Numerically Efficient Robustness Analysis of Trajectory Tracking for Nonlinear Systems

Jorge E. Tierno Richard M. Murray John C. Doyle
Irene M. Gregory
Division of Engineering and Applied Science
California Institute of Technology
Pasadena, California 91125

Submitted, AIAA Journal of Guidance, Control, and Dynamics.
November 29, 1995.

1 Introduction

The engineering motivation behind the theory of robust control is based on two unavoidable facts: first, all analysis and synthesis methods are based on the use of models. In most cases it is not practical to try the designs in prototypes at the early stages. Furthermore, critical control systems must have a reasonable expectation of good performance before they are implemented, since their failure can lead to significant losses. These factors force us to extensively analyze systems using models before we proceed to the implementation stage. The second fact is that models are, by nature, incomplete and inaccurate. They are incomplete because they are deduced from a finite observation of the system. They are inaccurate because the observation is noisy and, what is even more important, because we are trying to project a physical system into a particular class of mathematical models we believe to be representative (e. g., the class of linear, finite dimensional model).

Theoretical and computational tools for analysis and synthesis of robust controllers for linear systems are well developed in a variety of instances. Controllers generated with these tools can provide guaranteed performance in the presence of structured uncertainty, and the worst case disturbances for a given controller can be determined.

There are several possible choices of performance measures and uncertainty sets for which we can determine robustness. For linear time invariant (LTI) systems with structured uncertainty, analysis of robust performance can be reduced to computing a single scalar function of the system known as the structured singular value and denoted μ [8]. Although this function cannot be computed exactly, we are able to find computationally efficient algorithms to compute upper and lower bounds, such as the power algorithm for the μ lower bound [9, 15], without doing an explicit parameter search involving repeated simulation. This works because the system is linear and the performance and uncertainty descriptions are chosen so as to give computationally attractive solutions, even for large problems.

Analysis of nonlinear systems, on the other hand, has stayed mainly at the theoretical level. A large body of work has been devoted to extend the linear analysis techniques to nonlinear systems. As part of this work, conditions for robust stability and robust performance similar to the structured singular value have been established. Robustness for nonlinear systems was proved to be equivalent to the existence of solution to Hamilton-Jacobi equations [14] or nonlinear matrix inequalities [5]. However, computational methods to establish the existence of these solutions have not been developed to a level comparable to their linear counterpart (i. e., existence of solutions of Riccati equations and linear matrix inequalities).

The state of the art in industry still consists in obtaining lower bounds to the performance indices through extensive simulation or local optimization techniques. However these methods require large amounts of computation; standard optimization techniques fail even for small problems, and a search over parameter space exhibits exponential growth with the number of parameters. The methods actually used in industry share two main characteristics: performance specifications are made over a finite time horizon, and the interface between the analysis method and the system is a simulation.

The intention of the work in this paper is to extend the robustness analysis techniques of linear systems, and in particular the associated computational methods, to nonlinear systems. Given the diversity of nonlinear behavior, it is clear that this cannot be done in complete generality and still maintain the efficiency and usability of the methods. In this paper we will develop analysis methods for a specific nonlinear robust performance problem. We will show that an efficient numerical algorithm can be developed to compute a lower bound on the corresponding performance index. The problem, and the corresponding algorithm, share the characteristics of current industrial practice mentioned earlier.

The problem we will study is that of tracking a trajectory in the presence of noise and uncertainty. Many nonlinear analysis problems of engineering interest can be reduced to such a problem. A common example is an airplane performing an automatic change of altitude and heading. The pilot enters the new heading and altitude and the flight computer determines nominal commands to perform it. A second control loop maintains the airplane around the planned trajectory. The designer has in mind an appropriate path to be completed in a finite predetermined time, and designs the control system accordingly. Since the real system is not exactly the one used for the design, and since it is also subject to noise, the system will not follow the intended trajectory. The question of interest becomes: will the real trajectory, under the worst conditions possible, remain close enough to the nominal one in an appropriate norm? We will call this question the robust trajectory tracking problem. (We will give it a more formal definition in the next section.)

In this paper we develop a power algorithm to compute a lower bound on the performance index associated with the robust trajectory tracking problem (i. e., the distance from the actual to the nominal trajectory). This algorithm is similar in nature to the one developed for the structured singular value [8, 15], and has similar behavior. Since, as was the case for linear systems, the algorithm is not guaranteed to converge in general, its analysis is done empirically. We test this algorithm by applying it to simulations of real systems. We carry out several different performance tests on two different platforms: the Caltech ducted fan experiment and a simplified model of an F-16 jet fighter. The results of these tests are reported in Section 5. These results indicate that without significant additional computation, and avoiding computationally expensive parameter searches, a lower bound on the given performance index can be computed that gives more information on the worst case behavior of the system than the standard Monte Carlo procedures.

2 Problem Formulation

We will be concerned with the study of perturbations of systems around a prespecified nominal trajectory over a finite time horizon with initial time t_i and final time t_f. The performance specifications and all characterizations of the noise signals, under-modeled components, and uncertain parameters affecting the system will be given over this time interval.

We will restrict our study to nonlinear systems whose dynamics can be represented by a smooth function F of the states x, a set of signals u corresponding to external disturbances, a set of signals v corresponding to the effect of under-modeled components, a set of parameters δ, and a set of nominal commands U responsible for steering the nominal system along the nominal trajectory. The evolution of the system will thus be the solution to

$$\dot{x} = F(x, u, v, \delta, U, t),$$

with initial conditions $x = x_o$. The trajectory will be described by a set of coordinates Y, given by a smooth function g of the same variables

$$Y = G(x, u, v, \delta, U, t).$$

The nominal trajectory Y_n, will thus be given by

$$Y_n = G(x, 0, 0, \delta_n, U, t),$$

where δ_n are the nominal values for the parameters.

The signals v are not independent of the other signals, but the exact dependence is not known (hence the name under-modeled dynamics.) We know however that v is the image under a structured, norm bounded operator of a set of variables z, given as a smooth function of the other variables in the problem

$$z = H(x, u, v, \delta, U, t).$$

We will denote Δ the mapping from z to u, $\mathbf{\Delta}$ the class of operators with same block diagonal structure. Since we will analyze the system for a fixed nominal trajectory, we can incorporate the nominal commands into the functions F, G, and H, and define

$$\begin{aligned} f_U(x,u,v,\delta,t) &= F(x,u,v,\delta,U,t) \\ h_U(x,u,v,\delta,t) &= H(x,u,v,\delta,U,t). \end{aligned}$$

Since we are only interested in the error between the nominal trajectory and the actual one, we will also define

$$g_U(x,u,v,\delta,t) = G(x,u,v,\delta,U,t) - Y_n$$

Whenever no confusion is possible we will drop the subscript U. The structure of the operator Δ we will consider is block diagonal, and will thus induce a partition in the sets of signals v and z

$$\begin{aligned} v &= [v_1, v_2, \ldots, v_p] \\ z &= [z_1, z_2, \ldots, z_p]. \end{aligned}$$

Note that v and z are divided in the same number of blocks of signals, but correspondent blocks do not have to have the same number of signals. The size of the noise signals entering the problem will also require the introduction of a partition of the signal set u

$$u = [u_1, u_2, \ldots, u_m].$$

Figure 1 shows a diagram of the system interconnection. There are several similarities with the linear μ framework reviewed in the previous chapter, and some differences. The under-modeled dynamical components are feedback loops as is the case in linear systems. However the effect of the parameters on the system can be more general. The dynamics are not only nonlinear, but they are also autonomous: the system includes explicit functions of time.

We will measure the sizes of all signals in the two norm defined as usual over a finite time horizon

$$||a||_2 = \left[\frac{1}{t_f - t_i}\int_{t_i}^{t_f} a^t a \; dt\right]^{\frac{1}{2}}.$$

The performance index is given by the 2-norm of the error signal

$$J = ||y||_2.$$

The size of the noise signals will be specified for each block of signals in the partition given

$$||u_i||_2 = N_i \qquad\qquad i = 1, 2, \ldots, m.$$

All the information we have on the under-modeled dynamical components is that Δ is in $\mathbf{B\Delta}$. We will use the induced 2-norm as a measure of the size of Δ.

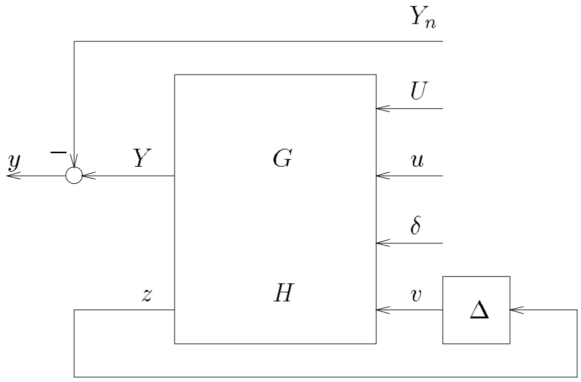

Figure 1: Uncertain system interconnection.

This restriction is then equivalent to imposing the following relations between the signals v and z

$$\|v_i\|_2 = \|z_i\|_2 \qquad i = 1, 2, \ldots, p.$$

We will allow the parameters δ to vary in closed intervals

$$d_i \leq \delta_i \leq D_i \qquad i = 1, 2, \ldots, r.$$

Finally we will also allow some or all of the initial conditions to vary in given closed intervals

$$c_i \leq x_i(t_i) \leq C_i \qquad i = 1, 2, \ldots, n.$$

The preceding performance, noise, and uncertainty descriptions given in this section can be summarized as a constrained optimization problem. Since we only use one norm will drop the subscript 2 from now on to keep the notation unencumbered. We now state the following

Problem 1 (Robust Trajectory Tracking) *Given an uncertain nonlinear system, driven by the equation*

$$\dot{x} = f_U(x, u, v, \delta, t)$$

and the nominal command signal U, with initial conditions satisfying

$$c_i \leq x_i(t_i) \leq C_i \qquad i = 1, 2, \ldots, n,$$

what is the maximum value of the norm of the error signal

$$\|y\| = \|g_U(x, u, v, \delta, t)\|$$

subject to the constraints

$$\begin{array}{rcll} \|u_i\|_2 & = & N_i & \qquad i = 1, 2, \ldots, m \\ \|v_i\|_2 & = & \|z_i\|_2 & \qquad i = 1, 2, \ldots, p \\ d_i \leq & \delta_i & \leq D_i & \qquad i = 1, 2, \ldots, r \quad ? \end{array}$$

This problem may seem too specific to be of any consequence to real life applications. When analyzing linear systems, we use different kinds of multiplicative weighting functions (weights) to impose a frequency content to noise signals, to determine the bandwidth of an uncertain operator, or to determine the region in frequency domain over which we would like to impose the performance specification. The use of these weights adds a flexibility to the linear analysis techniques, and is in general fundamental to establish the validity of the method. With similar techniques we can reduce many interesting application problems to a particular instance of Problem 1.

3 Necessary Conditions

The first step in deriving a power algorithm is establishing conditions characterizing local maxima. First order conditions for extrema of dynamical systems have been developed for different optimization indices and signal constraints. (See for example [6] or [2].) We derive the necessary conditions for a local maximum of Problem 1 from the Euler-Lagrange optimization setup. In this section we first review briefly the standard Euler-Lagrange optimization framework for dynamical systems and we show how the robust trajectory tracking problem reduces to an instance of the general Euler-Lagrange problem.

The following theorem summarizes the Euler-Lagrange framework:

Theorem 1 [2] *For a dynamical system described by the equations:*

$$\dot{x} = f(x,u,t) \quad x(0) \text{ given}, t_i \leq t \leq t_f$$

a performance index of the form

$$J = \int_{t_i}^{t_f} L(x,u,t)\, dt$$

and restrictions on the final state

$$G(x(t_f)) = c$$

if the signal u_o achieves an extremum of J, then there exists a vector of constants ζ and a solution to the two point boundary value problem

$$\begin{aligned} \dot{x} &= f(x,u,t) \\ \dot{\lambda} &= -\left(\frac{\partial f}{\partial x}\right)^t \lambda - \left(\frac{\partial L}{\partial x}\right)^t && (1) \\ 0 &= \left(\frac{\partial L}{\partial u}\right) + \left(\frac{\partial f}{\partial u}\right)^t \lambda && (2) \end{aligned}$$

with boundary conditions:

$$\begin{aligned} x(0) & \quad \text{given} \\ \lambda(t_f) &= \left(\frac{\partial G}{\partial x(t_f)}\right)^t \zeta. \end{aligned}$$

Furthermore, if these conditions are met we will have

$$\lambda(t_i) = \frac{\partial J}{\partial x(0)}. \tag{3}$$

This theorem states necessary conditions for a set of signals being a first order extremum of the performance index, in terms of the existence of a solution of an associated adjoint system (Equation 1. Furthermore, the solutions of the original and adjoint system verify the constraint equation 2. We will see that this constraint can be interpreted as an alignment condition between the inputs of the original nonlinear system and a set of outputs of the adjoint one. This structure will be exploited when we develop the power algorithm.

3.1 Reduction to the Euler-Lagrange setup

To use the Euler-Lagrange necessary conditions in the robust trajectory tracking problem, we have to write the performance index, noise signals, uncertain parameters and under-modeled dynamical components of the system as constraints compatible with the hypothesis of Theorem 1.

Performance index

The performance index we described is very naturally written in the form required by Theorem 1. Letting

$$L = \frac{1}{2} y^t y,$$

then optimizing $||y||$ is equivalent to optimizing

$$J = \int_{t_i}^{t_f} L \, dt =: \frac{1}{2} ||y||^2.$$

Noise signals

The only constraints allowed by Theorem 1 are in the final values of states. The norm restrictions for the noise signals and the uncertain operator have then to be imposed through final conditions of additional states, created for that purpose. We will describe the case for one noise signal only. However the generalization to several signals is obtained simply by repeating the single signal case. To impose the 2-norm condition on the noise signals, we add to the system a new state named x_u governed by the differential equation

$$\dot{x}_u = \frac{1}{2} u^t u \qquad\qquad x_u(t_i) = 0.$$

Then $||u|| = N$ if and only if $x_u(t_f) = \frac{1}{2} N$.

Under-modeled dynamical component

The under-modeled dynamical block is characterized by a constraint on the norms of its input signals v and output signals z

$$\|v\| = \|z\|.$$

To impose this equality we add to the system a state x_Δ, governed by the differential equation

$$\dot{x}_\Delta = \frac{1}{2}(z^t z - v^t v) \qquad\qquad x_\Delta(t_f) = 0.$$

Then $\|v\| = \|z\|$ if and only if $x_\Delta(t_f) = 0$.

3.2 Uncertain parameters and uncertain initial conditions

Optimality with respect to uncertain initial conditions or uncertain parameters will be established through the gradient condition given by Equation (3). To treat parameters as initial conditions we create a state that tracks the parameter δ. Let x_δ follow the equation

$$\dot{x}_\delta = 0 \qquad\qquad x_\delta(t_i) = \delta.$$

At a local maximum, either this derivative of the performance index with respect to the value of the parameter or the state initial condition is zero, or it is negative and the parameter is at the lower end of the interval or it is positive and the parameter is at the higher end of the interval.

3.3 Summary

Summarizing, the robust trajectory tracking problem is equivalent to optimizing the performance index

$$J = \int_{t_i}^{t_f} L\, dt = \frac{1}{2}\|y\|^2$$

for the system verifying the differential equation

$$\begin{array}{rcll}
\dot{x} &=& f(x, u, v, x_\delta) & \text{(Dynamics)} \\
\dot{x}_u &=& \frac{1}{2}u^t u & \text{(Noise constraint)} \\
\dot{x}_\Delta &=& \frac{1}{2}(z^t z - v^t v) & \text{(Uncertainty constraint)} \\
\dot{x}_\delta &=& 0, & \text{(Uncertain parameter)}
\end{array}$$

where

$$\begin{array}{rcll}
y &=& g(x, u, v, x_\delta) & \text{(Performance output)} \\
z &=& h(x, u, v, x_\delta) & \text{(Uncertainty output)}
\end{array}$$

with given initial conditions

$$x(t_i) = x_0,\ x_u(t_i) = 0,\ x_\Delta(t_i) = 0,\ x_\delta(t_i) = \delta$$

and final conditions

$$x_u(t_f) = \frac{1}{2}, \; x_\Delta(t_f) = 0.$$

This problem is in the form required by Theorem (1). So a set of signals u, v, and a parameter δ achieve the worst case value of the performance index J only if there exists λ_x, λ_u, λ_Δ, λ_δ, verifying

$$\begin{array}{rcl} \dot{\lambda} & = & -\left(\frac{\partial f}{\partial x}\right)^t \lambda - \left(\frac{\partial h}{\partial x}\right)^t z\lambda_\Delta - \left(\frac{\partial g}{\partial x}\right)^t y \\ \dot{\lambda}_\delta & = & -\left(\frac{\partial f}{\partial \delta}\right)^t \lambda - \left(\frac{\partial h}{\partial \delta}\right)^t z\lambda_\Delta - \left(\frac{\partial g}{\partial \delta}\right)^t y \\ \dot{\lambda}_u & = & 0 \\ \dot{\lambda}_\Delta & = & 0, \end{array} \tag{4}$$

with final state conditions

$$\begin{array}{rcl} \lambda(t_f) & = & 0 \\ \lambda_\delta(t_f) & = & 0, \end{array}$$

verifying the following alignment conditions

$$\begin{array}{rcl} \left(\frac{\partial f}{\partial u}\right)^t \lambda + u\lambda_u + \left(\frac{\partial h}{\partial u}\right)^t z\lambda_\Delta + \left(\frac{\partial g}{\partial u}\right)^t y & = & 0 \\ \left(\frac{\partial f}{\partial v}\right)^t \lambda + \left(\left(\frac{\partial h}{\partial v}\right)^t z - v\right)\lambda_\Delta + \left(\frac{\partial g}{\partial v}\right)^t y & = & 0, \end{array} \tag{5}$$

and such that the initial state verifies

$$\lambda_\delta(t_i) = 0, \text{ or } \left\{ \begin{array}{c} \delta = -1 \\ \text{and} \\ \lambda_\delta(t_i) < 0 \end{array} \right. \text{ or } \left\{ \begin{array}{c} \delta = 1 \\ \text{and} \\ \lambda_\delta(t_i) > 0. \end{array} \right. \tag{6}$$

The nature of the performance index chosen, and the fact that none of the functions f, g, or h depend explicitly on the additional states x_u and x_Δ give Equations (4) and (5) a particular structure. The states λ_u and λ_Δ will be constants. And the choice of the 2-norm as a performance measure, is reflected in the fact that the adjoint system dynamics are driven by the performance output of the system. In the following sections we will make this structure clearer by introducing convenient notation, and we will show how a natural power like iteration can be derived from it.

3.4 Dynamical systems interpretation

For a given trajectory of the original nonlinear system, define the time varying matrices

$$A(t) = \begin{bmatrix} \left(\frac{\partial f}{\partial x}\right)^t & 0 \\ \left(\frac{\partial f}{\partial \delta}\right)^t & 0 \end{bmatrix} \qquad B(t) = \begin{bmatrix} \left(\frac{\partial g}{\partial x}\right)^t & \left(\frac{\partial h}{\partial x}\right)^t \\ \left(\frac{\partial g}{\partial \delta}\right)^t & \left(\frac{\partial h}{\partial \delta}\right)^t \end{bmatrix}$$

$$C(t) = \begin{bmatrix} \left(\frac{\partial f}{\partial u}\right)^t & 0 \\ \left(\frac{\partial f}{\partial v}\right)^t & 0 \end{bmatrix} \qquad D(t) = \begin{bmatrix} \left(\frac{\partial g}{\partial u}\right)^t & \left(\frac{\partial h}{\partial u}\right)^t \\ \left(\frac{\partial g}{\partial v}\right)^t & \left(\frac{\partial h}{\partial v}\right)^t \end{bmatrix},$$

and consider the dynamical system driven by the equations

$$\begin{aligned} -\dot{\Lambda} &= A\Lambda + B \begin{bmatrix} \nu \\ \xi \end{bmatrix} \\ \begin{bmatrix} -\gamma \\ \kappa \end{bmatrix} &= C\Lambda + D \begin{bmatrix} \nu \\ \xi \end{bmatrix}, \end{aligned} \tag{7}$$

with zero final conditions. Then if the signals u and v and the value of the parameter δ achieve a local maximum, there exist constants λ_u and λ_Δ, such that the following alignment conditions are verified between inputs and outputs of the direct and adjoint dynamical systems

$$\begin{aligned} \nu &= y \\ \xi &= \lambda_\Delta z \end{aligned} \tag{8}$$

and

$$\begin{aligned} \lambda_u u &= \gamma \\ \lambda_\Delta v &= \kappa. \end{aligned} \tag{9}$$

Equation (6) states that at an optimum, either the derivative of the performance index with respect to the value of the parameter is zero, or it is negative and the parameter is at the lower end of the interval or it is positive and the parameter is at the higher end of the interval.

This interconnection structure is represented in Figure 2. Note that although the two point boundary value problem has both initial and final state conditions, in this case they separate into two sets. The initial conditions are imposed on the states of the original nonlinear system and the final conditions are imposed on the states of the adjoint system. It is this particular structure of the two point boundary value problem, that will allow us to develop a power algorithm to solve it.

4 A Power Algorithm

Power algorithms have been successfully used to compute among other things, fixed points and local maximums of nonlinear functions and eigenvalues and singular values of matrices. The main advantage of power algorithms, when compared to other iterative methods, is the simplicity of each iteration. Iterations usually consist of little more than a function evaluation. In the case of

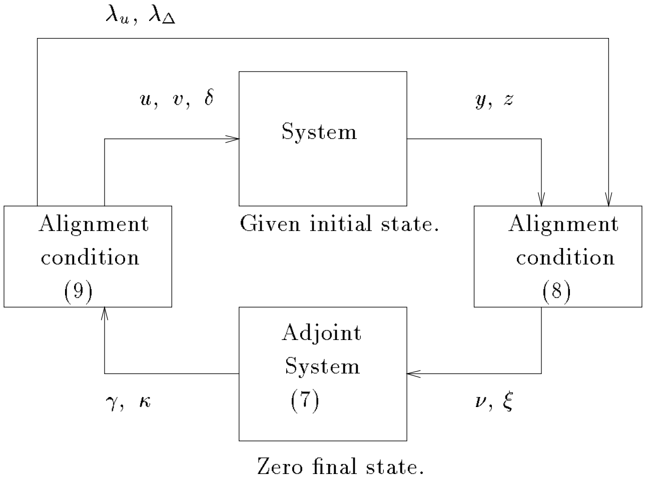

Figure 2: Dynamical systems interpretation of the equilibrium conditions.

eigenvalue computations, iterations usually require simply computing a matrix vector product. Another important characteristic of power methods is their tendency to explore a large region of the search space in their early stages. When applied to computing lower bounds for the linear performance index μ, power algorithms have proven to be fast and reliable. The robust trajectory tracking problem is very similar in nature to the structured singular value problem. (When applied to linear systems the robust trajectory tracking problem actually reduces to a special case of μ [12].) This motivates the development of a power like algorithm to compute a lower bound for Problem 1. In this section we will first give a qualitative description of power algorithms when applied to matrix problems; we will briefly describe the application of the standard μ power algorithm to linear systems over a finite time horizon; finally we will show how these ideas can be extended in an elegant way to the general nonlinear problem.

4.1 Heuristics behind power algorithms for linear matrices

Power algorithms are normally used to compute eigenvalues and singular values of matrices. Suppose that the matrix M is diagonalizable, and that its largest eigenvalue is unique. Starting from a random point ν_0 the power algorithm produces a sequence as follows [4]:

$$\begin{aligned} z^{(k)} &:= M\nu^{(k-1)} \\ \lambda^{(k)} &:= \|z^{(k)}\|_2 \\ \nu^{(k)} &:= z^{(k)}/\lambda^{(k)}. \end{aligned}$$

Let x_i be the eigenvectors of the matrix M. If ν_0 has a component along the maximum eigenvalue direction

$$\nu^{(0)} = a_1 x_1 + \sum_{i=2}^{n} a_i x_i$$

then it follows that

$$M^k \nu^{(0)} = a_1 \lambda_1^k \left[x_1 + \sum_{i=2}^{n} \frac{a_i}{a_1} \left(\frac{\lambda_i}{\lambda_1} \right)^k x_j \right].$$

If the terms

$$\frac{\lambda_i}{\lambda_1}$$

are sufficiently small, then the component of $\nu^{(0)}$ along the maximum eigenvector direction is amplified, while the others are contracted, and the procedure converges to the maximum eigenvalue. Since singular values of M are eigenvectors of M^*M the same procedure can be used to compute the maximum singular value. In this case we will have to perform alternating power steps, by M and M^*.

Power type algorithms are naturally suited for searching for directions of maximum gain. This fact makes them appealing to the computation of performance measures in disturbance rejection settings. Computing the structured singular value, for example implies computing the largest of the spectral radii of the matrices in a set. There are two search directions: one within the set of matrices to find the one that has the largest spectral radius, and one for a given matrix to find its eigen-direction of maximum gain. The power algorithm for μ does both searches simultaneously. From a qualitative point of view, the power steps, alternating multiplications by M and M^*, help look for directions of maximum gain of the particular matrix under study. The alignment conditions forced between power steps perform the search in the matrix set. A discrete time linear system, considered over a finite time horizon, can be represented by the matrix of the map between inputs and outputs. Many robust performance questions asked of this type of systems can be solved by computing the structured singular value of the map matrix with respect to an adequate structure [12]. It is interesting to note that in this case the power step with respect to M is taken by integrating the difference equations of the system forwards in time, and the power step by M^* is computed by simulating an adjoint dynamical system backwards in time. The succession of simulations of linear systems amplifies the component the chosen input has along the direction of maximum gain, while the operations carried out in between power steps look for the system in the class with maximum gain.

Conceptually, all the operations necessary to carry out this procedure are still well defined when the system is nonlinear. The substitutions needed to convert one case into the other are natural. Multiplication by the system matrix is equivalent to integration of the equations of motion. Multiplication by the adjoint matrix is equivalent to integration of the transposed linearized system. Although numerically harder, the operations are not conceptually different than in the linear case. And in principle, the heuristics explaining why this type of algorithm is efficient at finding directions of maximum gain still hold even when the system is nonlinear. In the next section we will prove that these substitutions are the adequate ones, and from them we will develop a power algorithm to solve the nonlinear robust trajectory tracking problem.

4.2 Power algorithm for nonlinear systems

Earlier in this section we showed how the first order necessary conditions for robust performance could be interpreted as an interconnection of dynamical systems. Four maps can be identified in the diagram of Figure 2. The first map integrates the equations of motion of the original system, with the given initial conditions and for a set of input signals u, v. The constants λ_u and λ_Δ are not changed. (These are included in the definition of the map to simplify the final expression.)

$$\Phi : (u(t), v(t), \delta, \lambda_u, \lambda_\Delta) \mapsto (y(t), z(t), \delta, \lambda_u, \lambda_\Delta).$$

The position of the map Φ in the system interconnection corresponding to the necessary conditions is shown in Figure 3.

The second map generates the input signals for the adjoint system by forcing the alignment conditions in Equation (8).

$$\Pi : (y(t), z(t), \delta, \lambda_u, \lambda_\Delta) \mapsto (y(t), \lambda_\Delta z(t), \lambda_u, \lambda_\Delta).$$

Figure 4 indicates the position of this map in the general interconnection.

The third map in the sequence will integrate the equations for the adjoint system along the current trajectory with the given inputs.

$$\Psi : (\nu, \xi, \lambda_u, \lambda_\Delta) \mapsto (\gamma, \kappa, \lambda_u, \lambda_\Delta)$$

The final map we will define computes new inputs for the original system by using the alignment condition in Equation (9), and will also compute new values for δ, λ_u, λ_Δ. There is more than one way of carrying out this evaluation. We propose here one possibility

$$\begin{aligned}
\Theta \quad &: \quad (\gamma, \kappa, \lambda_u, \lambda_\Delta) \mapsto (u(t), v(t), \delta, \lambda_u, \lambda_\Delta) \\
\lambda_u \quad &:= \quad \frac{||\gamma||}{U_e} \frac{\gamma \cdot u}{||\gamma|| U_e} \\
u \quad &:= \quad \frac{\gamma}{\lambda_u} \\
\lambda_\Delta \quad &:= \quad \frac{||\kappa||}{||z||} \frac{\kappa \cdot v}{||\kappa|| ||v||} \\
v \quad &:= \quad \frac{\kappa}{\lambda_\Delta}.
\end{aligned}$$

and δ is computed as follows:

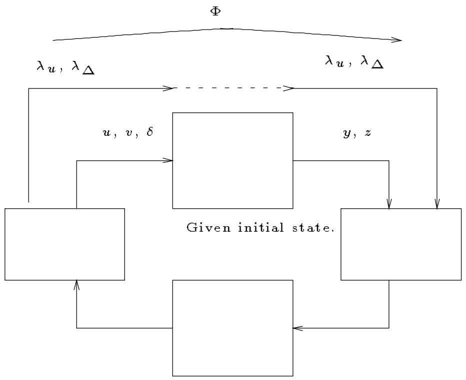

Figure 3: Φ operator: Integration of system equations.

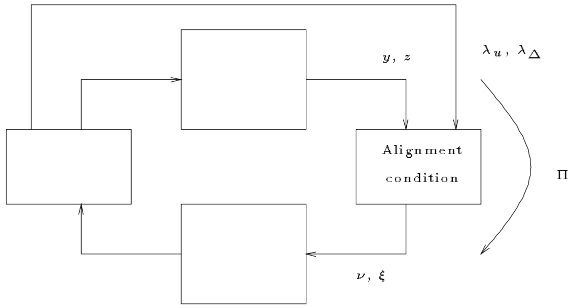

Figure 4: Π operator: Forcing the alignment conditions.

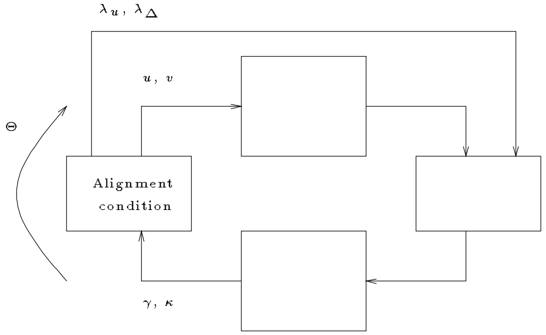

Figure 5: Θ operator: Forcing alignment conditions.

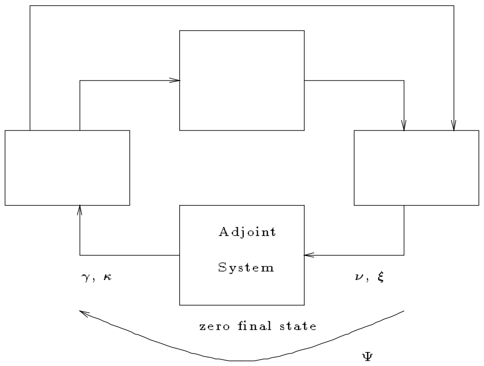

Figure 6: Ψ operator: Integration of adjoint system equations.

$$
\begin{aligned}
\chi &:= \delta + \lambda_\delta(t_i) \\
\delta &:= \begin{cases} -1 & \chi < -1 \\ \chi & -1 \le \chi \le 1 \\ 1 & \chi > 1. \end{cases}
\end{aligned}
$$

Figures 6 and 5 show these maps relative to the necessary conditions interconnection.

The mappings Φ and Ψ are, as described in the previous section, the nonlinear equivalent of power steps or matrix vector multiplications. The operations carried out in the mappings Π and Θ, impose the alignment conditions for optimality. The diagram in Figure 2 commutes when the signals u and v and the constants δ, λ_u, and λ_Δ are the ones corresponding to an extremum point. In other words, by going once around the diagram following the maps Φ, Π, Ψ, and Θ we should return to the starting point. This fact is formalized in the following

Lemma 2 *The signals u and v, and the parameter δ verify the optimality conditions if and only if there exists λ_u, λ_Δ such that*

$$(u, v, \delta, \lambda_u, \lambda_\Delta)$$

is a fixed point of the composition $\Theta \circ \Psi \circ \Pi \circ \Phi$.

We can now use an iterative algorithm to search for the fixed points of this composition. The standard form of such an algorithm is given by the pseudocode in Table 1.

$x^{(1)}$`:=random;`
repeat
$\quad x^{(i+1)} := \Theta \circ \Psi \circ \Pi \circ \Phi(x^{(i)})$;
until $(|x^{(i+1)} - x^{(i)}| < \epsilon |x^{(i)}|)$

Table 1: Power algorithm pseudo-code.

If the algorithm converges, it converges to a fixed point of the composition $\Theta \circ \Psi \circ \Pi \circ \Phi$ and thus, according to Lemma 2, to a set of signals that meet the necessary conditions for a critical point.

Remarks: In order to prove convergence we would have to prove that the composition $\Theta \circ \Psi \circ \Pi \circ \Phi$ is a contraction around local maximums. We can only prove this under very limiting conditions. The standard power algorithm for the lower bound of μ has not been proven to be stable for a general uncertainty structure, and is in fact known to be unstable in some cases [7]. Hence the evaluation of the algorithm will have to be done empirically.

The algorithm developed has many of the characteristics of the power algorithm for linear systems. In particular, if the system is linear, the adjoint system is linear time invariant, and in this case the algorithm reduces to the standard power iteration alternating multiplication by M and M^*.

The iteration presented is not the only one possible. In particular it is not necessary to update the adjoint system every iteration. The adjoint can be computed once every n iterations for example. The effect of this modification depends on the characteristics of the nonlinear dynamics, the size of the disturbances and the particular trajectory under consideration. It could in principle be harmful or beneficial. Since the behavior of nonlinear system is a lot more diverse than in the linear case, it is important to tailor the analysis tools to the particular problem at hand. An advantage of the power algorithm is that due to its simplicity, it easily accepts modifications that adapt it to the application being considered. Given the wide diversity in the behavior of nonlinear systems, it is fair to say that no one optimization algorithm will work always. This fact makes comparison between different algorithms hard. It also makes it difficult to prove that any given algorithm is suitable for a general purpose. Evaluation of the performance of the algorithms has to be done on a case by case basis. Nonlinear systems however can still be classified in families of problems. All aircraft models for example are very similar, and variations in the possible behaviors are more of a quantitative than a qualitative nature. Evaluation of the algorithm can be done on test cases for classes of problems. These examples have to be chosen so as to contain all the important characteristics of their class.

5 Numerical Examples

In this chapter we present the results obtained from trying the power algorithm presented in 4 on two different systems for several different disturbance and performance specifications. The systems are nonlinear, parts of the models are obtained from first principles (equations of motion of rigid bodies), and parts of the models are obtained through measurements and implemented with look-up tables. The models include command and rate saturations. These systems, and the performance problems we set up for them, have many of the characteristics of typical aircraft (or other types of vehicles) applications.

The results obtained are compared with different alternative approaches to solving the problems. These results prove the viability of the algorithm. Further evaluation of its behavior and its applicability to real life problems can only be obtained through extensive use in industrial applications.

This chapter will be divided in two sections, corresponding to each of the systems analyzed. A brief description of the dynamics of the system is included at the beginning of each section, followed by the different problems solved and the results obtained.

5.1 The Caltech Experimental Ducted Fan

The first example we will treat tests the noise rejection capabilities of a full state feedback LQR design for an experimental ducted fan platform (Figure 7).

Also, information on the experimental setup used, including copies of the simulation files can be obtained from `http://avalon.caltech.edu/~dfan`. The

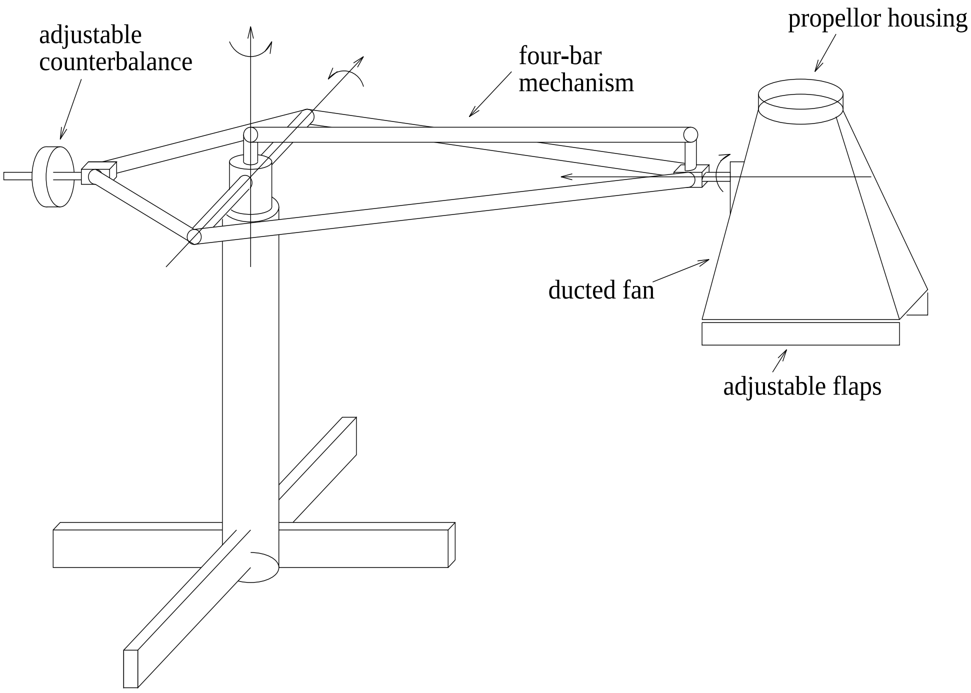

Figure 7: Overview of the experimental setup.

experimental vehicle consists of a relatively simple ducted fan aircraft that can provide two dimensional vectored and reverse thrust. The aircraft is bolted to a rotating arm, which limits its motion to three degrees of freedom: one rotational and two translational, approximately on the surface of a sphere defined by the arm. With this geometry, the ducted fan is completely controllable with just the vectored thrust. The ducted fan itself is a wooden duct powered by a variable speed electric motor driving a propeller. A detachable flap assembly, mounted at the end of the duct gives the fan vectored and reverse thrust. The setup and its nonlinear model are described in more detail in [3].

We consider a controller whose objective is to make the vehicle track a step in vertical position, i. e., we want to move it from hover at zero altitude to hover at another pre-specified altitude. The performance measure for the disturbance rejection problem will be the 2-norm of the distance from the nominal to the actual trajectory in the y position of the vehicle from beginning to end of the maneuver. A time domain weight multiplies the error signal to capture the fact that we consider the error at the final time to be more important than at the beginning. We will have three disturbances entering as torques along the three axes of rotation. The 2-norm of each of this noise signals is set to be 0.006 Nm over the 9 second time horizon (i. e., the rms value of the torque is

0.006 Nm). The uncertain block tries to capture the unmodeled dynamics of the electrical motor that is driving the fan. The input of the uncertain block is the command signal given to the motor filter through a high pass filter. The uncertain parameter represents the application point of the reaction force on the vehicle, since this point changes with, among other factors the position of the flaps. This distance is considered to be 0.25m + δ with $|\delta| < 0.1$m.

The maneuver specifies that we start at hover, i.e. with zero velocity in all directions, and end at hover. Furthermore, it is not only important to reach the desired height, we also want to keep the x position constant. In order to impose these restrictions we will use as performance signals the error in all the six state variables for the vehicle. (These are the mechanical states. We are not considering the states added by the controller and the different filters.) All other specifications remained the same.

After 30 iterations the algorithm converged to within 2 percent, and reports a value of $J = 0.411$. The algorithm reached a point within a few percent of the maximum reported after 10 iterations. We believe that this is due to the fact that there exists a subspace of signals for which the value of the optimization index is fairly constant. The algorithm then rotates in this subspace. The signals change, but the optimization index remains within small margins. This is a characteristic of power algorithms. A similar situation can happen when using power algorithms to compute the maximum eigenvalue of a matrix. If the corresponding eigenspace is not of dimension 1, the algorithm will converge to the value of the spectral radius, but not necessarily to one eigenvector. We compared our algorithm in this case with the one developed in [10]. Each iteration of this algorithm is more expensive than the one carried out by the power algorithm. After 500 iterations the algorithm in [10] reported a value for the local maximum of $J = 0.431$. After 30 iterations it reported a value of $J = 0.165$. The comparison between power type and gradient based algorithms shown here is very similar to the one observed in the linear case [7]. The latter kind of algorithms requires a much larger number of more expensive iterations to give a similar answer than the former one.

Although gradient methods provides a more accurate answer, it requires a far larger amount of time to do so. In practical applications the tradeoff of 5 percent in accuracy for reducing the number of iterations by 10 is reasonable, and in many cases necessary.

Figures 8 and 9 compare the disturbance signals obtained by the gradient and power algorithms respectively. Their similarity confirms that both algorithms converged to the same local extremum.

5.2 F-16 Maneuver

We want to determine whether the algorithm is suitable for aerospace applications. As a first step, the algorithm's ability to handle a model that includes a number of nonlinear equations and tabular data with a relatively high number of parameters,all characteristic of a typical aircraft, must be ascertained.

The aircraft used in this example application is an F16. The aerodynamic

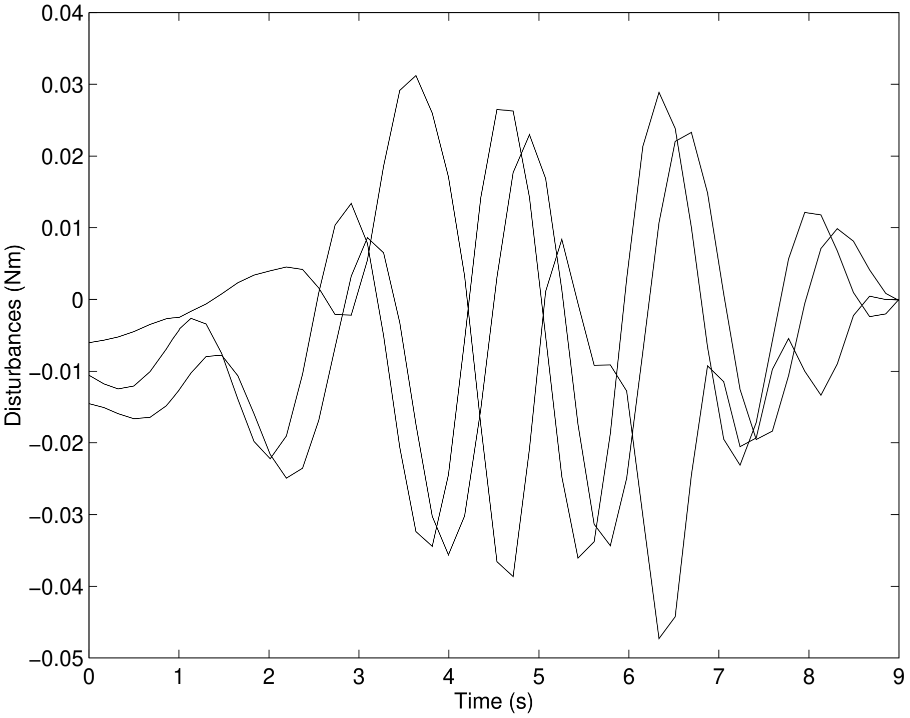

Figure 8: Worst case signals: gradient algorithm.

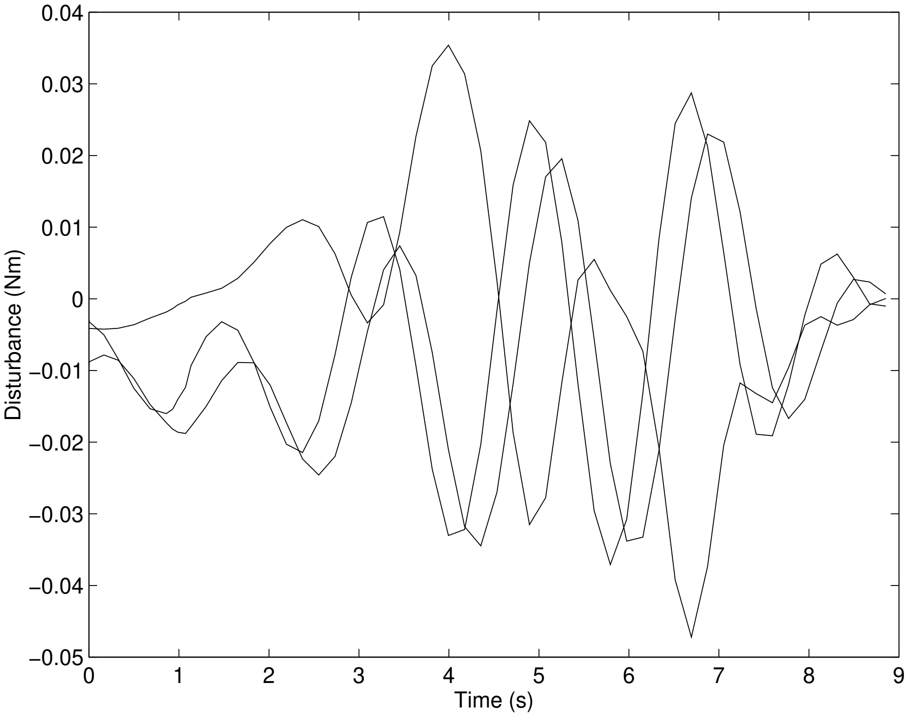

Figure 9: Worst case signals: power algorithm.

model is a reduced version of the full model obtained in wind tunnel tests at NASA Langley in 1979 [11]. It consists of tabular data with typical interpolation routines and nonlinear equations of motion. The engine model is that of an after-burning turbofan. The airplane model utilized in this application is defined for speed range of up to Mach 0.6 and angle of attack interval between -10 and 45 degrees. The model includes four traditional controls (elevator, aileron, rudder, and throttle) and thirteen states (velocity vector, attitude angles, angular velocities, navigational position, altitude, and engine power). Furthermore, the aerodynamic coefficients are build up in a traditional way and the equations of motion are full nonlinear flat Earth equations.

The trajectory considered is a constant climb, constant g coordinated turn. The aircraft initiates the maneuver at 10,000 ft flying at 500 ft/s. The F16 is then trimmed to climb at 50 ft/s while maintaining a 4.5 g coordinated turn. We study the maneuver over a 30s interval. Figure 11 illustrates the nominal trajectory. (All other nominal conditions are identical to the ones in the previous example.)

During the maneuver the aircraft is subjected to atmospheric turbulence in vertical, horizontal, and lateral directions modeled by Von Karman spectra and implemented by Dryden filters [1]. In addition, seven parameters in the model are allowed to vary individually on a closed interval. These parameters include variation in Cg position as well as uncertainty in the aerodynamic forces and moments along each axis. For the example presented here the numerical values for the variations are shown in Table 2. The bounds given for the center of mass are relative to the mean aerodynamic chord. The bounds given for the aerodynamic forces and moments are as a fraction of the value given by the lookup tables.

Parameter	Name	Lower Bound	Upper Bound
Center of mass	Cg	$0.195\overline{c}$	$0.205\overline{c}$
oX Force	Cx	.975	1.025
oY Force	Cy	.985	1.015
oZ Force	Cz	.97	1.03
Rolling Moment	Cl	.95	1.05
Pitching Moment	Cm	.95	1.05
Yawing Moment	Cn	.95	1.05

Table 2: Allowed variations for the uncertain parameters.

The algorithm is asked to find the combination of parameters and wind gusts that produce the largest norm of the performance variable vector, i.e. turning radius and altitude error. The worst case combination produced by the algorithm gives the value of each of the parameters at the end point of the allowable interval of variation. Table 3 summarizes the values found for the parameters.

The resulting 2-norm of the performance variables is 230 ft. The model

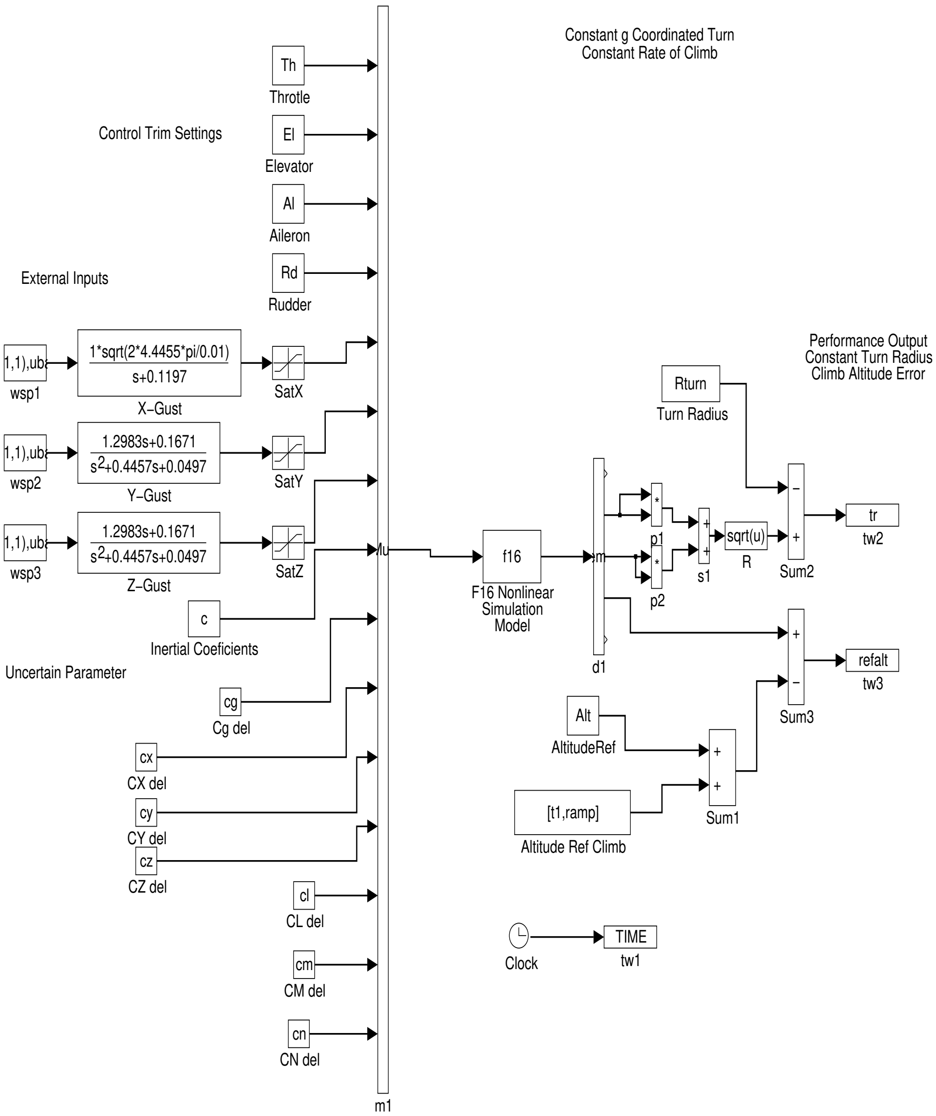

Figure 10: Simulink diagram for the F16 robust performance analysis.

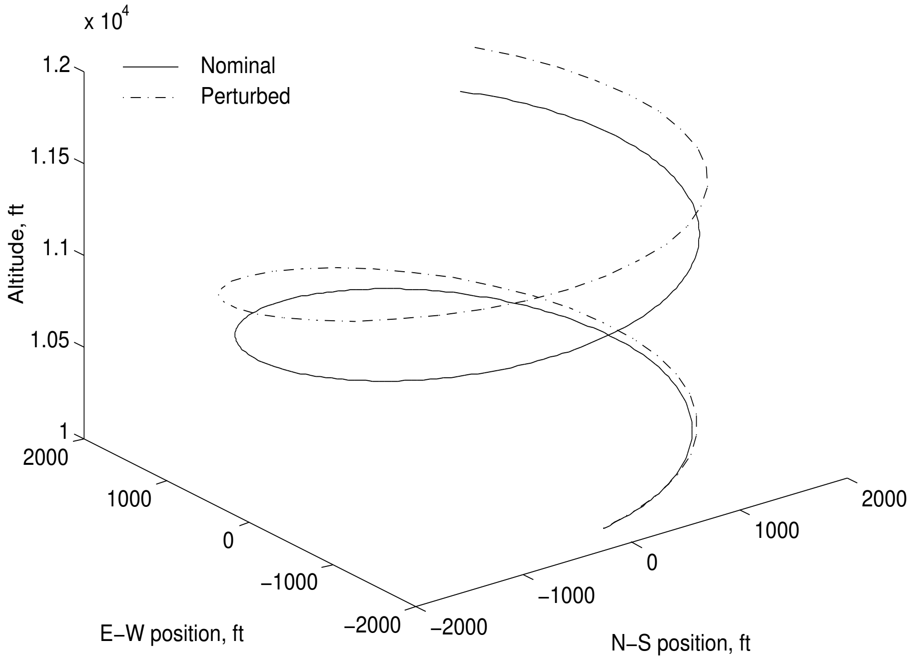

Figure 11: Spatial view of the trajectory.

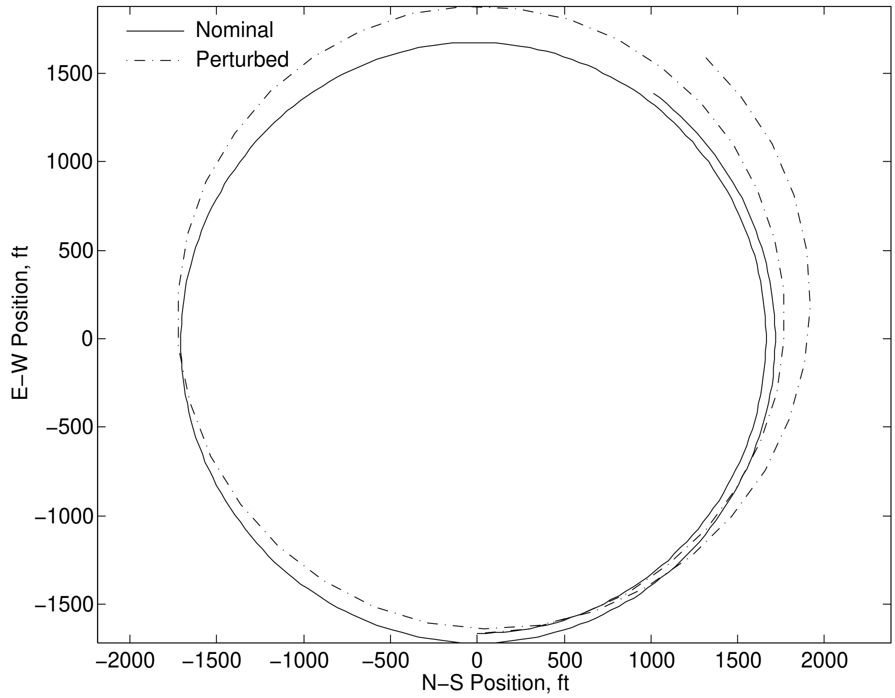

Figure 12: Ground track of the trajectory.

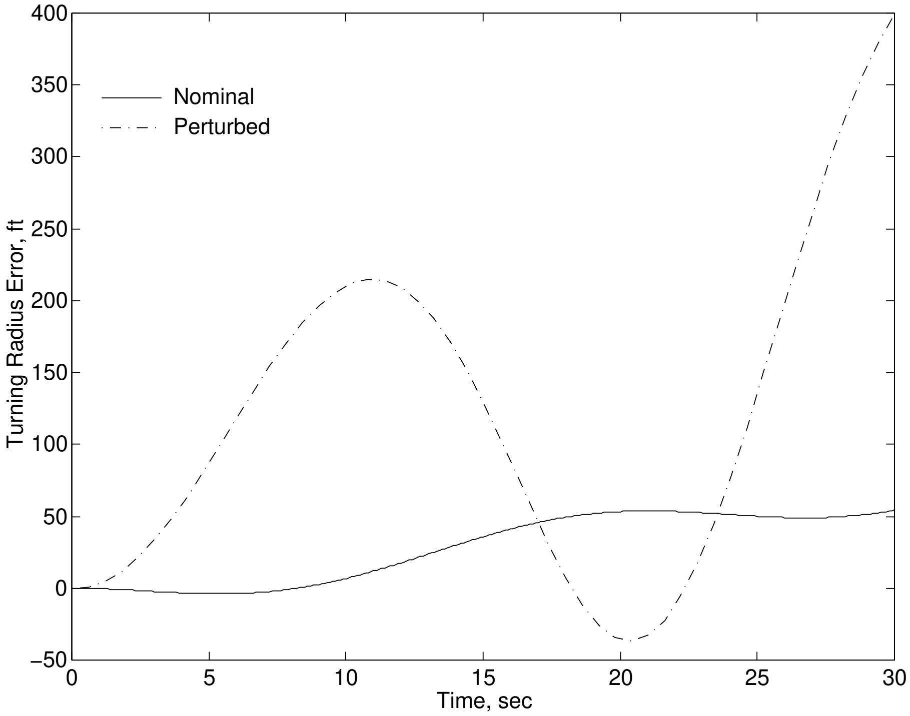

Figure 13: Turning radius error.

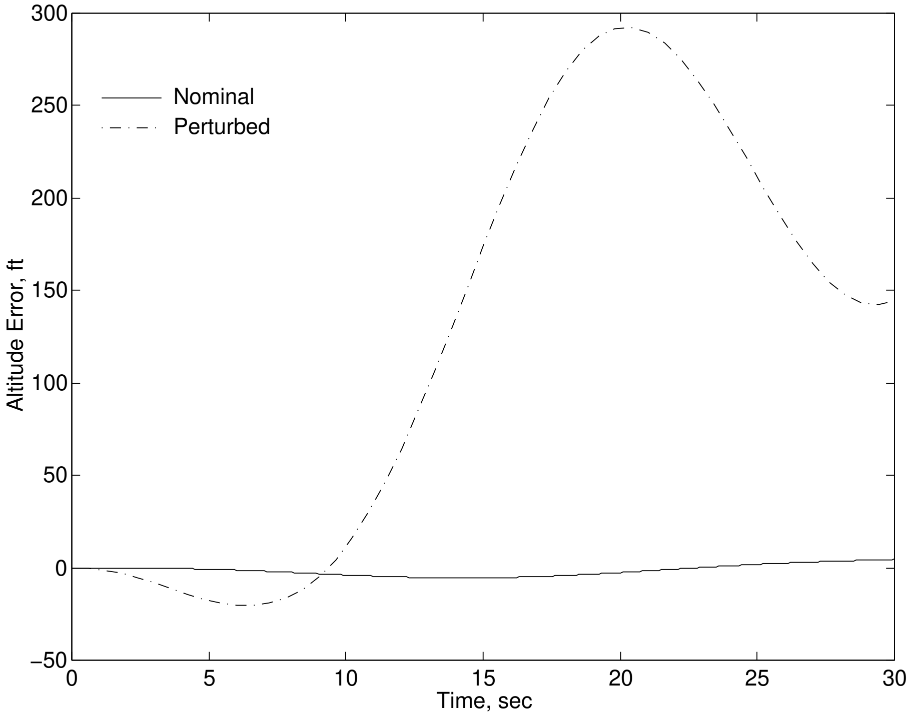

Figure 14: Altitude error.

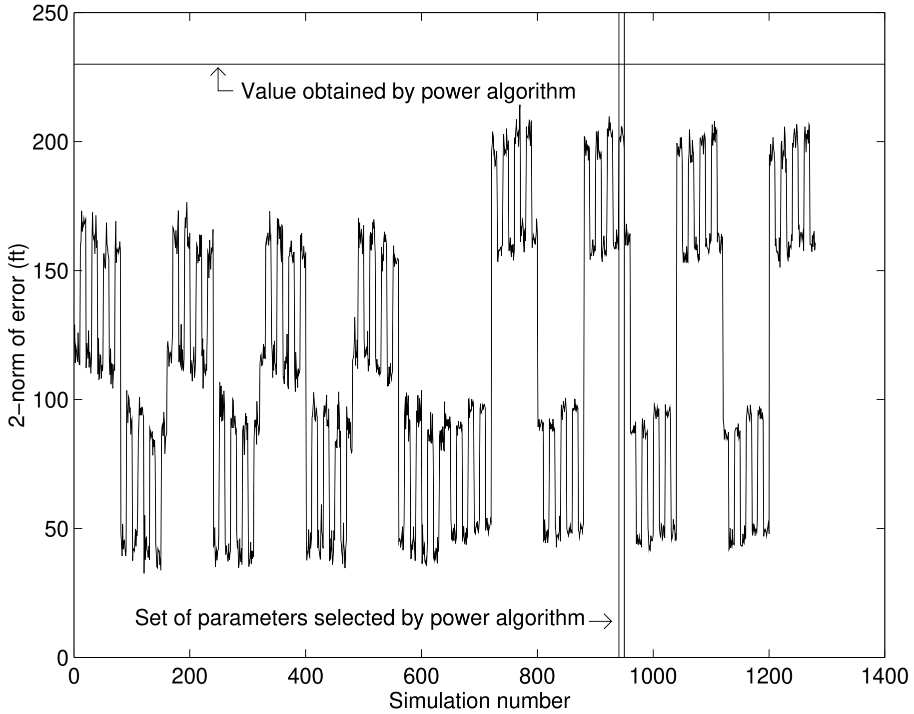

Figure 15: Comparison of stochastic and worst case analysis.

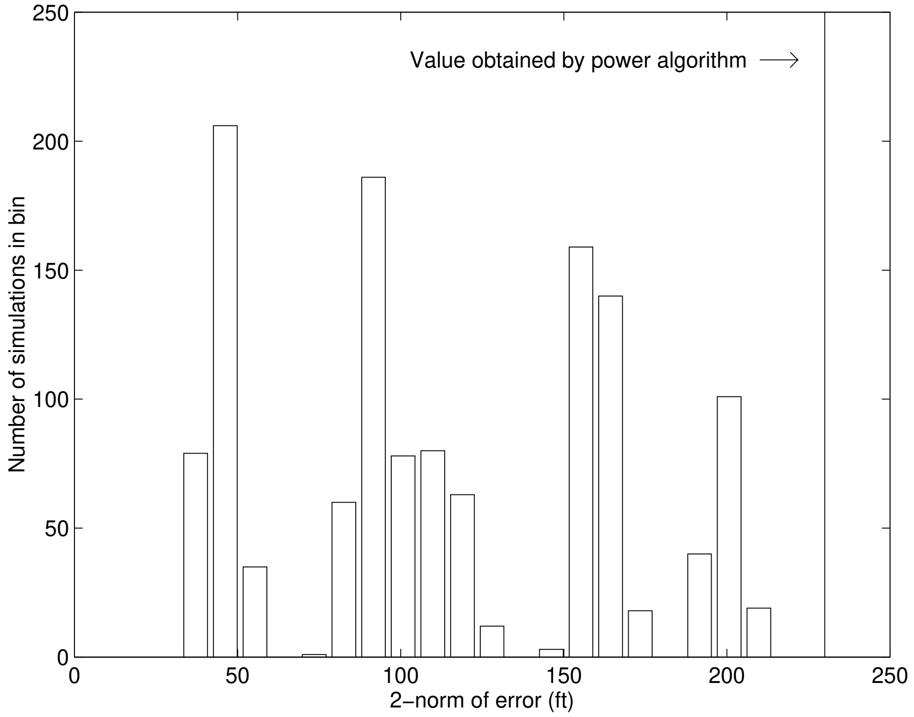

Figure 16: Comparison of stochastic and worst case analysis.

Parameter	Value
Cg	$0.195\overline{c}$
Cx	1.025
Cy	.985
Cz	.97
Cl	.95
Cm	.95
Cn	1.05

Table 3: Final values for the uncertain parameters.

simulation used by the algorithm was built in a Simulink diagram shown in Figure 10. The behavior of the airplane under the worst case parameter variation selected by the algorithm is illustrated in Figures 11 through 14. The solid line in all the figures represent nominal trajectory while the dashed line represents the perturbed trajectory.

To compare with more traditional ways of evaluating nonlinear system behavior, Monte Carlo simulations were run. For each parameter the endpoints of the interval of variation were selected as allowable values. A system simulation with random turbulence subjected to the same restrictions, is run for each possible combination of parameter values, 128 in this case. For each of these parameter combinations 10 simulations are performed. The resulting 2-norm of each simulation is plotted in Figure 15. The figure shows the 2-norm of the performance vector for each of the simulations as well as the worst case 2-norm. A total of 1280 simulations were performed. The power algorithm took four and a half hours for the signals to converge within 0.1 percent (51 iterations). However a value within 0.1 percent of the final performance index had already been achieved after 13 iterations. The Monte Carlo simulations were computed in slightly over four hours in the same platform.

As can be seen from Figure 15, the 2-norm of the worst case parameter combination with atmospheric winds shaped by the algorithm is indeed larger than any combination of parameters with random atmospheric winds. The two vertical lines demarcate the interval that corresponds to the same combination of parameters as that selected by the algorithm for the worst case. While this combination is not unique, as is evident from the figure, it does provide us with a better lower bound on the worst case behavior of the airplane for the allowable set of parameter variations than the Monte Carlo method. In terms of computational efficiency, the worst case algorithm is at least four times faster than the Monte Carlo simulations in this particular case. Figure 16 presents a frequency distribution of the error 2-norms obtained from the Monte Carlo simulations. The vertical line indicates the worst case value.

6 Conclusions

The development of numerical tools for robustness analysis of linear systems has been successful. In the last 15 years these tools have been adopted by control engineers, to a large extent replacing the traditional graphical tools developed in the 1950's. Central to the success of these techniques has been the fact that robustness properties of linear systems can be established by computing functions of finite matrices.

The main motivation behind the work in this paper is to extend these methods to nonlinear problems. Due to the wide variety of behavior in nonlinear systems, in order to develop practical computation algorithms, it is necessary to work with a restricted class of problems and systems.

The performance problem we deal with in this paper is robust trajectory tracking: a system is designed to complete a pre-specified path in a known finite time. Since the real system is not exactly the one used for the design, and since it is also subject to noise, the system will not follow the intended trajectory. The question of interest becomes: will the real trajectory, under the worst conditions possible, remain close to the nominal one in an appropriate norm? In order achieve the same tradeoff between generality, applicability and computational efficiency that the structured singular value framework achieves for linear systems, we use the 2-norm as the measure for the noise signals, the under-modeled components gain, and the performance objectives. Since the underlying optimization problem is not convex — as is also the case in linear system analysis — we are not able to compute the worst case error. We have to settle for upper and lower bounds.

We develop an algorithm to compute a lower bound on the performance index. This algorithm is similar in nature to the power algorithm for μ and shares with it many of its numerical properties. Although the algorithm is not proven to converge, a numerical study of its behavior shows that it is well behaved when applied to some practical examples, and that it outperform alternative methods on those same problems.

More experience is needed with the behavior of the lower bound power algorithm. It has to be tried in a larger variety of nonlinear systems, and with a larger array of uncertainty descriptions. It is important to understand the bounds on its performance in order to develop improvements for it; this is the way the power algorithm for linear systems was perfected. It is also important to gain experience with the setup for the robust trajectory tracking problem: how to choose uncertainty descriptions, time and frequency domain weights, noise bounds, and performance criteria.

It is also important to develop computable upper bounds. For a restricted set of systems, an upper bound on the robust trajectory tracking problem performance index can also be computed [13], in the form of a linear matrix inequality. However better numerical tools are needed to solve the resulting LMIs given their size and structure.

Acknowledgements

The authors would like to thank Adam Schwartz for carrying out comparison runs with other optimization techniques, and to Dr. Gene Morelli for providing the F16 simulation code. This work was partially supported by NASA and AFOSR.

References

[1] Federal Aviation Administration. Advisory circular: Automatic landing systems, January 1971. AC No 20-57.

[2] A. E. Bryson and Yu-Chi Ho. *Applied Optimal Control: Optimization, Estimation, and Control.* Halsted Press, 1975.

[3] H. Choi, P. Sturdza, and R. M. Murray. Design and construction of a small ducted fan engine for nonlinear control experiments. In *Proceedings of the American Control Conference*, pages 2618–2622, 1994.

[4] G. H. Golub and C. F. Van Loan. *Matrix Computations.* The John Hopkins University Press, 1983.

[5] W. M. Lu and J. C. Doyle. Robustness analysis and synthesis of nonlinear uncertain systems. Technical Report CIT-CDS-94-010, California Institute of Technology, 1994.

[6] D. G. Luenberger. *Optimization by Vector Space Methods.* John Wiley & Sons, 1968.

[7] M. P. Newlin and S. Glavaski. Advances in the computation of the μ lower bound. In *Proceedings of the American Control Conference*, pages 442–446, 1995.

[8] A. Packard and J. C. Doyle. The complex structured singular value. *Automatica*, 29(1):71–109, 1993.

[9] A. Packard, M. K. H. Fan, and J. C. Doyle. A power method for the structured singular value. In *Proceedings of the* 27^{th} *Conference on Decision and Control*, pages 2132–2137. IEEE, 1988.

[10] A. L. Schwartz. Ph. D. Dissertation, U. C. Berkeley. Document in preparation, 1996.

[11] B. L. Stevens and F. L. Lewis. *Aircraft Control and Simulation.* John Wiley & Sons, 1992.

[12] J. E. Tierno and J. C. Doyle. Finite time horizon robust performance analysis. In *Proceedings of the* 33^{rd} *Conference on Decision and Control*, pages 3080–3085. IEEE, 1994.

[13] J. E. Tierno and R. M. Murray. Robust performance analysis for a class of uncertain nonlinear systems. IEEE, 1995. To appear 34^{th} CDC.

[14] A. J. van der Schaft. L_2-gain analysis of nonlinear systems and nonlinear state feedback H_∞ control. *IEEE Transactions on Automatic Control*, 37(6):770–784, 1992.

[15] P. M. Young and J. C. Doyle. Computation of μ with real and complex uncertainties. In *Proceedings of the 29^{th} Conference on Decision and Control*, pages 1230–1235. IEEE, 1990.