JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 19, No. 3, May–June 1996

Direct Optimization of Gain Scheduled Controllers via Genetic Algorithms

Stuart C. Kramer
U.S. Air Force Institute of Technology, Wright–Patterson Air Force Base, Ohio 45433-7765
and
Robert C. Martin IV
U.S. Air Force Wright Laboratory, Wright–Patterson Air Force Base, Ohio 45433-7505

A new approach to designing gain scheduled controllers in which controller performance is directly optimized over the operating envelope of the system is presented. Rather than interpolating between point designs, the basic controller form is selected, and then the controller's parameter schedule is computed to minimize the selected cost functional. The resulting optimization problem is difficult and ill behaved but can be solved using genetic algorithms. The feasibility of the approach and the resulting performance improvements are demonstrated by designing several controllers for the longitudinal dynamics of an F-18 aircraft and comparing the results to a traditionally designed controller.

I. Introduction

GAIN scheduling is a time-honored adaptive control technique useful in situations where the plant dynamics vary nonlinearly as a function of some relatively slowly changing external variables, as, for example, the dependence of aircraft dynamics on altitude and airspeed. By making the parameters of the controller dependent on some or all of the external variables, the controller can change to match the changing dynamics of the plant. The external variables are the scheduling variables, the controller parameters that change are the scheduled variables or parameters, and the numerical relation between the two is the gain schedule. The operating point of the plant at any time is identified by the values of the scheduling variables at that time, and the operating envelope is the expected range of the scheduling variables. The basic objective in gain scheduling is to compensate for changes in the plant dynamics so that the overall system has the desired response over the entire operating envelope. In other words, given a nominal closed-loop behavior (in our case the response at a nominal operating point), we want to design a gain schedule that minimizes the gain scheduling error, the difference between the nominal response of the system and the actual response throughout the envelope. The desired response may or may not be independent of the operating point; either case can be accommodated by appropriate definition of the gain scheduling error.

There are three general concerns in designing such controllers: maintaining stability throughout the operating envelope, minimizing gain scheduling error (maximizing consistency) over the operating envelope, and minimizing gain schedule complexity. (Note that this in no way implies minimizing complexity of the overall controller. That is largely dependent on the complexity of the basic unscheduled controller, which, as we note later, is assumed to be given.) Only the first has been rigorously considered.[1–3] The other two are currently addressed by utilizing some general rules of thumb and engineering judgment.[4–6] Furthermore, while it is clear that consistency and complexity tend to be conflicting, little or no attention has been paid to the tradeoff between the two. Our objective in this paper is to demonstrate an approach that allows the designer to explicitly address these last two concerns as well as their interaction.

The existing design process for gain scheduled controllers consists of four basic steps.[2,6] First, select appropriate scheduling variables and pick a number of constant operating points covering the

Stuart Kramer received his B.S.E.E. from Colorado State University in 1976, his M.S.S.E. from the U.S. Air Force Institute of Technology in 1978, and his Ph.D. from the University of California at San Diego in 1985. He has served in a number of positions, including weapons systems analyst and system program manager. He is currently Associate Professor of Systems Engineering in the Department of Aeronautics and Astronautics at the Air Force Institute of Technology and is the curriculum chair for the Graduate Systems Engineering Program.

Robert C. Martin IV received a B.S. in Aeronautical Engineering from the University of Cincinnati in 1993. After receiving his commission in the U.S. Air Force, he attended the Air Force Institute of Technology, where he completed an M.S. in Aeronautical Engineering in December 1994. He is currently assigned to the Flight Dynamics Laboratory.

Received April 12, 1995; revision received Dec. 12, 1995; accepted for publication Feb. 8, 1996.

operating envelope. Second, select a controller form and design a controller of that given structure for each of the operating points. Third, use the parameter values from these point designs to form a control schedule that fills in between the design points while trying to maintain performance at the design points. Two simple options for the scheduling functions are piecewise constant in the neighborhood of each design point and linear interpolation between design points. As a more sophisticated alternative, the designer could fit a low-order polynomial or other simple function to the parameter values from the individual design points. Finally, check the performance of the system at off-design operating points. Although this procedure results in generally acceptable controllers, it is essentially just a concatenation of point designs.

This paper presents an alternative method, which directly optimizes the gain schedule over the entire operating envelope. Reviewing the process, we see that the designer makes some structural choices (e.g., the scheduling variables, the form of the controller, the form of the relation between the scheduling variable and the controller parameters) and several numerical choices (e.g., the number and location of the design points, the values of the parameters in the gain schedule). We propose making the structural decisions using current practice, but making the numerical choices so as to minimize the gain scheduling error over the entire operating envelope. In this sense, this is a global rather than point design process. This approach explicitly addresses overall system performance over the entire operating envelope and, as will be shown, allows us to trade off performance and complexity.

The numerical optimization problem inherent in this approach can be very complex, of high dimensionality, and with ill-behaved or nonexistent gradients.[7,8] Genetic algorithms (GAs) are known to be robust in these circumstances,[9,10] and so the authors chose to use a GA for the optimization. GAs also easily accommodate integer valued parameters and allow the number of parameters to vary, which allows us to optimize a broader range of parameters. Other approaches such as simulated annealing may work as well but were not explored in this research.

The next section contains a brief overview of GAs and a discussion of the specific algorithm parameters used in this work. More detailed descriptions can be found in any of a number of sources, such as the text by Michalewicz.[10] The following sections describe the proposed approach in detail and compare controllers designed using this approach to a traditionally designed one for an F-18 aircraft.

II. Overview of GAs

GAs have been used to find global optima in problems with both nonlinearities and high dimensionality where calculus-based methods have failed, such as aircraft structural parameter design[11] and various control optimizations.[12–16] GAs differ from traditional optimization techniques in four basic ways: 1) they code the parameter set instead of the parameters themselves, 2) they search a population of points instead of a single point in the solution space, 3) they do not require an initial guess of the solution, and 4) they use probabilistic transition rules instead of deterministic transition rules. They have a number of advantages in complex problems. GAs are zero-order methods and do not use the derivative of the objective function, and so they can easily handle sharp constraints and penalty functions. The computation time for a GA typically grows linearly with the dimension of the problem, whereas calculus-based methods grow at least quadratically. Since the solution is not dependent on an initial guess, a GA is more likely to find the global optimum.

GAs are probabilistic search algorithms based on survival of the fittest where fitness is determined by a solution's score on the objective function and meeting the constraints. They are implemented by generating an initial random population of possible solution vectors and allowing them to evolve by applying crossover, mutation, and reproduction. Each solution vector is represented by a binary string (chromosome), which is a concatenation of coded representations of each design variable (genes). Each gene gives a value between pre-established minimum and maximum values for the variable, with precision determined by the length of the gene. Increasing the length increases precision, but it also increases computation time.

The crossover operator exchanges substrings, known as schemata, between two solution strings in the population. For example, take the two chromosomes [1 1 0 0 1] and [0 1 0 1 0]. If the crossover point was chosen to be between the third and fourth bits, the resulting chromosomes would be [1 1 0 1 0] and [0 1 0 0 1]. The crossover rate is controlled by the user defined probability of crossover p_c. In general, the higher the probability of crossover, the faster the GA exploits the schemata in the current population, but the more likely it is to prematurely converge. Mutation changes one bit in a chromosome to its complement value, potentially introducing new schemata into the population. For example, if the second bit were selected for mutation, [1 1 0 1 0] would become [1 0 0 1 0]. This rate is determined by the probability of mutation p_m. Typically, p_m is small so that the disruption from mutation does not prevent the GA from converging. As p_m increases, the GA approaches a random search.

The reproduction operator acts on the population only after both mutation and crossover because reproduction is dependent on the resulting fitness of each solution. Each solution reproduces with a probability proportional to its fitness, so strings with above average fitness are reproduced at higher rates than strings with below average fitness. Consequently, poor solutions tend to die out. By not eliminating the worst solutions immediately, the population remains diverse and does not converge prematurely to a local optimum. Since reproduction is random, good solutions occasionally die. To avoid this, the GA included an elitist rule that guaranteed the best solution would survive each generation.

Selecting good values for the various parameters requires some educated guessing and experimentation. For the optimizations presented the population size was 100, the string length was varied to maintain three significant digits precision, the probability of crossover was 0.95, and the probability of mutation was 0.01. These values were chosen based on results from Grefenstette[17] and preliminary experimentation by the authors.[8]

III. Approach

As just alluded to, our proposed approach is to extend the current practice of designing gain scheduled controllers by including explicit optimization. There are three main steps.

1) Establish the structural attributes of the controller. This first step is essentially unchanged from current practice. The process begins by identifying the scheduling variable or variables. Guidelines for this are well established.[2–6] Next the designer must choose the generic form of the controller, which will be a matter of engineering judgment. We must emphasize that the choice of a generic controller design is a prelude to the proposed optimization and is functionally independent from it; the objective of the approach is to optimize the gain schedule given a particular generic controller design as an input. The designer then identifies the parameters of the generic controller that will change with the system operating point and will have to be scheduled, and specifies the general form of the relation between those parameters (the scheduled variables) and the scheduling variable. The free parameters in the scheduling functions are the basic design variables in the optimization. Finally, the designer may identify other system parameters that are to be included as design variables.

2) Optimize the gain schedule. In this step we depart from standard practice by explicitly computing the gain schedule to minimize some specified error. The first task is to characterize the gain scheduling error by defining an objective function. This may include side constraints, penalty functions, or secondary objectives. Next the designer must set up the optimization algorithm. For a GA, this means specifying a few algorithm parameters and the range and resolution of each design variable, as discussed in Sec. II. To determine the ranges of the design variables, we recommend designing a small number of point controllers at extreme points in the operating envelope. The final part of this step is to run the algorithm.

3) Verify the operation of the controller. If the objective function was evaluated only at a selection of discrete points, this should include checking system performance at some additional points within the envelope.

IV. Example

The following example illustrates the proposed process and the performance improvement that results.

A. Problem Description

We consider the design of a longitudinal controller for an F-18 aircraft. The full aircraft dynamics are described by a nonlinear multi-input/multi-output transfer function. For this work, our truth model was a complex simulation program maintained by the Air Force Wright Laboratories. It is a full six-degree-of-freedom model using aerodynamic coefficients derived from recent flight test data. Standard simplifications and perturbation analysis results in a locally linear approximation that is a strong function of both altitude and airspeed. The longitudinal dynamics are dominated by the short period mode, which can be approximated by

$$\begin{bmatrix}\dot{\alpha}\\ \dot{q}\end{bmatrix} = A\begin{bmatrix}\alpha\\ q\end{bmatrix} + \begin{bmatrix}0\\ 1\end{bmatrix}\delta$$

where δ is normalized elevator deflection, α is angle of attack, and q is pitch rate. The dynamic matrix A depends on the flight conditions; numerical values at the various operating points used in this paper are listed in Appendix A. Elevator actuator dynamics are included by prepending the transfer function

$$\delta(s) = \frac{(s/82.9)^2 + 2(0.068)s/82.9 + 1.0}{[(s/36.4)^2 + 2(0.41)s/36.4 + 1][(s/105.3)^2 + 2(0.59)s/105.3 + 1]}\dot{q}_i(s)$$

where $\dot{q}_i$ is the commanded pitch acceleration. The reader is referred to the baseline design documentation[18] for details of the derivation.

B. Baseline Design

The baseline for the example is an existing gain scheduled controller designed using traditional methods by a team at the Air Force Wright Laboratories.[18] This section summarizes the main elements of their design procedure and results.

The original team began by selecting dynamic pressure ($\bar{q}$) as the scheduling variable since it varies slowly relative to the time constants in the longitudinal dynamics and is easily measured. Although there is independent variation with both airspeed and altitude, most of the variation can be accounted for as a function of dynamic pressure. For this aircraft, the operating envelope corresponds to a dynamic pressure range of 0–1000 psf.

The overall controller was based on an inner/outer loop design as shown in Fig. 1. The only objective of the inner-loop design was make the inner closed-loop response as consistent as possible, specifically, as close as possible to the response at a specific nominal operating point. The inner loop, therefore, had to account for all of the operating point dependent dynamics of the aircraft. This suggested using a gain scheduled design for the inner loop. Since the closed inner loop would have (at least approximately) an unvarying transfer function, the outer-loop controller K_{outer} (see Appendix B for details) could be designed just to meet performance objectives given the transfer function P_{inner} at the nominal operating point. Ideally, this combination should result in good closed-loop response over the entire envelope. For the baseline nominal operating point, the original designers chose Mach 0.95 airspeed at 20-kft altitude ($\bar{q} = 614$ psf), based on their engineering judgment. Note that the nominal operating point could be considered a design parameter,[19] although the original designers did not do so.

They next selected 11 other operating points distributed over the envelope (see Appendix A). At each point they designed a minimal order H_∞ inner-loop controller to minimize the relative error $\|\bar{\sigma}(\tilde{\Delta})\|_\infty$, where $\bar{\sigma}$ is the maximum singular value, $\|\cdot\|_\infty$ is the infinity norm, $\tilde{\Delta} \doteq (P - P_0)P_0^{-1}$, P_0 is the closed inner-loop frequency response at the nominal operating point, and P is the closed inner-loop frequency response at the design point. This resulted in the inner-loop controller

$$\dot{z} = (F - GN)z + K_f\begin{bmatrix}\alpha\\ q\end{bmatrix}; \qquad \dot{q}_f = Nz + M\begin{bmatrix}\alpha\\ q\end{bmatrix}$$

The parameters F, K_f, and G were found to be essentially constant over the envelope with $F = -40$, $K_f = [1\ \ 1]$, and $G = 0.0247$. M and N were scheduled by fitting a linear function of dynamic pressure to the parameter values obtained from the point designs, resulting in $M = [50.5 - 0.058\bar{q}\ \ 8.11 - 0.006\bar{q}]$ and $N = 461 - 0.312\bar{q}$.

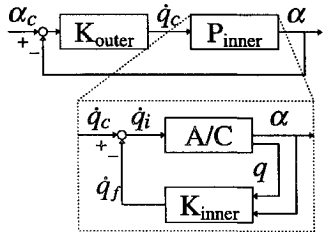

Fig. 1 Inner/outer loop structure.

C. Proposed Method

To illustrate our proposed design approach, we designed several alternative controllers for the situation just described. These resulted from making different choices within the process, as noted subsequently. The designs proceeded as follows with the major steps numbered to correspond with the earlier description.

1) The approach to choosing the scheduling variable and controller form is not changed from current practice. For ease of comparison, we used dynamic pressure as the scheduling variable, retained the inner/outer loop design with the same inner-loop controller structure, and scheduled the same controller parameters. This was essentially an arbitrary choice, made in this case based on our objective to illustrate the improvement possible with direct optimization. Other generic controller structures are certainly possible and could yield better performance. Two scheduling functions were considered: piecewise constant and piecewise linear. The intervals for the piecewise constant and piecewise linear functions were held constant in some designs and allowed to vary in length and number in others. In some cases the nominal operating point was allowed to vary as a design variable; when it was fixed, it was at the same point used in the baseline design.

2) We used several objective functions to explore various tradeoffs of interest, all based on the relative error criterion used in the baseline design. The logical extension from minimizing the error at each point design is to minimize its integral over the envelope. As this was impractical, we approximated it by summing over a number of sample points, the original 12 points used in the baseline design plus 8 more chosen to evenly cover the operating envelope (see Appendix A). Thus, the initial cost functional was

$$J_1 = \sum_{i=1}^{20} \|\bar{\sigma}(\tilde{\Delta}_i)\|_\infty \tag{1}$$

where the terms are as defined earlier. If the nominal operating point is included in the sample set, as it is in this example, one of the error terms will be zero.

As noted earlier, the current method does not explicitly address the complexity of the controller schedule. For the piecewise constant and piecewise linear scheduling functions, complexity is directly related to the number of intervals used. This can be included by appending a penalty on the number of intervals, as in

$$J_2 = \sum_{i=1}^{20} \|\bar{\sigma}(\tilde{\Delta}_i)\|_\infty + f(N) \tag{2}$$

where $f(N)$ is a functional weighting on the number of intervals. For this example, the authors chose to use $f(N) = N$, but others are, of course, possible.

The basic objective function in Eq. (1) is a frequency-domain measure. Including time-domain objectives is often desirable, but

frequently difficult because of the requirements of calculus-based optimization schemes. We added a time-domain relative error based on the closed outer loop α_c to α behavior. For simplicity, we just used the same K_{outer} controller developed by the original designers. The time response relative error is defined as

$$e_i = \int_0^{t_f} |\alpha_i(t) - \alpha_0(t)|\, dt$$

where α_0 is the closed-loop time response to a unit step at the nominal operating point and α_i is the response at the ith sample point. Thus,

$$J_3 = \sum_{i=1}^{20} \left[\|\tilde{\sigma}(\tilde{\Delta}_i)\|_\infty + \tfrac{1}{5} e_i(t) \right] \tag{3}$$

The factor $\frac{1}{5}$ was chosen to roughly balance the magnitudes of the frequency- and time-domain error terms. Other values could be used for different relative tradeoffs between the two errors.

The various parameters required to run the GA were selected after some minor experiments on simplified problems, as noted in Sec. II. The ranges of the scheduled parameters were set by extrapolating from the baseline controller schedule. Had these not existed, we would have done a few point designs at the extremes of the envelope to find reasonable limits.

3) For simplicity, only one of the designs was validated. We selected several additional evaluation points and compared the resulting closed-loop time and frequency responses with those at the nominal operating point.

V. Results

The authors chose the following cases from among the various combinations of design variables and objective functions mentioned earlier to illustrate the range of tradeoffs possible with this technique. The cases generally progress by increasing the number of design variables each time. The first four cases all assume piecewise constant scheduling functions. In the first case, the intervals over which the functions are constant were four equal-sized segments covering the operating range from $q = 0$ to $q = 1000$. The width of the intervals is allowed to vary in the second case, whereas in the third the number is also changeable. The fourth case includes varying the nominal operating point. The last two cases demonstrate both the ability to use more complicated scheduling functions and the ability to trade off between frequency- and time-domain objectives. Table 1 summarizes the main points of each of the cases. It is worth noting that despite the varying complexity of the resulting optimization problems, there was little variation in the rate of convergence or total computational effort of the GA.

The numerical results for these cases are summarized in Table 2, which includes the baseline as case 0 for comparison. Note that J_1 and J_3 were computed for all cases regardless of which was used as the actual objective. We will consider each case in turn.

Case 0, the original design, is included as a baseline reference. The summed relative error [maximum singular value of $(P - P_0)P_0^{-1}$] for the original design is $J_1 = 6.37$. Recall that this controller was obtained by curve fitting a schedule to a set of controllers, which were each designed to minimize the relative error at their respective operating point. Hence, this seems an appropriate measure for the resulting scheduled controller. Figure 2 shows the log magnitude of the relative error vs frequency at the 19 off-nominal operating points. Smaller singular values (lower curves) are more desirable; clustering or uniformity is unimportant. The closed-loop angle-of-attack α response to a step input in commanded angle of attack at each of the sample points is shown in Fig. 3. Here uniformity is most important since we want the inner-loop response to be as consistent as possible. Other features such as overshoot or steady-state error can be corrected by modifying the outer-loop controller and so are unimportant in this evaluation.

For case I, the only design variables are the constant values assigned each of the scheduled variables for the four intervals. The nominal operating point is left at the baseline value. Remarkably, this simple schedule results in a 50% reduction in the summed relative error. Interestingly, the time error increases slightly, as can be seen by noting that the decrease in J_3 is less than the decrease in its frequency-domain component J_1.

In case II, the interval width is allowed to vary. This yields additional improvement, as would be expected since the design degrees of freedom increased. The change is not as striking as in the first

Table 1 Summary of example cases

Case	Objective functions	Scheduling functions	Interval width	Interval number	Nominal operating point
I	J_1	const	equal	4	base
II	J_1	const	var	4	base
III	J_2	const	var	var	base
IV	J_1	const	var	var	var
V	J_1	linear	var	var	var
VI	J_3	linear	var	var	var

Table 2 Comparison of optimization results

Case	J_1	J_2	J_3	No. intervals	P_0 (M, kft)
0	6.37	——	9.74	——	0.95, 20
I	3.09[a]	——	6.85	4	0.95, 20
II	2.51[a]	——	6.53	4	0.95, 20
III	3.24	5.24[a,b]	6.75	2	0.95, 20
IV	2.07[a]	——	7.02	5	0.4, 22
V	1.98[a,b]	——	5.47	3	0.7, 18.5
VI	2.44	——	5.16[a,b]	2	0.6, 15

[a] Denotes objective function used in each case.
[b] Denotes minimum for each objective function.

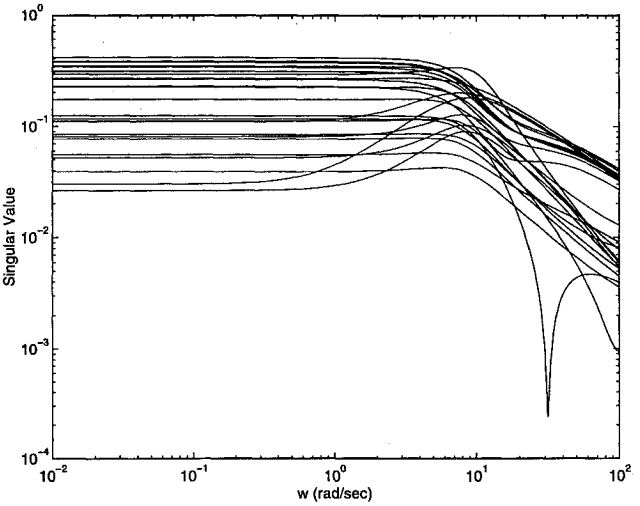

Fig. 2 Case 0 relative error.

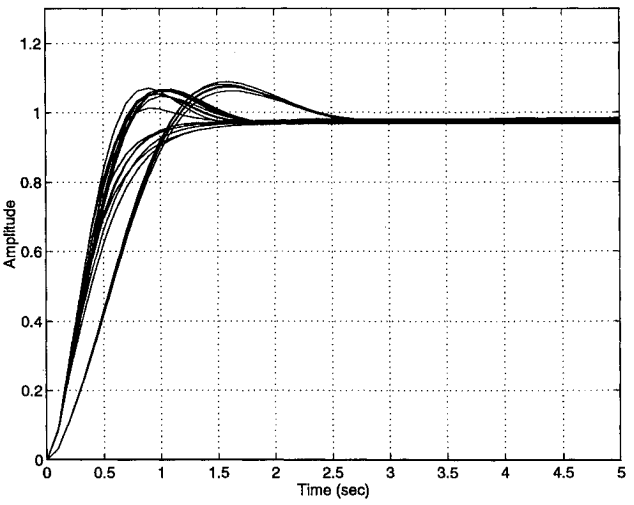

Fig. 3 Case 0 step response.

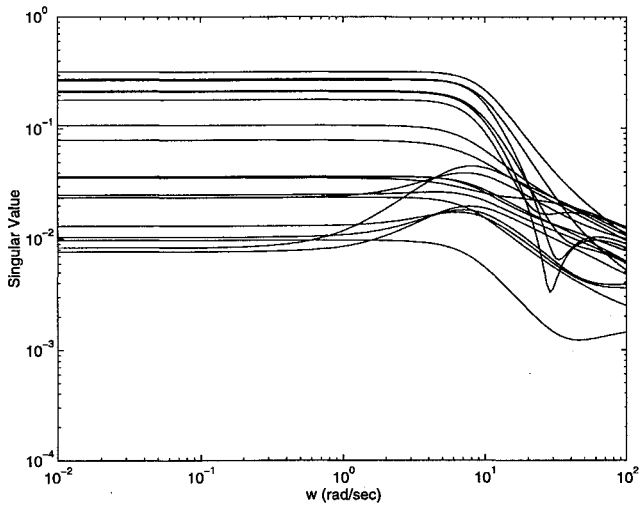

Fig. 4 Case V relative error.

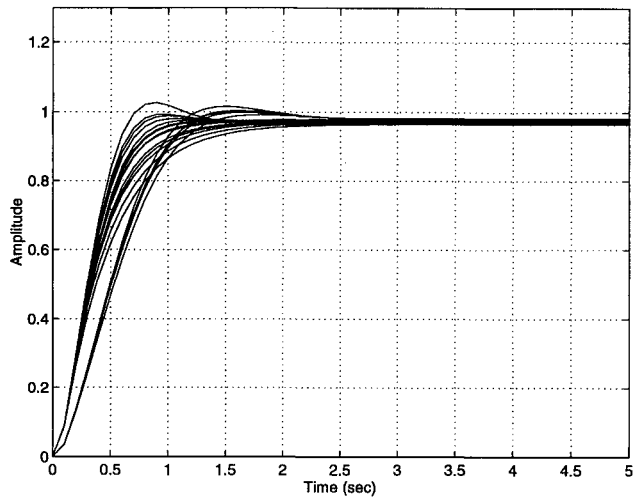

Fig. 7 Case VI step response.

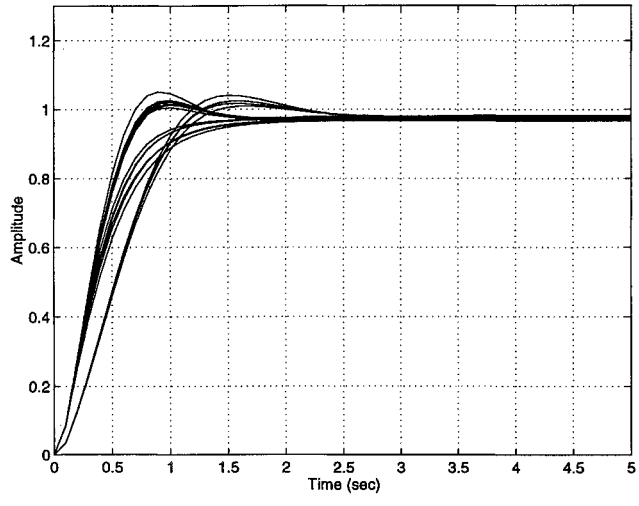

Fig. 5 Case V step response.

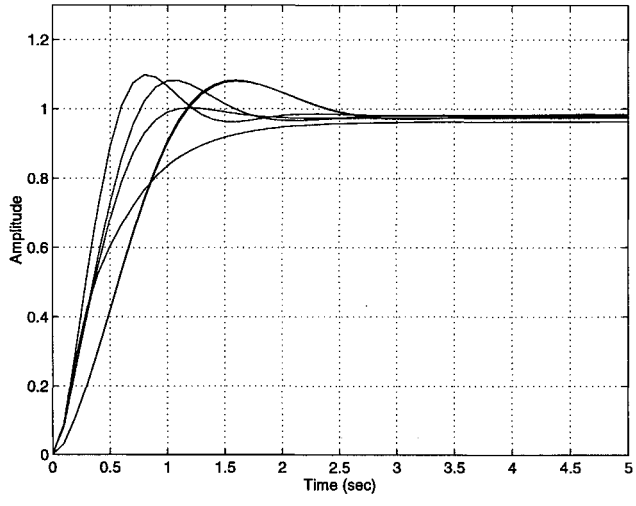

Fig. 8 Baseline validation step response.

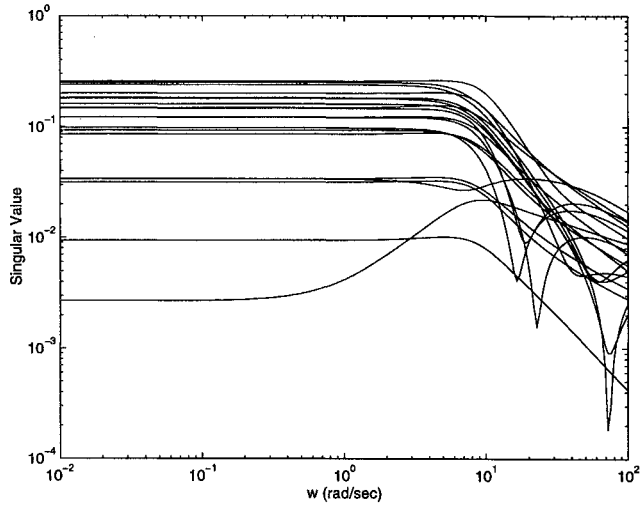

Fig. 6 Case VI relative error.

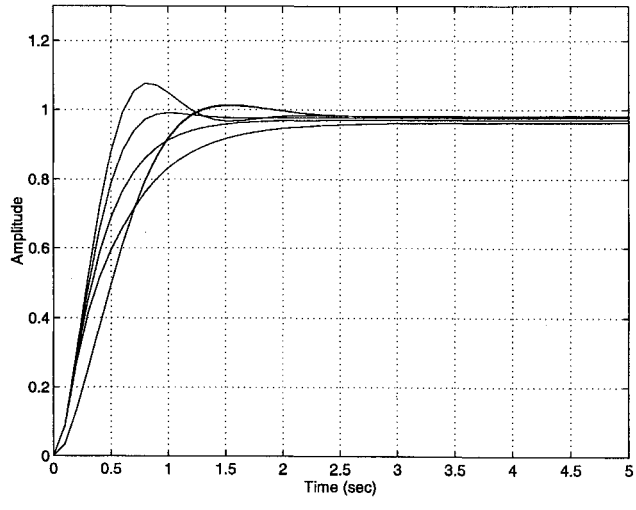

Fig. 9 Case VI validation step response.

case, but is still significant. Again, there is some increase in the time response error. The resulting schedule has narrower intervals at low $\bar{q}$ but is quite similar to the first case.

For case III, controller complexity is represented by the number of intervals used in the scheduling function. This case, then, shows the tradeoff of performance and complexity by allowing the number of intervals to vary between 2 and 9 and adding a penalty on the number of intervals used. Since the penalty function value ranges from 2 to 9 while the relative error is about 3, this represents a fairly strong penalty on the number of intervals, and the GA pushes the number of intervals to the minimum allowed. The cost is a modest increase in the relative error to just slightly more than half of the baseline value.

Next is case IV. As noted earlier, the choice of nominal operating point is a judgment and, hence, to some extent arbitrary. Since we are using GAs, adding the nominal operating point as another design variable is very simple. This additional degree of freedom allows the objective function to be reduced again, this time to about one third

of the baseline value. The GA selected Mach 0.4 and 22 kft as the nominal operating point, which is a striking change from the baseline selection. Since there was no penalty on the number of intervals, the GA increased it to five, but did not reach the maximum allowed.

Piecewise constant scheduling functions are very simple and easy to implement as table lookups, but are not particularly desirable because of their discontinuous behavior. To avoid discontinuities, we can use continuous, piecewise linear functions in case V. The resulting relative error is the lowest of all the cases, and there is also a reasonable reduction in the time error. The improvements are quite visible by comparing Fig. 2 with Fig. 4 and Fig. 3 with Fig. 5.

Despite considerable reduction in the relative error, there is not much change in the consistency of the time responses as measured by the integrated time error in the previous cases. Case VI shows how that concern can be addressed by including both time- and frequency-domain objectives. As one would expect, this results in the lowest combined objective function value, as well as the lowest time error, at a small cost in the relative error. The relative error graph in Fig. 6 looks very similar to case V in Fig. 4. On the other hand, the step response graph in Fig. 7 clearly shows more consistent responses as compared to Fig. 5.

After the initial design phase, the controller should be tested appropriately. For this example, the authors selected several additional operating points at which to check the controller performance. Six points (noted in Appendix A) were chosen to cover the extremes of the envelope and fill in some open spaces. Only the baseline and case VI controllers were evaluated. The summed relative error for the optimized controller is less than half that of the baseline, 1.10 vs 2.87. The improvement in step response consistency is evident by comparing Figs. 8 and 9. Since the optimized controller performs so well on these points, we have much increased confidence in its general performance.

As a final evaluation, we compared our results to a robustified version of the same generic controller. Robustification has been explored as an alternative way of dealing with plant variations. The idea is to select a single set of controller parameters to minimize variation in performance over the operating envelope, or in this case to minimize total relative error. Like our approach, it can result in a numerically difficult problem that may require a GA or related algorithm.[15] Robustification amounts to choosing a piecewise constant gain schedule with a single interval, and so was easy to implement in our analysis. One might expect that it would not be as good as a multiple interval schedule or a linear schedule over a single interval simply because it has fewer design degrees of freedom. In this case, the optimum robustified controller yielded a total relative error of $J_1 = 8.34$, and a corresponding combined error of $J_3 = 14.0$, both substantially worse than the various scheduled controllers.

VI. Conclusions

Direct optimization of gain scheduled controllers is clearly feasible using the proposed approach and produces controllers with significantly improved performance. The process allows the designer to explicitly define the desired objectives, including tradeoffs between complexity and performance and between time and frequency domains, and increases the design degrees of freedom available to the designer. GAs are well suited to this optimization problem and are robust with regard to the range of design variables and objective functions that may occur in this application. This approach should be a valuable addition to the controller designer's tool kit.

Appendix A: Operating Points

Table A1 lists the flight conditions used as design and validation points in this study and the corresponding plant dynamic matrices.

Table A1 Operating points and corresponding dynamic matrices

$n = 1$ $v = 0.3$ $h = 26$ $a = 25.2$	$A = \begin{bmatrix} -0.2296 & 0.9931 \\ 0.02436 & -0.2046 \end{bmatrix}$	$n = 14$ $v = 0.5$ $h = 10$ $a = 3.5$	$A = \begin{bmatrix} -0.8930 & 0.9852 \\ -4.1582 & -0.6873 \end{bmatrix}$
$n = 2$ $v = 0.5$ $h = 40$ $a = 16.8$	$A = \begin{bmatrix} -0.2423 & 0.9964 \\ -2.342 & -0.1737 \end{bmatrix}$	$n = 15$ $v = 0.6$ $h = 15$ $a = 2.9$	$A = \begin{bmatrix} -0.9181 & 0.9872 \\ -6.2419 & -0.6920 \end{bmatrix}$
$n = 3$ $v = 0.6$ $h = 30$ $a = 5.2$	$A = \begin{bmatrix} -0.5088 & 0.994 \\ -1.131 & -0.2804 \end{bmatrix}$	$n = 16$ $v = 0.7$ $h = 18.5$ $a = 2.4$	$A = \begin{bmatrix} -0.9920 & 0.9888 \\ -7.8450 & -0.7525 \end{bmatrix}$
$n = 4$ $v = 0.4$ $h = 6$ $a = 6.0$	$A = \begin{bmatrix} -0.8018 & 0.9847 \\ -1.521 & -0.5944 \end{bmatrix}$	$n = 17$ $v = 0.6$ $h = 2$ $a = 1.8$	$A = \begin{bmatrix} -1.4710 & 0.9808 \\ -11.5022 & -1.0846 \end{bmatrix}$
$n = 5$ $v = 0.7$ $h = 14$ $a = 2.6$	$A = \begin{bmatrix} -1.175 & 0.9871 \\ -8.458 & -0.8776 \end{bmatrix}$	$n = 18$ $v = 0.8$ $h = 14$ $a = 1.4$	$A = \begin{bmatrix} -1.4406 & 0.9868 \\ -14.2709 & -1.0645 \end{bmatrix}$
$n = 6$ $v = 0.8$ $h = 12$ $a = 1.9$	$A = \begin{bmatrix} -1.562 & 0.9862 \\ -14.94 & -1.132 \end{bmatrix}$	$n = 19$ $v = 0.9$ $h = 14$ $a = 1.2$	$A = \begin{bmatrix} -2.1163 & 0.9872 \\ -32.6459 & -1.1826 \end{bmatrix}$
$n = 7$ $v = 0.95$ $h = 20$ $a = 1.6$	$A = \begin{bmatrix} -1.905 & 0.9895 \\ -33.88 & -0.9872 \end{bmatrix}$	$n = 20$ $v = 0.95$ $h = 9$ $a = 0.9$	$A = \begin{bmatrix} -2.8375 & 0.9855 \\ -51.8325 & -1.4037 \end{bmatrix}$
$n = 8$ $v = 0.8$ $h = 10$ $a = 1.7$	$A = \begin{bmatrix} -1.675 & 0.9853 \\ -16.16 & -1.212 \end{bmatrix}$	$n = 21$ $v = 0.98$ $h = 40$ $a = 3.1$	$A = \begin{bmatrix} -0.7055 & 0.9949 \\ -17.5821 & -0.4583 \end{bmatrix}$
$n = 9$ $v = 0.8$ $h = 5$ $a = 1.5$	$A = \begin{bmatrix} -1.994 & 0.9828 \\ -19.44 & -1.427 \end{bmatrix}$	$n = 22$ $v = 0.99$ $h = 10$ $a = 1.0$	$A = \begin{bmatrix} -2.6317 & 0.9860 \\ -76.1833 & -1.3868 \end{bmatrix}$
$n = 10$ $v = 0.9$ $h = 10$ $a = 1.4$	$A = \begin{bmatrix} -2.452 & 0.9856 \\ -38.61 & -1.34 \end{bmatrix}$	$n = 23$ $v = 0.5$ $h = 20$ $a = 5.2$	$A = \begin{bmatrix} -0.6199 & 0.9900 \\ -1.8909 & -0.4433 \end{bmatrix}$
$n = 11$ $v = 0.85$ $h = 5$ $a = 1.4$	$A = \begin{bmatrix} -2.328 & 0.9831 \\ -30.44 & -1.493 \end{bmatrix}$	$n = 24$ $v = 0.3$ $h = 15$ $a = 12.2$	$A = \begin{bmatrix} -0.3664 & 0.9887 \\ 0.0321 & -0.3425 \end{bmatrix}$
$n = 12$ $v = 0.9$ $h = 5$ $a = 1.3$	$A = \begin{bmatrix} -2.911 & 0.9835 \\ -46.47 & -1.553 \end{bmatrix}$	$n = 25$ $v = 0.2$ $h = 1$ $a = 20.8$	$A = \begin{bmatrix} -0.3678 & 0.9843 \\ 0.3351 & -0.3022 \end{bmatrix}$
$n = 13$ $v = 0.4$ $h = 22$ $a = 8.7$	$A = \begin{bmatrix} -0.4285 & 0.9916 \\ -0.7473 & -0.3123 \end{bmatrix}$	$n = 26$ $v = 0.8$ $h = 1$ $a = 1.0$	$A = \begin{bmatrix} -2.2679 & 0.9803 \\ -22.8644 & -1.6265 \end{bmatrix}$

Points 1–12 are the 12 points used in the baseline design, 13–20 are the added design points, and 21–26 are the points used in the validation. Altitude h is given in kilofeet, angle of attack α in degrees, and velocity v in Mach.

Appendix B: Outer-Loop Controller

The outer-loop controller K_{outer} is a reduced-order μ-synthesis design with blending between designs for low and high angle of attack.

$$K_{\text{outer}} = K_{\text{low}}C + K_{\text{high}}(1 - C)$$

$$C = \begin{cases} 1 & \alpha < 12.5 \\ \dfrac{17.5 - \alpha}{5} & 12.5 \le \alpha \le 17.5 \\ 0 & \alpha > 17.5 \end{cases}$$

$$K_{\text{low}} = \frac{130(s + 602)(s + 3.84 \pm 6.25j)}{(s + 59.1 \pm 52j)(s + 0.0363)(s + 6.64)}$$

$$K_{\text{high}} = \frac{120(s + 2420)(s + 5.4 \pm 7.26j)}{(s + 71.8 \pm 40.5j)(s + 0.0723)(s + 2.05)}$$

References

[1]Rugh, W., "Analytical Framework for Gain Scheduling," *IEEE Control Systems Magazine*, Vol. 11, No. 1, 1991, pp. 79–84.

[2]Lawrence, D. A., and Rugh, W. J., "On a Stability Theorem for Nonlinear Systems with Slowly-Varying Inputs," *IEEE Transactions on Automatic Control*, Vol. 35, No. 7, 1990, pp. 860–864.

[3]Shamma, J., and Athans, M., "Guaranteed Properties of Gain Scheduled Control for Linear Parameter-Varying Plants," *Automatica*, Vol. 27, No. 3, 1991, pp. 559–564.

[4]Shamma, J., and Athans, M., "Gain Scheduling: Potential Hazards and Possible Remedies," *IEEE Control Systems Magazine*, Vol. 12, No. 3, 1992, pp. 101–107.

[5]Shamma, J., and Athans, M., "Analysis of Gain Scheduled Control for Nonlinear Plants," *IEEE Transactions on Automatic Control*, Vol. 35, No. 8, 1990, pp. 898–907.

[6]Lawrence, D. A., and Rugh, W. J., "Gain Scheduling Dynamic Linear Controllers for a Nonlinear Plant," *Proceedings of the 32nd IEEE Conference on Decision and Control*, Inst. of Electrical and Electronics Engineers, New York, 1993, pp. 1024–1029.

[7]Martin, R., "Gain Scheduling Optimization Method Using Genetic Algorithms," M.S. Thesis, Dept. of Aeronautics and Astronautics, U.S. Air Force Inst. of Technology, Wright–Patterson AFB, OH, Dec. 1994.

[8]Martin, R., and Kramer, S., "Gain Scheduling Optimization by Genetic Algorithms," *Proceedings of the 1995 American Control Conference*, Inst. of Electrical and Electronics Engineers, New York, 1995, pp. 3041, 3042.

[9]Dymec, A., "Design and Implementation of Parallel Genetic Algorithms for Solving Large Scale Optimization Problems," M.S. Thesis, Dept. of Electrical and Computer Engineering, U.S. Air Force Inst. of Technology, Wright–Patterson AFB, OH, March 1992.

[10]Michalewicz, Z., *Genetic Algorithms + Data Structures = Evolution Programs*, Springer–Verlag, New York, 1992.

[11]Bramlette, M., and Cusin, R., "Comparative Evaluation of Search Methods Applied to Parametric Design," *Proceedings of the Third International Conference on Genetic Algorithms*, Morgan Kaufmann, San Mateo, CA, 1989, pp. 213–218.

[12]Krishnakumar, K., and Goldberg, D., "Control System Optimization Using Genetic Algorithms," *Journal of Guidance, Control, and Dynamics*, Vol. 15, No. 3, 1992, pp. 735–739.

[13]McGregor, D., Odetayo, M., and Dasgupta, D., "Adaptive Control of a Dynamic System Using Genetic-Based Methods," *Proceedings of the 1992 IEEE Symposium on Intelligent Control*, Inst. of Electrical and Electronics Engineers, New York, 1992, pp. 521–525.

[14]Michalewicz, Z., Janikow, C., and Drawczyk, J., "Modified Genetic Algorithm of Optimal Control Problems," *Computers, Mathematics, and Applications*, Vol. 23, No. 12, 1992, pp. 83–94.

[15]Porter, B., and Hicks, D., "Genetic Robustification of Digital Model-Following Flight Control Systems," *Proceedings of the IEEE National Aerospace and Electronics Conference*, Inst. of Electrical and Electronics Engineers, New York, 1994, pp. 556–563.

[16]Porter, B., and Hicks, D., "Perfomance Measures in the Genetic Design of Digital Model-Following Flight Control Systems," *Proceedings of the IEEE National Aerospace and Electronics Conference*, Inst. of Electrical and Electronics Engineers, New York, 1994, pp. 564–570.

[17]Grefenstette, J., "Optimization of Control Parameters for Genetic Algorithms," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-16, No. 1, 1986, pp. 122–128.

[18]Adams, R., Buffington, J., Sparks, A., and Banda, S., "Introduction to Multivariable Flight Control System Design," U.S. Air Force Wright Lab., TR WL-TR-92-3110, Wright–Patterson AFB, OH, Oct. 1992.

[19]Martin, R., and Kramer, S., "Using the Nominal Operating Point as a Design Variable for Gain Scheduled Controllers," *Proceedings of the IEEE 1995 National Aerospace and Electronics Conference*, Inst. of Electrical and Electronics Engineers, New York, 1995, pp. 486–490.