'PcntEGR_{error}';
PI_11.OutputName = 'VGT_{vane}';
[~,Kc,tauI] = simc_tuning(g22,tau_c);
PI_22 = Kc*(1 + 1/(tauI*s));
PI_22.InputName = 'InCylAFR_{error}';
PI_22.OutputName = 'EGR_{vlv}';

%%  Create individual closed loop systems and response plots
%   Percent EGR Control Loop
Tref_11 = 1/(tau_c*s + 1);
Tref_11.InputName = 'PcntEGR_{setpt}';
Tref_11.OutputName = 'PcntEGR';
PcntEGRErrorBlk = sumblk('%e = %r - %y',PI_11.u,Tref_11.u,g11.y);
T_11 = connect(g11,PI_11,PcntEGRErrorBlk,Tref_11.u,g11.y);
T_11_full = connect(g11_full,PI_11,PcntEGRErrorBlk,Tref_11.u,g11.y);
figure; h = stepplot(Tref_11,T_11,T_11_full);
setoptions(h,'Grid','on');
%   In-Cylinder AFR Control Loop
Tref_22 = 1/(tau_c*s + 1);
Tref_22.InputName = 'InCylAFR_{setpt}';
Tref_22.OutputName = 'InCylAFR';
InCylAFRErrorBlk = sumblk('%e = %r - %y',PI_22.u,Tref_22.u,g22.y);
T_22 = connect(g22,PI_22,InCylAFRErrorBlk,Tref_22.u,g22.y);
T_22_full = connect(g22_full,PI_22,InCylAFRErrorBlk,Tref_22.u,g22.y);
```

```
figure; h = stepplot(Tref_22,T_22,T_22_full);
setoptions(h,'Grid','on');
```

Controller designs:

```
%%  Load data
load('20130322_thesis_workspace_initial_tuning');

Tref = append(Tref_11,Tref_22);
clear Tref_11 Tref_22;

t = [0:0.001:10]';
y_ref = step(Tref,t);

%%  Connect Diagonal PI Controller & Control Loop (Baseline)
K_diag = append(PI_11,PI_22);
ErrorBlk = sumblk('%e = %r - %y',K_diag.u,Tref.u,linear_air_sys.y);
T_diag = minreal(connect(linear_air_sys,K_diag,ErrorBlk,Tref.u,linear_air_sys.y));
S_diag = minreal(connect(linear_air_sys,K_diag,ErrorBlk,Tref.u,linear_air_sys.u));
figure; h = stepplot(Tref,T_diag,t);
setoptions(h,'Grid','on');
%   calculate metrics
y_diag = step(T_diag,t);
u_diag = step(S_diag,t);
ISE_y_diag = 0.001*cumsum((y_ref - y_diag).^2);
TV_u_diag = cumsum(abs([zeros(1,2,2); diff(u_diag)]))/0.001;

%%  Connect Feed-forward Decoupling Controller & Control Loop
PI_11.OutputName = 'VGT_{vane,PI}';
PI_22.OutputName = 'EGR_{vlv,PI}';
% delay is neglected, << tau_c
Gc_12 = minreal(-(set(g12,'InputDelay',0)*PI_22)/set(g11,'InputDelay',0));
Gc_12.InputName = PI_22.u; Gc_12.OutputName = 'VGT_{vane,FF}';
% delay is neglected, ease of calculations
Gc_21 = minreal(-(set(g21,'InputDelay',0)*PI_11)/set(g22,'InputDelay',0));
```

```
Gc_21.InputName = PI_11.u; Gc_21.OutputName = 'EGR_{vlv,FF}';
VGTSumBlk = sumblk('%u = %u_pi + %u_ff',g11.u,PI_11.y,Gc_12.y);
EGRSumBlk = sumblk('%u = %u_pi + %u_ff',g22.u,PI_22.y,Gc_21.y);
K_ff = connect(PI_11,PI_22,Gc_12,Gc_21,...
    VGTSumBlk,EGRSumBlk,K_diag.u,linear_air_sys.u);
ErrorBlk = sumblk('%e = %r - %y',K_ff.u,Tref.u,linear_air_sys.y);
T_ff = minreal(connect(linear_air_sys,K_ff,ErrorBlk,Tref.u,linear_air_sys.y));
S_ff = minreal(connect(linear_air_sys,K_ff,ErrorBlk,Tref.u,linear_air_sys.u));
figure; h = stepplot(Tref,T_ff,t);
setoptions(h,'Grid','on');
%   calculate metrics
y_ff = step(T_ff,t);
u_ff = step(S_ff,t);
ISE_y_ff = 0.001*cumsum((y_ref - y_ff).^2);
TV_u_ff = cumsum(abs([zeros(1,2,2); diff(u_ff)]))/0.001;

%%  Connect CIF Decoupling Controller & Control Loop
%   reload individual PI controllers
load('20130322_thesis_workspace_initial_tuning.mat','PI_11','PI_22');
PI_11.InputName = 'PcntEGR_{error,adj}';
PI_22.InputName = 'InCylAFR_{error,adj}';
% delay is neglected, << tau_c
W_12 = minreal(-set(g12,'InputDelay',0)/(set(g11,'InputDelay',0)*PI_11));
W_12.InputName = PI_22.y; W_12.OutputName = 'InCylAFR_{cif}';
% delay is neglected, ease of calculations
W_21 = minreal(-set(g21,'InputDelay',0)/(set(g22,'InputDelay',0)*PI_22));
W_21.InputName = PI_11.y; W_21.OutputName = 'PcntEGR_{cif}';
PcntEGRSumBlk = sumblk('%e_adj = %e + %u_cif',PI_11.u,'PcntEGR_{error}',W_12.y);
InCylAFRSumBlk = sumblk('%e_adj = %e + %u_cif',PI_22.u,'InCylAFR_{error}',W_21.y);
K_cif = minreal(connect(PI_11,PI_22,W_12,W_21,...
    PcntEGRSumBlk,InCylAFRSumBlk,K_diag.u,linear_air_sys.u));
ErrorBlk = sumblk('%e = %r - %y',K_cif.u,Tref.u,linear_air_sys.y);
T_cif = minreal(connect(linear_air_sys,K_cif,ErrorBlk,Tref.u,linear_air_sys.y));
S_cif = minreal(connect(linear_air_sys,K_cif,ErrorBlk,Tref.u,linear_air_sys.u));
figure; h = stepplot(Tref,T_cif,t);
```

```
setoptions(h,'Grid','on');
%   calculate metrics
y_cif = step(T_cif,t);
u_cif = step(S_cif,t);
ISE_y_cif = 0.001*cumsum((y_ref - y_cif).^2);
TV_u_cif = cumsum(abs([zeros(1,2,2); diff(u_cif)]))/0.001;

%%  Connect Hinf Loop Shaping Observer Based 2 DoF Controller & Control Loop
%   create shaped plant, including alignment at bandwidth
%   (align algorithm of Kouvaritakis, as mentioned by Skogestad & Postlethwaite)
Wa = mvc_align(freqresp(linear_air_sys*K_diag,2));    %   align at 2 rad/sec
Wg = blkdiag(1,0.3);
W1 = K_diag*Wa*Wg; W1.InputName = K_diag.InputName;
Gs = connect(W1,linear_air_sys,K_diag.u,linear_air_sys.y);
%   synthesize matrices to satisfy Hinf criteria, create controller
[~,Fs,Fr,Hs] = hinf2dof_obs(Gs,Tref,1.10);
[As,Bs,Cs,~] = ssdata(Gs);  [Ar,Br,~,~] = ssdata(Tref);
K1 = ss(Ar,Br,Fr,0);
K1.InputName = Tref.InputName;
K1.OutputName = {'VGT_{vane,ref}','EGR_{vlv,ref}'};
K2 = ss(As+Hs*Cs,[Bs -Hs],Fs,0);
K2.InputName = [Gs.InputName; Gs.OutputName];
K2.OutputName = {'VGT_{vane,obs}','EGR_{vlv,obs}'};
HinfSumBlk = sumblk('%u = -%u_r - %u_obs',Gs.u,K1.y,K2.y);
%   connect initial controller/system, determine presacle gain
K_hinf = connect(K1,K2,HinfSumBlk,[K1.InputName; Gs.OutputName],Gs.InputName);
T_hinf = minreal(connect(K_hinf,Gs,Tref.u,Gs.y));
Wi = dcgain(Tref)/dcgain(T_hinf);
%   reassemble final controller/system
K1 = ss(Ar,Br*Wi,Fr,0);
K1.InputName = Tref.InputName;
K1.OutputName = {'VGT_{vane,ref}','EGR_{vlv,ref}'};
K_hinf = minreal(connect(K1,K2,HinfSumBlk,[K1.InputName; Gs.OutputName],Gs.InputName));
T_hinf = minreal(connect(K_hinf,Gs,Tref.u,Gs.y));
S_hinf = minreal(connect(K1,K2,HinfSumBlk,W1,linear_air_sys,Tref.u,linear_air_sys.u));
```

```
101
102 %   create reduced Hinf design
103 Gs_red = reduce(Gs,2);
104 Gs_red.OutputName = Gs.OutputName;
105 Gs_red.InputName = Gs.InputName;
106 %   synthesize matrices to satisfy Hinf criteria, create controller
107 [~,Fs_red,Fr_red,Hs_red] = hinf2dof_obs(Gs_red,Tref,1.10);
108 [As_red,Bs_red,Cs_red,~] = ssdata(Gs_red);
109 K1_red = ss(Ar,Br,Fr_red,0);
110 K1_red.InputName = Tref.InputName;
111 K1_red.OutputName = {'VGT_{vane,ref}','EGR_{vlv,ref}'};
112 K2_red = ss(As_red+Hs_red*Cs_red,[Bs_red -Hs_red],Fs_red,0);
113 K2_red.InputName = [Gs_red.InputName; Gs_red.OutputName];
114 K2_red.OutputName = {'VGT_{vane,obs}','EGR_{vlv,obs}'};
115 %   connect initial controller/system, determine presacle gain
116 K_hinf_red = connect(K1_red,K2_red,HinfSumBlk,...
117     [K1_red.InputName; Gs_red.OutputName],Gs_red.InputName);
118 T_hinf_red = minreal(connect(K_hinf_red,Gs_red,Tref.u,Gs_red.y));
119 Wi_red = dcgain(Tref)/dcgain(T_hinf_red);
120 %   reassemble final controller/system
121 K1_red = ss(Ar,Br*Wi_red,Fr_red,0);
122 K1_red.InputName = Tref.InputName;
123 K1_red.OutputName = {'VGT_{vane,ref}','EGR_{vlv,ref}'};
124 K_hinf_red = minreal(connect(K1_red,K2_red,HinfSumBlk,...
125     [K1_red.InputName; Gs_red.OutputName],Gs_red.InputName));
126 T_hinf_red = minreal(connect(K_hinf_red,Gs_red,Tref.u,Gs_red.y));
127 S_hinf_red = minreal(connect(K1_red,K2_red,HinfSumBlk,...
128     W1,linear_air_sys,Tref.u,linear_air_sys.u));
129
130 %   create plots for full & reduced designs
131 figure; h = stepplot(Tref,T_hinf,T_hinf_red,t);
132 setoptions(h,'Grid','on');
133 %   calculate metrics
134 y_hinf = step(T_hinf,t);
135 y_hinf_red = step(T_hinf_red,t);
```

```
u_hinf = step(S_hinf,t);
u_hinf_red = step(S_hinf_red,t);
ISE_y_hinf = 0.001*cumsum((y_ref - y_hinf).^2);
ISE_y_hinf_red = 0.001*cumsum((y_ref - y_hinf_red).^2);
TV_u_hinf = cumsum(abs([zeros(1,2,2); diff(u_hinf)]))/0.001;
TV_u_hinf_red = cumsum(abs([zeros(1,2,2); diff(u_hinf_red)]))/0.001;

%%  Create comparison plots
figure; pzmap(T_diag,T_ff,T_cif,T_hinf,T_hinf_red,Tref,'k');
figure; stepplot(T_diag,T_ff,T_cif,T_hinf,T_hinf_red,Tref,'k');
figure; h = sigmaplot(T_diag,T_ff,T_cif,T_hinf,T_hinf_red,Tref,'k');
setoptions(h,'Grid','on');
```

Engine Simulation Analysis Code

The closed loop engine simulation case study results were obtained by running the simulation commands shown below.

```
%%  Load data & model
load('20130323_thesis_workspace_all_ctrl_systems.mat');
cl_eng_air_mdl = load_system('GTPowerCoSim_Thesis2_ClosedLoop');
warm_start;

%%  Diagonal PI Controller (Baseline) Simulation
set_param('GTPowerCoSim_Thesis2_ClosedLoop/ECU/ControllerVariants',...
    'OverrideUsingVariant','DIAGONAL_PI');
sim('GTPowerCoSim_Thesis2_ClosedLoop');
logsout_DiagPI = logsout;

%%  Feed-forward Decoupling Controller Simulation
set_param('GTPowerCoSim_Thesis2_ClosedLoop/ECU/ControllerVariants',...
    'OverrideUsingVariant','FEEDFWD_DECOUP');
sim('GTPowerCoSim_Thesis2_ClosedLoop');
logsout_FeedFwd = logsout;
```

```
17
18 %%  Cross-Interaction Filter Controller Simulation
19 set_param('GTPowerCoSim_Thesis2_ClosedLoop/ECU/ControllerVariants',...
20     'OverrideUsingVariant','CROSS_INT_FILT');
21 sim('GTPowerCoSim_Thesis2_ClosedLoop');
22 logsout_CIF = logsout;
23
24 %%  Hinf 2DoF LS Observer Based Simulation
25 set_param('GTPowerCoSim_Thesis2_ClosedLoop/ECU/ControllerVariants',...
26     'OverrideUsingVariant','HINF_2DOF_OBS_LS');
27 sim('GTPowerCoSim_Thesis2_ClosedLoop');
28 logsout_Hinf = logsout;
29
30 %%  Hinf 2DoF LS Observer Based (Reduced) Simulation
31 set_param('GTPowerCoSim_Thesis2_ClosedLoop/ECU/ControllerVariants',...
32     'OverrideUsingVariant','HINF_2DOF_OBS_LS_RED');
33 sim('GTPowerCoSim_Thesis2_ClosedLoop');
34 logsout_Hinf_Red = logsout;
35
36 %%  Collect & save off all data
37 clear logsout;
38 save('20130323_case_study_simulation_results','logsout_DiagPI',...
39     'logsout_FeedFwd','logsout_CIF','logsout_Hinf','logsout_Hinf_Red');
```

The closed loop engine simulation results were then analyzed to generate the plots and metrics of Chapters 3 and 4 with the following commands.

```
1 %%  Load data
2 load('20130323_case_study_simulation_results');
3
4 %%  Process results
5 logsout_DiagPI = ProcessThesisCoSimResults(logsout_DiagPI);
6 logsout_FeedFwd = ProcessThesisCoSimResults(logsout_FeedFwd);
7 logsout_CIF = ProcessThesisCoSimResults(logsout_CIF);
8 logsout_Hinf = ProcessThesisCoSimResults(logsout_Hinf);
```

```
logsout_Hinf_Red = ProcessThesisCoSimResults(logsout_Hinf_Red);

%%  Plot Diagonal PI architecture (Baseline) results
figure; hold on; grid on;
plot(logsout_DiagPI.getElement('PcntEGRSetpoint').Values,'k');
plot(logsout_DiagPI.getElement('PercentEGR').Values,'g');
plot(logsout_DiagPI.getElement('PercentEGRVariable_ref').Values,'b');
set(gca,'Xlim',[0 1240],'Ylim',[0.1 0.4]); ylabel('Percent EGR');
figure; hold on; grid on;
plot(logsout_DiagPI.getElement('InCylAFRSetpoint').Values,'k');
plot(logsout_DiagPI.getElement('InCylAFR').Values,'g');
plot(logsout_DiagPI.getElement('InCylAFRVariable_ref').Values,'b');
set(gca,'Xlim',[0 1240],'Ylim',[0.1 0.5]); ylabel('In-Cylinder AFR');
figure; hold on; grid on;
plot(logsout_DiagPI.getElement('VGTVanePosition').Values);
set(gca,'Xlim',[0 1240],'Ylim',[-0.2 0.8]); ylabel('VGT Vane Position'); title('');
figure; hold on; grid on;
plot(logsout_DiagPI.getElement('EGRValvePosition').Values);
set(gca,'Xlim',[0 1240],'Ylim',[0.0 1.0]); ylabel('EGR Valve Position'); title('');

%%  Plot Feed-forward Decoupling architecture results
figure; hold on; grid on;
plot(logsout_FeedFwd.getElement('PcntEGRSetpoint').Values,'k');
plot(logsout_FeedFwd.getElement('PercentEGR').Values,'g');
plot(logsout_FeedFwd.getElement('PercentEGRVariable_ref').Values,'b');
set(gca,'Xlim',[0 1240],'Ylim',[0.1 0.4]); ylabel('Percent EGR');
figure; hold on; grid on;
plot(logsout_FeedFwd.getElement('InCylAFRSetpoint').Values,'k');
plot(logsout_FeedFwd.getElement('InCylAFR').Values,'g');
plot(logsout_FeedFwd.getElement('InCylAFRVariable_ref').Values,'b');
set(gca,'Xlim',[0 1240],'Ylim',[0.1 0.5]); ylabel('In-Cylinder AFR');
figure; hold on; grid on;
plot(logsout_FeedFwd.getElement('VGTVanePosition').Values);
set(gca,'Xlim',[0 1240],'Ylim',[-0.2 0.8]); ylabel('VGT Vane Position'); title('');
figure; hold on; grid on;
```

```
plot(logsout_FeedFwd.getElement('EGRValvePosition').Values);
set(gca,'Xlim',[0 1240],'Ylim',[0.0 1.0]); ylabel('EGR Valve Position'); title('');

%%  Plot Cross Interaction Filter architecture results
figure; hold on; grid on;
plot(logsout_CIF.getElement('PcntEGRSetpoint').Values,'k');
plot(logsout_CIF.getElement('PercentEGR').Values,'g');
plot(logsout_CIF.getElement('PercentEGRVariable_ref').Values,'b');
set(gca,'Xlim',[0 1240],'Ylim',[0.1 0.4]); ylabel('Percent EGR');
figure; hold on; grid on;
plot(logsout_CIF.getElement('InCylAFRSetpoint').Values,'k');
plot(logsout_CIF.getElement('InCylAFR').Values,'g');
plot(logsout_CIF.getElement('InCylAFRVariable_ref').Values,'b');
set(gca,'Xlim',[0 1240],'Ylim',[0.1 0.5]); ylabel('In-Cylinder AFR');
figure; hold on; grid on;
plot(logsout_CIF.getElement('VGTVanePosition').Values);
set(gca,'Xlim',[0 1240],'Ylim',[-0.2 0.8]); ylabel('VGT Vane Position'); title('');
figure; hold on; grid on;
plot(logsout_CIF.getElement('EGRValvePosition').Values);
set(gca,'Xlim',[0 1240],'Ylim',[0.0 1.0]); ylabel('EGR Valve Position'); title('');

%%  Plot Hinf 2DoF Loop Shaping Observer Based architecture results
figure; hold on; grid on;
plot(logsout_Hinf.getElement('PcntEGRSetpoint').Values,'k');
plot(logsout_Hinf.getElement('PercentEGR').Values,'g');
plot(logsout_Hinf.getElement('PercentEGRVariable_ref').Values,'b');
set(gca,'Xlim',[0 1240],'Ylim',[0.1 0.4]); ylabel('Percent EGR');
figure; hold on; grid on;
plot(logsout_Hinf.getElement('InCylAFRSetpoint').Values,'k');
plot(logsout_Hinf.getElement('InCylAFR').Values,'g');
plot(logsout_Hinf.getElement('InCylAFRVariable_ref').Values,'b');
set(gca,'Xlim',[0 1240],'Ylim',[0.1 0.5]); ylabel('In-Cylinder AFR');
figure; hold on; grid on;
plot(logsout_Hinf.getElement('VGTVanePosition').Values);
set(gca,'Xlim',[0 1240],'Ylim',[-0.2 0.8]); ylabel('VGT Vane Position'); title('');
```

```
figure; hold on; grid on;
plot(logsout_Hinf.getElement('EGRValvePosition').Values);
set(gca,'Xlim',[0 1240],'Ylim',[0.0 1.0]); ylabel('EGR Valve Position'); title('');

%%  Plot Hinf 2DoF Loop Shaping Observer Based Reduced architecture results
figure; hold on; grid on;
plot(logsout_Hinf_Red.getElement('PcntEGRSetpoint').Values,'k');
plot(logsout_Hinf_Red.getElement('PercentEGR').Values,'g');
plot(logsout_Hinf_Red.getElement('PercentEGRVariable_ref').Values,'b');
set(gca,'Xlim',[0 1240],'Ylim',[0.1 0.4]); ylabel('Percent EGR');
figure; hold on; grid on;
plot(logsout_Hinf_Red.getElement('InCylAFRSetpoint').Values,'k');
plot(logsout_Hinf_Red.getElement('InCylAFR').Values,'g');
plot(logsout_Hinf_Red.getElement('InCylAFRVariable_ref').Values,'b');
set(gca,'Xlim',[0 1240],'Ylim',[0.1 0.5]); ylabel('In-Cylinder AFR');
figure; hold on; grid on;
plot(logsout_Hinf_Red.getElement('VGTVanePosition').Values);
set(gca,'Xlim',[0 1240],'Ylim',[-0.2 0.8]); ylabel('VGT Vane Position'); title('');
figure; hold on; grid on;
plot(logsout_Hinf_Red.getElement('EGRValvePosition').Values);
set(gca,'Xlim',[0 1240],'Ylim',[0.0 1.0]); ylabel('EGR Valve Position'); title('');

%%  Plot comparisons
figure; hold on; plot(logsout_DiagPI.getElement('ISE_PcntEGR').Values.Time,...
    logsout_DiagPI.getElement('ISE_PcntEGR').Values.Data,...
    logsout_FeedFwd.getElement('ISE_PcntEGR').Values.Time,...
    logsout_FeedFwd.getElement('ISE_PcntEGR').Values.Data,...
    logsout_CIF.getElement('ISE_PcntEGR').Values.Time,...
    logsout_CIF.getElement('ISE_PcntEGR').Values.Data,...
    logsout_Hinf.getElement('ISE_PcntEGR').Values.Time,...
    logsout_Hinf.getElement('ISE_PcntEGR').Values.Data,...
    logsout_Hinf_Red.getElement('ISE_PcntEGR').Values.Time,...
    logsout_Hinf_Red.getElement('ISE_PcntEGR').Values.Data);
set(gca,'Xlim',[0 1240],'Ylim',[0 0.4]); grid on;
figure; hold on; plot(logsout_DiagPI.getElement('ISE_InCylAFR').Values.Time,...
```

```
114     logsout_DiagPI.getElement('ISE_InCylAFR').Values.Data,...
115     logsout_FeedFwd.getElement('ISE_InCylAFR').Values.Time,...
116     logsout_FeedFwd.getElement('ISE_InCylAFR').Values.Data,...
117     logsout_CIF.getElement('ISE_InCylAFR').Values.Time,...
118     logsout_CIF.getElement('ISE_InCylAFR').Values.Data,...
119     logsout_Hinf.getElement('ISE_InCylAFR').Values.Time,...
120     logsout_Hinf.getElement('ISE_InCylAFR').Values.Data,...
121     logsout_Hinf_Red.getElement('ISE_InCylAFR').Values.Time,...
122     logsout_Hinf_Red.getElement('ISE_InCylAFR').Values.Data);
123 set(gca,'Xlim',[0 1240],'Ylim',[0 0.8]); grid on;
124 figure; hold on; plot(logsout_DiagPI.getElement('TV_VGTVane').Values.Time,...
125     logsout_DiagPI.getElement('TV_VGTVane').Values.Data,...
126     logsout_FeedFwd.getElement('TV_VGTVane').Values.Time,...
127     logsout_FeedFwd.getElement('TV_VGTVane').Values.Data,...
128     logsout_CIF.getElement('TV_VGTVane').Values.Time,...
129     logsout_CIF.getElement('TV_VGTVane').Values.Data,...
130     logsout_Hinf.getElement('TV_VGTVane').Values.Time,...
131     logsout_Hinf.getElement('TV_VGTVane').Values.Data,...
132     logsout_Hinf_Red.getElement('TV_VGTVane').Values.Time,...
133     logsout_Hinf_Red.getElement('TV_VGTVane').Values.Data);
134 set(gca,'Xlim',[0 1240]); grid on;
135 figure; hold on; plot(logsout_DiagPI.getElement('TV_EGRValve').Values.Time,...
136     logsout_DiagPI.getElement('TV_EGRValve').Values.Data,...
137     logsout_FeedFwd.getElement('TV_EGRValve').Values.Time,...
138     logsout_FeedFwd.getElement('TV_EGRValve').Values.Data,...
139     logsout_CIF.getElement('TV_EGRValve').Values.Time,...
140     logsout_CIF.getElement('TV_EGRValve').Values.Data,...
141     logsout_Hinf.getElement('TV_EGRValve').Values.Time,...
142     logsout_Hinf.getElement('TV_EGRValve').Values.Data,...
143     logsout_Hinf_Red.getElement('TV_EGRValve').Values.Time,...
144     logsout_Hinf_Red.getElement('TV_EGRValve').Values.Data);
145 set(gca,'Xlim',[0 1240]); grid on;
```

APPENDIX B. ENGINE SYSTEM SIMULATION SETUP

Simulink/GT-SUITE Engine Model Interfaces

A standard GT-SUITE model develop by engineers at John Deere Power Systems was used for the analysis of the various air system controllers. It has an interface model wrapped around it to simplify the use of the system, and to prevent disclosure of any proprietary material (such as the normalization scalings and the names of the proprietary control variables used). To collect the open loop response data for system identification, the model setup in Figure B.1 was used. To perform closed loop simulations of the various controller designs and the engine model, see Figure B.2.

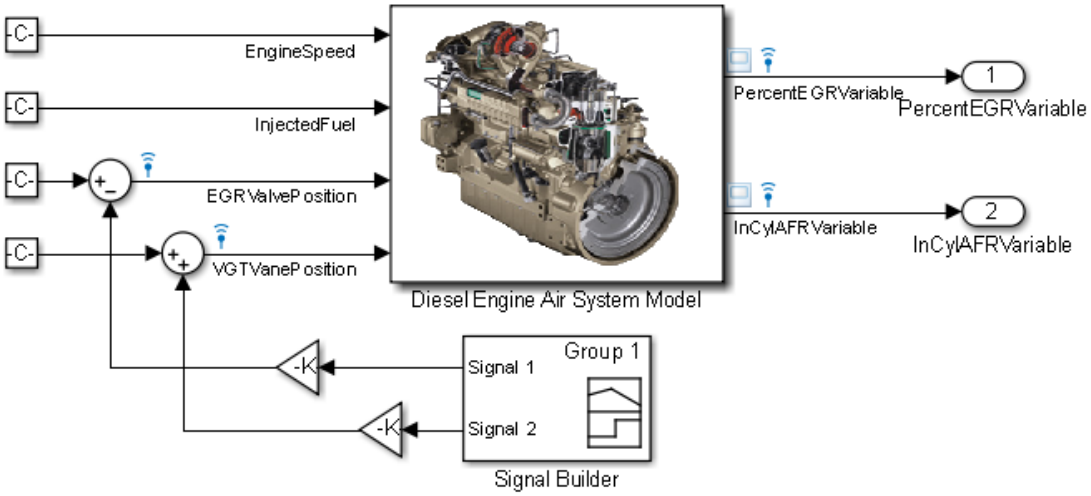

Figure B.1 Open Loop Engine Model Simulation Setup

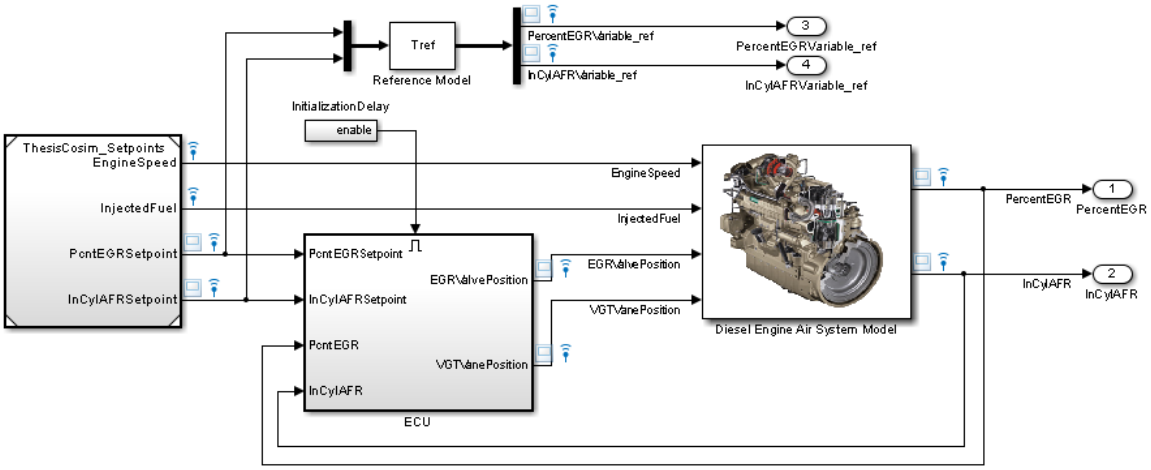

Figure B.2 Closed Loop Engine Model Simulation Setup

Implemented Controller Models

The Simulink models of the controller designs studied in this paper are shown below, including realistic and necessary discontinuities of actuator command saturation and integrator wind-up clamping. The implemented controllers for the H_∞ full order and reduced order have the same structure, the only aspect that is different is the size of the matrices A_s, B_s, C_s, F_s, H_s, and the values in the model matching gain W_i.

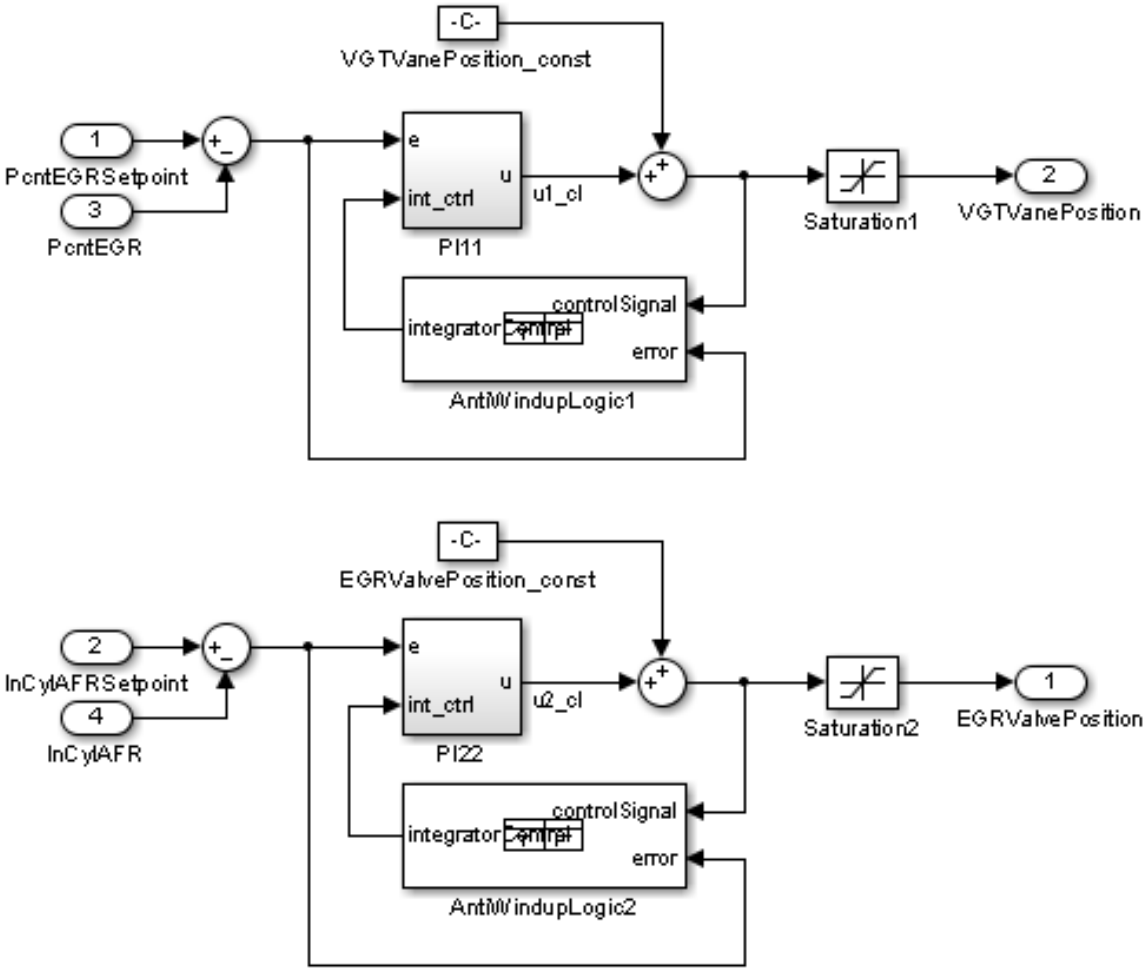

Figure B.3 Diagonal PI Controller Implementation

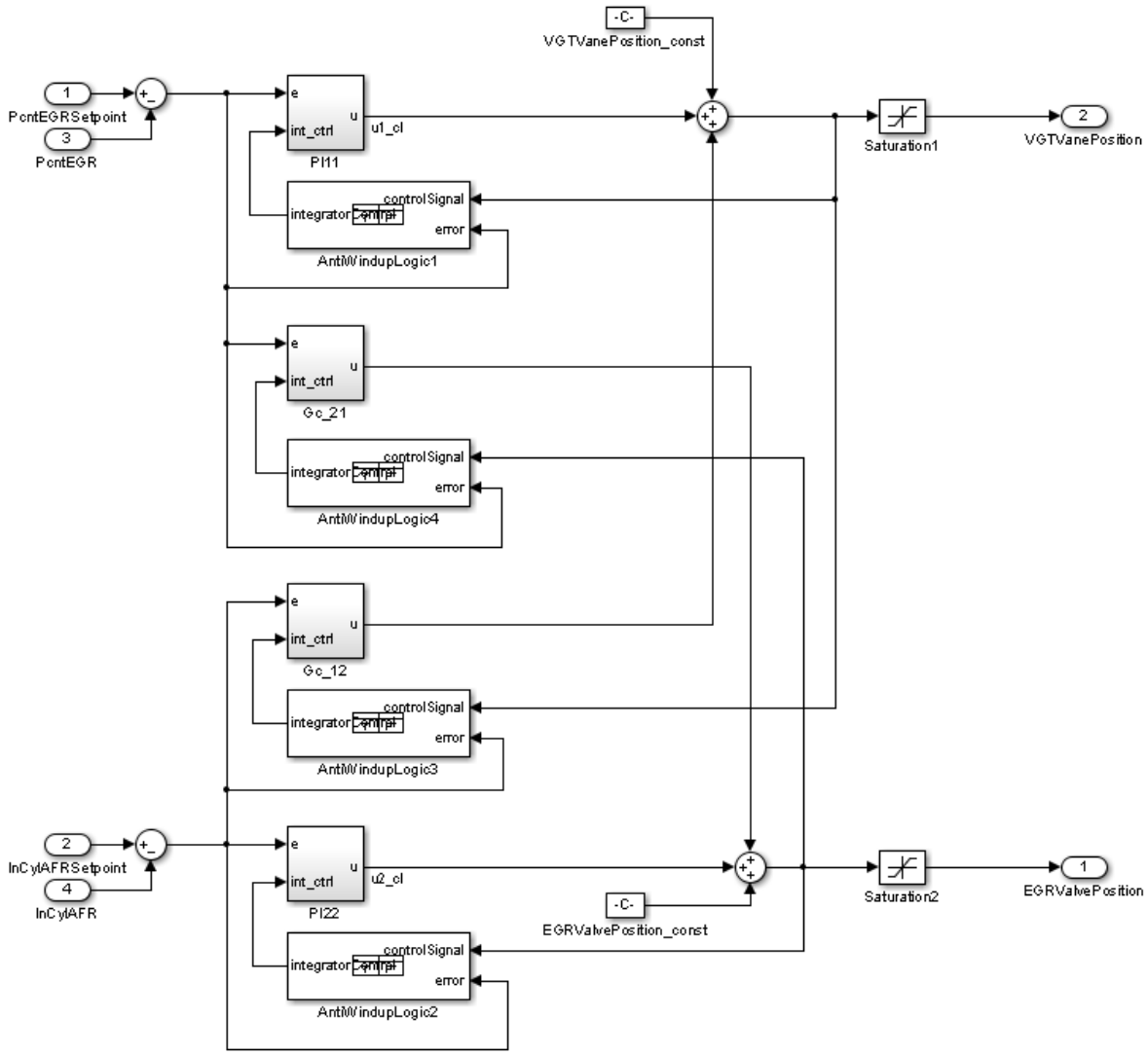

Figure B.4 Feed-forward Decoupling Controller Implementation

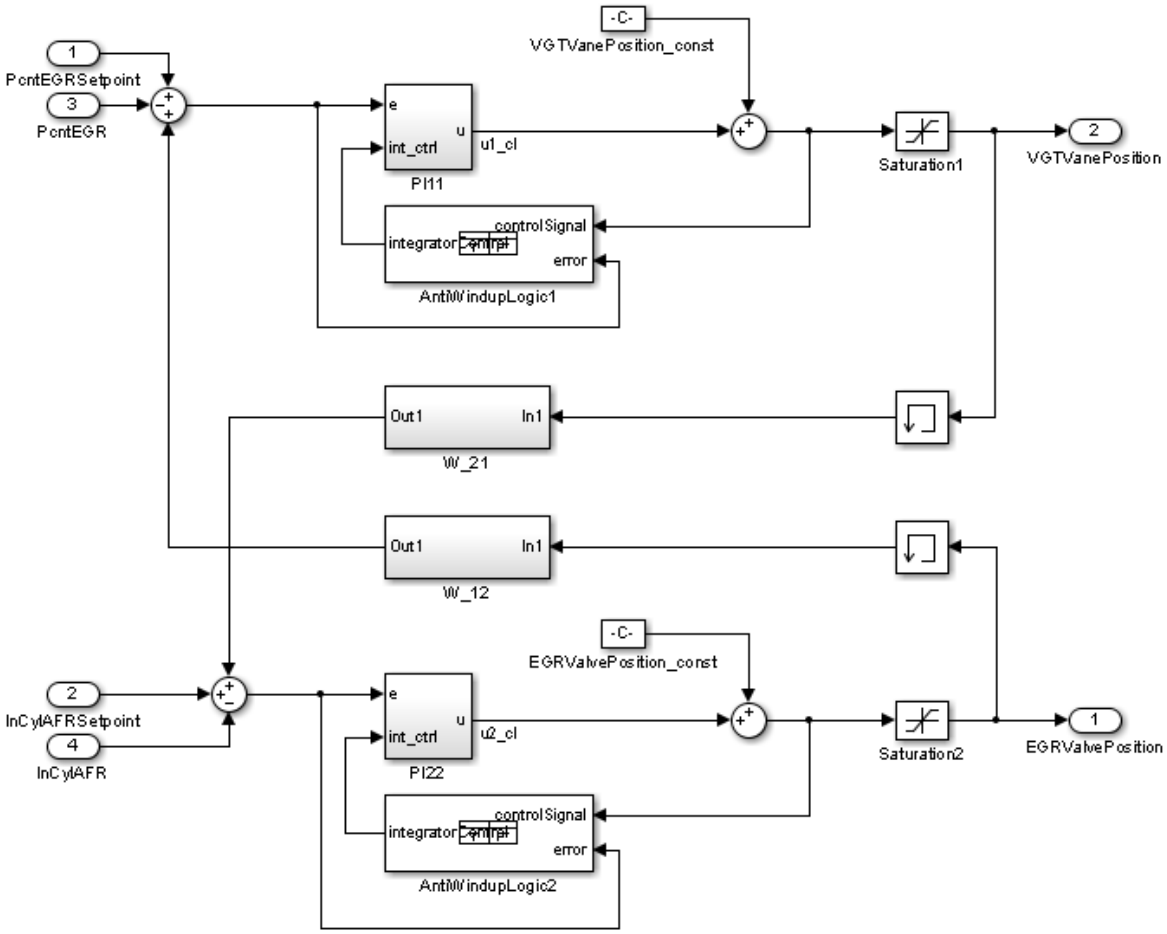

Figure B.5 Cross Interaction Filter Controller Implementation

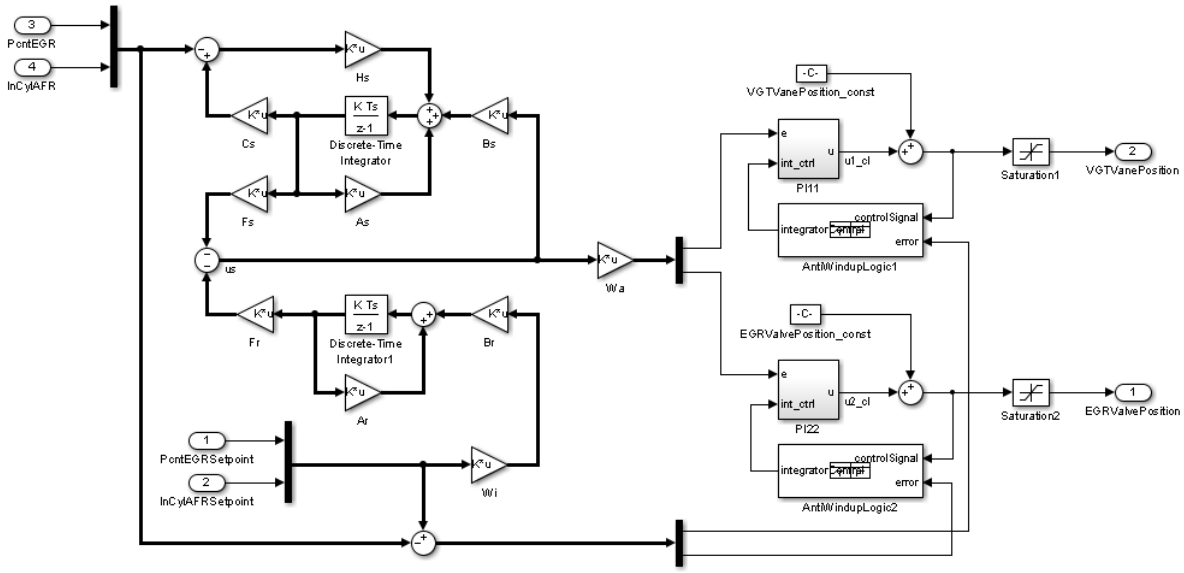

Figure B.6 H_∞ 2 DoF Loop Shaping Observer-Based Controller Implementation

APPENDIX C. MATHEMATICAL EQUATIONS AND DERIVATIONS

SIMC Model Reduction and Tuning Rules

The model reduction rules and tuning recommendations were first published in Skogestad (2003) and later reviewed in part in Skogestad and Postlethwaite (2005), and have been reproduced here for reference. Equation C.3 has been modified from the original publication based on errata notes available for download from the author's webpage.

Half Rule - Reduction of Real Poles and RHP Zeros

If the original model is of the form in Equation C.1, where the time constants and inverse response time constants (negative numerator time constants) have been ordered from largest to smallest, then a first order system with time delay can be obtained by applying half of the second most dominant lag to the dominant lag, and the remaining half plus all inverse response time constants to the delay (Equation C.2).

$$\frac{\prod_j \left(-T_{j0}^{inv} s + 1\right)}{\prod_i \left(\tau_{i0} s + 1\right)} e^{-\theta_0 s} \tag{C.1}$$

$$\tau_1 = \tau_{10} + \frac{\tau_{20}}{2}; \quad \theta = \theta_0 + \frac{\tau_{20}}{2} + \sum_{i \geq 3} \tau_{i0} + \prod_j -T_{j0}^{inv} + h/2; \tag{C.2}$$

Approximation of LHP Zeros

To reduce the LHP zeros of the initial system, Skogestad (2003) proposes to cancel the numerator term with a "neighboring" denominator term, and provides the following approximations (more discussions, derivations, and examples of the RHP reductions are provided in Skogestad (2003)). In Equation C.3, τ_c is the desired first order closed loop response time

constant.

$$\frac{T_0 s+1}{\tau_0 s+1} \approx \begin{cases} \frac{T_0}{\tau_0} & for\ T_0 \geq \tau_0 \geq \tau_c & (Rule\ T1) \\ \frac{T_0}{\tau_c} & for\ T_0 \geq \tau_c \geq \tau_0 & (Rule\ T1a) \\ 1 & for\ \tau_c \geq T_0 \geq \tau_0 & (Rule\ T1b) \\ \frac{T_0}{\tau_0} & for\ \tau_0 \geq T_0 \geq 5\tau_c & (Rule\ T2) \\ \frac{\tilde{\tau_0}/\tau_0}{(\tilde{\tau_0}-T_0)s+1} & for\ \tilde{\tau_0} \equiv min\,(\tau_0, 5\tau_c) \geq T_0 & (Rule\ T3) \end{cases} \tag{C.3}$$

It would appear to be a logical sequence (based on the approximation rules above and the examples from Skogestad (2003)) that the LHP zero reductions would be applied first until the system is in a form of C.1 before applying the Half Rule; this is the procedure followed for the engine air system study and seen in the Matlab code excerpts below.

Engine air system model reduction:

```
%%  Convert state-space model to transfer function array elements
g11_full = zpk(linear_air_sys(1,1));
g11_full.DisplayFormat = 'time constant';
g12_full = zpk(linear_air_sys(1,2));
g12_full.DisplayFormat = 'time constant';
g21_full = zpk(linear_air_sys(2,1));
g21_full.DisplayFormat = 'time constant';
g22_full = zpk(linear_air_sys(2,2));
g22_full.DisplayFormat = 'time constant';

%%  Reduce each element using SIMC rules
tau_c = 0.5;    % 2 rad/sec bandwidth recommendation
[g11,g11_rpt] = simc_reduce(g11_full,tau_c,true,'PI');
[g12,g12_rpt] = simc_reduce(g12_full,tau_c,true,'PI');
[g21,g21_rpt] = simc_reduce(g21_full,tau_c,true,'PI');
[g22,g22_rpt] = simc_reduce(g22_full,tau_c,true,'PI');
```

SIMC reduction code:

```
function [sys_r,rpt] = simc_reduce(sys,tau_c,tau_flag,ctrl_type)
```

```
2       %   default settings for arguments not provided
3       switch nargin
4           case 1  %   system only provided
5               tau_c = simc_getTotalDelay(sys);
6               tau_flag = false;
7               ctrl_type = 'PI';
8           case 2  %   system & desired closed loop time constant provided
9               tau_flag = false;
10              ctrl_type = 'PI';
11          case 3
12              ctrl_type = 'PI';
13      end
14
15      %   initialize report
16      rpt = {};
17
18      %   get system zero-pole-gain data, including initial time delay
19      [z_,p_,k_prime] = zpkdata(sys,'v');
20      theta0 = simc_getTotalDelay(sys);
21
22      %   convert values to time constant format - TODO: how to handle
23      %   complex conjugates for both numerator & denominator
24      re = real(z_);
25      im = imag(z_);
26      z = sign(re).*sqrt(re.^2 + im.^2);
27      tau_z = sort(-1./z,1,'descend');
28      re = real(p_);
29      im = imag(p_);
30      p = sign(re).*sqrt(re.^2 + im.^2);
31      tau_p = sort(-1./p,1,'descend');
32      k = k_prime * prod(-z)/prod(-p);
33
34      %   separate LHP (pos. tau) and RHP (neg. tau) zeros
35      tau_z_lhp = tau_z(tau_z > 0);
36      tau_z_rhp = tau_z(tau_z < 0);
```

```
37
38      %   TODO: setup logic to decide between using estimate of final theta
39      %   or tau_c in decision logic for numerator reduction
40      if tau_flag
41          theta = tau_c;
42      else
43          theta = theta0 + tau_p(2)/2 + sum(tau_p(3:end));
44      end
45      %   reduce out positive numerator time constants, starting with largest
46      while ~isempty(tau_z_lhp)
47          T0 = tau_z_lhp(1);
48          tau0a_idx = find(T0 < tau_p,1,'last');
49          tau0a = tau_p(tau0a_idx);
50          tau0b_idx = find(T0 > tau_p,1,'first');
51          tau0b = tau_p(tau0b_idx);
52          if ~isempty(tau0a) && ~isempty(tau0b)
53              %   if two pole time constants on either side of the zero time
54              %   constant are found, use the rule below to determine which
55              %   pole time constant to use
56              use_smallest = ((T0/tau0b < tau0a/T0) && (T0/tau0b < 1.6));
57          else
58              %   otherwise default the flag false
59              use_smallest = false;
60          end
61
62          if all(T0 > tau_p) || use_smallest
63              %   select next smallest denominator time constant
64              tau0_idx = tau0b_idx;
65              tau0 = tau0b;
66
67              if T0 >= tau0 && tau0 >= theta
68                  %   use rule T1
69                  k_eq = T0/tau0;
70                  rpt = [rpt; {'Numerator approx. rule T1 used'}]; %#ok<*AGROW>
71              elseif T0 >= theta && theta >= tau0
```

```
72                  %   use rule T1a
73                  k_eq = T0/theta;
74                  rpt = [rpt; {'Numerator approx. rule T1a used'}];
75              elseif theta >= T0 && T0 >= tau0
76                  %   use rule T1b
77                  k_eq = 1;
78                  rpt = [rpt; {'Numerator approx. rule T1b used'}];
79              else
80                  %   use gain of 1
81                  k_eq = 1;
82                  rpt = [rpt; {'Rules T1, T1a, T1b (next smallest den. tau) not applied'}];
83              end
84          else
85              %   otherwise use next largest denominator time constant
86              tau0_idx = tau0a_idx;
87              tau0 = tau0a;
88
89              if tau0 >= T0 && T0 >= 5*theta
90                  %   use rule T2
91                  k_eq = T0/tau0;
92                  rpt = [rpt; {'Numerator approx. rule T2 used'}];
93              elseif min(tau0,5*theta) >= T0
94                  %   use rule T3
95                  tau_tilde = min(tau0,5*theta);
96                  k_eq = (tau_tilde/tau0);
97
98                  %   create additional pole
99                  new_tau_p = tau_tilde - T0;
100                 %   append to end of list so as to not disturb indicies,
101                 %   re-sort list later
102                 tau_p = [tau_p; new_tau_p];
103                 rpt = [rpt; {'Numerator approx. rule T3 used'}];
104             else
105                 %   use gain of 1
106                 k_eq = 1;
```

```
107                 rpt = [rpt; {'Rules T2, T3 (next largest den. tau) not applied'}];
108             end
109         end
110
111         %   remove time constants from arrays; calculate new gain
112         tau_z_lhp(1) = [];
113         tau_p(tau0_idx) = [];
114         tau_p = sort(tau_p,1,'descend');
115         k = k * k_eq;
116     end
117
118     %   reduce to order specified, if possible
119     if (numel(tau_p) < 2) && strcmpi(ctrl_type,'PID')
120         %   force to be 1st order
121         ctrl_type = 'PI';
122         disp(['User-defined reduction order of 2 '...
123             '(for PID) overridden to 1 (for PI)']);
124         rpt = [rpt; {'User-defined reduction order of 2 '...
125             '(for PID) overridden to 1 (for PI)'}];
126     end
127
128     %   create reduced system using half rule to approximate RHP numerator
129     %   zeros and neglected time constants
130     s = tf('s');
131     theta = theta0;
132     if ~isempty(tau_z_rhp)
133         theta = theta + sum(-tau_z_rhp);
134     end
135     switch(ctrl_type)
136         case 'PID'  %   system for PID tuning
137             tau1 = tau_p(1);
138             switch numel(tau_p)
139                 case 2
140                     tau2 = tau_p(2);
141                 case 3
```

```
142                     tau2 = tau_p(2) + tau_p(3)/2;
143                     theta = theta + tau_p(3)/2;
144                 otherwise
145                     tau2 = tau_p(2) + tau_p(3)/2;
146                     theta = theta + tau_p(3)/2 + sum(tau_p(4:end));
147             end
148
149             sys_r = (k/((tau1*s + 1)*(tau2*s + 1))) * exp(-s*theta);
150         otherwise  %   system for PI tuning
151             switch numel(tau_p)
152                 case 1
153                     tau1 = tau_p(1);
154                 case 2
155                     tau1 = tau_p(1) + tau_p(2)/2;
156                     theta = theta + tau_p(2)/2;
157                 otherwise
158                     tau1 = tau_p(1) + tau_p(2)/2;
159                     theta = theta + tau_p(2)/2 + sum(tau_p(3:end));
160             end
161
162             sys_r = (k/(tau1*s + 1)) * exp(-s*theta);
163     end
164
165     %   copy input & output names from original system to reduced system
166     sys_r.OutputName = sys.OutputName;
167     sys_r.InputName = sys.InputName;
168 end
```

One last item to mention about the model reduction method applied here is that - as seen in Figure 2.2 - there are complex poles, and for the individual SISO transfer functions, complex zeros as well. Since the method briefly outlined above only accounts for reducing real poles and zeros, an approximation needed to be made. It was decided that the complex poles and zeros would be replaced with two real roots at the same frequency. While this does eliminate the oscillations and frequency peaks in the system responses, it does keep the same steady-state

level and the same high frequency roll-off. This approximation is validated by closeness of the open loop step responses in Figure 2.1, and by the success of the individual PI tuning results.

BIBLIOGRAPHY

Chauvin, J., Corde, G., and Petit, N. (2007). Transient control of a diesel engine airpath. In *American Control Conference, 2007. ACC'07*, pages 4394–4400. IEEE.

Friedrich, I., Liu, C.-S., and Oehlerking, D. (2009). Coordinated egr-rate model-based controls of turbocharged diesel engines via an intake throttle and an egr valve. In *Vehicle Power and Propulsion Conference, 2009. VPPC'09. IEEE*, pages 340–347. IEEE.

Herceg, M., Raff, T., Findeisen, R., and Allgowe, F. (2006). Nonlinear model predictive control of a turbocharged diesel engine. In *Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control, 2006 IEEE*, pages 2766–2771. IEEE.

Heywood, J. B. (1988). *Internal Combustion Engine Fundamentals*. McGraw-Hill, Inc., New York.

Hoyle, D., Hyde, R., and Limebeer, D. (1991). An h_∞ approach to two degree of freedom design. In *Decision and Control, 1991., Proceedings of the 30th IEEE Conference on*, pages 1581–1585. IEEE.

Jung, J. and Nam, K. (1999). A dynamic decoupling control scheme for high-speed operation of induction motors. *Industrial Electronics, IEEE Transactions on*, 46(1):100–110.

Limebeer, D., Kasenally, E., and Perkins, J. (1993). On the design of robust two degree of freedom controllers. *Automatica*, 29(1):157–168.

McFarlane, D. and Glover, K. (1992). A loop-shaping design procedure using h_∞ synthesis. *Automatic Control, IEEE Transactions on*, 37(6):759–769.

Omran, R., Younes, R., and Champoussin, J.-C. (2009). Optimal control of a variable geometry turbocharged diesel engine using neural networks: Applications on the etc test cycle. *Control Systems Technology, IEEE Transactions on*, 17(2):380–393.

Pfeiffer, J. M. and Hedrick, J. K. (1999). Nonlinear algorithms for simultaneous speed tracking and air-fuel ratio control in an automobile engine. *SAE transactions*, 108(3):783–788.

Shirakawa, T., Itoyama, H., and Miwa, H. (2001). Study of strategy for model-based cooperative control of egr and vgt in a diesel engine. *JSAE Review*, 22(1):3–8.

Skogestad, S. (2003). Simple analytic rules for model reduction and pid controller tuning. *Journal of process control*, 13(4):291–309.

Skogestad, S. (2006). Tuning for smooth pid control with acceptable disturbance rejection. *Industrial & engineering chemistry research*, 45(23):7817–7822.

Skogestad, S. and Postlethwaite, I. (2005). *Multivariable Feedback Control: Analysis and Design*. John Wiley and Sons, Ltd., Chichester, 2nd edition.

Tham, M. (1999). Multivariable control: An introduction to decoupling control. *Department of Chemical and Process Engineering, University of Newcastle upon Tyne, Newcastle upon Tyne*. Published originally as Chapter 8 in "Industrial Digital Control Systems", Eds. K. Warwick and D. Rees, IEE Control Engineering Series 37, Peter Peregrinus, 1988.

Wahlström, J. and Eriksson, L. (2009). *Non-linear Compensator for handling non-linear Effects in EGR VGT Diesel Engines*. PhD thesis, Linköpings universitet, SE-581 83 Linköping, Sweden.

Wahlström, J. and Eriksson, L. (2010). Nonlinear input transformation for egr and vgt control in diesel engines. *SAE International Journal of Engines*, 3(2):288–305.

Wahlström, J., Eriksson, L., and Nielsen, L. (2009). *System analysis of a Diesel Engine with VGT and EGR*. PhD thesis, Linköpings universitet, SE-581 83 Linköping, Sweden.

Wahlström, J., Eriksson, L., and Nielsen, L. (2010). Egr-vgt control and tuning for pumping work minimization and emission control. *Control Systems Technology, IEEE Transactions on*, 18(4):993–1003.

Walker, D. (1996). On the structure of a two-degree-of-freedom h_∞ loop shaping controller. *International Journal of Control*, 63(6):1105–1127.