DESIGN OF CONTROLLERS FOR A MULTIPLE INPUT MULTIPLE OUTPUT SYSTEM

A Dissertation

by

AMANDA LYNNE HARRIS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2012

Major Subject: Electrical Engineering

Design of Controllers for a Multiple Input

Multiple Output System

DESIGN OF CONTROLLERS FOR A

MULTIPLE INPUT MULTIPLE OUTPUT SYSTEM

A Dissertation

by

AMANDA LYNNE HARRIS

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Approved by:

Chair of Committee,	Shankar Bhattacharyya
Committee Members,	Aniruddha Datta
	John Valasek
	Narasimha Reddy
Head of Department,	Costas Georghiades

May 2012

Major Subject: Electrical Engineering

ABSTRACT

Design of Controllers for a Multiple Input

Multiple Output System. (May 2012)

Amanda Lynne Harris, B.S., The University of Kansas;

MSc, The University of Edinburgh

Chair of Advisory Committee: Dr. Shankar Bhattacharyya

A method of controller design for multiple input multiple output (MIMO) system is needed that will not give the high order controllers of modern control theory but will be more systematic than the “ad hoc” method. The objective of this method of design for multiple input multiple output systems is to find a controller of fixed order with performance specifications taken into consideration. An inner approximation of the stabilizing set is found through the algorithm discussed in Keel and Bhattacharyya’s "Fixed order multivariable controller synthesis: A new algorithm." The set satisfying the performance is then approximated through one of two algorithms; a hybrid of two optimization algorithms or the grid algorithm found in Lampton’s "Reinforcement Learning of a Morphing Airfoil-Policy and Discrete Learning Analysis." The method is then applied to five models of four aircraft; Commander 700, X-29, X-38, and F-5A using controllers of first and second orders. The examples show that when the method finds a stabilizing set, it will also find the performance set if it exists. However, it is possible for the method to not find a stabilizing for a stabilizable system.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

INTRODUCTION

When adding controllers to a multiple input multiple output (MIMO) system, options are limited. Modern control theory gives an elegant controller, but often has too high of an order for practical use [1]. The modern control methods that use a dynamic compensator will always find a stabilizing controller but it will have the same order as the plant [2]. Most MIMO systems are of high order; a two input two output system with each transfer function having an order of just three will have a system order of twelve. This means that any stabilizing controller found using these methods will also be twelfth order. Alternatively, the "ad hoc" methods use a fixed order controller with a given number of variables. Instead of determining the values to set the variables through a controller design method, the controller is inserted into the system and the variables are changed until the system is stable and exhibits the desired performance. There are some tuning techniques so the user does not have to use guess and check. They involve putting a test input in the system and the response gives an idea where to start the tuning. These classical PID tuning methods are: the Ziegler-Nichols open-loop or recess reaction method, the Ziegler-Nichols closed-loop method, the Cohen-Coon Method, the Internal Model Control method, and the Auto Tune Variation [3]. As these methods involve inputting a signal to the system, the system can go unstable in the process. While it is advantageous that these methods don't require knowledge of the system model, they do

---

This dissertation follows the style of *IEEE Control Systems*.

require some insight about the system as they each work for specific types of systems [3]. Both the modern control theory and "ad hoc" methods have their uses, but each has issues. If the modern control theory result has a reasonable order or the order of the controller is not a limiting factor, that controller can be used. However, if it is of too high an order for use, the only other option is to try a different method [2]. The "ad hoc" method is useful because if the fixed order controller is of an order that can stabilize the system, eventually an acceptable response can be found. Unfortunately, it takes time and effort to run through a large set of values for each variable and if the tuning methods are used, the experimentation can still take time [3]. To resolve the issue, the fixed order controller should be used, and instead of using the "ad hoc" method to find the control variables, an alternative design method is needed.

In searching for recent work on fixed order controllers for MIMO systems, four methods were determined to be relevant. All four methods focused on H∞ controller design modified for a fixed order system. The simplest method to use was discussed in [4]. The HIFOO package for Matlab was the basis of the method as it is readily available and relatively easy to use. By using state space representation and addition constraints on the system, HIFOO was able to find controllers of a lower order than the plant while still having the advantages of H∞. However, the program gives an optimized result and therefore the designer is given only one controller instead of a range to choose from. In [5], a design method giving H∞ performance while achieving D-stability and reducing the coupling of the channels in the MIMO system is presented. LMIs are then used to optimize the controller. [6] proposes a similar method using matrix inequalities with a

free parameterization matrix which can be solved using a convergent iterative algorithm. These three methods require the plant model to be known. However, the design method in [7] can be used with the frequency response also known through data. The method is based on an approximation of the Generalized Nyquist Stablility criterion and results in convex constraints on the control variables. However, a known stabilizing controller is needed if the plant is unstable. Unlike the method in [4], the design methods developed in [5,6,7] are mathematically intensive.

The objective of the proposed research is to develop a method of design for multiple input multiple output systems. The controller is of fixed order, and performance specifications should be taken into consideration. The starting point for such a design method would be to find the range of each control variable that would stabilize the system. Doing so would give the option of trying many different order controllers to determine the more practical option for a given system. In addition, having the range that stabilizes the system will allow the variables to be chosen so that they are not close to the edge of the range. Choosing the variables in such a way will reduce the chance of small errors in the model causing instability. However, the algorithm used needs to find an inner approximation for the stable range to ensure all points in the range are in fact stable. The next step is to determine any performance specifications and the set of controller values within the stabilizing range which satisfy the specifications given.

The simplest, longest, and most computationally extensive method for finding both the stability range and the performance set would be to check every point within the possible range of controller values. When working with only two controller variables,

this is possible though still lengthy. Once more variables are added, this method becomes too extensive to be practical. Consequently, it is critical to find algorithms that will find the stable set and the performance set without checking every point in the range of controller values.

METHODOLOGY

As stated before, the first step to develop a method for design is to find an algorithm that gives the stable range of a MIMO system. The robust positivity algorithm from [8] is essentially an extension of Kharitonov's Theorem. It only requires the characteristic equation of a system to test for stability over a range of variables. However, this requirement means that any system for which the characteristic equation cannot be found, the design method will not work. The following explains this algorithm and how it will be used in the design method.

$$\mathcal{P} := \{P(s,\mathbf{x}) : \mathbf{x} \in B \tag{1}$$

is a polynomial family where B is a box in the first orthant.

$$P(s) = a_0(\mathbf{x}) + a_1(\mathbf{x})s + a_2(\mathbf{x})s^2 + a_3(\mathbf{x})s^3 + a_4(\mathbf{x})s^4 + \cdots \tag{2}$$

is a typical element of the family where ai(x)are polynomial functions of x for i=0,1,…,n ai(x)= ai+(x)- ai-(x). Assuming that an(x)≠0 for all $\mathbf{x} \in B$. In this application, P(s) is the characteristic polynomial of the system and the ai(x) are the k values ranging from 0 to positive infinity. For ease of calculations, the state space representation is used to find the characteristic equation as using the transfer function would require decomposition. Therefore the characteristic equation is found using

$$\text{P(s)=det(sI-A).} \tag{3}$$

Where

$$A = \begin{bmatrix} A_P & B_P C_C \\ 0 & A_C \end{bmatrix} - \begin{bmatrix} B_P D_C \\ B_C \end{bmatrix} ones[I + D_P D_C ones]^{-1} [C_P \quad D_P C_C] \tag{4}$$

and

$$A_C = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, B_C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, C_C = \begin{pmatrix} k_{i_1} & 0 \\ 0 & k_{i_2} \end{pmatrix}, D_C = \begin{pmatrix} k_{p_1} & 0 \\ 0 & k_{p_2} \end{pmatrix} \tag{5}$$

A_p, B_p, C_p, D_p are the minimal realization of the plant.

Define

$$P^+_{even}(s^2, \mathbf{x}) := a^+_0(\mathbf{x}) - a^-_2(\mathbf{x})s^2 + a^+_4(\mathbf{x})s^4 - \cdots \tag{6}$$

$$P^-_{even}(s^2, \mathbf{x}) := a^-_0(\mathbf{x}) - a^+_2(\mathbf{x})s^2 + a^-_4(\mathbf{x})s^4 - \cdots \tag{7}$$

$$sP^+_{odd}(s^2, \mathbf{x}) := s[a^+_1(\mathbf{x}) - a^-_3(\mathbf{x})s^2 + a^+_5(\mathbf{x})s^4 - \cdots] \tag{8}$$

$$sP^-_{odd}(s^2, \mathbf{x}) := s[a^-_1(\mathbf{x}) - a^+_3(\mathbf{x})s^2 + a^-_5(\mathbf{x})s^4 - \cdots] \tag{9}$$

and

$$P_{even}(s^2, \mathbf{x}) := P^+_{even}(s^2, \mathbf{x}) - P^-_{even}(s^2, \mathbf{x}) \tag{10}$$

$$sP_{odd}(s^2, \mathbf{x}) := sP^+_{odd}(s^2, \mathbf{x}) - sP^-_{odd}(s^2, \mathbf{x}) \tag{11}$$

Lastly, let

$$\overline{P}_{even}(s^2) := P^+_{even}(s^2, \mathbf{x}^+) - P^-_{even}(s^2, \mathbf{x}^-) \tag{12}$$

$$\underline{P}_{even}(s^2) := P^+_{even}(s^2, \mathbf{x}^-) - P^-_{even}(s^2, \mathbf{x}^+) \tag{13}$$

$$s\overline{P}_{odd}(s^2) := sP^+_{odd}(s^2, \mathbf{x}) - sP^-_{odd}(s^2, \mathbf{x}) \tag{14}$$

$$s\underline{P}_{odd}(s^2) := sP^+_{odd}(s^2, \mathbf{x}) - sP^-_{odd}(s^2, \mathbf{x}) \tag{15}$$

The family P is robustly Hurwitz stable if the following four fixed polynomials are Hurwitz stable.

$$P_1(s) := \underline{P}_{even}(s^2) - s\underline{P}_{odd}(s^2) \tag{16}$$

$$P_2(s) := \underline{P}_{even}(s^2) - s\overline{P}_{odd}(s^2) \tag{17}$$

$$P_3(s) := \overline{P}_{even}(s^2) - s\overline{P}_{odd}(s^2) \tag{18}$$

$$P_4(s) := \overline{P}_{even}(s^2) - s\underline{P}_{odd}(s^2) \tag{19}$$

Using the minimum and maximum of the ranges, find P1, P2, P3, and P4. If all four equations are stable, the entire range is stable but if any are not stable, nothing is known about the range. Otherwise, bisect the range along each variable and find P1, P2, P3, and P4 for each new block. Those that are stable are left alone and cut the remaining blocks. Repeat the cutting until either a minimum block size is reached or until all blocks are stable. The smaller the block size is allowed to be, the closer the stable range will be to the actual set of stabilizing values. However, the smaller the block size the longer the algorithm will take and the more memory will be used. Therefore it is a tradeoff on time/memory and precision. Before moving to the next step in the design method, it should be noted that this algorithm can be connected to a well-known theorem in control theory. This proof is taken straight from [8]. Consider the interval family of polynomials

$$P(s) = x_0 + x_1 s + x_2 s^2 + x_3 s^3 + \cdots \tag{20}$$

where

$$0 < x_i^- < x_i < x_i^+ \tag{21}$$

Note that, using the previous notations,

$$a_i = x_i = a_i^+, a_i^- = 0 \tag{22}$$

$$P_{even}^{+}(s^2) = x_0 + x_4 s^4 + x_8 s^8 + \cdots \tag{23}$$

$$P_{even}^{-}(s^2) = -x_2 s^2 - x_6 s^6 - x_{10} s^{10} \cdots \tag{24}$$

$$sP_{odd}^{+}(s^2) = x_1 s + x_5 s^5 + x_9 s^9 + \cdots \tag{25}$$

$$sP_{odd}^{-}(s^2) = -x_3 s^3 - x_7 s^7 - x_{11} s^{11} \cdots \tag{26}$$

and

$$\begin{aligned} \bar{P}_{even}(s^2) &= P_{even}^{+}(s^2, \boldsymbol{x}^+) - P_{even}^{-}(s^2, \boldsymbol{x}^-) \\ &= x_0^+ + x_2^- s^2 + x_4^+ s^4 + x_6^- s^6 + x_8^+ s^8 + \cdots \end{aligned} \tag{27}$$

$$\begin{aligned} \underline{P}_{even}(s^2) &= P_{even}^{+}(s^2, \boldsymbol{x}^-) - P_{even}^{-}(s^2, \boldsymbol{x}^+) \\ &= x_0^- + x_2^+ s^2 + x_4^- s^4 + x_6^+ s^6 + x_8^- s^8 + \cdots \end{aligned} \tag{28}$$

$$\begin{aligned} s\bar{P}_{odd}(s^2) &= P_{odd}^{+}(s^2, \boldsymbol{x}^+) - P_{odd}^{-}(s^2, \boldsymbol{x}^-) \\ &= x_1^+ s + x_3^- s^3 + x_5^+ s^5 + x_7^- s^7 + x_9^+ s^9 + \cdots \end{aligned} \tag{29}$$

$$\begin{aligned} s\underline{P}_{odd}(s^2) &= P_{odd}^{+}(s^2, \boldsymbol{x}^-) - P_{odd}^{-}(s^2, \boldsymbol{x}^+) \\ &= x_1^- s + x_3^+ s^3 + x_5^- s^5 + x_7^+ s^7 + x_9^- s^9 + \cdots \end{aligned} \tag{30}$$

Therefore,

$$\begin{aligned} P^1(s^2) &= \underline{P}_{even}(s^2) + s\underline{P}_{odd}(s^2) \\ &= x_0^- + x_1^- s + x_2^+ s^2 + x_3^+ s^3 + x_4^- s^4 + \cdots \end{aligned} \tag{31}$$

$$P^2(s^2) = \underline{P}_{even}(s^2) + s\bar{P}_{odd}(s^2)$$

$$= x_0^- + x_1^+ s + x_2^+ s^2 + x_3^- s^3 + x_4^- s^4 + \cdots \tag{32}$$

$$P^3(s^2) = \overline{P}_{even}(s^2) + s\overline{P}_{odd}(s^2)$$

$$= x_0^+ + x_1^+ s + x_2^- s^2 + x_3^- s^3 + x_4^+ s^4 + \cdots \tag{33}$$

$$P^4(s^2) = \overline{P}_{even}(s^2) + s\underline{P}_{odd}(s^2)$$

$$= x_0^+ + x_1^- s + x_2^- s^2 + x_3^+ s^3 + x_4^+ s^4 + \cdots \tag{34}$$

Therefore, Kharitonov's theorem has been recovered from the algorithm. By this proof tying the positivity algorithm back to Kharitonov's theorem, the confidence in the positivity algorithm is increased which is important as the work on it is only a few years old.

The second step of the design technique is to find the set of points that satisfy given performance specifications. There are two methods included: a method using two optimization algorithms and a method using an adaptive grid. For both methods, define the limits for the performance parameters to be included in the optimization and decide which if any control variables will be set. These are determined by the designer. The application for which the system is to be used will aid in determining the performance specifications and the number of control variables allowed to vary should be either two or three as the sets are best viewed graphically. Those variables set will be chosen to fall within the stable range found in the first step.

For the optimization method, obtain a random set of points within the stable range. Deciding how to choose the random points is an important step. If taken randomly

throughout the stable range, the full set of performance satisfying points will not be found as all the random points will move towards one optimum point. However, if each block within the stable range had a few of the random points and those points were optimized over that box only, the probability of obtaining the full performance set is much higher. Therefore, there will be at least three points in the smallest section of the stable range and as the section size increases so does the number of optimization points. Then, the performance of each point is determined for the parameters set. Separate the points into three sections: good, swarm, and genetics. The good section includes all points that satisfy the performance specifications. The swarm section is all the points that will use the swarm optimization technique [9]. The points are either within half the minimum or double the maximum for each parameter or for each parameter outside that range there is a parameter that is satisfied.

The genetics section covers all other points in the random set and will use genetics optimization [10]. Each section is optimized separately. For those points in the good section, they are left alone. Those points in the swarm sections will be used by a type of swarm optimization to move these points towards the point in the good section with the best performance. However, if the new point is not in the stable range, the old point will be moved slightly instead of using the value found through swarm theory. The points in the genetic section have the worst performance and therefore genetic optimization is used to create new points to replace these points. If the new points are not in the stable range, the old point is modified slightly. After ten iterations, check

every point within a sphere of radius 10 around each point in the good section to obtain a more complete set.

Next is the adaptive grid method [11]. A grid at a specific distance, i.e. 8, is created across the variable range. If a point is not in the stable range, throw it away. Then find the performance at each point. If it does satisfy the performance specifications, set the point to 1, if not -1. If the surrounding orthogonal points of a 1 valued point are also 1, make the center point a 2. For all points still one, between the orthogonal surrounding points that are not 1 or 2, create a grid of 4. Once the small grids of distance 4 are created, repeat the process to find small areas to grid at distance 2 and then 1. The resulting sets are controller values that are stable and fulfill the performance specifications.

EXAMPLES

For ease of demonstration, the two input two output system shown in figure 1 is used for the following examples. PI controllers are at each input giving four control variables and G(s) is a given model of the plant

$$G(s)=\begin{bmatrix} G_{11}(s) & G_{12}(s) \\ G_{21}(s) & G_{22}(s) \end{bmatrix} \tag{35}$$

Also for ease of demonstration, the range for each control value is given as 0-100. However, if a system is unstable at the origin, the range is changed to 1-100.

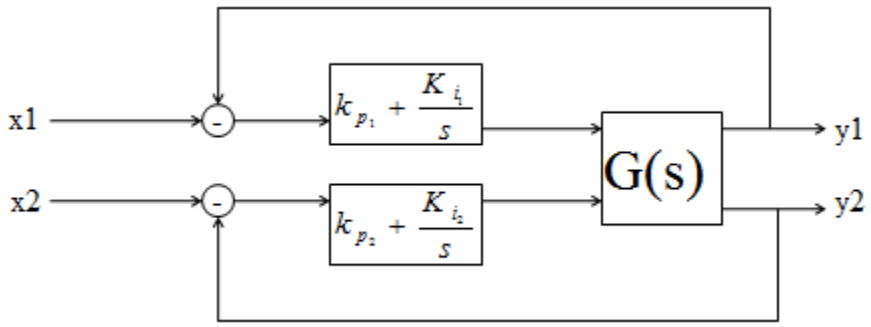

Figure 1 Two input two output system used in all examples

The first example is a single input single output (SISO) plant

$$G(s) = \frac{s+1}{s^2-4} \tag{36}$$

When in state space representation, the plant is

$$A_p = \begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix} B_p = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} C_p = \begin{bmatrix} 1 & .5 \\ 0 & 0 \end{bmatrix} D_p = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{37}$$

This results in an An matrix of

$$A_n = \begin{bmatrix} -k_{p_1} & 2 - \frac{1}{2}k_{p_1} & k_{i_1} \\ 2 & 0 & 0 \\ -1 & -\frac{1}{2} & 0 \end{bmatrix} \tag{38}$$

giving a characteristic equation of

$$P(s) = s^3 + k_{p_1}s^2 + s\left(k_{i_1} + k_{p_1} - 4\right) + k_{i_1} \tag{39}$$

Therefore

$$P^{+}_{even}(s^2, \mathbf{x}) = 2\mathrm{k}_{\mathrm{i}_1} \tag{40}$$

$$P^{-}_{even}(s^2, \mathbf{x}) = (-23 - \mathrm{k}_{\mathrm{p}_1})s^2 \tag{41}$$

$$sP^{+}_{odd}(s^2, \mathbf{x}) = s[2\mathrm{k}_{\mathrm{p}_1} + \mathrm{k}_{\mathrm{i}_1} + 27] \tag{42}$$

$$sP^{-}_{odd}(s^2, \mathbf{x}) = -s^3 \tag{43}$$

$$\overline{P}_{even}(s^2) = 23\mathrm{k}^{+}_{\mathrm{p}_1}s^2 + 2\mathrm{k}^{-}_{\mathrm{i}_1} \tag{44}$$

$$\underline{P}_{even}(s^2) = 23\mathrm{k}^{-}_{\mathrm{i}_1}\mathrm{s}^2 + 2\mathrm{k}^{+}_{\mathrm{i}_1} \tag{45}$$

$$s\overline{P}_{odd}(s^2) = s[(2\mathrm{k}^{-}_{\mathrm{p}_1} + \mathrm{k}^{-}_{\mathrm{i}_1} + 27) + s^2] \tag{46}$$

$$s\underline{P}_{odd}(s^2) = s[(2\mathrm{k}^{+}_{\mathrm{p}_1} + \mathrm{k}^{+}_{\mathrm{i}_1} + 27) + s^2] \tag{47}$$

Finally

$$P_1(s) = s^3 + k_{p_1}{}^{+}s^2 + s\left(k_{p_1}{}^{-} + k_{i_1}{}^{-} - 4\right) + k_{i_1}{}^{-} \tag{48}$$

$$P_2(s) = s^3 + k_{p_1}{}^{+}s^2 + s\left(k_{p_1}{}^{+} + k_{i_1}{}^{+} - 4\right) + k_{i_1}{}^{-} \tag{49}$$

$$P_3(s) = s^3 + k_{p_1}{}^- s^2 + s\left(k_{p_1}{}^+ + k_{i_1}{}^+ - 4\right) + k_{i_1}{}^+ \quad (50)$$

$$P_4(s) = s^3 + k_{p_1}{}^- s^2 + s\left(k_{p_1}{}^- + k_{i_1}{}^- - 4\right) + k_{i_1}{}^+ \quad (51)$$

Figure 2 shows the stable range of the controllers when using the Routh method at each point. This method is inefficient because it checks each point in the range. It will only be used to show the corresponding results from the positivity method.

Matlab was used for all the examples mostly because of previous knowledge of the program, but also for the built-in functions available for most of the performance parameters. A GUI was designed to make working with each example as quick and easy as possible. Half of this interface is shown in figure 3 and the other half in figure 4. As this interface was designed for the PI controller used in this section, the only inputs to find the stable range were the numerator and denominator of the plant and the desired threshold for the block size. The inputs for the graphs were merely values for one or two control variables and the performance inputs were the desired performance specifications and values for whichever control variables were to be set. From figure 3, the threshold is set at ten making the smallest bisection is just larger than six. Therefore, as the only sections of figure 3 that are not in the stable range contain the area not stable in figure 2, the method is shown to be accurate within the threshold. For the remainder of the figures, the blue boxes show the stability range and the red points are those satisfying the performance specifications. Figure 4 shows the points within the stable range that have an overshoot of 10 percent or less using the optimization method. Figures 5-8 show the grid method at each iteration of the grid: figure 5 shows the grid at

distance of 8, figure 6 shows the additional points at a distance of 4, figure 7 has a distance of 2, and figure 8 has a distance of 1.

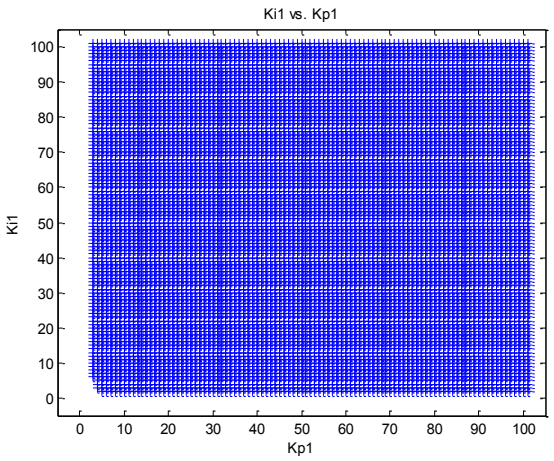

Figure 2 Stable range using Routh method for the SISO example.

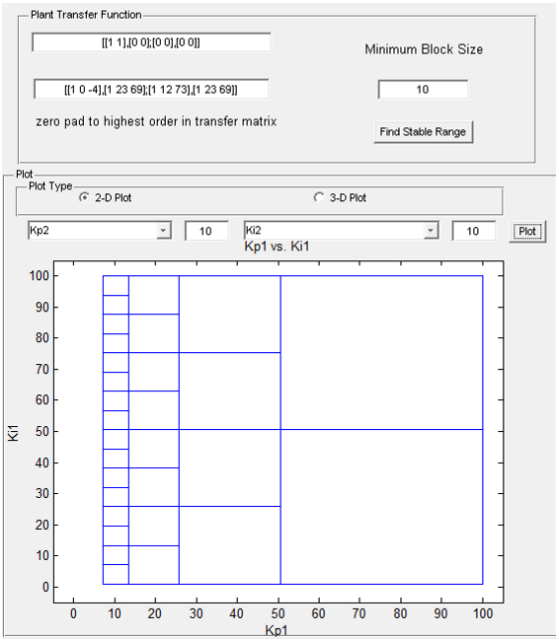

Figure 3 Stable range using the design method for the SISO example.

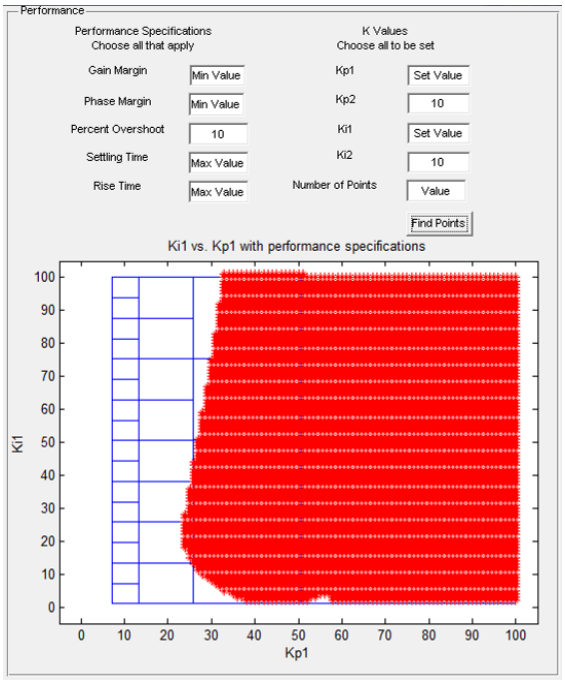

Figure 4 Performance satisfying range using the optimization method for the SISO example.

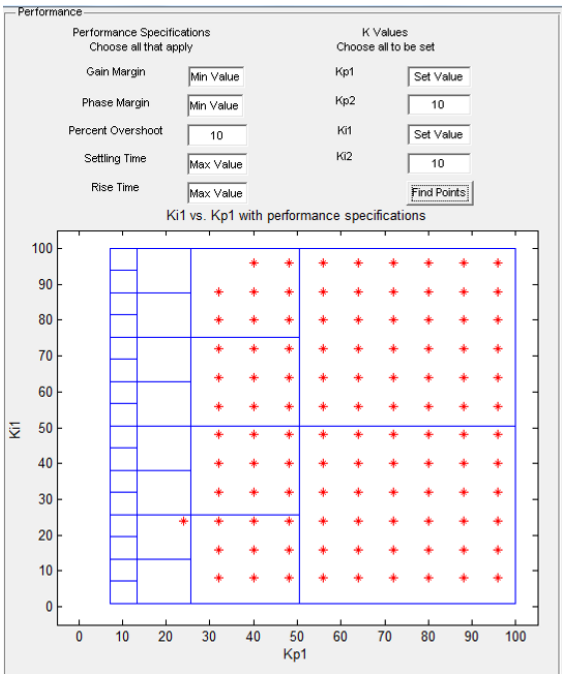

Figure 5 Performance satisfying range using the grid method at first iteration for the SISO example.

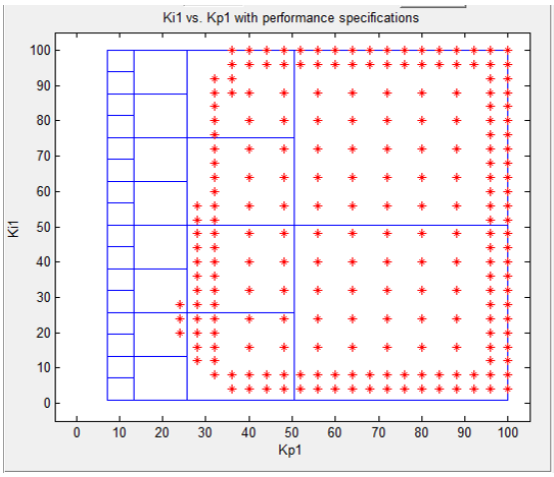

Figure 6 Performance satisfying range using the adaptive grid method at second iteration for the SISO example.

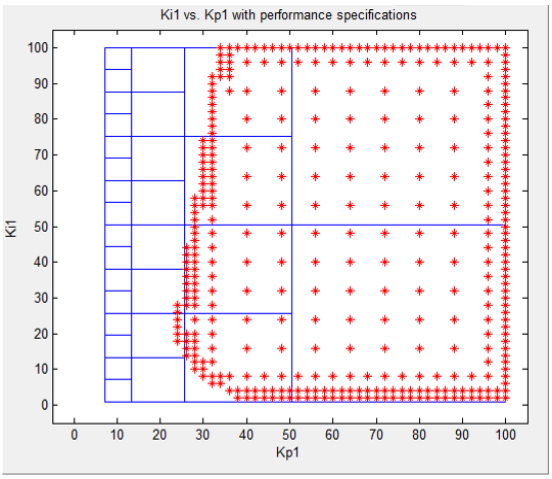

Figure 7 Performance satisfying range using the adaptive grid method at third iteration for the SISO example.

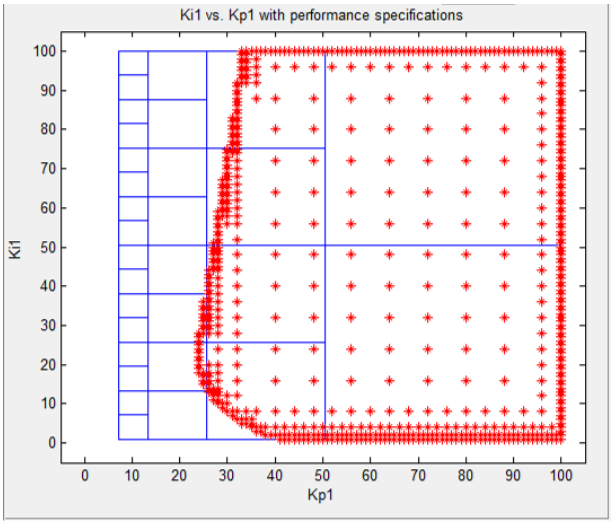

Figure 8 Performance satisfying range using the adaptive grid method at fourth iteration for the SISO example.

The second example is the two input two output system

$$G(s) = \begin{bmatrix} \frac{s+16}{s^2+12s+73} & \frac{s-2}{s^2+23s+69} \\ \frac{s+24}{s^2+12s+73} & \frac{s+3}{s^2+23s+69} \end{bmatrix} \tag{52}$$

which gives a state space representation of

$$A_p = \begin{bmatrix} -12 & -9.125 & 0 & 0 \\ 8 & 0 & 0 & 0 \\ 0 & 0 & -23 & -17.25 \\ 0 & 0 & 4 & 0 \end{bmatrix}, B_p = \begin{bmatrix} 2 & 0 \\ 0 & 0 \\ 0 & 2 \\ 0 & 0 \end{bmatrix},$$

$$C_p = \begin{bmatrix} 0.5 & 1 & 0.5 & -0.25 \\ 0.5 & 1.5 & 0.5 & 0.375 \end{bmatrix}, D_p = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{53}$$

The An matrix is then

$$A_n = \begin{bmatrix} -12 - k_{p_1} & -\frac{73}{8} - 2k_{p_1} & -k_{p_1} & \frac{1}{2k_{p_1}} & 2k_{i_1} & 0 \\ 8 & 0 & 0 & 0 & 0 & 0 \\ -k_{p_2} & -3k_{p_2} & -23 - k_{p_2} & -\frac{69}{4} - \frac{3}{4k_{p_2}} & 0 & 2k_i \\ 0 & 0 & 4 & 0 & 0 & 0 \\ -\frac{1}{2} & -1 & -\frac{1}{2} & \frac{1}{4} & 0 & 0 \\ -\frac{1}{2} & -\frac{3}{2} & -\frac{1}{2} & -\frac{3}{8} & 0 & 0 \end{bmatrix} \quad (54)$$

which gives a characteristic equation

$$P(s) = s^6 + s^5\left(k_{p_2} + k_{p_1} + 35\right) + s^4\left(418 + 15k_{p_2} + k_{i_2} + k_{i_1} + 39k_{p_1}\right) + s^3\left(2507 + 109k_{p_2} + 437k_{p_1} + 39k_{i_1} - 3k_{p_1}k_{p_2} + 15k_{i_2}\right) + s^2\left(109k_{i_2} - 3k_{p_1}k_{i_2} + 1104k_{p_1} + 219k_{p_2} - 3k_{i_1}k_{p_2} + 96k_{p_1}k_{p_2} + 5037 + 437k_{i_1}\right) + s\left(1104k_{i_1} + 96k_{p_1}k_{i_2} + 96k_{i_1}k_{p_2} - 219k_{i_2} - 3k_{i_1}k_{i_2}\right) + 96k_{i_1}k_{i_2} \quad (55)$$

Figure 9 shows the stable range output of the system. The order of the outputs is the high then low values of kp1, kp2, ki1, ki2. Also, the larger blocks are shown first then the smaller. From this output, one or two control variables should be set. Figures 10 and 11 show the optimization and grid methods when they have similar quality results. This size set is approximately the cutoff point for the two methods; if the set is larger the grid method should be used, if the set is smaller the optimization method should be used.

Command Window

```
stable_range =

  100.0000   87.6250   50.5000   38.1250  100.0000   87.6250  100.0000   87.6250
  100.0000   87.6250   50.5000   38.1250  100.0000   87.6250   87.6250   75.2500
  100.0000   87.6250   50.5000   38.1250   87.6250   75.2500  100.0000   87.6250
  100.0000   87.6250   50.5000   38.1250   87.6250   75.2500   87.6250   75.2500
  100.0000   87.6250   38.1250   25.7500  100.0000   87.6250  100.0000   87.6250
  100.0000   87.6250   38.1250   25.7500  100.0000   87.6250   87.6250   75.2500
  100.0000   87.6250   38.1250   25.7500   87.6250   75.2500  100.0000   87.6250
  100.0000   87.6250   38.1250   25.7500   87.6250   75.2500   87.6250   75.2500
   87.6250   75.2500   50.5000   38.1250  100.0000   87.6250   87.6250   75.2500
   87.6250   75.2500   50.5000   38.1250   87.6250   75.2500   87.6250   75.2500
   87.6250   75.2500   38.1250   25.7500  100.0000   87.6250  100.0000   87.6250
   87.6250   75.2500   38.1250   25.7500  100.0000   87.6250   87.6250   75.2500
   87.6250   75.2500   38.1250   25.7500   87.6250   75.2500  100.0000   87.6250
   87.6250   75.2500   38.1250   25.7500   87.6250   75.2500   87.6250   75.2500
  100.0000   87.6250   50.5000   38.1250  100.0000   87.6250   75.2500   62.8750
  100.0000   87.6250   50.5000   38.1250  100.0000   87.6250   62.8750   50.5000
  100.0000   87.6250   50.5000   38.1250   87.6250   75.2500   75.2500   62.8750
```

Figure 9 The stable range as seen in the command window for the second example.

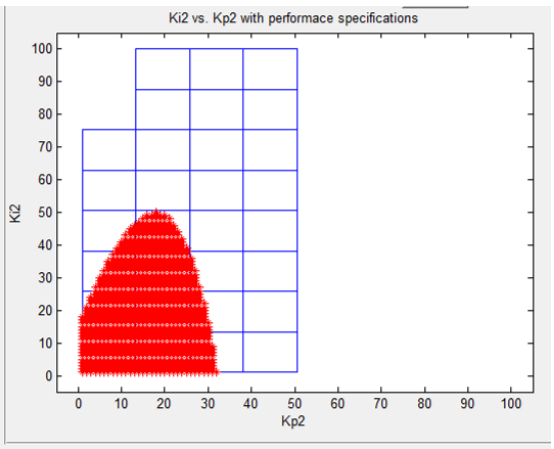

Figure 10 Performance satisfying range using the optimization method for the second example with a higher gain margin.

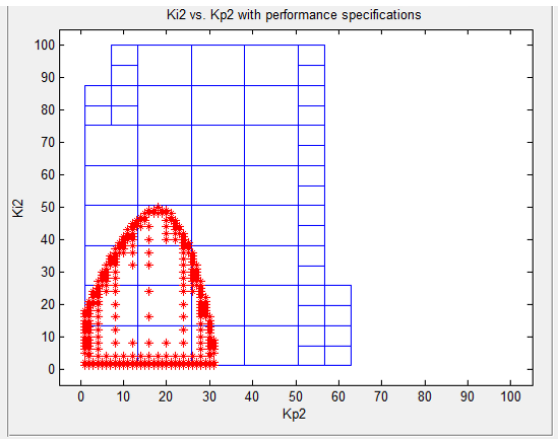

Figure 11 Performance satisfying range using the adaptive grid method for the second example with a higher gain margin.

The last example is a simple third order plant

$$G(s)=\begin{bmatrix}\dfrac{s^2+5s+6}{s^3+40s^2+491s+1820} & \dfrac{s+10}{s^3+33s^2+342s+1080}\\ \dfrac{s-4}{s^3+33s^2+327s+935} & \dfrac{s^2+10s+9}{s^3+43s^2+574s+2352}\end{bmatrix} \tag{56}$$

which has a characteristic equation of

$$P = s^{14} + s^{13}(149 + k_{p_1} + k_{p_2}) + s^{12}(k_{i_1} + 10021 + k_{p_1}k_{p_2} + k_{i_2} + 114k_{p_1} + 116k_{p_2}) + s^{11}(402061 + k_{i_1}k_{p_2} + 5721k_{p_1} + k_{p_1}k_{i_2} + 116k_{i_2} + 81k_{p_1}k_{p_2} + 5958k_{p_2} + 114k_{i_1}) + s^{10}(81k_{p_1}k_{i_2} + 178482k_{p_2} + 10712339 + 166426k_{p_1} + k_{i_1}k_{i_2} + 2812k_{p_1}k_{p_2} + 5721k_{i_1} + 81k_{i_1}k_{p_2} + 5958k_{i_2}) + s^{9}(2812k_{i_1}k_{p_2} + 2812k_{p_1}k_{i_2} + 166426k_{i_1} + 3109239k_{p_1} + 3455688k_{p_2} + 54768k_{p_1}k_{p_2} + 199560163 + 81k_{i_1}k_{i_2} + 178482k_{i_2}) + s^{8}(2812k_{i_1}k_{i_2} + 657720k_{p_1}k_{p_2} + 3455688k_{i_2} + 2663714119 + 3109239k_{i_1} + 45245634k_{p_2} + 54768k_{i_1}k_{p_2} + 39037278k_{p_1} + 54768k_{p_1}k_{i_2}) + s^{7}(657720k_{i_1}k_{p_2} + 657720k_{p_1}k_{i_2} + 406809962k_{p_2} + 334766179k_{p_1} + 54768k_{i_1}k_{i_2} + 45245634k_{i_2} + 39037278k_{i_1} + 25652630399 + 5040364k_{p_1}k_{p_2}) + s^{6}(176786204320 + 657720k_{i_1}k_{i_2} + 5040364k_{i_1}k_{p_2} + 1951796334k_{p_1} + 2494212598k_{p_2} + 406809962k_{i_2} + 5040364k_{p_1}k_{i_2} + 24687483k_{p_1}k_{p_2} + 334766179k_{i_1}) + s^{5}(849711065148 + 1951796334k_{i_1} + 24687483k_{p_1}k_{i_2} + 7541747820k_{p_1} + 24687483k_{i_1}k_{p_2} + 10108144311k_{p_2} + 75918207k_{p_1}k_{p_2} + 5040364k_{i_1}k_{i_2} + 2494212598k_{i_2}) + s^{4}(75918207k_{p_1}k_{i_2} + 18281510568k_{p_1} + 25301514690k_{p_2} + 146613664k_{p_1}k_{p_2} + 24687483k_{i_1}k_{i_2} + 7541747820k_{i_1} + 10108144311k_{i_2} + 2702104765920 + 75918207k_{i_1}k_{p_2}) + \quad (57)$$

Figure 12 shows the stable range and performance set using the grid method. A point to note, just because a set is disjointed does not mean an error has occurred. Figure 13 is a good example of this. Gaps can be found within a set because those points do not satisfy the performance specifications and these gaps become more common the more performance specifications there are. The optimization method is not included for the 3D images as the previous examples have shown it to be inferior to the grid method for three variables. Until this example, all the figures have shown the grid method to be superior to the optimization method. If this were the case in every example, the optimization method would be removed from the design method. In large sets, the optimization method has been shown to occasionally miss points since the set is comprised of spherical sets of points. While for these same examples, the grid method has demonstrated a complete outer edge of the set. The next four figures show sets that are fairly small. Figure 14 has no missing points even though it is the optimization method. Figure 15, however, does not have the complete edge that the previous examples have shown the grid method to have. One reason this occurred is because the original grid had a distance of eight and the edge is built around points where there is a change between it and its neighbors. Therefore, parts of the edge can be missed if the edge falls at a specific place on the grid. Figures 16 and 17 show another example of when the optimization method gives a more complete set than the grid method. If the set is narrow enough, it can be between points on the grid and remain undetected. For the most part, this happens with the portion of the set has a width less than eight and no part of the grid falls in the set.

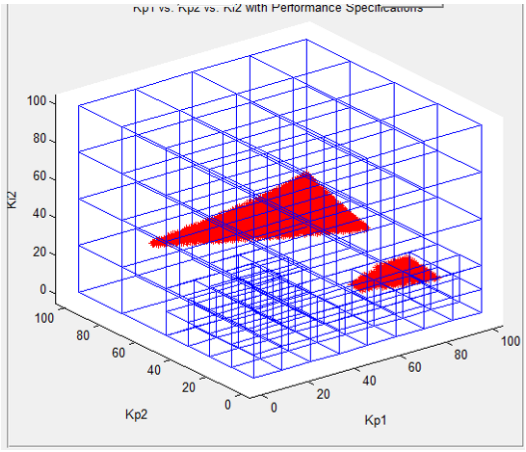

Figure 12 Performance satisfying range using the adaptive grid method for example three with two performance specifications.

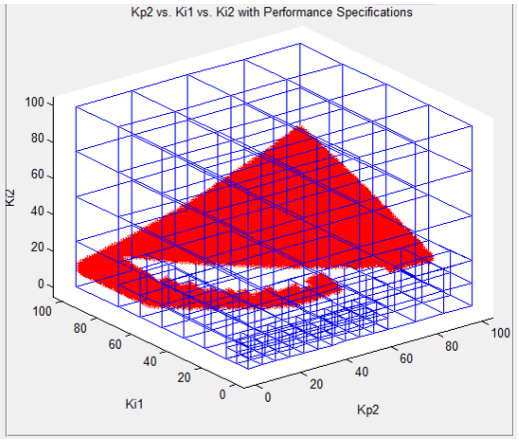

Figure 13 Performance satisfying range using the adaptive grid method for example three with three performance specifications.

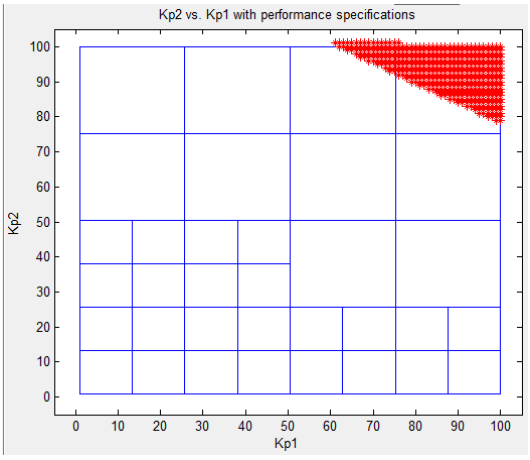

Figure 14 Performance satisfying range using the optimization method for example three with two performance specifications.

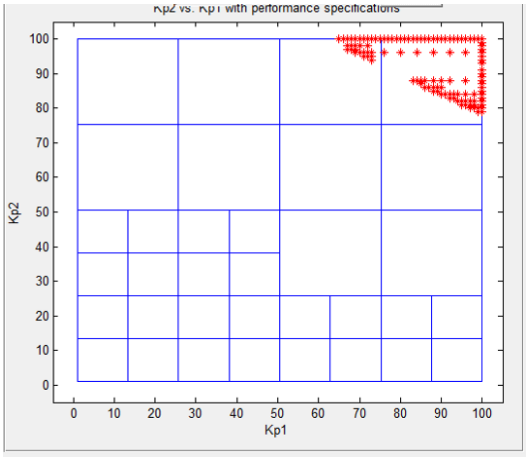

Figure 15 Performance satisfying range using the adaptive grid method for example three with two performance specifications.

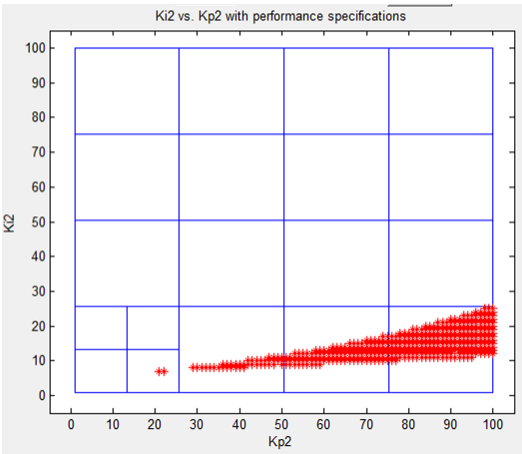

Figure 16 Performance satisfying range using the optimization method for example three with two performance specifications and differing K values from figure 14.

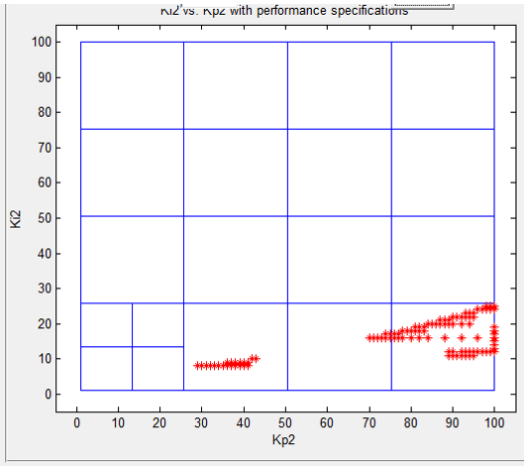

Figure 17 Performance satisfying range using the adaptive grid method for example three with two performance specifications and differing K values from figure 15.

"REAL WORLD" EXAMPLES

All the previous examples have been "made up" ones, but to show the design method works, "real world" examples are needed. The method is applied to five models of four aircraft; Commander 700, X-29, X-38, and F-5A. First is the longitudinal model of the X-38 as it is a SISO system. The system is given in state space as follows;

$$A = \begin{bmatrix} -.0335 & -22.5 & 0 & -32.2 \\ 0 & -.0944 & 1.0 & 0 \\ 0 & -1.94 & -.188 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, B = \begin{bmatrix} -8.83 \\ -.0196 \\ -2.02 \\ 0 \end{bmatrix}, C = [0010], \text{and } D = [0] \tag{58}$$

However, the program needs the transfer function which is

$$tf = \frac{-2.02s - 0.1527}{s^2 + 0.2824s + 1.958} \tag{59}$$

The PI controller from the previous examples is not of high enough order to stabilize the following MIMO systems. Therefore even though the PI controller could stabilize this SISO, the PID controller is used

$$C = \begin{bmatrix} \frac{s^2 k_{d_1} + s k_{p_1} + k_{i_1}}{s(\frac{s}{T}+1)} & 0 \\ 0 & \frac{s^2 k_{d_2} + s k_{p_2} + k_{i_2}}{s(\frac{s}{T}+1)} \end{bmatrix} \tag{60}$$

where T=1000. It has a state space representation of

$$Ac = \begin{bmatrix} -1000 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & -1000 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}, Bc = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \end{bmatrix},$$

$$Cc = \begin{bmatrix} 1000(k_{p_1} - 100) - 1000000(k_{d_1} - 100) & 1000(k_{i_1} - 100) & 0 & 0 \\ 0 & 0 & 1000(k_{p_2} - 100) - 1000000(k_{d_2} - 100) & 1000(k_{i_2} - 100) \end{bmatrix}$$

$$Dc = \begin{bmatrix} 1000(k_{d_1} - 100) & 0 \\ 0 & 1000(k_{d_2} - 100) \end{bmatrix} \quad (61)$$

So, the characteristic equation for this SISO system is

$$P_{PID} = s^4 + (.4854 - 0.00202k_{p_1})s^3 + (2.145 - 0.0018693k_{p_1} + 0.20200e-5k_{d_1})s^2 + (-0.00018693k_{p_1} - 0.18869 - 0.1527e-6k_{d_1} + 0.00202k_{k_1})s + (0.0001527k_{i_1} - 0.01527) \quad (62)$$

All figures for these next examples use the grid method because all but figure 18 is 3D and as previously stated the optimization method is not appropriate for those graphs. Figure 18 shows the resulting set when Kd1 is zero and the performance specifications are that the minimum gain and phase margins are 10 and 60° and the maximum overshoot, settling time, and rise time are 15%, 10 seconds, and 2 seconds. As the blue box shows, the stable range for this system is large. In fact for all the examples in this section, the stable range is -100 to 100 for all control variables. The reason for this occurring with the PID and not the PI is the large T variable in the PID controller. Figure 19 has the same performance specifications, but Kd1 is allowed to vary. A note about Figure 19 that will occur again, when the adaptive grid method is used for the three variable graphs, it becomes possible to run out of memory when most of the stable range is in the performance set. Therefore, the minimum grid size is 4 for this figure and this will be the case for the rest of the graphs that take up most of the stable range.

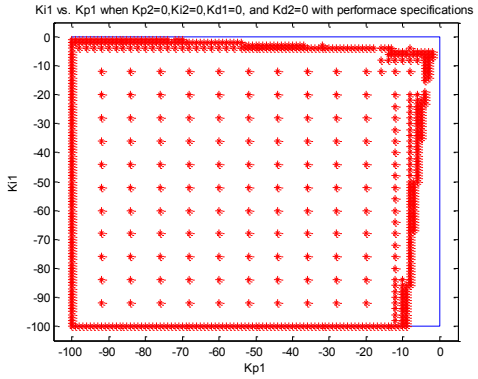

Figure 18 Performance satisfying range using the adaptive grid method for the X-38 with PID.

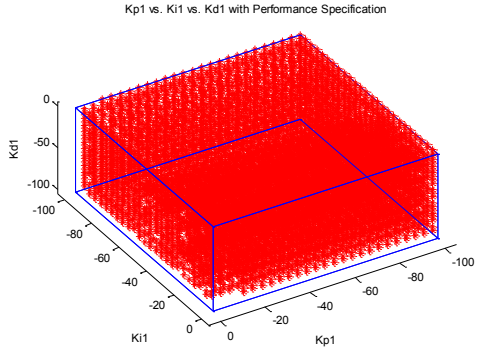

Figure 19 Performance satisfying range using the adaptive grid method at the second iteration for the X-38 with PID.

The next system is also the X-38 but is the lateral/directional model. The state space representation is

$$A = \begin{bmatrix} -.048 & 0 & -1 & .032 \\ -29.35 & -.15 & .082 & 0 \\ .11 & .012 & -.091 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} B = \begin{bmatrix} 0 & .0048 \\ 3.13 & 3.35 \\ -.22 & -.91 \\ 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} D = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad (63)$$

which results in a transfer function of

$$tf = \begin{bmatrix} \frac{3.13s^3 + 0.417s^2 - 6.1s - 8.657e^{-17}}{s^4 + 0.289s^3 + 0.1342s^2 + 0.6041s + 0.08518} & \frac{-0.22s^3 - 0.006s^2 + 0.0002189s - 0.1956}{s^4 + 0.289s^3 + 0.1342s^2 + 0.6041s + 0.08518} \\ \frac{3.35s^3 + 0.2501s^2 - 26.34s + 1.179e^{-16}}{s^4 + 0.289s^3 + 0.1342s^2 + 0.6041s + 0.08518} & \frac{-0.91s^3 - 0.1395s^2 - 0.006234s - 0.8429}{s^4 + 0.289s^3 + 0.1342s^2 + 0.6041s + 0.08518} \end{bmatrix} \quad (64)$$

The denominator of each term is the same because it was derived from the state space representation; this will be the case for the rest of the examples. Figure 20 shows the set when Kp1 = 50, Kp2 = 80, and Ki1 = 20 and the maximum overshoot, settling time, and rise time are 15%, 5 seconds, and 2 seconds.

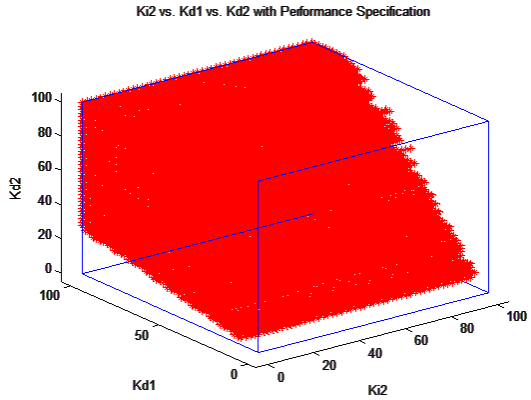

Figure 20 Performance satisfying range using the adaptive grid method for the X-38 Lat/D with PID.

Figures 21 and 22 are the performance set of the wing leveler model of the Commander 700 aircraft which has a transfer function of

$$tf = \begin{bmatrix} \frac{-2.09s^3 - 1.098s^2 - 5.334s + .02039}{s^4 + 2.811s^3 + 3.887s^2 + 6.343s - .1021} & \frac{.1207s^3 + 1.328s^2 - .06059s - 1.082}{s^4 + 2.811s^3 + 3.887s^2 + 6.343s - .1021} \\ \frac{.2706s^3 - .1104s^2 - .3915s + .00275}{s^4 + 2.811s^3 + 3.887s^2 + 6.343s - .1021} & \frac{-1.66s^3 - 4.062s^2 - .3154s - .05245}{s^4 + 2.811s^3 + 3.887s^2 + 6.343s - .1021} \end{bmatrix} \quad (65)$$

The K variables are set such that Kp1 = -45, Kp2 = -5, and Ki1 = -90 and the performance specifications are that the minimum gain and phase margins are 10 and 60° and the maximum settling time and rise time are 10 and 2 seconds. The positive graph of figure 21 is included for completeness because Kd1 and Kd2 must be negative or zero. The next system is a model of the X-29 aircraft and has a transfer function of

$$tf = \begin{bmatrix} \frac{77s^3 + 23.37s^2 + 601.6s - 1.901}{s^4 + 2.824s^3 + 8.583s^2 + 19.39s - .3476} & \frac{13s^3 - 2.183s^2 + 26.57s - .08522}{s^4 + 2.824s^3 + 8.583s^2 + 19.39s - .3476} \\ \frac{5.1s^3 + 6.145s^2 + 43.44s + 26.4}{s^4 + 2.824s^3 + 8.583s^2 + 19.39s - .3476} & \frac{-4.1s^3 - 12.35s^2 + .7712s + 1.184}{s^4 + 2.824s^3 + 8.583s^2 + 19.39s - .3476} \end{bmatrix} \quad (66)$$

For figures 23 and 24, Ki2 = 75, Kd1 =25, and Kd2 = -25 while the minimum gain and phase margins are again 10 and 60° and the maximum overshoot, settling time, and rise

time is 15%, 5 seconds and 2 seconds. As with the previous example, figure 23 is only included for completeness because in this case Kp1 and Ki1 must be negative or zero.

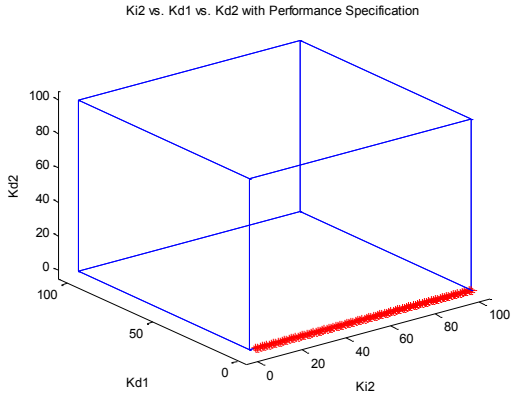

Figure 21 Performance satisfying range using the adaptive grid method for the Commander 700.

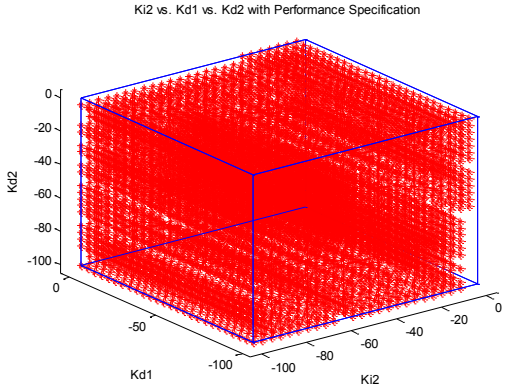

Figure 22 Performance satisfying range using the adaptive grid method for the Commander 700, second iteration.

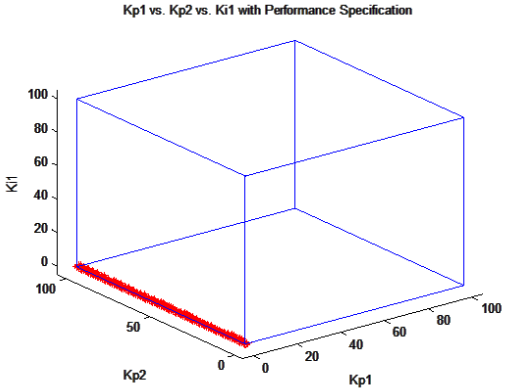

Figure 23 Performance satisfying range using the adaptive grid method for the X-29.

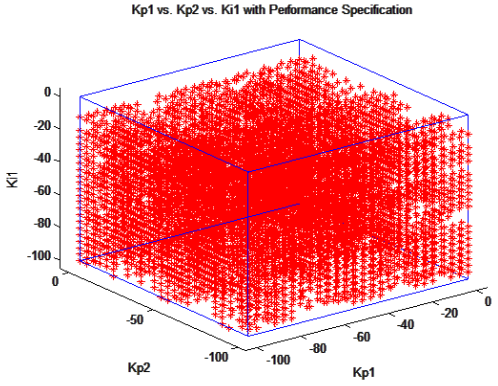

Figure 24 Performance satisfying range using the adaptive grid method for the X-29, second iteration.

The final system is a lateral/directional linear model for the F-5A aircraft which has a transfer function of

$$tf = \begin{bmatrix} \frac{1.01s^3 + .7716s^2 + 13.06s - .002764}{s^4 + 4.652s^3 + 15.63s^2 + 47.53s + .528} & \frac{.223s^3 + .04456s^2 - 2.544s + .0005429}{s^4 + 4.652s^3 + 15.63s^2 + 47.53s + .528} \\ \frac{.0091s^3 + .1026s^2 + .08864s + .5035}{s^4 + 4.652s^3 + 15.63s^2 + 47.53s + .528} & \frac{-.11s^3 - .434s^2 - .08798s - .0989}{s^4 + 4.652s^3 + 15.63s^2 + 47.53s + .528} \end{bmatrix} \quad (67)$$

For figures 25 and 26, Kp1 = 80, Ki1 = -40, and Kd1 = 20 and the maximum overshoot, settling time, and rise time are 15%, 10 seconds, and 2 seconds. This example would be a perfect case for the optimization method for performance when another control variable was set and the images were two dimensional.

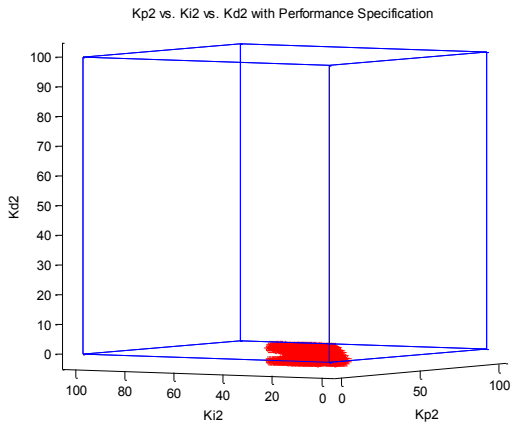

Figure 25 Positive performance satisfying range using the adaptive grid method for the F-5A.

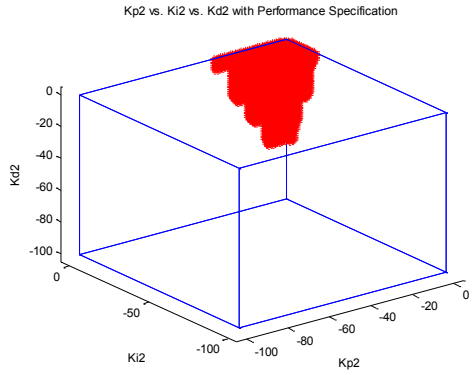

Figure 26 Negative performance satisfying range using the adaptive grid method for the F-5A.

CONCLUSIONS

As shown in the previous examples, the design method can give a stabilizing range and performance satisfying set. However, when the method does not find a range or set, it does not mean a set does not exist. It merely means that the method could not find it at the threshold given or in the range specified. As shown with the PI examples, the smaller the threshold the more sections of the stable range can be found, but at the risk of running out of time or memory. In fact the sections can get down to a single point which destroys the point of the algorithm completely but would always find the full set within the given range. The PID examples do not show this because of the large T variable. The optimization method for performance is based on two optimization methods. Both methods search for a local maximum or minimum, which is why many points in a large set can be missed with this method. The adaptive grid method works well with large sets, but can miss points or the complete set if it is small enough to not fall on the grid. If the design method works, a stable range and performance set is found. In comparison to the methods discussed in the introduction, modern control theory will always find a stabilizing controller which this design method will not but this method is not restricted to the same order controller as the plant. Like the “ad hoc” method, this design method will find the stabilizing control values if they exist in the variable range. Though with this method, a set a variables is found instead of tuning to one that happens to work.

In relation to the recent work previously mentioned, the HIFOO method from [4] is the only method with less intensive mathematics though this is mainly because it uses a readily available program. Like the design method in this paper, the HIFOO method can have an issue with run-time but it has a parameter to control the run-time while the threshold of the stability range and the iterations of the performance determine the run-time for the current method. The methods in [5] and [6] are similar in the complexity as well as the methodology. As they both use optimization, the likelihood of finding a stabilizing controller might be better than with the current method if the stable range is small. However, the optimization will result in a single controller result instead of the range of controllers given here. As mentioned before, the method in [7] is the only one to work without a model. Therefore for certain applications it is superior to the other methods discussed. However, the design method developed here gives ranges of stability which can help reduce the effect of modeling errors. And like the methods in [4,5,6], [7] also uses optimization resulting in one controller.

For further work, there are two areas that should be improved. First, in this method, the variable range is set and the stable range found within it. It would be advantageous to work out from a set range and find an approximation of the entire set. However, it is difficult to do with the positivity algorithm as the block sizes can be very small so expanding isn't always possible. A thought would be to only expand and bisect the outer boxes of the stable range. Also, the performance methods included give their information best in a graph form. A method that would give the performance in the same block style as the positivity algorithm would be helpful as a graph would not be needed

and all control variables could be included instead of a maximum of three. In conclusion, there are a few options when the design method doesn't work. One, reduce the threshold and try again. Two, increase the order of the controller and try again. Lastly, try a different design method.

REFERENCES

[1] L.H. Keel, S.P. Bhattacharyya, "Data driven synthesis of three term digital controllers," *American Control Conference, 2006* , pp.6, 14-16 June 2006

[2] James Bennett, Ajay Bhasin, Jamila Grant, and Wen Chung Lim, "PID Tuning Classical," *The University of Michigan Chemical Engineering Process Dynamics and Controls Open Textbook*, 16 Oct. 2007. Web. 06 Mar. 2012. [Online] <https://controls.engin.umich.edu/wiki/index.php/PIDTuningClassical>.

[3] N. Andrei, "Modern Control Theory - A historical perspective," *Studies in Informatics and Control*, vol.10, no.1, March 2006, pp.51-62.

[4] S Gumussoy, M. Millstone, M.L. Overton, "H∞ strong stabilization via HIFOO, a package for fixed-order controller design," *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on* , vol., no., pp.4135-4140, 9-11 Dec. 2008.

[5] Qiongbin Lin, Fuwen Yang, "Multiobjective fixed-order controllers for MIMO systems," *Control, Automation, Robotics and Vision, 2008. ICARCV 2008. 10th International Conference on* , vol., no., pp.1494-1497, 17-20 Dec. 2008.

[6] Zhan Shu; Lam, J.; Ping Li; , "Simultaneous $\mathcal{H}\infty$ stabilization via fixed-order controllers: Equivalence and computation," *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on* , pp.3244-3249, 15-18 Dec. 2009.

[7] Gorka Galdos, Alireza Karimi, Roland Longchamp, "H∞ Controller design for spectral MIMO models by convex optimization," *Journal of Process Control*, vol. 20, no. 10, Dec 2010, pp 1175-1182,

[8] L.H. Keel, S.P. Bhattacharyya, "Fixed order multivariable controller synthesis: A new algorithm," *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on* , pp.977-982, 9-11 Dec. 2008

[9] J. Kennedy, R. Eberhart, "Particle swarm optimization," *Neural Networks, 1995. Proceedings., IEEE International Conference on* , vol.4, Nov/Dec 1995, pp.1942-1948

[10] Vladislav Gorbunov, "Genetic Optimization," *Trade Smart Research*, [Online]. 27 Sept. 2011. <http://www.traderssoft.com/publications/go/>.

[11] Lampton, Amanda, Niksch, Adam, and Valasek, John, "Reinforcement Learning of a Morphing Airfoil-Policy and Discrete Learning Analysis," *Journal of Aerospace Computing, Information, and Communication*, vol. 7, no. 8, August 2010, pp. 241-260.

VITA

In May 2006, Amanda Lynne Harris recieved her Bachelor of Science degree in Electrical Engineering from the University of Kansas. She then completed her Masters of Science with distinction in Signal Processing and Communications in Nov. 2007 at the University of Edinburgh. In Aug. 2007 Ms. Harris began her work on a Doctor of Philosophy at Texas A&M. During her time at KU, she completed a summer 2004 internship at the Army Corps of Engineers in Kansas City, MO. She was also employed by the University of Kansas Center for Remote Sensing of Ice Sheets (CReSIS) her senior year. While at Texas A&M, Ms. Harris was a Teaching Assistant for four semester. In the summer of 2008, Ms. Harris interned through the Office of Naval Research (ONR) Naval Research Enterprise with NAVAIR weapons department, located at China Lake Naval Base, CA. In addition, she interned through Office of Under Secretary of Defense with Acquisition, Technology and Logistics/Joint Interoperability at the Pentagon in the summer of 2009. In 2009, Ms. Harris was awarded the Science, Mathematics, & Research for Transformation Scholarship (SMART) managed by Naval Postgraduate School (NPS) and American Society for Engineering Education (ASEE) on behalf of the Office of the Secretary of Defense. Through this award, she worked summers 2010 and 2011 with TRADOC Analysis Center at White Sands Missile Range, NM. Ms. Harris can be reached through Department of Electrical and Computer Engineering, Texas A&M University, 214 Zachry Engineering Center, TAMU 3128 College Station, Texas 77843-3128