USING GENETIC ALGORITHMS TO OPTIMIZE CONTROL LYAPUNOV FUNCTIONS

By

BRIAN K. HARGRAVE

Bachelor of Science in Mechanical Engineering

Oklahoma State University

Stillwater, OK

1999

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
MASTER OF SCIENCE
July, 2008

USING GENETIC ALGORITHMS TO OPTIMIZE

CONTROL LYAPUNOV FUNCTIONS

Thesis Approved:

Dr. L. L. Hoberock

Thesis Adviser

Dr. P. R. Pagilla

Dr. G. E. Young

Dr. A. Gordon Emslie

Dean of the Graduate College

Table of Contents

List of Tables

Table Page

List of Tables (Cont'd)

Table Page

List of Figures

Figure Page

List of Figures (Cont'd)

Figure Page

List of Figures (Cont'd)

Figure Page

List of Figures (Cont'd)

Figure Page

List of Symbols

Symbol	Description
x	system state
f, g	nonlinear system dynamics
A, B	linearized system dynamics
u	system control
$\Re^n$	n-dimensional vector space over the real line
C^1	space of all real-valued differentiable functions
V	Lyapunov function
P	Lyapunov function parameters
K	controller parameters
Q	quadratic performance index weighting matrix
inf	"infimum" of set theory
sup	"supremum" of set theory
$\gamma_{\min}$	minimum rate of convergence
X^0	region of interest
X	region of attraction
X_d^0	checking set
$\lvert u \rvert_{\max}$	maximum control effort
$\lVert \bullet \rVert$	2-norm
ε	distance function in the region of interest
L_f	system Lipschitz constant
E	optimization parameters
$\underline{\lambda}$	minimum eigenvalue
$\overline{\lambda}$	maximum eigenvalue
p_r	probability of copy
p_c	probability of crossover
p_m	probability of mutation
s	fitness bias parameter
μ_γ	mean rate of convergence
σ_γ	standard deviation of the rate of convergence
$\mu_{\lvert u \rvert}$	mean control effort
$\sigma_{\lvert u \rvert}$	standard deviation of control effort

1 Introduction

1.1 Motivation

1.1.1 CLF Design Theory

Linear $\mathcal{H}_2$ and $\mathcal{H}_\infty$ design methods are proven tools for designing control laws for guaranteed optimal and robust performance of linear systems. Zhou (1997) & Levine, et al. (1996) provide an introduction to these subjects. Linear structure allows controller design methods applicable to *all* linear systems that are controllable and observable. Although nonlinear systems do not lend themselves to a single design method, many engineering systems are linearizable and can be fit into the linear $\mathcal{H}_2$ and $\mathcal{H}_\infty$ design framework. The drawback is that the resulting controllers exhibit degraded performance from being robust against the nonlinearities.

A more general and less conservative controller design method for nonlinear systems is the use of control Lyapunov functions (CLFs) (Khalil, 1996), (Krstic, 1995). This approach forces the "energy" of the system to decrease with time. "Energy" is represented by a positive definite function of the system states, called the CLF. As the states increase in magnitude, the CLF increases. As an example, in a mechanical system, energy increases with increasing positions and velocities, such that an appropriate CLF also increases with the positions and velocities. The CLF time derivative can then be made a function of the control input(s), such that any control law that causes the CLF to constantly decrease over time (in a closed set of state space) is guaranteed to drive the

states to the equilibrium point(s). If the energy of the system continuously decreases, the system must settle to an equilibrium point. Lyapunov stability theory can be used to find a satisfactory control law from two broad approaches (Kokotovic & Arcak, 2001): 1. A control law is selected and a search for a Lyapunov function is conducted to *prove* the performance of the closed loop system, 2. A *control* Lyapunov function is selected, from which a control law is derived that *yields* certain performance measures. The difference between the two approaches is that the first relies primarily upon *analysis,* while the second relies on *synthesis*. Specifically, in the first approach the control law is fixed, such that the performance measures are invariant to the selection of the Lyapunov function. Hence the Lyapunov function acts merely as a tool to guarantee stability, and sometimes estimate the performance measures. In the second approach, the control law arises from guaranteed performance measures of the CLF (i.e. rate of convergence and region of attraction). The problem is that the CLF may not be a good selection for determining the desired performance measure because the CLF topology and parameter values are incapable of yielding "globally optimal" performance, where "global" means the set of all possible CLF's. The work of Johansen (2000, a & b) offers a computational procedure for generating non-quadratic Lyapunov functions which can be used to estimate the performance of smooth nonlinear systems. The work shows that any Lyapunov function may be represented to arbitrary accuracy by a sufficiently large finite summation of quadratic functions weighted by smooth switching functions. Johansen's work indirectly supports the need to appropriately tune a CLF so that an accurate estimate of the performance of the system may be obtained and used by the optimization algorithm.

1.1.2 CLF Design Examples

The *manner* in which the system states and control signal settle to equilibrium is the focus herein and will be referred to as the performance measure of the pair, control law and CLF. Typical performance measures include the RMS and maximum values of the system states and control signals, the rate of convergence of the system, and the maximum region of attraction, loosely defined as the set of points in state space such that the state returns to equilibrium (defined precisely below). Because of the dependence of the CLF on the control signals, the performance measures depend on both the selection of the CLF and the control law. In the work that follows, we first select the CLF and control law functions and use a numerical method for tuning the parameters of these functions to satisfy the performance requirements. We investigate the simultaneous numerical tuning of both the CLF and the control law parameters. By allowing both entities to vary when searching for a solution, shortcomings of other methods may be overcome. Henceforth, the resulting CLF and control law will correspond to the pair *(P,K)* where *P* stands for the vector of parameters of the CLF and *K* for the vector of parameters of the control law. In some situations, *P* and *K* may not be independent of each other due to the selection of CLF and controller topologies (e.g. $K = -\frac{1}{2} B^T P$, $B \in \Re^{nx1}$, $P \in \Re^{nxn}$). To demonstrate the utility for this approach, consider as an example a parameterized 2nd order linear time invariant system given by the equation

$$\dot{x} = \begin{bmatrix} 0 & 1 \\ -1 & -a \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \qquad a > 0\,,\ x = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T \in \Re^{2x1},\ u \in \Re \qquad (1.1)$$

where dot notation indicates time differentiation. We wish to use CLF theory to design a stabilizing control, *u,* of the states, *x*. We select a parameterized form of the CLF, *V(x)*, given by

$$V(x,b) = x^T \begin{bmatrix} b & 0 \\ 0 & 1-b \end{bmatrix} x, \qquad 0 < b < 1 \tag{1.2}$$

We use this form of CLF so that we may vary the influence of x_1 and x_2 on the CLF. Suppose we seek to find a control such that

$$\dot{V}(x) = -x_2^2 \tag{1.3}$$

Computing $\dot{V}$ yields

$$\dot{V}(x,a,b) = 2(2b-1)x_1x_2 + 2a(b-1)x_2^2 + 2(1-b)x_2u \tag{1.4}$$

It is straightforward to show that (1.3) can be satisfied by selecting *u* as

$$u(x,a,b) = 2x_1 + \frac{1}{b-1}\left(ab - a + \frac{1}{2}\right)x_2 \tag{1.5}$$

The pair *(P,K)* becomes *(b, [a b])*. Now suppose we desire to minimize the control effort inside the region defined by $S = \left\{x \in \Re^2 : x_1 \in \left[-1 \quad 1\right], x_2 \in \left[-1 \quad 1\right]\right\}$ (a unit square centered at the origin in state-space) by varying the free parameter *b*. Considering *a* fixed, a suitable performance measure could be defined as

$$J(b) = \int_{-1}^{1}\int_{-1}^{1} u(x,a,b)^2\, dx_1 dx_2 \tag{1.6}$$

in which we wish to minimize the control effort in *S*. It may be shown for the ranges of *a* and *b,* the minimum of (1.6) exists at the point

$$b^* = \left\{b \in \Re : \frac{dJ}{db} = 0\right\}. \tag{1.7}$$

The derivative of *J* with respect to *b* is

$$\frac{dJ}{db} = -\frac{2}{3}\frac{2a(b-1)+1}{(b-1)^3} \tag{1.8}$$

In this special case, solving for b^* in *(1.7)* yields

$$b^* = 1 - \frac{1}{2a} \tag{1.9}$$

We see that the selection of the CLF in (1.2) has an effect on the *optimality* of the resulting control. Our example minimized the control effort in a region of state space. Had we selected an arbitrary value for *b,* other than b^* of (1.9), a suboptimal controller would have resulted. In addition, $a \le \frac{1}{2}$ implies $b^* \le 0$, violating the requirement $0 < b < 1$, hence 1.3 cannot be satisfied if $a \le \frac{1}{2}$.

Now consider a similar investigation of a 2nd order *nonlinear* time invariant system. The system equations are

$$\dot{x} = f(x) + g(x)u, \quad f(x) = \begin{bmatrix} x_1^3 & 0 \end{bmatrix}^T, g(x) = \begin{bmatrix} 0 & 1 \end{bmatrix}^T, u \in \Re \tag{1.10}$$

In this example, we begin by selecting Sontag's Universal Formula (Sontag, 1989) as the control law

$$u = \begin{cases} -\dfrac{\dfrac{\partial V}{\partial x} f + \sqrt{\left(\dfrac{\partial V}{\partial x} f\right)^2 + \left(\dfrac{\partial V}{\partial x} g\right)^4}}{\dfrac{\partial V}{\partial x} g} & \text{if } \dfrac{\partial V}{\partial x} g \neq 0 \\ 0 & \text{if } \dfrac{\partial V}{\partial x} g = 0 \end{cases}, \tag{1.11}$$

where $V : \Re^2 \to \Re$.

Assuming a CLF, *V(x)*, exists, the control law of (1.11) is a smooth globally asymptotic stabilizing control for (1.10) because it forces the state trajectories of (1.10) to follow the

gradient of the CLF. A problem with such a control law is the "$\frac{\partial V}{\partial x} g$" term in the denominator may cause u to become very large. To investigate the role of the CLF, we assume the quadratic function in

$$V(x,b,\theta) = x^T \begin{bmatrix} b & 0 \\ 0 & 1-b \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} x, \qquad 0 < b < 1,\ 0 \le \theta < \pi/2 \tag{1.12}$$

is a "local CLF" (specifically defined in the next section). Equation (1.12) is general quadratic function similar to (1.2), except that we introduce a new parameter, θ, which rotates the eigenvectors about the origin. The local CLF gradient is

$$\frac{\partial V}{\partial x} = x^T \begin{bmatrix} b\cos\theta & -b\sin\theta \\ (1-b)\sin\theta & (1-b)\cos\theta \end{bmatrix} \tag{1.13}$$

The specific expression for the control law (1.11) now becomes

$$u = \begin{cases} \dfrac{x_1^4 b\cos\theta + x_1^3 x_2 (1-b)\sin\theta + \sqrt{\left(x_1^4 b\cos\theta + x_1^3 x_2(1-b)\sin\theta\right)^2 + \left(-x_1 b\sin\theta + x_2(1-b)\cos\theta\right)^4}}{x_1 b\sin\theta - x_2(1-b)\cos\theta} & if \quad x_1 b\sin\theta - x_2(1-b)\cos\theta \neq 0 \\ 0 & f \quad x_1 b\sin\theta - x_2(1-b)\cos\theta = 0 \end{cases} \tag{1.14}$$

Ensuring that (1.14) does not grow too large might involve minimizing the maximum control effort on some set S by tuning b and θ. Since a closed form optimal solution probably does not exist or is at least unknown, u would have to be computed and evaluated at all the points in S for every iteration of the optimization algorithm. Even for such a relatively simple system and control law, we clearly have a multimodal optimization problem which requires many function evaluations. It is much more desirable to find an optimization algorithm that scales nicely with problem complexity.

1.2 Definitions, Problem Statement, and Background Theory

1.2.1 Scope of Systems and CLF's

In the following, we assume differentiable time-invariant nonlinear systems and controls of the form

$$\dot{x} = f(x) + g(x)u(x,K) \tag{1.15}$$

where $x \in \Re^n$ is the *n-dimensional* state vector, $K \in \Re^p$ is the *p-dimensional* control gain vector, $u : \Re^{n+p} \to \Re$ is the control input, $f, g : \Re^n \to \Re^n$ are the system dynamics, and $u, f, g \in C^1$, *f(0)=0*, where C^1 is the space of all differentiable functions. Chapter 2 discusses the specific control laws selected for the study.

We restrict the analysis to quadratic Lyapunov functions of the form

$$V(x) = x^T Px \tag{1.16}$$

where $P \in \Re^{nxn}, P = P^T > 0$. The optimization parameters shall be represented by the pair *(P,K)*.

Adapted from (Sontag, 1989), a *local* CLF is defined in this work as a smooth, proper, and positive definite function $V : \Re^n \to \Re$, such that $\inf_{u \in \Re, x \in X^0 - \{0\}} \dot{V}(x,u) < 0$, where

$$X^0 \subset \Re^n \cup \{0\} \tag{1.17}$$

is the *region of interest*. The region of interest is defined as a subset of state-space where the controller is optimized that contains a neighborhood of the origin and *only one equilibrium point*. We shall use "local CLF" and "CLF" interchangeably from this point on in the discussion.

1.2.2 Performance Measures

The performance measures considered herein are: 1) minimum rate of convergence, $\gamma_{\min}$; 2) region of attraction, X; and 3) maximum control effort, $|u|_{\max}$. These performance measures are defined below for all *x* in a region of state-space, X^0.

The minimum rate of convergence, $\gamma_{\min}$, is defined by the expression (Johansen, 2000 a)

$$\|x(t)\| \le \sqrt{\frac{c_2}{c_1}}\|x(0)\|e^{-\frac{\gamma_{\min}}{2}t}, \qquad \gamma_{\min} > 0, \quad t \ge 0 \tag{1.18}$$

where, $\|\bullet\|$ denotes the *2-norm*, $V(x) \ge c_1\|x\|^2$, $\dot{V}(x) \le -\gamma_{\min}V(x)$, and $c_2 = \bar{\lambda}(P)$ the magnitude of the largest eigen-value of *P*. This performance measure means that the norm of the states will converge no slower than an exponential decay with time constant $2/\gamma_{\min}$, and is based on uniform exponential stability.

The region of attraction, *X*, is defined as (Johansen, 2000 a)

$$X = \left\{ x \in X^0 \middle| V(x) \le \inf_{\xi \in \partial X^0} V(\xi) \right\} \tag{1.19}$$

where *inf* denotes "infimum" of set theory and ∂X^0 represents the border of the set X^0. This definition is based on the fact that the trajectory of the system cannot cross a level-set Ω_a of the Lyapunov function, $\Omega_\alpha = \{x \in X^0 | V(x) = \alpha, \alpha > 0\}$, because the Lyapunov function is a decreasing function of time. The value for α in this case is $\alpha = \inf_{\xi \in \partial X^0} V(\xi)$.

Maximum control effort, $u_{\max}$, will be defined as

$$u_{\max} = \sup_{x \in X}|u| \tag{1.20}$$

where *u* is the output of the controller and *sup* denotes "supremum" of set theory. The utility of this performance measure is that, along with the minimum region of attraction performance measure (1.19), it may be used to design controllers for systems with sensor and actuator saturations by ensuring the saturation levels are avoided.

1.2.3 Problem Statement

The objective of this thesis is to demonstrate the effectiveness of tuning both the control law and the local CLF simultaneously to maximize the rate of convergence and minimize the control effort of nonlinear systems. The controller design intent is to find a controller tune specific to the nonlinear system that is less conservative than the tune based on robust linear systems theory. Two control laws, defined in Chapter 2, will be tested: 1. an LQR full-state feedback control law, and 2. a Sontag-like nonlinear full-state feedback control law. It is assumed that a quadratic Lyapunov function is a local CLF for the nonlinear systems studied herein. The proposed solution method does not offer a strict guarantee on performance level. However, it is suggested that with enough random performance sampling, the system will achieve the estimated performance level with sufficiently high confidence, making the proposed method a practical solution for real-world controller design.

1.2.4 Literature Review

Convex optimization techniques were used in Johansen (2000, a & b) to numerically compute a generalized non-quadratic Lyapunov function for Lipschitz nonlinear systems. The work was based solely on the use of Lyapunov functions as tools to measure the performance of smooth (locally Lipschitz) nonlinear systems. Extensions to using the method for controller design were not addressed, however. Ghaoui &

Balakrishnan (1994) proposed the so-called "V-K" iteration technique for linear control systems (analogous to the "D-K" iteration of mu-synthesis (Zhou, 1997)), where the control gains, K, are held fixed and the quadratic Lyapunov matrix, P, is found using LMI techniques. This minimizes the time derivative of the Lyapunov function, and then P is held fixed while K is used to minimize the time derivative of the Lyapunov function.

Although Johansen's work was not used for controller design, an extension is fairly straightforward. While not covered herein, an interesting avenue of future research would be to replace the quadratic Lyapunov functions and linear control laws of Ghaoui & Balakrishnans' V-K iteration method with the general non-quadratic Lyapunov functions of Johansen's method, then employ control laws linear-in-the-parameters to extend the V-K iteration method to Lipschitz nonlinear systems that are affine in the control law. Johansen's method exploits the structure of the selected Lyapunov function, which is linear-in-the-parameters. If a control affine system and a linear-in-the-parameters control law were assumed, the controller parameters would also show up linearly in $\dot{V}$, so that a convex optimization method could be used to alternately tune both the CLF and controller parameters.

1.2.5 Restrictions of Current Methods

Ghaoui & Balakrishnans' method is restricted to quadratic Lyapunov functions and linear systems. The possible extension to Johansen's method outlined above is restricted to affine in the control nonlinear systems with linear-in-the-parameters control laws. In addition, these methods require that the CLF and the control law be tuned separately. Fixing one set of parameters and tuning another is a restriction in the optimization method, which may hinder the parameter search. Considering the

optimization mechanism known as "hill climbing", holding some parameters constant while varying others is analogous to traveling up the hill in alternating orthogonal directions, whereas, tuning parameters simultaneously is analogous to following the gradient all the way up the hill. Following the gradient can be a more efficient search mode because it is instantaneously the fastest way to increase elevation. The approach taken herein exploits the interaction between the CLF and the control law as they are both varied simultaneously, with the intention of finding better controllers and/or a better optimization method.

1.2.6 Checking Performance Measures for Nonlinear Systems

The performance measures must be met everywhere in X^0. However, checking every point is impossible because X^0, although a compact set, is a dense uncountably infinite set of points. We must therefore find a way to sample X^0 and check the performance measures on a discrete finite subset, X_d^0, while guaranteeing that the points between the "checking points" of X_d^0 also satisfy the requirements.

The following is taken from "Theorem 2" of Johansen (2000 a) and may be used to guarantee the Lyapunov conditions are met for all $x \in X^0$ given that they are satisfied for all $x \in X_d^0$. We first present some preliminary definitions used in the theorem. Define the checking set density function $\varepsilon(x)$ by

$$\varepsilon(x) = \inf_{x_d \in X_d^0} \|x - x_d\| \tag{1.21}$$

which is the *distance from some point in the region of interest,* $x \in X^0$*, to the closest neighbor in the checking set,* $x_d \in X_d^0$. Define the Lipschitz constant for f as L_f. The *size*

parameter will be defined as $S = \sup_{x \in X^0, k \in \mathrm{K}} \| f(x,k) \|$, where K is the set of admissible values for the control gains, $\overline{P} = \overline{\lambda}(P)$, and $\overline{X}^0 = \sup_{x,y \in X^0} \| x - y \|$.

Theorem 1.1 (adapted from Theorem 2 of Johansen (2000 a))

Suppose X^0 is a compact set, $V(x) > 0$ for all $x \in X^0 - \{0\}$ and f is a bounded, locally Lipschitz function. Let $\gamma > 0$ be given and suppose there exists an $\alpha > \gamma > 0$ such that for all $x \in X_d^0$

$$\dot{V}(x) \le -\alpha V(x) \tag{1.22}$$

Assume the checking set grid density is such that

$$\varepsilon(x) \le (\alpha - \gamma_{\min}) \frac{V(x)}{Q} \tag{1.23}$$

where

$$Q = 2S\overline{P} + 2\overline{X}^0 \overline{P} L_f + 2\alpha \overline{P} \overline{X}^0 \tag{1.24}$$

Then for all $x \in X^0$

$$\dot{V}(x) \le -\gamma_{\min} V(x) \tag{1.25}$$

Proof: (see Johansen 2000a)

Johansen's theorem allows us to ensure stability for the points not sampled. The theorem suggests that if the state space is sampled fine enough, then an accurate estimate of $\gamma_{\min}$ may be achieved. The required "closeness" of the sampling points is related through Q, a measure of how fast the system states change (proportional to the system Lipschitz constant). By (1.22), the size of $\gamma_{\min}$ in (1.25) is directly related to α, ε, V, and Q. We note, however, that computing Q is very difficult in practice because for highly

nonlinear systems, finding L_f can be very difficult. In addition, both $\overline{P}$ and L_f are variables in the search for *(P,K)*. Therefore any $\overline{P}$ and L_f values used initially that satisfy (1.23) might change enough during the search to invalidate (1.23), causing the need to refine the checking grid and possibly invalidate the progress made by the search algorithm. We therefore propose a more practical method to fix $\varepsilon(x) = \varepsilon, \varepsilon > 0$ and estimate $\gamma_{\min}$ via random sampling of X^0 (described in Chapter 2). The estimation of $\gamma_{\min}$ of course occurs after a *(P,K)* pair is found that satisfies (1.22).

Finding a positive value of $\gamma_{\min}$ does not imply that the performance measures of (1.17)-(1.19) have been met. However, it does imply the satisfaction of performance measures of the *form* defined by (1.18)-(1.20), in the following ways: 1) a positive value of $\gamma_{\min}$ is precisely a minimum rate of convergence (1.18); 2) $\gamma_{\min}$ is defined on some region of X^0 containing the origin which must contain a connected level set of the Lyapunov function whose interior's minimum rate of convergence is determined by $\gamma_{\min}$ in (1.18), assuring the region X^0 contains a minimum region of attraction (1.19); and 3) $|u|$ has an upper bound on X since u is differentiable and X is bounded. In other words, finding a *(P,K)* pair which yields $\gamma_{\min} > 0$ means that on X^0, a decay rate, a region of attraction, and a maximum control effort exists, but not necessarily satisfying the desired amounts.

A Method for Relaxing the Checking Grid Density Requirements Around the Origin

Theorem 1.2

Suppose we have a system of the form (1.15) and CLF of the form (1.16). Let X^0 *be a compact set containing the origin and, without loss of generality, assume the system is locally stable and u=0. Define* $A = \left.\frac{\partial f}{\partial x}\right|_{x=0}$ *and*

$$\gamma_{\min} = \inf_{x \in X^0 - \{0\}} \frac{-\dot{V}}{V} \tag{1.26}$$

$$\gamma_{\min}^{L} = \inf_{x \in X^0 - \{0\}} \frac{-x^T \left(A^T P + PA\right) x}{x^T P x} \tag{1.27}$$

Then as $\overline{X}^0 = \sup_{x,y \in X^0} \|x - y\| \to 0$, $\gamma_{\min} \to \gamma_{\min}^{L}$ (1.28)

Proof:

The Taylor series expansion of *f* about the origin is

$$f(x) = Ax + D(x) \tag{1.29}$$

where *D(x)* represents the higher order terms of the expansion. The time derivative of *V* becomes

$$\frac{\dot{V}}{V} = \frac{\frac{\partial V}{\partial x} Ax + \frac{\partial V}{\partial x} D(x)}{V} = \frac{-x^T \left(A^T P + PA\right) x - 2x^T P D(x)}{x^T P x} \tag{1.30}$$

decomposing $\gamma_{\min}$ into two parts

$$\gamma_{\min} = \inf_{x \in X^0 - \{0\}} \left(\frac{-x^T \left(A^T P + PA\right) x}{x^T P x} + \frac{-2x^T P D(x)}{x^T P x} \right) \tag{1.31}$$

and using the facts that $\inf_{a,b}(a+b) \le \inf_{a,b}(a + \|b\|)$ and $\inf_{a,b}(a + \|b\|) \le \inf_{a}(a) + \|b\|$, and the definition of $\gamma_{\min}^{L}$ (1.27) one may arrive at the inequality

$$\gamma_{\min} = \inf_{x \in X^0 - \{0\}} \left(\frac{-x^T \left(A^T P + PA\right) x}{x^T P x} + \frac{-2x^T P D(x)}{x^T P x} \right)$$
$$\leq \inf_{x \in X^0 - \{0\}} \left(\frac{-x^T \left(A^T P + PA\right) x}{x^T P x} + 2\left\| \frac{x^T P D(x)}{x^T P x} \right\| \right) \tag{1.32}$$
$$\leq \inf_{x \in X^0 - \{0\}} \left(\frac{-x^T \left(A^T P + PA\right) x}{x^T P x} \right) + 2\left\| \frac{x^T P D(x)}{x^T P x} \right\| = \gamma_L + 2\left\| \frac{x^T P D(x)}{x^T P x} \right\|$$

The inequality (1.32) implies

$$\left\| \gamma_{\min} - \gamma_{\min}^L \right\| \leq 2\left\| \frac{x^T P D(x)}{x^T P x} \right\| \tag{1.33}$$

Since the lowest order terms in the polynomial $D(x)$ are quadratic, the lowest order terms in the polynomial "$x^T P D(x)$" are cubic ensuring that as $\overline{X}^0 = \sup_{x,y \in X^0} \|x - y\| \to 0$,

$\left\| \frac{x^T P D(x)}{x^T P x} \right\| \to 0$ implying $\gamma_{\min} \to \gamma_{\min}^L$. ■

Because of the large number of sample points usually required to obtain a reliable estimate of $\gamma_{\min}$, it is more efficient to compute the eigenvalues of P and $A^T P + PA$ and instead use the bounding relationship

$$\frac{-\overline{\lambda}\left(A^T P + PA\right)}{\overline{\lambda}(P)} \leq \gamma_{\min}^L \tag{1.34}$$

By Theorem 1.2, for a sufficiently small region about the origin, the difference between $\gamma_{\min}$ and $\gamma_{\min}^L$ is small, making the left side of (1.34) a good estimate of $\gamma_{\min}$ for points near the origin and dramatically reducing the required size of the checking set X_d^0.

1.3 Thesis Outline

Chapter 1 motivates the search for an optimization method for tuning CLF's and their corresponding control laws. Chapter 2 poses the specific optimization problem and

describes the specific genetic algorithm used to solve the optimization problem. Chapter 3 provides examples of how to use the genetic algorithm of Chapter 2 to tune full state feedback CLF controllers. Chapter 4 provides an outline of future research that explores the addition of uncertainty and adaptation in the control systems, and also addresses the use of more generalized CLFs, as well as discrete time control.

2 A Genetic Algorithm for CLF Optimization

2.1 Genetic Algorithm Motivation

Genetic algorithms (GA's) are ideal candidates for a general optimization method for solving the dual tuning problem described in chapter 1. The reader is referred to Flemming & Purshouse (2002) for an excellent survey on evolutionary algorithms in control engineering. GA's are parallel global stochastic search algorithms that do not depend on derivatives to perform the optimization – their most attractive feature. The stochastic nature of the algorithm allows it to make random jumps into hard-to-reach regions of search space that may contain a local extrema. These regions would otherwise be inaccessible using only local gradient information. Metaphorically speaking, they are capable of jumping over valleys onto other mountains in search of the highest peak. The *parallel* search feature emerges from the *population* of search points being spread-out among the parameter search space. This allows the simultaneous exploration of several regions in search space containing local maxima. Non-differentiable *fitness functions* may be used as the optimization objective function because the search movements are not based on gradients, but occur from either random jumps called *mutations* or selective combinations of two or more highly fit individuals called *crossovers*. The crossovers effectively serve as interpolations and extrapolations of the existing search points in the GA population.

The GA is capable of making fast progress in the parameter search effort for difficult problems. However, unlike gradient-based methods, GA's lack a guarantee of

making steady progress towards a local maxima. A remedy to this problem is decreasing the mutation step size. This mimics the small incremental progress made by a gradient method because a small random jump in some situations is likely to have a positive projection upon the gradient direction so that a step in the right direction is made. Gradient and other methods that use only local objective function information are not attractive here for three major reasons: (1) the multimodal nature of the objective function leads to convergence to local extrema; (2) the objective function evaluation points drift in time (explained below), giving the objective function a time varying nature which calls for a variable step rate to balance numerical stability with making fast enough progress, adding an unnecessary degree of difficulty to the problem; and (3) the non-differentiability of an objective function is incompatible with a gradient method calling for the use of a finite difference approximation to the gradient which can be inefficient for a large parameter space (many function evaluations must be made just to move a small amount in parameter space). A general rule of thumb in ensuring GA's make sufficient progress is that the population have sufficient size and time. This typically makes them inefficient for searching low dimensional parameter spaces. However, out-weighing this drawback is their ability to make fast progress in large parameter search spaces, as well as their ability to optimize both the parameters and topology of functions, as with genetic programming, a subset of genetic algorithms (Koza et al., 1999).

2.2 Tailoring to the Specific Problem

2.2.1 Optimization Objectives

The objective of the optimization algorithm is to find a *(P,K)* pair that satisfies the exponential stability conditions

$$V(x) \geq \nu \|x\|^2 \qquad \nu > 0 \tag{2.1}$$

and

$$\dot{V}(x) \leq -\gamma_{\min} V(x) \qquad \gamma_{\min} > 0 \tag{2.2}$$

for all sampled points $x \in X_d^0$. In addition to (2.1) and (2.2), we also consider the maximum allowable control effort on the sample set labeled $|u|_{\max}$,

$$|u|_{\max} \geq \sup_{x \in X_d^0} |u| \tag{2.3}$$

We define an *admissible (P,K)* pair as the set of parameter values that make the system (1.15) and Lyapunov function (1.16) satisfy (2.1) , (2.2), and (2.3) for some user specified triple $\left(\nu, \gamma_{\min}, |u|_{\max}\right)$ on the checking set X_d^0. Typically, desired values of $\left(\nu, \gamma_{\min}, |u|_{\max}\right)$ are not known, so they are allowed to "float" with progressive improvement during the optimization, and the user decides if the values obtained are good enough at the end of the optimization run.

2.2.2 Simplifying the Search

We restrict our investigation to quadratic Lyapunov functions. Due to computational restrictions and the use of Theorem 1.2 to reduce the required size of X_d^0, we also restrict the design optimization to closed convex sets about the origin. To add an additional degree of local robustness and optimality to the closed loop nonlinear system, we also assume the CLF is locally H_2 *inverse optimal* (Kokotovic & Arcak, 2001). That is, the CLF matrix *P* is the solution to the LQR problem for the linearized system for some set of states and control effort weighting matrices in the objective function. The

problem reduces to using the GA to find $E \in \Re^{nxn}$ such that

$$Q = E^T E + cI \ , \ c \geq 0 \tag{2.4}$$

so that $\underline{\lambda}(Q) \geq c$. We compute a $P = P^T > 0$ that satisfies the *Hamilton-Jacobi-Bellman* (HJB) equation (also known as the Algebraic Riccati Equation) for linear time invariant systems (Levine et al., 1996)

$$A^T P + PA - PBB^T P + Q = 0 \tag{2.5}$$

The solution to (2.5) satisfies the minimization of the performance index $J = \int_t^\infty x^T Qx + u^2 dt$ when we assign the control as

$$u = -B^T Px \tag{2.6}$$

We thus use E to generate the CLF $V = x^T Px$. An additional utility to the approach is the direct computation of a full state feedback linear control (2.6), if such a control law is desired. One may argue that selecting (2.6) for the control is like using the gradient direction of V normalized to satisfy the HJB equation (Primbs, et al 2000). Therefore, similar to the argument presented in Theorem 1.2, any control law that approaches $u = -B^T Px$ as x approaches the origin will also be a locally optimal controller for the nonlinear system. In addition, because we sample the performance index for the nonlinear system on the set X_d^0, the controller shall be optimal on X_d^0. In summary, we attempt to maximize the rate of convergence and minimize the control effort on X_d^0 using controllers whose linearization is inverse optimal for the linearized system dynamics.

A more general version of the control (2.6) based on the proposed control of Primbs, et al (2000) is

$$u = \begin{cases} -\dfrac{\dfrac{\partial V}{\partial x} f + \sqrt{\left(\dfrac{\partial V}{\partial x} f\right)^2 + q(x)\left(\dfrac{\partial V}{\partial x} g\right)^2}}{\dfrac{\partial V}{\partial x} g} & \dfrac{\partial V}{\partial x} g \neq 0 \\ 0 & \dfrac{\partial V}{\partial x} g = 0 \end{cases} \tag{2.7}$$

The control is a modified version of Sontag's Universal Formula (Sontag, 1989) for nonlinear systems that are single input and affine in the control. The performance index minimized by (2.7) has the form $J = \int_0^\infty q(x) + u^2 dt$, $q : \Re^n \to \Re$, $q(x) > 0, \forall x \neq 0, q(0) = 0$, and arises from the solution of a more general version of the HJB equation (Primbs, et al 2000)

$$\frac{\partial V}{\partial x} f - \frac{1}{4}\left(\frac{\partial V}{\partial x} g\right)^2 + q(x) = 0 \tag{2.8}$$

Notice (2.7) and (2.8) are equivalent to (2.6) and (2.5), respectively, for the case of LTI systems. Using the control of (2.7) with a quadratic CLF and the linearized dynamics of the nonlinear system reduces to the LQR control of (2.6). Therefore (2.7) combined with a quadratic CLF and the nonlinear system dynamics yields globally asymptotically stable dynamics with local optimality. In this work, we shall consider (2.6) only for linear controllers and (2.7) only for nonlinear controllers.

2.3 Genetic Algorithm Description

The objective of the genetic algorithm is to maximize the *fitness function* which is used to measure how well the controller performs on X^0. We choose to maximize the minimum rate of convergence while minimizing the maximum control effort on the discrete checking set X_d^0. Therefore, the fitness function is defined as

$$Fitness(E^i) = \frac{w_1}{w_1 + w_2}\left(\frac{\phi_\gamma^i - \underline{\phi}_\gamma}{\overline{\phi}_\gamma - \underline{\phi}_\gamma}\right) + \frac{w_2}{w_1 + w_2}\left(1 - \frac{\phi_u^i - \underline{\phi}_u}{\overline{\phi}_u - \underline{\phi}_u}\right),\ w_1, w_2 > 0 \qquad (2.13)$$

where $\phi_\gamma^i = \min_{x_d \in X_d^0}\left(-\frac{\dot{V}(x_d, E^i)}{V(x_d, E^i)}\right)$, $\phi_u^i = \max_{x_d \in X_d^0}\left(\left\|u\left(x_d, E^i\right)\right\|\right)$, $\underline{\phi}_\gamma = \min_i \phi_\gamma^i$, $\overline{\phi}_\gamma = \max_i \phi_\gamma^i$,

$\underline{\phi}_u = \min_i \phi_u^i$, $\overline{\phi}_u = \max_i \phi_u^i$. The E^i is left in the function arguments to remind the reader that the i^{th} CLF and control are dependent upon the i^{th} matrix E^i defined by (2.4) in the population. The two major components of the fitness function, $\left(\frac{\phi_\gamma^i - \underline{\phi}_\gamma}{\overline{\phi}_\gamma - \underline{\phi}_\gamma}\right)$ and $\left(1 - \frac{\phi_u^i - \underline{\phi}_u}{\overline{\phi}_u - \underline{\phi}_u}\right)$, are considered *sub-fitness functions* and are used to normalize the rate of convergence and control effort to eliminate numerical problems due to scale mismatch between these two quantities. The selection process tends to select species good at achieving both high rate of convergence and low control effort. The weights, *w*, are used to emphasize more selection pressure towards a high rate of convergence, ϕ_γ, or low control effort, ϕ_u. They may be left as constants or changed dynamically as the algorithm progresses. In fact, it was found that pulsing the *w's* dynamically helps control the composition of the GA population so that members of the population or *species* that satisfy one sub-fitness function well are combined with those that satisfy the other sub-fitness function well, speeding up the search process for species that do both tasks well. Future work could quantify this improvement. To aid in understanding the interdependencies of the variables involved with the fitness function, the diagram (Figure 2-1) below illustrates the computational flow of the fitness function.

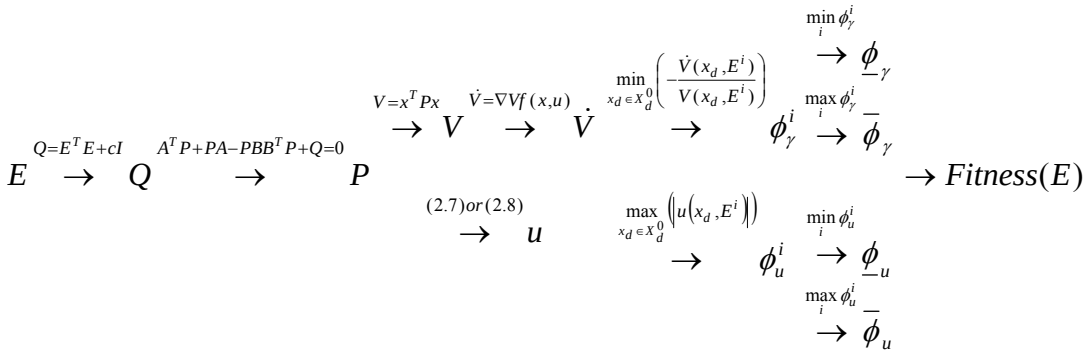

Figure 2-1: Flow of Fitness Computation

The genetic algorithm used in this work, employs the basic evolutionary operators "copy", "crossover", and "mutate". They are defined as

$$Copy(E) = E \tag{2.9}$$

$$Crossover(E^1, E^2) = \beta E^1 + (1-\beta)E^2 \tag{2.10}$$

where β is a uniformly distributed random number on the interval [0 2].

$$Mutate(E) = E + \Delta_E \tag{2.11}$$

where Δ_E is a uniformly distributed random matrix with dimensions matching those of *E*. The elements of Δ_E lie on the interval $\begin{bmatrix} -\sigma_E & \sigma_E \end{bmatrix}$ where σ_E is the user defined mutation step size of *E*. The *Copy* operator (also known as *reproduction* in the genetic algorithm literature) is used to preserve well fit individuals in the population so that they may be used in future generations for generating better individuals via the *Crossover* or *Mutation* operators. The *Crossover* operator takes two highly fit individuals and either creates a new individual that interpolates between the two when $\beta < 1$ or extrapolates along the line connecting the two individuals in *E* search space when $\beta > 1$. The *Mutation* operator takes a single highly fit individual and creates a new individual via randomly perturbing its elements about the hyper-cube of width Δ_E.

To demonstrate the simplicity of imposing constraints on the GA optimization and to further support the use of a genetic algorithm, we place an additional constraint on the control gains. All the elements of the LQR gain vector of (2.6) must be limited in magnitude by some upper bound, $K_{\max}$. This constrains the control effort as well as focuses the search to areas in parameter space that yield physically realizable control signals. For the nonlinear control law (2.7) this upper bound affects the gains of the linearized controller. By (2.4) and (2.5), decreasing the norm of E will decrease the norm of K. Therefore we select the following filter on E to ensure the maximum gain constraint is satisfied:

$$Gain\_Limit\_Filter(E) = \begin{cases} \eta E & \max_{i=1,2\ldots n} |K_i| > K_{\max} \\ E & otherwise \end{cases}, \; 0 < \eta < 1 \qquad (2.12)$$

The genetic algorithm uses this ad-hoc filter to update E such that controller gain limitations are enforced: if a gain is too high, then E is scaled down, indirectly scaling down the entire gain vector.

The GA performs a *parallel search* of the parameter search space (Jamshidi, et. al). That is, rather than search via a single point at a time, as with typical optimization methods, an entire set of points (i.e. the population) are considered at once. Members of the population are randomly selected for evolutionary operation to create the next population (or to move the population as a whole though the search space). To bias the population motion towards "progress", highly fit individuals are more likely to be chosen for evolutionary operation. The members of the population are first arranged by fitness in ascending order. The individuals with the highest fitness are placed at the beginning of the ordered list, denoted $I_{fit} \subset \{1,2,\ldots,N_{pop}\}$, where N_{pop} denotes the size of the

population. The probability density function (*pdf*) for the selection of the i^{th} member of I_{fit} is

$$P\left(I_{fit}^{i} \in Selected\right) = \frac{\left(\frac{i}{N_{pop}}\right)^{s}}{\sum_{j=1}^{N_{pop}}\left(\frac{j}{N_{pop}}\right)^{s}}, \qquad s > 0 \tag{2.14}$$

where *s* acts as a *fitness bias parameter*. The form of the *pdf* of (2.14) was selected for the following reason. By changing *s,* we may adjust the size of the most likely portion of the population we use during the selection process. The higher *s* becomes, the more we restrict selection to highly fit individuals. For example, if *s = 0* then we have a uniform probability of selecting *any* member of I_{fit}. If *s* is very large, the only member of I_{fit} likely selected is the individual with the highest fitness, i.e., the first member in the ordered list I_{fit}.

Once a species is selected, a random decision on which operation to perform must be made. The probability for *Copy*, *Crossover*, and *Mutation* are labeled p_r, p_c, and p_m, respectively. The restrictions for these parameters are

$$\begin{aligned} &p_r, p_c, p_m \geq 0 \\ &p_r + p_c + p_m = 1 \end{aligned} \tag{2.15}$$

Typically we insert the best fit individual into the next generation and set $p_r = 0$. This is because two copies of the same individual are often selected for the *Crossover* operation, which results in another copy of the two same individuals, implicitly implementing the *Copy* function. The usual settings for the evolutionary operation probabilities in this work are $p_r = 0, p_c = p_m = 0.5$.

2.3.1 Making X_d^0 Dynamic

The required size of the checking set X_d^0 grows exponentially with the dimension of the system state, hence the need to strategically assign X_d^0 such that a minimal number of points is used. The objective is to maximize the minimum value of $-\dot{V}/V$ on X_d^0, i.e. maximize $\gamma_{\min}$, and to minimize the maximum value of $|u|$ on X_d^0, i.e. minimize $|u|_{\max}$. For all practical purposes, if we knew where in X^0 these *max/min* and *min/max* conditions occurred, we could define X_d^0 as the set of these points, denoted *critical points*, so that the fitness function is evaluated only at the points that matter, i.e. the critical points. The argument is that we don't have to check all the points in X^0, just the critical points. Unfortunately, the critical points for each species in the population are unknown and are also functions of *(P,K)*, hence functions of E and change after every generation. The clusters of *E's* in any population are perturbations of an *average E* for that cluster, so taking the critical points for each E on the set X_d^0 is a way to create a perturbation cluster of critical points labeled C_d. The set C_d represents the best estimation of where the population of controllers (i.e. the *E's*) are yielding the minimum rate of convergence and maximum control effort on X^0. Therefore, it makes sense to evaluate the controllers of the next generation at these points. Hence, at the beginning of every generation, we redefine the checking set as $X_d^0 = C_d + R_d$: the critical points from the last generation (C_d) plus a set of randomly selected new points, R_d, that act as "exploration points" for finding *better* critical points for the current population. *Better*

means yielding a lower fitness value than any point in C_d. This method reduces the needed size of X_d^0 and increases the speed of the GA.

2.4 Stochastic Estimation of $\gamma_{\min}$

Once a satisfactory *(P,K)* pair is found, the stochastic checking method is used. The sampling method used to estimate the value of $\gamma_{\min}$ (1.18) is based on the Chebyshev inequality of probability theory (Resnick, 1999)(Stark & Woods, 1994). We assume the probability density function (*pdf*) of γ is a uniformly distributed random variable, Γ, with mean, μ_Γ, and variance, $\sigma_\Gamma{}^2$. The *pdf* for Γ is

$$p_\Gamma(z) = \begin{cases} \dfrac{1}{2\sqrt{3}\sigma_\Gamma} & \mu_\Gamma - \sqrt{3}\sigma_\Gamma \le z \le \mu_\Gamma + \sqrt{3}\sigma_\Gamma \\ 0 & \textit{otherwise} \end{cases} \tag{2.16}$$

This type of *pdf* is chosen because it is reasonable to assume we know only that the distribution of Γ is bounded from above and below, when we uniformly sample it on X^0. Future work could use a better estimate of the distribution of Γ by using the actual function $-\dfrac{\dot{V}}{V}$, since it is a known function of *x*.

The Chebyshev inequality may be expressed as

$$P\left[\left|\hat{\mu}_\Gamma - \mu_\Gamma\right| \ge \varepsilon\right] \le \frac{\sigma_\Gamma{}^2}{n\varepsilon^2} \tag{2.17}$$

where $P(A)$ denotes probability of *event A* and

$$\hat{\mu}_\Gamma = \frac{1}{n}\sum_{i=1}^{n}\Gamma_i \tag{2.18}$$

is the maximum likelihood estimate of μ_Γ (Stark & Woods, 1994), i.e. the sample average of Γ. If we knew the value of σ_Γ then we could use (2.17) to specify a *probable* lower bound on γ, which from the *pdf* (2.16) would be

$$\gamma_{\min} = \inf_{x \in X^0}(\gamma) = \hat{\mu}_\Gamma - \varepsilon - \sqrt{3}\sigma_\Gamma \tag{2.19}$$

The lower bound probability of (2.17), denoted $P_{\gamma_{\min}}$, is a free parameter which may be specified by setting *n* large enough in (2.17), i.e.

$$n \geq \frac{\sigma_\Gamma}{P_{\gamma_{\min}} \varepsilon^2} \tag{2.20}$$

We do not have the luxury of knowing σ_Γ, however, such that an upper bound of σ_Γ must be estimated. The maximum likelihood estimate of σ_Γ (Stark & Woods, 1994), denoted $\hat{\sigma}_\Gamma$, is given by

$$\hat{\sigma}_\Gamma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\Gamma_i - \hat{\mu}_\Gamma)^2} \tag{2.21}$$

along with its uncertainty, $u_{\hat{\sigma}}$, may be used to express the total value of σ_Γ, given by

$$\sigma_\Gamma = \hat{\sigma}_\Gamma + u_{\hat{\sigma}} \tag{2.22}$$

By Taylor series expansion of continuous functions, an approximation for the bound of $u_{\hat{\sigma}}$ (for sufficiently small ε) is

$$|u_{\hat{\sigma}}| \leq \left|\frac{\partial \hat{\sigma}_\Gamma}{\partial \hat{\mu}_\Gamma}\right| \varepsilon \tag{2.23}$$

where the term "$\left|\frac{\partial \hat{\sigma}_\Gamma}{\partial \hat{\mu}_\Gamma}\right|$" may be thought of as the *sensitivity* of $\hat{\sigma}_\Gamma$ to the $\hat{\mu}_\Gamma$ estimate, and ε as the uncertainty of $\hat{\mu}_\Gamma$. The absolute value of $\frac{\partial \hat{\sigma}_\Gamma}{\partial \hat{\mu}_\Gamma}$ is used to make a conservative

estimate of σ_Γ. Computing $\left|\frac{\partial \hat{\sigma}_\Gamma}{\partial \hat{\mu}_\Gamma}\right|$ and substituting it, along with the right sides of (2.21) – (2.23) into (2.20) yields

$$n^2 \geq \frac{\sum_{i=1}^{n}\left(\Gamma_i - \hat{\mu}_\Gamma\right)^2 + \varepsilon\left|\sum_{i=1}^{n}\left(\Gamma_i - \hat{\mu}_\Gamma\right)\right|}{P_{\gamma_{\min}}\varepsilon^2} \tag{2.24}$$

The result is a transcendental inequality, which when satisfied, guarantees that for small enough ε,

$$\gamma_{\min} = \hat{\mu}_\Gamma - \varepsilon - \frac{\sqrt{3}}{n}\left(\sum_{i=1}^{n}\left(\Gamma_i - \hat{\mu}_\Gamma\right)^2 + \varepsilon\left|\sum_{i=1}^{n}\left(\Gamma_i - \hat{\mu}_\Gamma\right)\right|\right) \tag{2.25}$$

with probability $1 - P_{\gamma_{\min}}$. To solve (2.24) we specify the desired values of ε and $P_{\gamma_{\min}}$, and guess a value for *n*. The right side of (2.24) is then computed and the inequality is checked. If the left side is greater than the right side, a larger value of *n* is chosen. This process repeats until *n* is large enough. After (2.24) is satisfied, (2.25) is computed.

If $\gamma_{\min}$ is positive, then we have, with probability $1 - P_{\gamma_{\min}}$, an exponentially stable control law on the domain $X \subset X^0$ with a Lyapunov function to prove it. Equation (2.25) in practice turns out to be an overly conservative estimate of $\gamma_{\min}$, however. This is because Γ is not truly a uniformly distributed random variable. It is a nonlinear function of a uniformly distributed random variable, making Γ a non-uniformly distributed random variable (Stark & Woods, 1994). We can roughly estimate $\gamma_{\min}$ directly by taking the lowest value of Γ while sampling X^0. Therefore, a better estimate of $\gamma_{\min}$ can be made by computing the uncertainty of the estimate and using it to compute a worst case $\gamma_{\min}$. Hence we define the sample minimum of Γ, $\Gamma_{\min}$ given by

$$\Gamma_{\min} = \min_{x \in X_d^0} (\Gamma) \tag{2.26}$$

and the estimate of $\gamma_{\min}$ as

$$\gamma_{\min} = \Gamma_{\min} \tag{2.27}$$

We have used the stochastic arguments above to derive a reasonable estimate of the size of the checking set for post-GA numerical estimation of the minimum rate of convergence. We shall assume that this checking set is sufficient for checking the maximum magnitude of the control signal as well. Note that for *pdf's* with substantially higher order moments (moments beyond the mean and variance), we expect this estimate to be invalid. It is, however, a good starting point for the size of the checking set and could be increased until the estimates of $\gamma_{\min}$ and $|u|_{\max}$ do not change substantially. Randomly sampling $\gamma_{\min}$ and $|u|_{\max}$ as discussed above, along with a few numerical simulations usually suffices as a confidence builder to the control engineer that the controller is indeed stable and with the allowable maximum control effort.

2.4.1 GA Flow-chart

The flow chart of the algorithm is given below in Figure 2-2. At the beginning of the algorithm, a randomly generated set of *E's* is inserted into the first population (i.e. generation 0). The set of *E's* is used to compute a set of *(P,K)* pairs, which in our case means solving the *HJB* equation (2.5) for *P* and using it to solve for *K* depending on the selection of the control law (2.6) or (2.7). The fitness computation of Figure 2.1 follows, and the fitness of the population is used to select the best individuals for performing the evolutionary operations defined in (2.9) through (2.11). The *E* with the highest fitness denoted "best *E*" is returned along with the corresponding *(P,K)* pair. The controller

performance, i.e. minimum rate of convergence and maximum control effort are numerically estimated by randomly sampling X^0 using the sample size defined by (2.24). If the performance is good enough then the CLF and controller are accepted as a solution to the optimization problem or used possibly in some other performance simulation. If the performance is unacceptable, the GA parameters are tuned or the controller performance requirements are relaxed and the process is repeated.

In the next chapter we use the ideas outlined above to tune the linear and nonlinear controllers of (2.6) and (2.7), respectively, for a set of nonlinear systems that are relatively difficult to control.

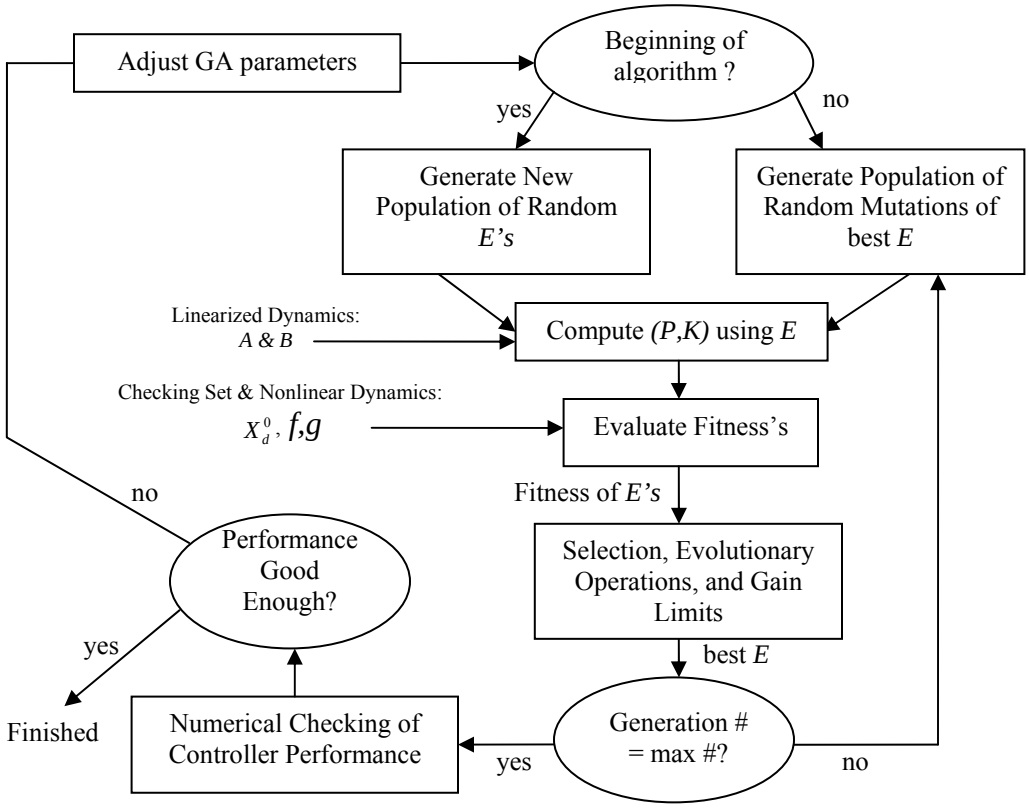

Figure 2-2: GA Optimization Algorithm

3 Full State-Feedback CLF Control

3.1 Selection of Benchmark Examples

The purpose of this chapter is to demonstrate how to use the ideas of Chapter 2 to tune full-state feedback CLF controllers. The reader is reminded that the discussion is limited to the two controller types of (2.6) and (2.7) re-listed below:

Linear (2.6): $$u = -B^T P x \tag{3.1}$$

Nonlinear (2.7): $$u = \begin{cases} -\dfrac{x^T Pf + \sqrt{\left(x^T Pf\right)^2 + \left(x^T Qx\right)\left(x^T Pg\right)^2}}{x^T Pg} & x^T Pg \neq 0 \\ 0 & x^T Pg = 0 \end{cases} \tag{3.2}$$

where P is the solution to the algebraic Ricatti equation $A^T P + PA - PBB^T P + Q = 0$ using the linearized dynamics *(A,B)* of the nonlinear control-affine system defined by $\dot{x} = f(x) + g(x)u$, $x \in \Re^n, u \in \Re, f, g : \Re^n \to \Re^n$, and the selected Q matrix defined by the quadratic integral performance index $\int_0^\infty x^T Qx + u^2 dt$.

The following example systems are taken from Ngamsom (2001). The theory developed by Ngamsom is based partially on maximizing the region of attraction (region of stability) of nonlinear systems using linear control laws. The drawback of the method proposed by Ngamsom is the controller is synthesized using an uncertain linearized model of a nonlinear system. Therefore, the theory is based on linear control of nonlinear systems, i.e. only local information about the system dynamics and worst case

assumptions on the effects of the nonlinearities are used. The controller design intent is to find a controller tune specific to the nonlinear system that is less conservative than the tune based on robust linear systems theory. We use the genetic algorithm described in Chapter 2 to tune linear controllers (3.1) and compare with those of Ngamsom. For nonlinear control, we use the augmented Sontag's universal control law (3.2) from Primbs, et al (2000) because of its inherent global stability and local inverse optimality. However, we show how the genetic algorithm's tuning of the CLF and quadratic performance index $q(x)=x^TQx$ results in using less control effort than the CLF resulting from linear dynamics-based methods. The main reason for the selection of the nonlinear control law is that it converges to the linear control law in a sufficiently small neighborhood about the origin. This allows the P matrix of Ngamsom's linear control method to be used directly in the nonlinear control law for an additional mode of comparison between Ngamsom's controller and the genetic algorithm tuned CLF controller. Ngamsom's linear control method happens to be a very effective design procedure for a robust linear controller. It is considered a good benchmark for the methods proposed in this thesis.

3.1.1 Comparing GAC to LARC

To facilitate comparison between Ngamsom's work and ours, we shall refer to the controller and CLF resulting from the method proposed herein as the "Genetic Algorithm Controller" (GAC). The GAC implementing the control law from (3.1) shall be referred to as the linear GAC (LGAC) and the GAC implementing the control law from (3.2) shall be referred to as the nonlinear GAC (NGAC). The linear controller that maximizes the region of stability is called the Lyapunov Attractive Region Controller (LARC), after

Ngamsom. Similar to the GAC terminology, the LARC implementing the control law from (3.1) shall be referred to as the linear LARC (LLARC) and the LARC implementing the control law from (3.2) shall be referred to as the nonlinear LARC (NLARC).

Ngamsom's design method is a systematic method for designing linear controllers for a class of nonlinear systems. The method uses the linearized dynamics of the nonlinear system along with assumptions on how the nonlinearities, treated as uncertainties, affect the linearized system. We observe that the size of the region of stability is a "side-effect" of the LARC. That is, the region of stability of the LARC is not explicitly selected – it is an inherent result of the combined nonlinear system and linear controller. The proposed GAC method offers the ability to explicitly select the region of stability, in addition to direct tuning of both the minimum rate of convergence and maximum control effort. This freedom comes with a price, however. For all of the examples examined in this study, the region of stability for the GAC was smaller than the region of stability for the LARC overall. On the other hand, the GAC allows the flexibility to select where in state space to attempt to achieve stability, as well as the ability to vary the rate of convergence while varying the control effort in this region.

3.1.2 Displaying Results

We display the performance of our example controllers in the region defined by X^0 in a set of Tables. The Tables list the estimated values for the minimum rate of convergence $\gamma_{\min}$, the mean rate of convergence μ_γ, the standard deviation of the rate of convergence σ_γ, the maximum control effort magnitude $|u|_{\max}$, the mean control effort $\mu_{|u|}$, and the standard deviation of control effort $\sigma_{|u|}$. These quantities are estimated by uniform random sampling in X^0. Equation (2.24) is used to determine a sufficient sample

size. For the second order system of the first example, we include 3D plots in X^0 of the level sets of the Lyapunov function, the value of γ, and the locations of the estimated critical points defined as the location estimates where $\gamma_{\min}$ and $|u|_{\max}$ occur for every controller in the GA population. The 3D plots aid in visualizing the effects the CLF shape has on the rate of convergence and the locations of the critical points. We leave blank the rate of convergence plot where it is negative or very close to the origin where it is numerically ill-conditioned. The blank spots where the rate of convergence is negative depict "holes" where the system becomes unstable with respect to the specific CLF.

3.2 Example 1: Artificial System

The equations of motion for the artificial 2nd order system considered by Ngamsom (2001) are:

$$\dot{x} = \begin{bmatrix} 10x_1 + 10x_2 + 10\sin(x_1)^2 \sin(x_2) - x_1^2 \\ 5x_1^2 + \sin(x_2) \end{bmatrix} + \begin{bmatrix} 0 \\ \cos(x_2) + 1.5 \end{bmatrix} u, \quad \begin{matrix} x = [x_1 \quad x_2]^T \in \Re^2 \\ u \in \Re \end{matrix} \tag{3.3}$$

The linearized dynamics about the origin are:

$$\dot{x} = \begin{bmatrix} 10 & 10 \\ 0 & 1 \end{bmatrix} x + \begin{bmatrix} 0 \\ 2.5 \end{bmatrix} u \tag{3.4}$$

Table 3-1 displays the results from the LARC method. The Lyapunov function matrix *P*, the linear control gains *K*, and the matrix for the quadratic performance index *Q*, are listed. The structure of *Q* in Ngamsom's work is typically fixed as a diagonal matrix, and the norm of *Q* is tuned via an optimization method. One advantage of the proposed GAC method lies in its ability to find a more general *Q* (non-zero off-diagonal elements) to optimize the performance index.

P		K		Q	
4160.8	125.6	-157.49	-68.244	16000	0
125.6	54.4			0	16000

Table 3-1: LLARC Parameters for System (3.3) (Ngamsom, 2001)

Table 3-2 displays the randomly sampled performance of the linear LARC in Table 3-1 on the artificial system (3.3). The range of X^0 is selected to be $x_1 \in [-25 \;\; 25]$ and $x_2 \in [-100 \;\; 100]$. Note the negative value for $\gamma_{\min}$, implying instability.

X^0 Range	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u \rvert_{\max}$	$\mu_{\lvert u \rvert}$	$\sigma_{\lvert u \rvert}$
$x_1 \in [-25 \;\; 25]$ $x_2 \in [-100 \;\; 100]$	-76.935	47.769	80.6	10709	3805.6	2515.5

Table 3-2: Performance of LLARC on System (3.3)

The randomly sampled performance of the nonlinear LARC on the artificial system (3.3) is displayed in Table 3-3. The minimum rate of convergence is positive by design, and the control effort is very high.

X^0 Range	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u \rvert_{\max}$	$\mu_{\lvert u \rvert}$	$\sigma_{\lvert u \rvert}$
$x_1 \in [-25 \;\; 25]$ $x_2 \in [-100 \;\; 100]$	1.0294	135.75	128.340	1.4878×10^7	17928	1.6251×10^5

Table 3-3: Performance of NLARC on System (3.3)

Table 3-4 displays the randomly sampled performance of the LLARC on the linearized artificial system dynamics. This Table is used to quantify the local performance of both controllers about the origin. The Table allows the comparison between the linear and nonlinear behavior of the system and helps quantify the effects the nonlinearities have on the system performance.

X^0 Range	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u \rvert_{\max}$	$\mu_{\lvert u \rvert}$	$\sigma_{\lvert u \rvert}$
$x_1 \in [-25 \;\; 25]$ $x_2 \in [-100 \;\; 100]$	3.7641	91.364	93.507	10709	3796.5	2518.9

Table 3-4: Performance of LLARC on Linearized System (3.4)

Consider the level sets of the CLF using *P* from Table 3-1 in Figure 3-1. The shape and orientation of the elliptical "rings" determine the path of the state trajectory when under the CLF control law. A true CLF control law forces the state trajectory to cross into successively smaller rings. The trajectory in Figure 3-1 actually increases the CLF value initially, yet eventually converges to the origin. This behavior explains the negative value for $\gamma_{\min}$ in Table 3-2. The CLF is therefore only a local CLF and not a CLF on X^0. The system is unstable with respect to the CLF, but may be locally stable for some other Lyapunov function; perhaps one whose level sets are turned slightly more counter-clockwise so that the state trajectory does not cross into a higher level set.

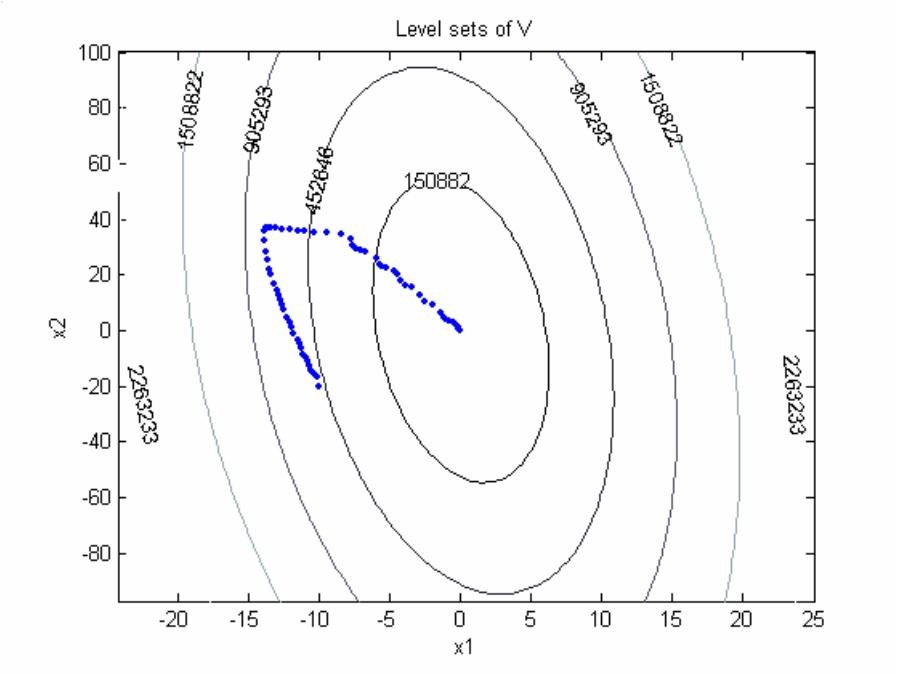

Figure 3-1: Level Sets of *V* for LLARC on System (3.3)

The rate of convergence of the CLF from Figure 3.1 is displayed in Figure 3-2. The points in X^0 where $\gamma < 0$ or points close to the origin that are numerically ill-conditioned due to division by a very small number, are left blank. The points where $\gamma < 0$ are points where the trajectory crosses into larger level sets and exit X^0, possibly never to return. We see that the region where the trajectory crosses into successively higher level sets, the rate of convergence is negative as expected.

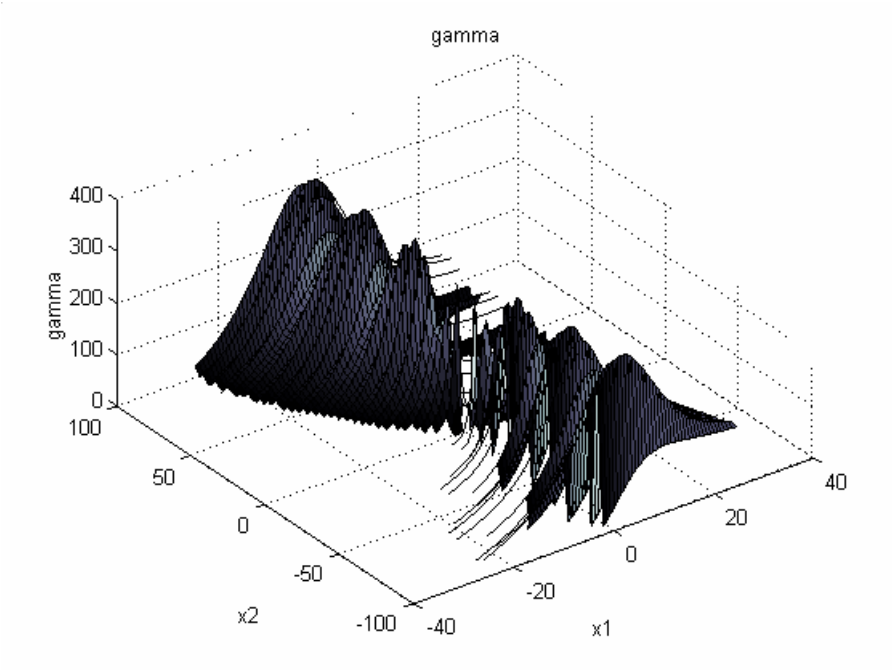

Figure 3-2: "Gamma" (γ) for LLARC on System (3.3)

Figure 3-3 is an estimated region of attraction for the controller reported in Ngamsom (2001) using Monte Carlo simulation. The shaded blocks represent regions that yield a stable trajectory when used as the region of the initial state. All other blocks represent regions that yield an unstable trajectory when used as the region of the initial state. A Lyapunov function with level sets that line up with the edge of stability in Figure 3-3 (where white and black boxes are in contact) would be a better selection to prove the

true region of stability for the artificial system (3.3) via Lyapunov Stability Theory. The values for γ here are negative, supporting the results of the Monte Carlo simulation used to generate Figure 3-3. The objective of the GA optimization is to make all values of γ positive and large while minimizing the amount of control effort it takes to do so.

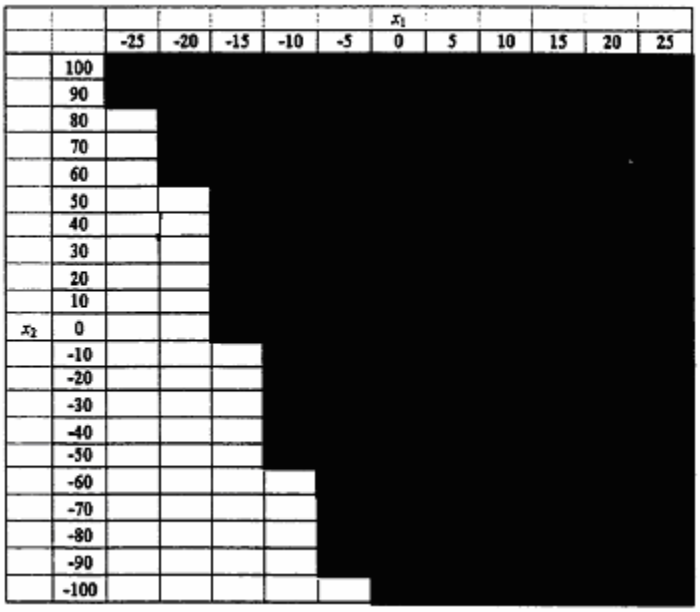

Figure 3-3: Stability Table from Ngamsom (2001)

Figure 3-4 and Figure 3-5 display the states and control signal for the linear LARC on the artificial system (3.3) for the initial condition (-10,-20). The states converge shortly after 0.2s while $|u|_{\max}$ reaches approximately 2700.

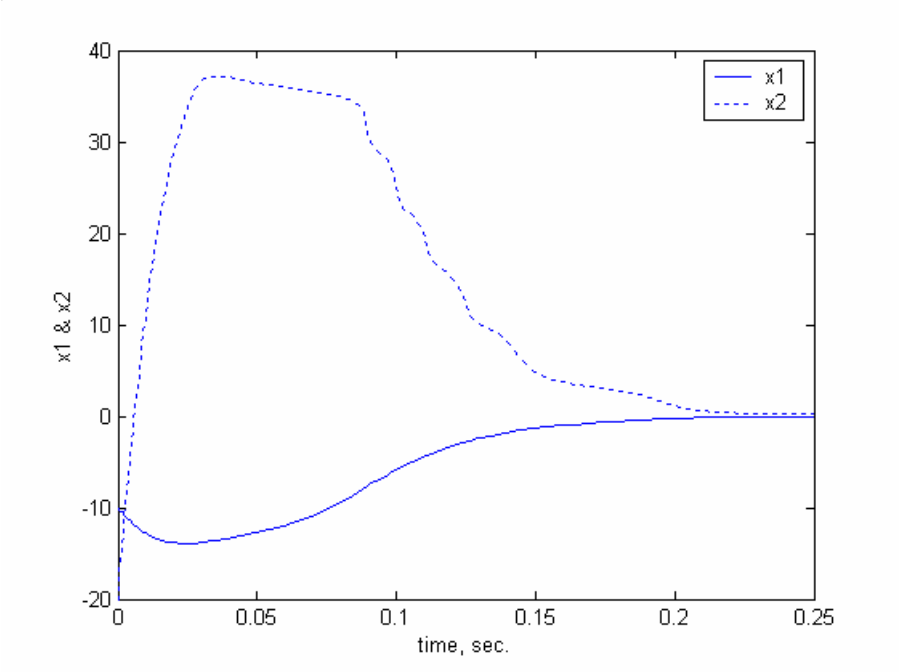

Figure 3-4: States of System (3.3) Using LLARC

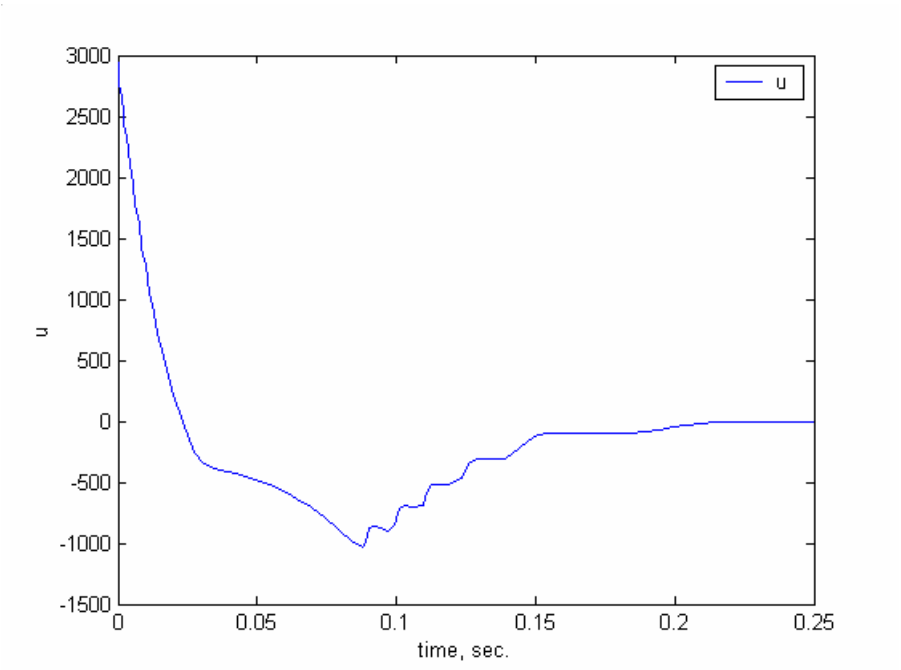

Figure 3-5: Control Signal of LLARC on System (3.3)

Figure 3-6 displays the level sets of the CLF for the nonlinear LARC and the state trajectory of the artificial system (3.3) for the initial condition (-10,-20). The state

trajectory is forced to cross successively lower levels of the CLF and yields very sudden discrete "switch-like" direction changes. Such behavior is due to $\dot{V}$ having a strict equality to a function of the states ($\dot{V} = \sqrt{\left(x^T Pf\right)^2 + \left(x^T Qx\right)\left(x^T Pg\right)^2}$). Viewing Figure 3-7, we see that the rate of convergence is positive everywhere by design of the control law. The peaks in the γ plot correspond to the fastest rates of change and they are evident in the state transition plot of Figure 3-8. The control law forces γ to be strictly positive so that the part of the trajectory in Figure 3-1 under the linear LARC that crosses into higher levels of the CLF is now steered inward toward successively lower levels of the CLF. Figure 3-8 and Figure 3-9 display the states and control signal for the (-10,-20) initial condition trajectory. The states do not converge much faster in the nonlinear LARC case than with the linear LARC case, however the maximum magnitude of x_2 is smaller for the nonlinear LARC because the trajectory is forced to stay inside the level set corresponding to the initial condition. The control effort is of course much larger for the nonlinear LARC. Figure 3-9 shows the cost of the renewed stability by using the nonlinear controller instead of the linear controller – excessively large control effort. The need for reorienting the CLF level sets to help decrease the large control effort is made evident in this example.

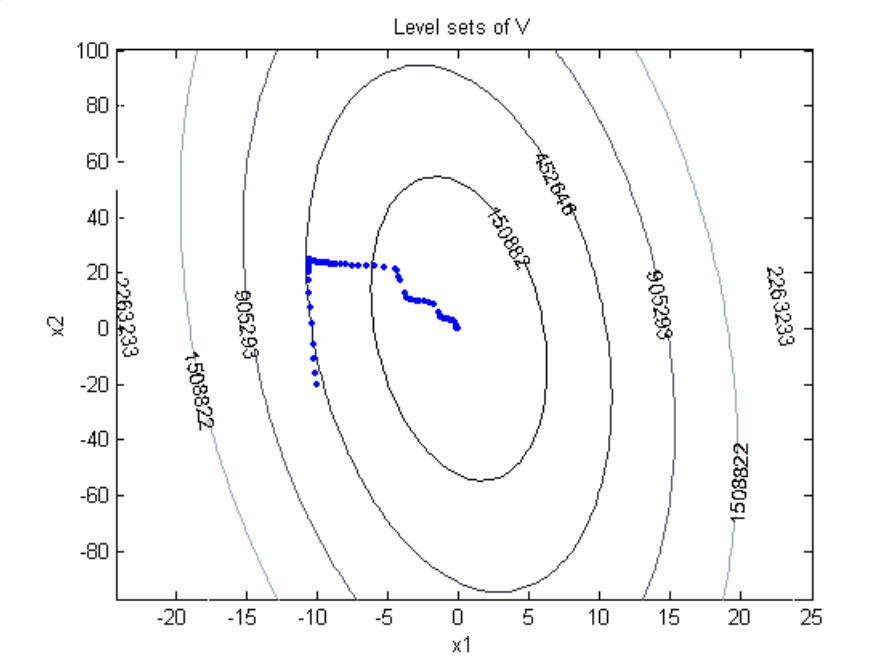

Figure 3-6: Level Sets of *V* for NLARC on System (3.3)

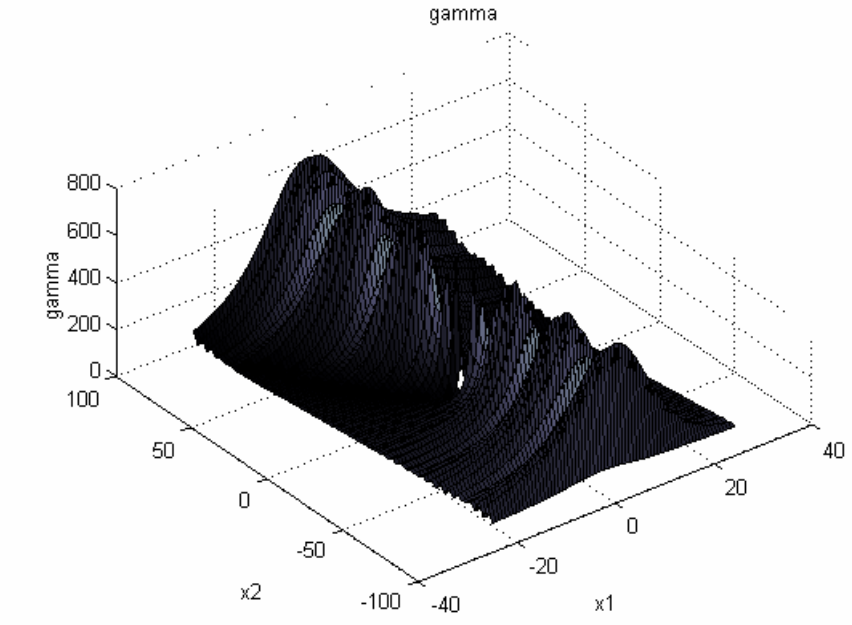

Figure 3-7: "Gamma" (γ) for NLARC on System (3.3)

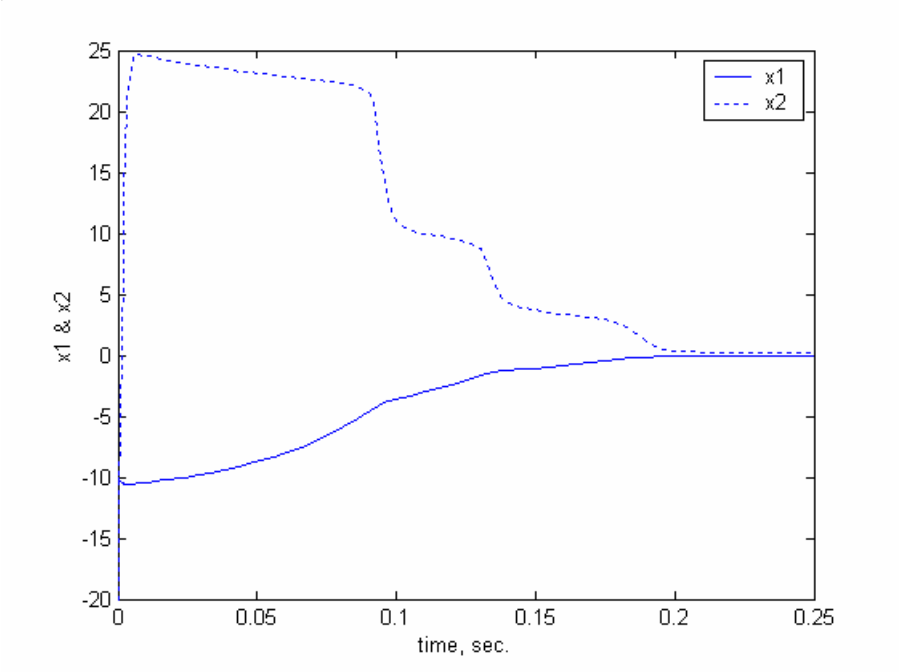

Figure 3-8: States of System (3.3) Using NLARC

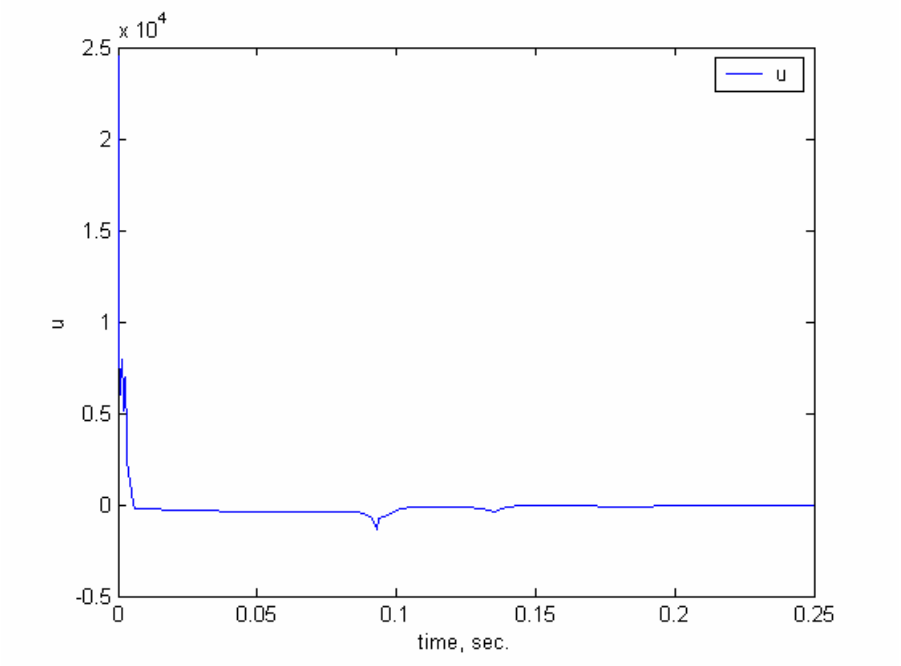

Figure 3-9: Control Signal of NLARC on System (3.3)

The LARC method does not directly use information about the rate of convergence nor information about the control effort of the nonlinear control system. In

fact, P is not explicitly treated as the matrix for the quadratic CLF, $V = x^T Px$, but it may be viewed as such. The following question is the primary motivation for this work.

Question 3.1

Can we achieve better control performance over that of Ngamsom (2001) for the nonlinear system in terms of minimum rate of convergence and maximum control effort in X^0 *by using the CLF,* $V = x^T Px$, *and directly checking these performance measures at all points on* X_d^0 *to maximize the performance index defined by the fitness function (2.13)?*

We now attempt to answer Question 3.1 by applying the proposed GAC method to the example systems.

3.2.1 GAC Case #1: Linear GAC

The genetic algorithm parameters for case #1 and the resulting GAC performance are given Table 3-5. This case implements linear control. The GA parameters are described in Chapter 2 and in the appendix where they are used in the Matlab code that implements the GA optimization procedure.

System: Artificial (3.3)	Controller Type: linear	Generations: 50	Population Size: 50	Checking points: 100
Critical Points: 33	Δ_E: 100	$K_{\max}$: 1000	c: 0	X^0 Range: $x_1 \in \lfloor -25\ 25 \rfloor$ $x_2 \in \lfloor -100\ 100 \rfloor$

Table 3-5: GA Parameters to Optimize LGAC for System (3.3)

The GA results of the optimization of the linear GAC for the artificial system (3.3) are listed in Table 3-6. The W vector contains the weighting factors used in the fitness function defined in (2.13) to vary the weight between rate of convergence and

control effort. The results in Table 3-6 include the three sets of weights $W = [1 \quad 0]$, $W = [1 \quad 1]$, and $W = [0 \quad 1]$. They represent 3 cases where the GA selects population members with a large $\gamma_{\min}$ and ignores $|u|_{\max}$, large $\gamma_{\min}$ and small $|u|_{\max}$, and lastly a small $|u|_{\max}$ while ignoring $\gamma_{\min}$, respectively. The reader is reminded that ultimately, the GA is tuning *Q* to yield a favorable set of *P* and *K*. Therefore comparing GAC to LARC begins with comparing their respective *Q* matrices. As can be seen the *Q's* of the linear GAC are very different than the *Q's* of the linear LARC. An obvious difference is the non-diagonal terms in the set of *Q's* resulting from the GA optimization.

Weight Case	Weights	*P*	*K*	*Q*
1	W=[1 0]	2768.8 347.3 347.3 71.8	-868.23 -179.38	$6.9845x10^5$ $1.1141x10^5$ $1.1141x10^5$ 20450
2	W=[1 1]	2993.0 352.2 352.2 62.1	-880.42 -155.16	715290 102800 102800 16910
3	W=[0 1]	3.9147 3.5413 3.5413 3.5308	-8.8532 -8.8269	0.0854 0.0452 0.0452 0.0267

Table 3-6 Optimized Parameters of LGAC for System (3.3)

Table 3-7 lists the randomly sampled performance of the GAC for the 3 weight cases. As one might expect, $\gamma_{\min}$ decreases and $|u|_{\max}$ increases as the emphasis on minimum rate of convergence shifts to an emphasis on maximum control effort. Unfortunately, this is not always the case as we shall see in the later examples. The first two cases in Table 3-7 yielded a much better performance than the LARC on X^0 for the minimum rate of convergence, but not for maximum control effort. Only the third case yielded a better value for maximum control effort, however the third case yielded an unstable system and therefore is not a fair comparison to the LARC control effort. Except for the last case, the controller gains are much larger for the GAC than the LARC. The design objective of achieving a simultaneous better rate of convergence and better

maximum control effort over the LARC is not met. Though not explored here, it is posited that a weight case that is somewhere between cases 2 and 3 may result in a controller that achieves the design objective or at least yield a controller closer to the design objective.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	34.564	301.74	208.37	39514	13381	9303.7
2	W=[1 1]	29.737	269.47	174.48	37392	12829	8786.4
3	W=[0 1]	-319.66	6.3853	76.592	1099.2	449.21	270.73

Table 3-7: Performance of LGAC on System (3.3)

Table 3-8 lists the randomly sampled performance of the GA controller on the linearized system (3.4). All minimum rates of convergence are positive, as expected for the linear system, because the linear GAC is a linear quadratic regulator. From Table 3-8 we observe that the local rate of convergence is faster, indicating the system nonlinearities work against the controller's effort to stabilize the system.

Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
W=[1 0]	76.5938	522.89	230.55	3933.4	13250	9287.8
W=[1 1]	92.0664	470.25	174.51	3730.0	12781	8801.2
W=[0 1]	0.05950	20.733	3.6937	1096.7	449.46	270.17

Table 3-8: Performance of LGAC on Linearized System (3.4)

Figure 3-10, Figure 3-11, and Figure 3-12 contain plots of the level sets of the CLF with the state trajectory for initial condition (-10,-20), the rate of convergence γ, and the locations of the estimated critical points for all controllers in the GA population. We see that $\gamma > 0$ on all of X^0, i.e. there are no "holes" in the γ plot unlike that of the LARC. Many critical points exist in the top left side of Figure 3-12 where the value for γ is low (see Figure 3-11) for many members of the GA population. Had we not used the

critical point estimation and just selected a fixed or random grid as the checking set X_d^0, many of these points would have been missed and much of the processing time would have been spent checking irrelevant points with a high γ value or a low $|u|_{\max}$ value. The number of critical points was set to 33. The placement of these points by the critical point search algorithm (Chapter 2 – "*Making* X_d^0 *Dynamic*") is displayed in Figure 3-12. The total number of points is 100; three times the number of critical points. The critical point searching algorithm clustered the majority of the critical points in the top left corner. It is reasonable to assume the spacing of these points is close to the spacing required to satisfy Theorem 1.1. If this is the case, then it would require many more points than the 100 search points used in X_d^0 to satisfy the spacing requirements of Theorem 1.1, hence the search procedure would be much slower. The heuristic formulated in Chapter 2 for moving the search points around to find the critical points seems to be effective.

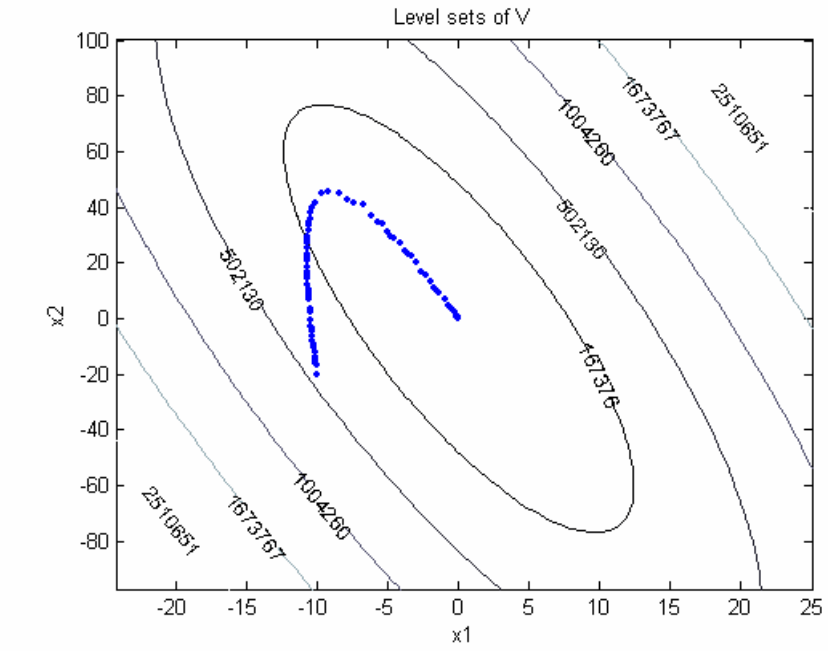

Figure 3-10: Level Sets of *V* for LGAC on System (3.3) (W=[1 0])

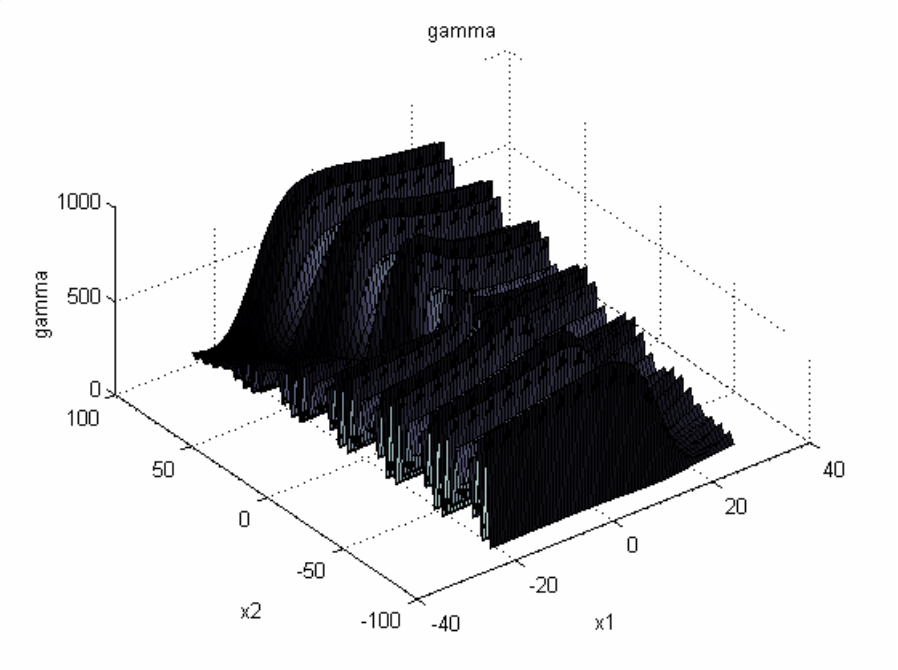

Figure 3-11: "Gamma" (γ) for LGAC on System (3.3) (W=[1 0])

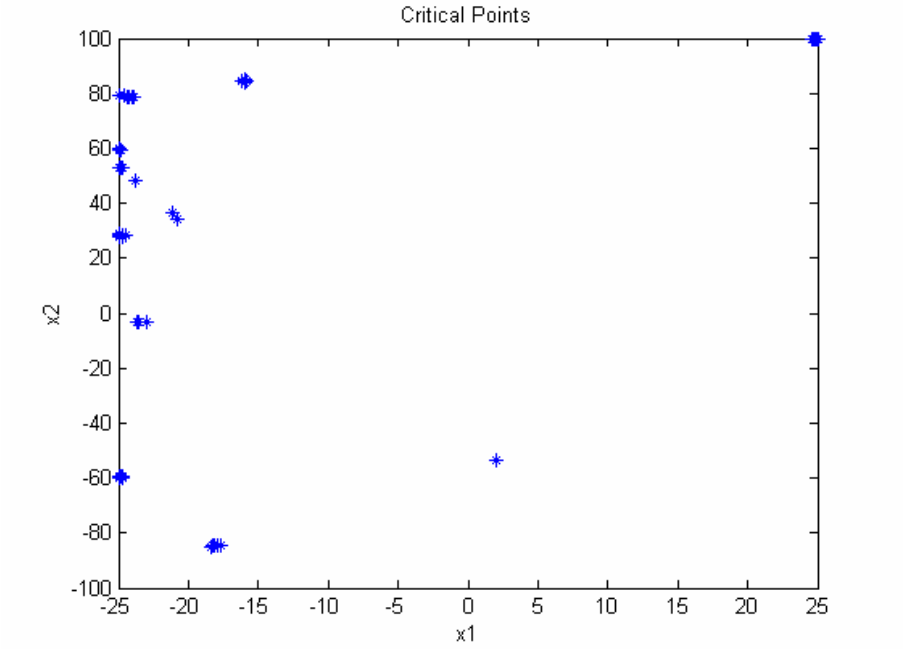

Figure 3-12: Critical Points for LGAC Population Using System (3.3) (W=[1 0])

It is apparent from Figure 3-8, Figure 3-9, Figure 3-13 and Figure 3-14 that the time of convergence is much faster for the linear GAC with W=[1 0] than the linear

LARC, but the control effort is higher. Such a result is acceptable because no weight was placed on the control effort during the optimization. Another notable observation is how the trajectory of Figure 3-10 uniformly crosses into successively lower level sets, even though the control law is linear and does not possess the restrictive effect on the state trajectories as the nonlinear controller. Figure 3-15 through Figure 3-19 are very similar to Figure 3-10 through Figure 3-14. This is expected given that the optimization yielded very similar sets of control gains. The control effort is slightly smaller because the control effort has an equal weight to the rate of convergence rather than zero in the previous case. It is interesting how sensitive the rate of convergence and control effort are to the weights. Figure 3-20 through Figure 3-24 are the results of not considering the rate of convergence during the optimization (i.e. W=[0 1]). For W=[0 1], a very low control effort is achieved but for much of X^0 we have $\gamma < 0$. Because we have open level sets, even if $\gamma > 0$, the trajectory can exit X^0 into the region of state space where the performance has not been checked and no stability guarantees can be made. Although more research must be performed, it is reasonable to assume that some weight vector between W=[1 1] and W=[0 1] (setting w_1 between 0 and 1) might yield a stable controller with a smaller value for $|u|_{\max}$ than the linear LARC. Such a nonlinear dependence on W for the performance measures makes tuning W a nontrivial task.

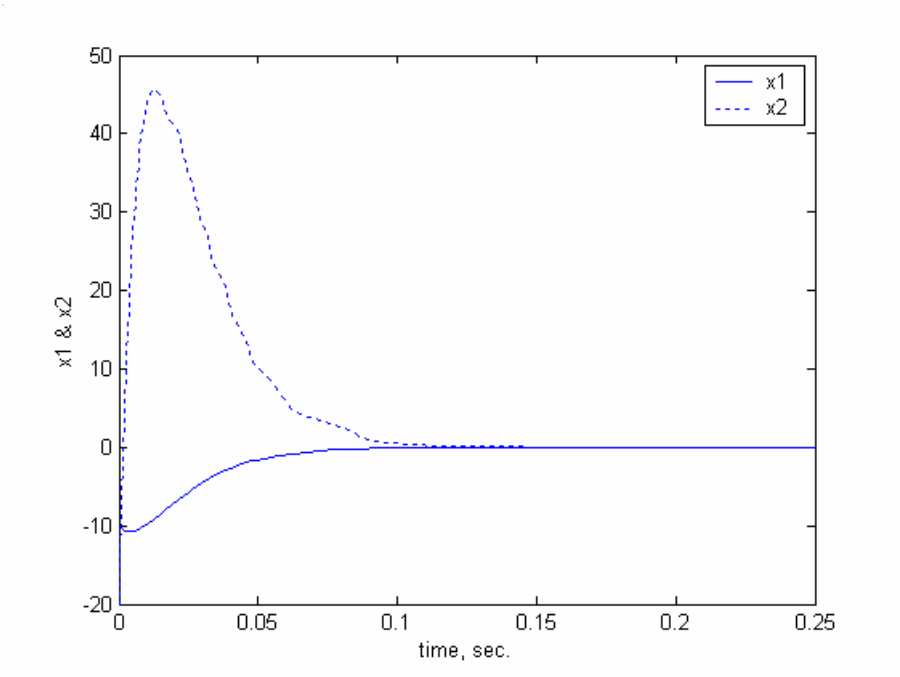

Figure 3-13: States of System (3.3) Using LGAC (W=[1 0])

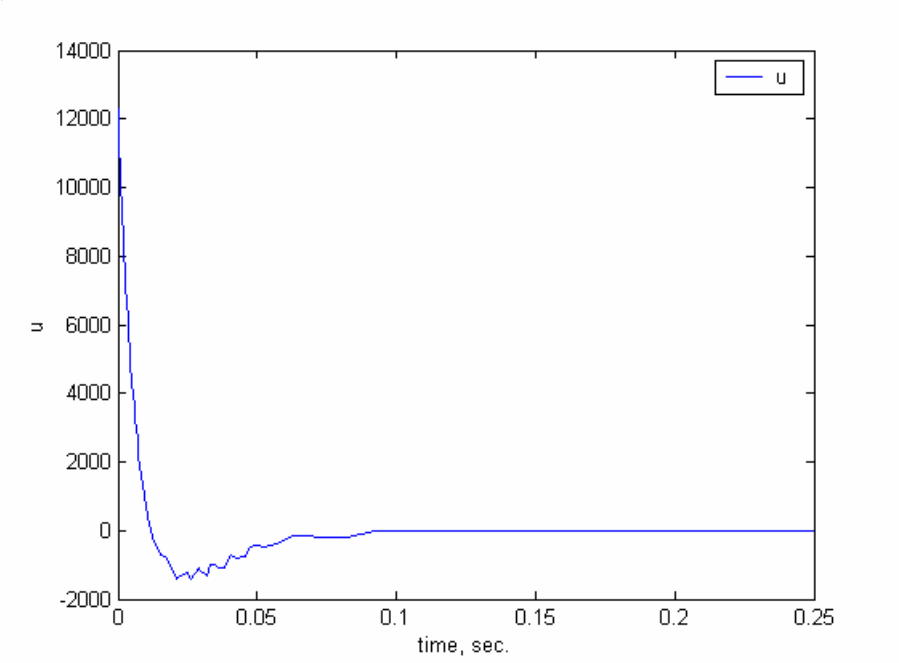

Figure 3-14: Control Signal of LGAC on System (3.3) (W=[1 0])

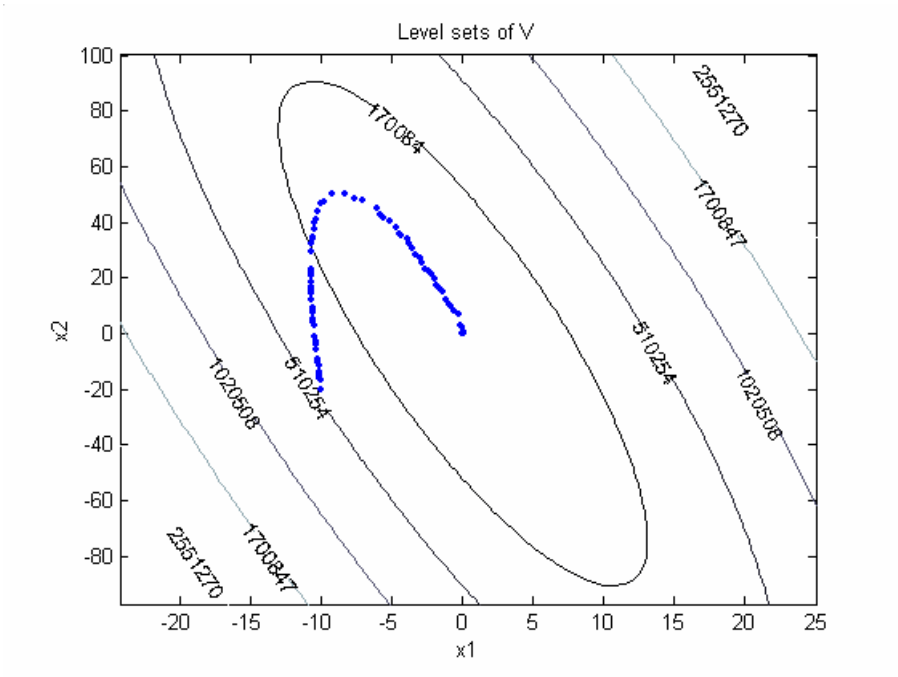

Figure 3-15: Level Sets of *V* for LGAC on System (3.3) (W=[1 1])

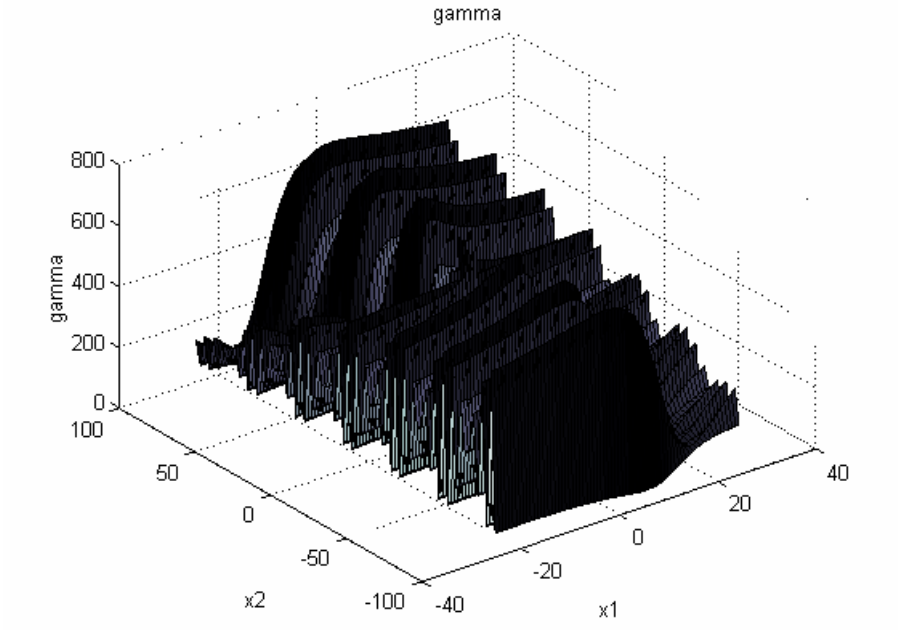

Figure 3-16: "Gamma" (γ) for LGAC on System (3.3) (W=[1 1])

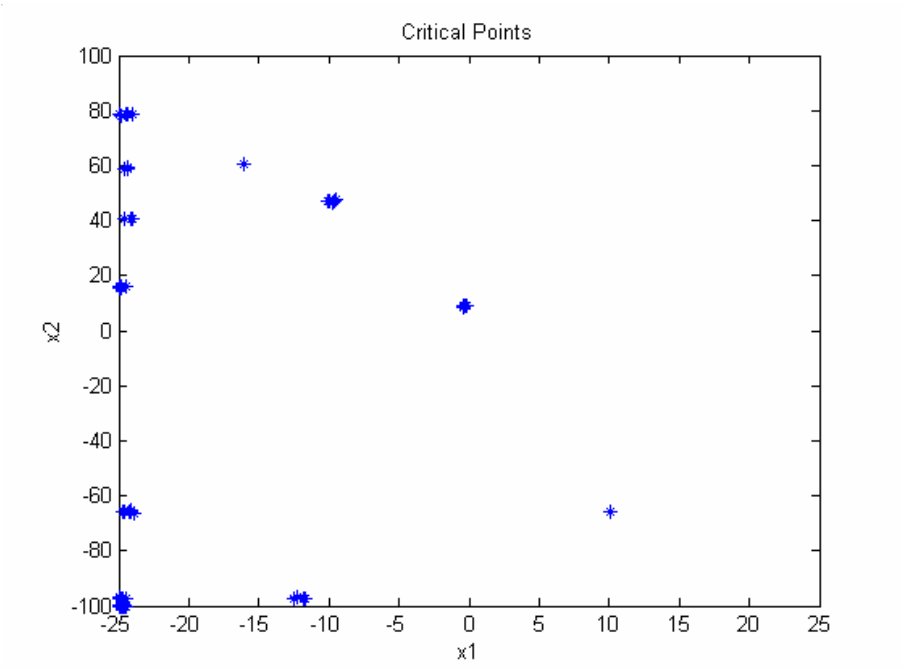

Figure 3-17: Critical Points for LGAC Population Using System (3.3) (W=[1 1])

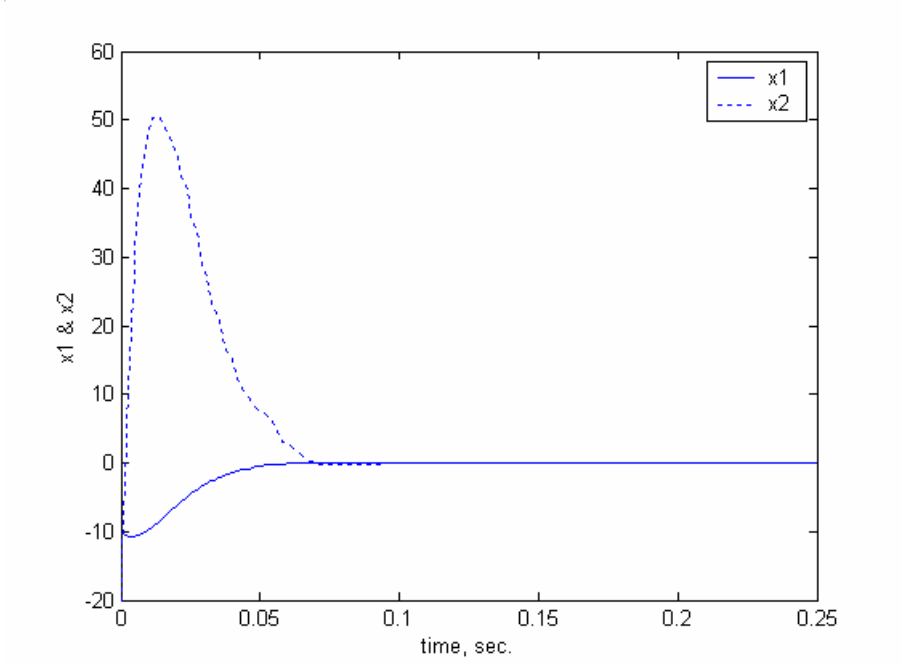

Figure 3-18: States of System (3.3) Using LGAC (W=[1 1])

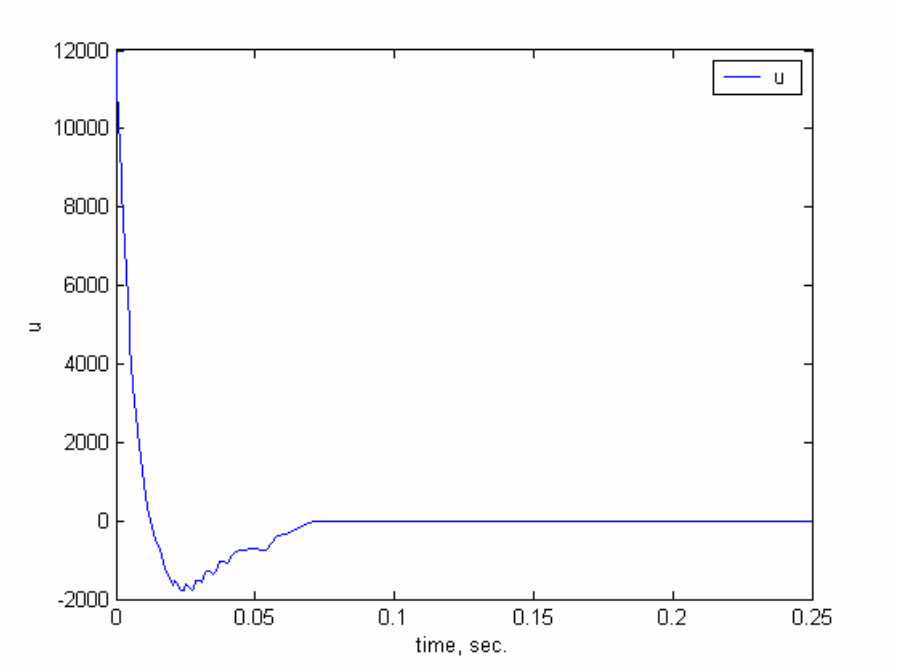

Figure 3-19: Control Signal Profile of LGAC on System (3.3) (W=[1 1])

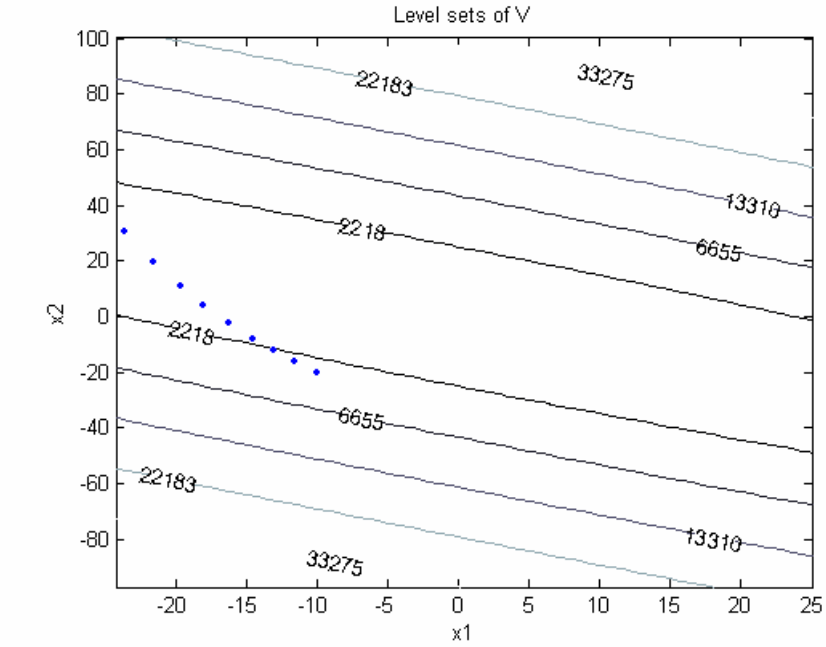

Figure 3-20: Level Sets of *V* for LGAC on System (3.3) (W=[0 1])

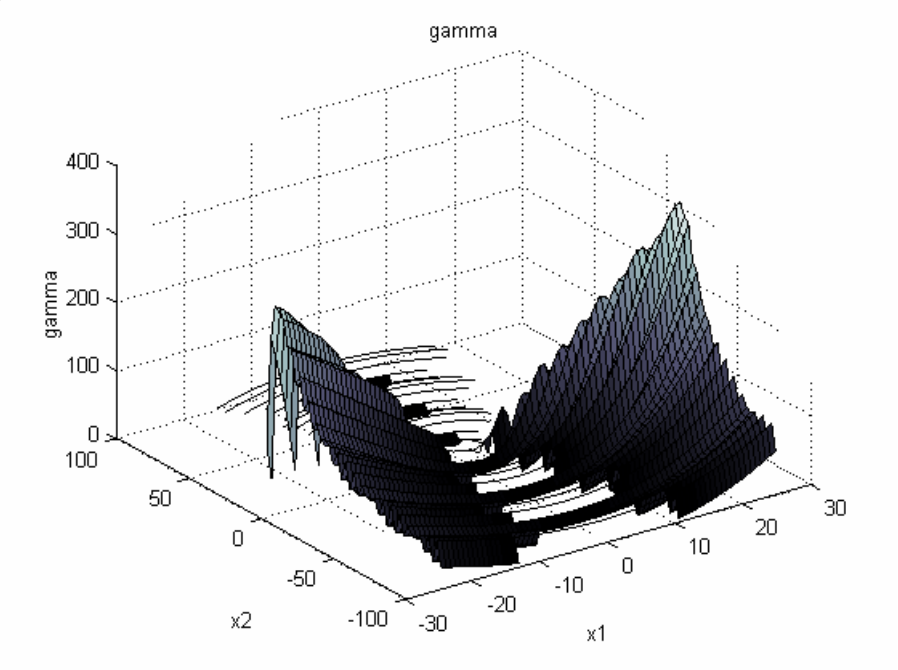

Figure 3-21: "Gamma" (γ) for LGAC on System (3.3) (W=[0 1]).

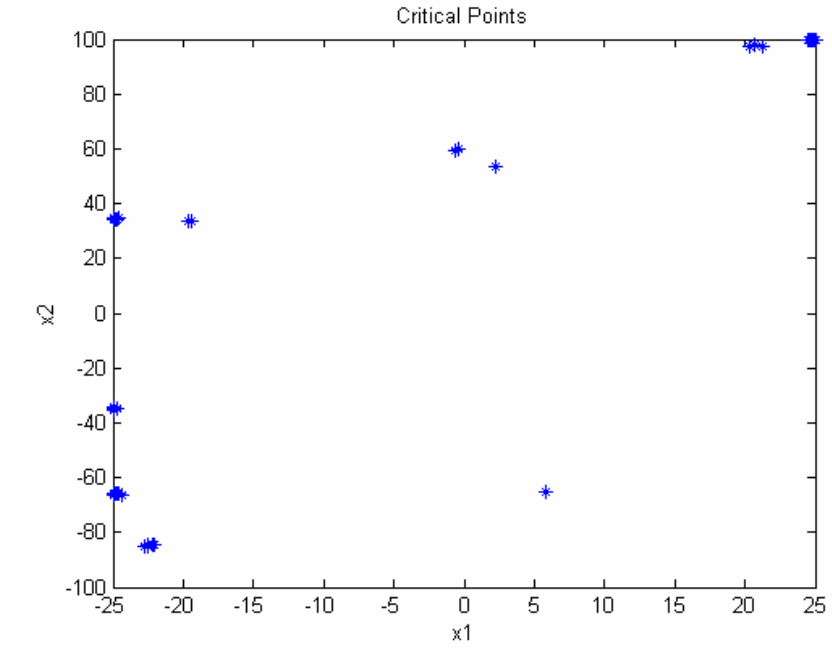

Figure 3-22: Critical Points for LGAC Population Using System (3.3) (W=[0 1])

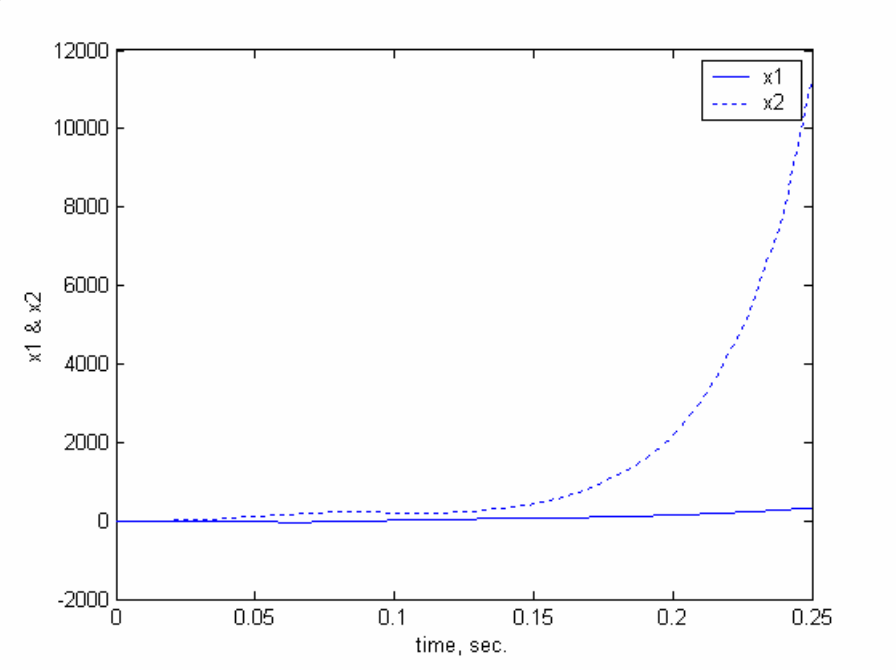

Figure 3-23: States of System (3.3) Using LGAC (W=[0 1])

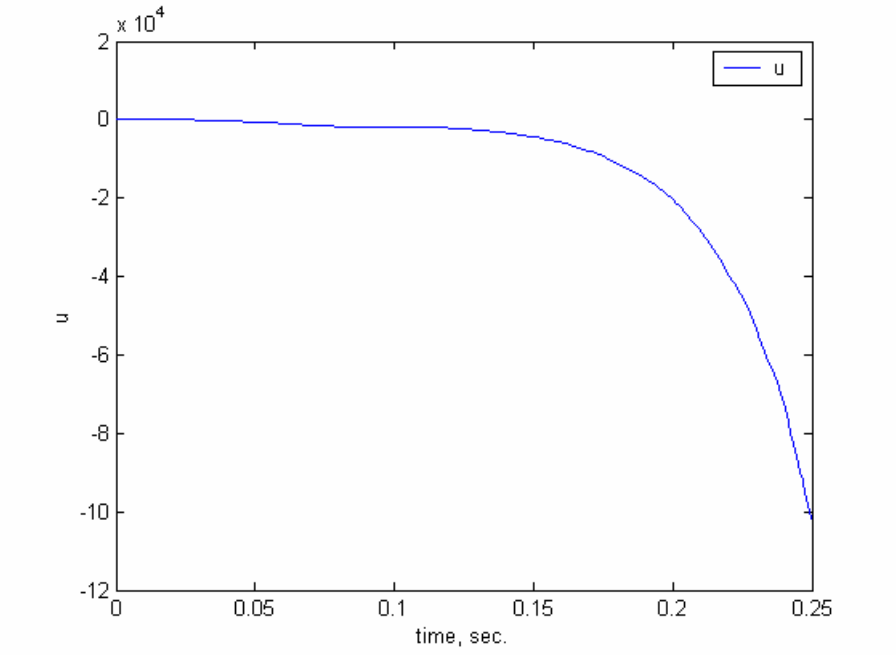

Figure 3-24: Control Signal of LGAC on System (3.3) (W=[0 1])

Summarizing the comparison between the linear LARC and the linear GAC on the artificial system (3.3), we make the following observations. The linear GAC yields a strictly positive γ on X^0 for the two instances that the rate of convergence has an effect on the fitness function (W=[1 0] & W=[1 1]). The minimum rate of convergence is higher for the linear GAC for the weight settings W=[1 0] and W=[1 1], but the maximum control effort is also much higher. The linear GAC for the weight setting W=[0 1] yields a much lower maximum control effort than the linear LARC but the controller is unstable which raises the question: With the correct setting of W can both $\gamma_{\min}$ and $|u|_{\max}$ be improved on X^0? Although further research would be needed, viewing the trend of $\gamma_{\min}$ and $|u|_{\max}$ with the settings of *W* in Table 3-7, it is reasonable to assume such a weight setting exists.

3.2.2 GAC Case #2: Nonlinear GAC

We now consider nonlinear control of the artificial system (3.3). The genetic algorithm parameters for this case are given in Table 3-9.

System: Artificial (3.3)	Controller Type: nonlinear	Generations: 50	Population Size: 50	Checking points: 100
Critical Points: 33	Δ_E: 100	$K_{\max}$: 1000	c: 0	X^0 Range: $x_1 \in \lfloor -25\ 25 \rfloor$ $x_2 \in \lfloor -100\ 100 \rfloor$

Table 3-9: GA Parameters to Optimize NGAC for System (3.3)

Table 3-10 lists the results of the optimization of the nonlinear GAC for the artificial system (3.3) using the parameters from Table 3-9. As with the case of the linear GAC, the GA found non-diagonal solutions for *Q* in all three cases.

Weight Case	Weights	*P*	*K*	*Q*
1	W=[1 0]	292.952 239.153 239.153 195.615	-597.88 -489.04	$3.5160x10^5$ $2.8683x10^5$ $2.8683x10^5$ $2.3398x10^5$
2	W=[1 1]	1664.4 348.60 348.60 77.100	-871.55 -192.71	$7.2631x10^5$ $1.3836x10^5$ $1.3836x10^5$ $0.2661x10^5$
3	W=[0 1]	3.9109 3.5506 3.5506 3.5455	-8.8766 -8.8638	0.5750 0.5136 0.5136 0.4625

Table 3-10: Optimized Parameters of NGAC for System (3.3)

Table 3-11 displays the randomly sampled performance of the nonlinear GAC on the artificial system (3.3). Here we see that the rates of convergence are positive even for the case where the rate of convergence is not a factor in the fitness function (*W*=[0 1]). The control effort is, however, much higher for the nonlinear GAC than the linear GAC, similar to the comparison between the LARC and the nonlinear controller using *P* from the LARC optimization. A much better $\gamma_{\min}$ is achieved along with a much better $|u|_{\max}$ using the nonlinear GAC, but the nonlinear GAC has a higher average control effort than the nonlinear LARC. If such a feature is undesirable to the control engineer, the GA fitness function may be easily augmented with a term that favors low average control effort, although more points in X_d^0 would be required to get a realistic estimate of the average control effort.

Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\|u\|_{\max}$	$\mu_{\|u\|}$	$\sigma_{\|u\|}$
W=[1 0]	12.685	1446.3	679.85	68306	25642	15525
W=[1 1]	7.8767	476.99	251.86	51756	14837	10615
W=[0 1]	0.0384	53.520	57.022	62062	1394.0	2103.7

Table 3-11: Performance of NGAC on System (3.3)

Table 3-12 presents the randomly sampled performance of the nonlinear GAC on the linearized artificial system (3.4). In all three cases, the local performance substantially differs from that across the entire set X^0.

Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u \rvert_{\max}$	$\mu_{\lvert u \rvert}$	$\sigma_{\lvert u \rvert}$
W=[1 0]	2.3226	2394.2	169.28	6362.4	25038	15347
W=[1 1]	45.683	786.90	205.97	40986	13668	9612.2
W=[0 1]	0.0287	20.980	3.6402	1102.7	452.00	269.85

Table 3-12: Performance of NGAC on Linearized System (3.4)

Figure 3-25, Figure 3-26, and Figure 3-27 contain plots of the CLF with the state trajectory for the initial condition (-10,-20), the rate of convergence γ, and the locations of the estimated critical points for all controllers in the GA population. The GA yielded open level sets on X^0. The sampled performance yielded $\gamma_{\min} = 2.3226$, however the true value is clearly negative given the unstable trajectory. This is a problem that occurs when the ratio $\underline{\lambda}(P)/\overline{\lambda}(P)$ is small (e.g. $\underline{\lambda}(P)/\overline{\lambda}(P) = 4.68x10^{-4}$ for the W=[1 0] case). The level sets are nearly open and γ is actually slightly negative at some points along the line that the trajectory follows to exit X^0 in Figure 3-25. Trajectories originating in this region are pulled into the crevice of negative γ values and follow it to the outside of X^0. The points along this line ("escape manifold") are hard to find with a small checking set, causing the GA to use a bad estimate of the true value of $\gamma_{\min}$ when evaluating the controllers. One solution is to simply increase the number of checking points in X_d^0. The next GA optimization run ("Case #3") in the discussion is based on this approach. The ultimate solution, recommended for future research, is to guarantee X^0 contains only closed level sets. That is, guarantee that $V(x) = V(y), \forall x, y \in \partial X^0$.

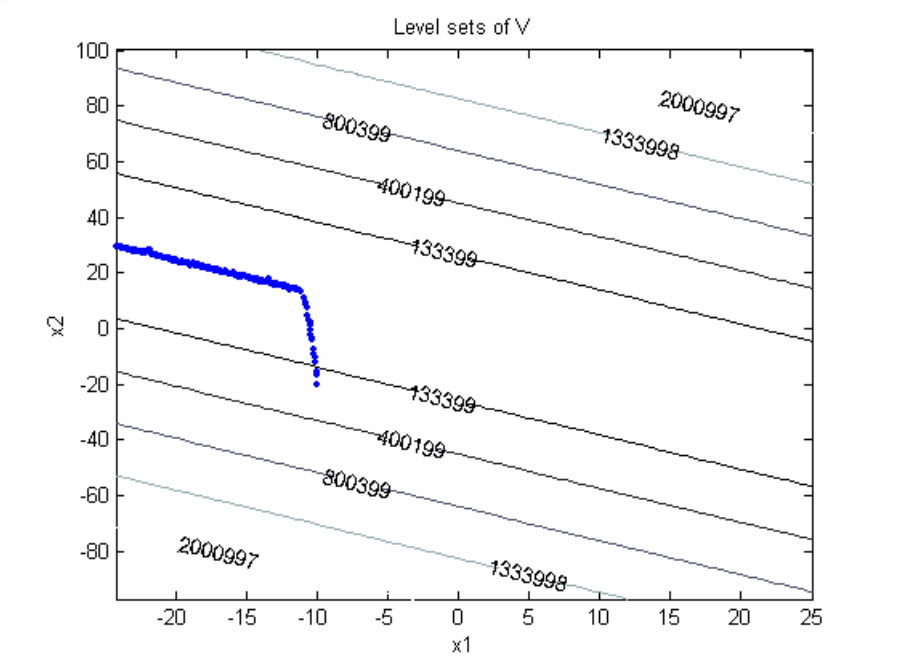

Figure 3-25: Level Sets of *V* for NGAC on System (3.3) (W=[1 0])

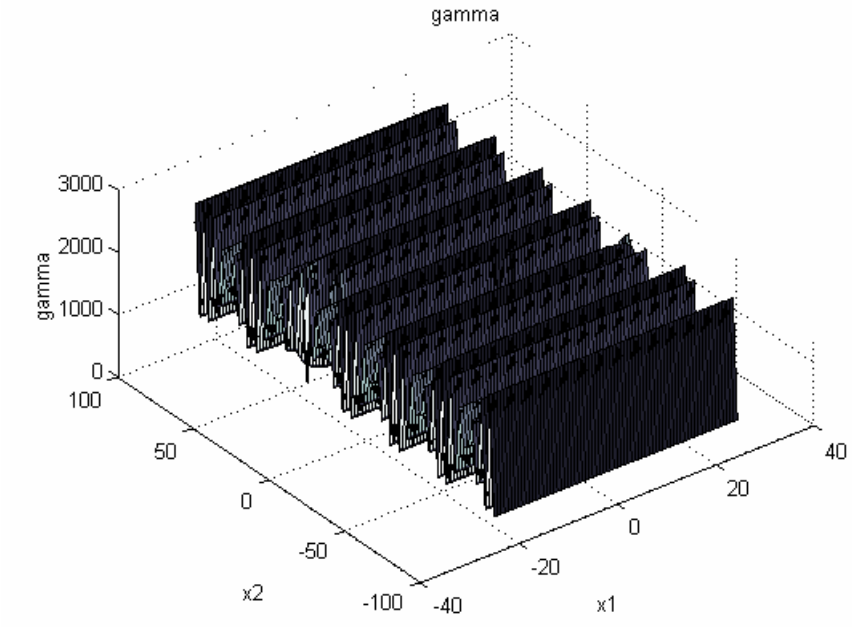

Figure 3-26: "Gamma" (γ) for NGAC on System (3.3) (W=[1 0])

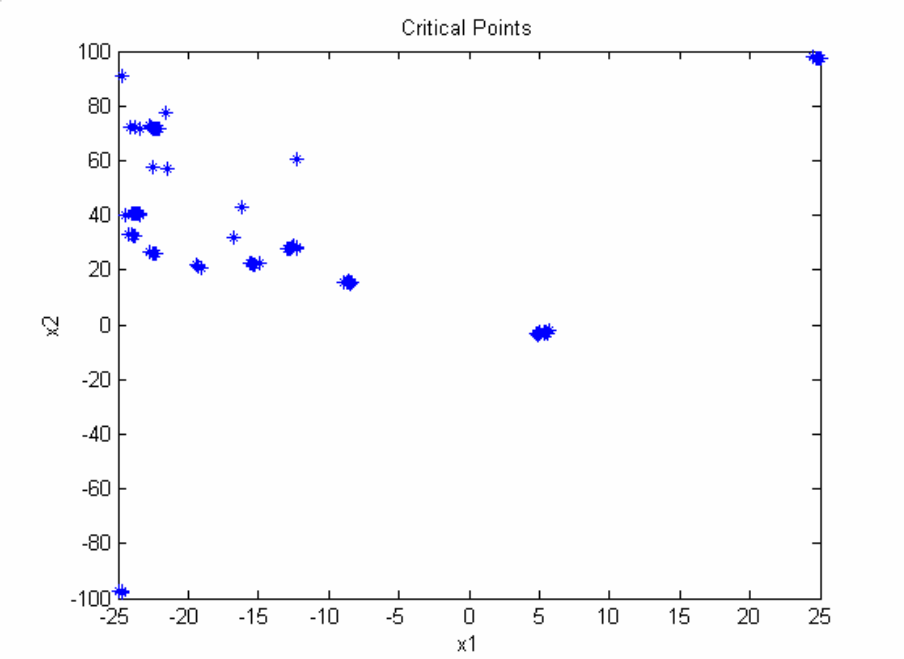

Figure 3-27: Critical Points for NGAC Population Using System (3.3) (W=[1 0])

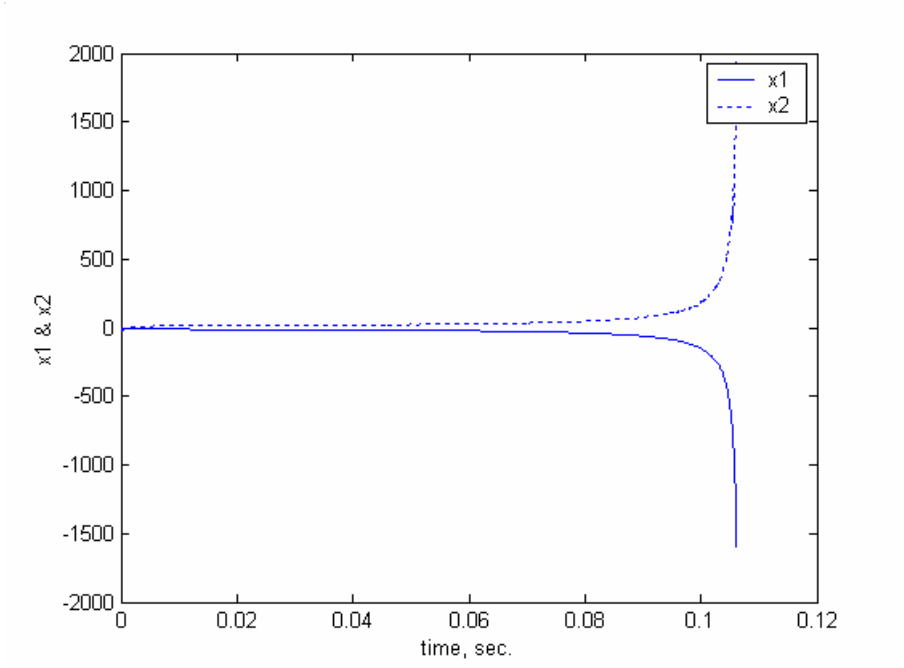

Figure 3-28: States of System (3.3) Using NGAC (W=[1 0])

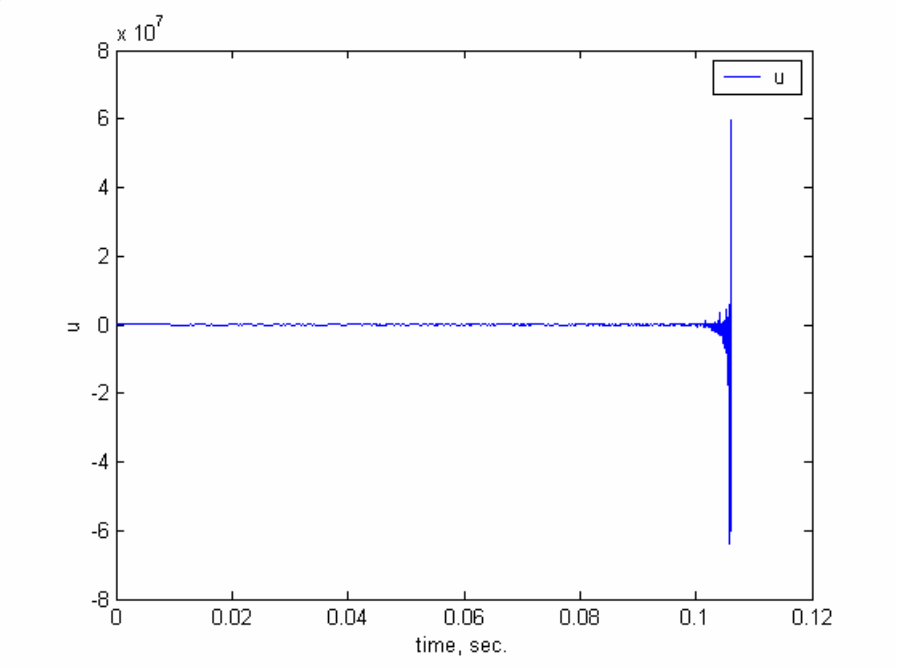

Figure 3-29: Control Signal of NGAC on System (3.3) (W=[1 0])

The case with W=[1 1] (Figure 3-30 through Figure 3-34) yields a stable trajectory. In fact, the nonlinear GAC with W=[1 1] yields a trajectory that converges about twice as fast with about half the maximum control effort than the nonlinear LARC. The design objective is therefore carried despite the problem of not using a large enough checking set X_d^0.

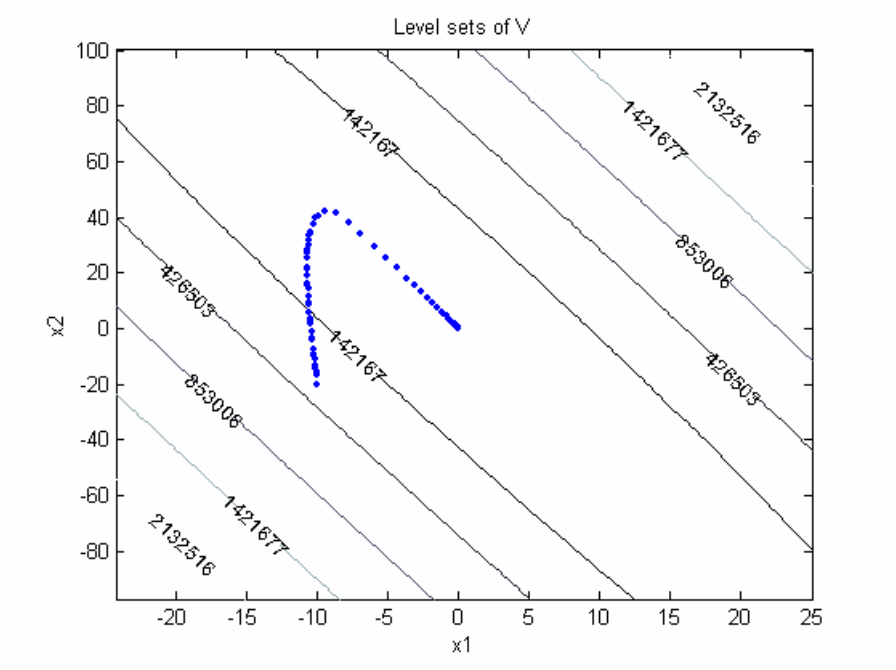

Figure 3-30: Level Sets of *V* for NGAC on System (3.3) (W=[1 1])

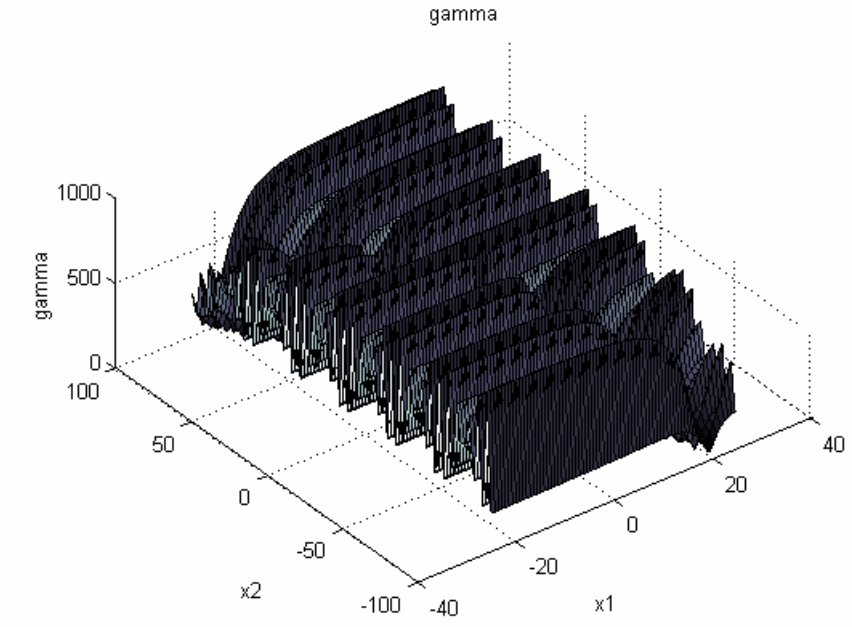

Figure 3-31: "Gamma" (γ) for NGAC on System (3.3) Using W=[1 1]

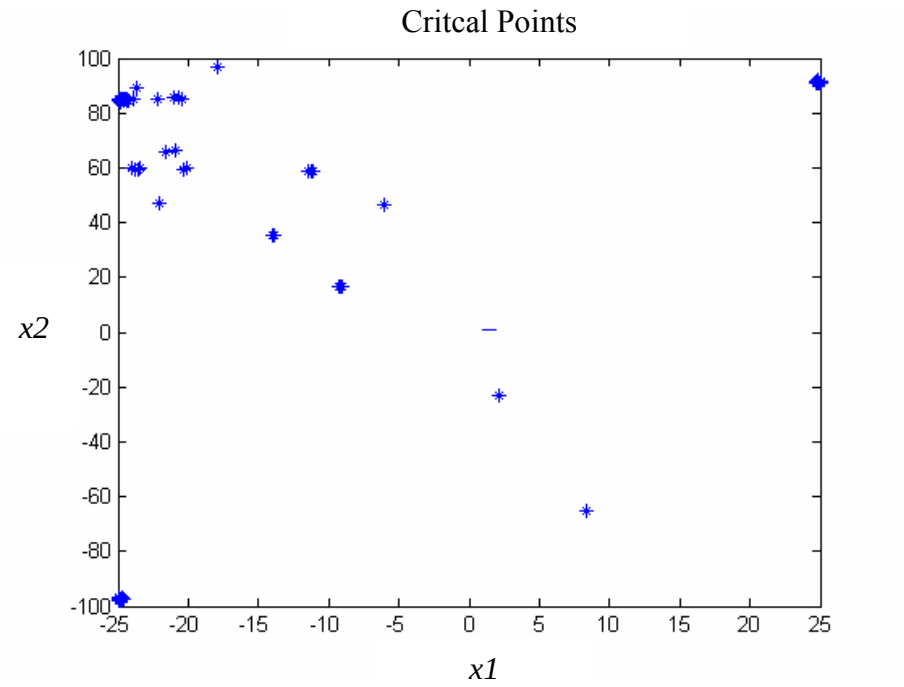

Figure 3-32: Critical Points for NGAC Population Using System (3.3) (W=[1 1])

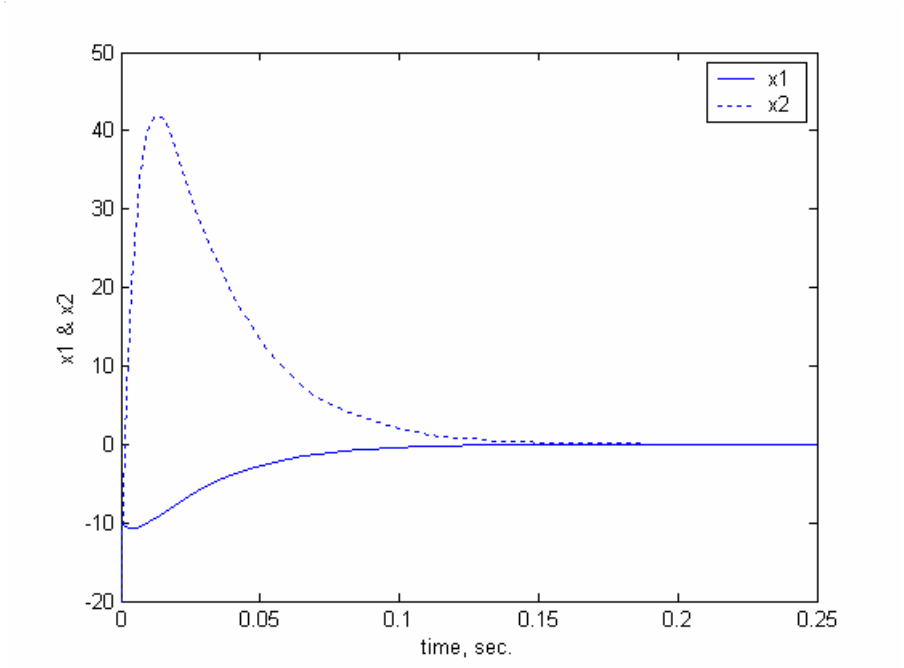

Figure 3-33: States of System (3.3) Using NGAC (W=[1 1])

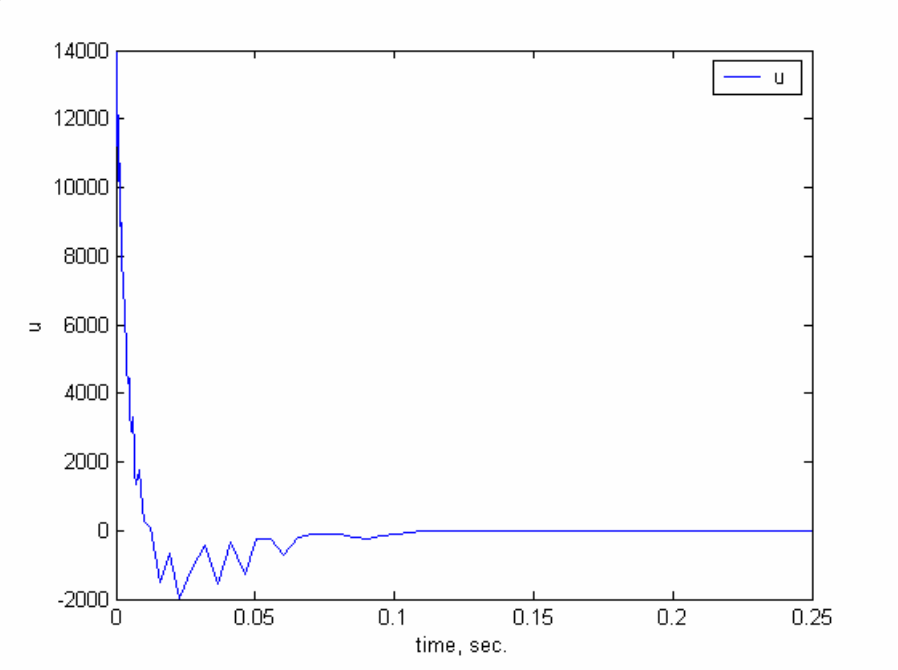

Figure 3-34: Control Signal of NGAC on System (3.3) (W=[1 1])

Weight case 3 suffers from the same problem as weight case 1 (Figure 3-35 through Figure 3-39). However, using more search points or guaranteeing $V(x) = V(y), \forall x, y \in \partial X^0$ would not necessarily guarantee a stable controller since $\gamma_{\min}$ has zero weight in the fitness function. Case 3 is used to show the GA's ability to minimize the maximum control effort of the given CLF and controller topologies.

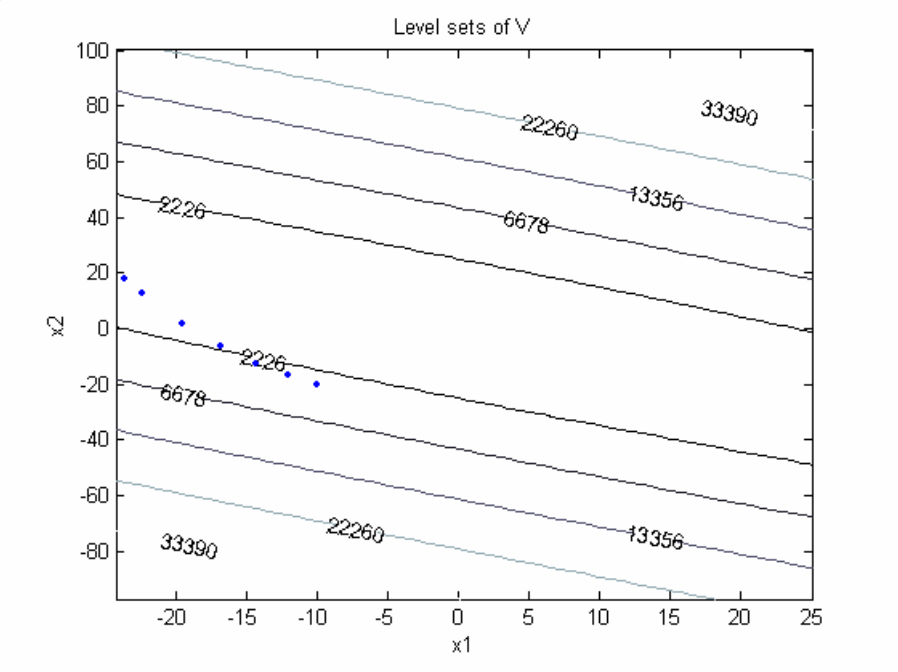

Figure 3-35: Level Sets of *V* for NGAC on System (3.3) (W=[0 1])

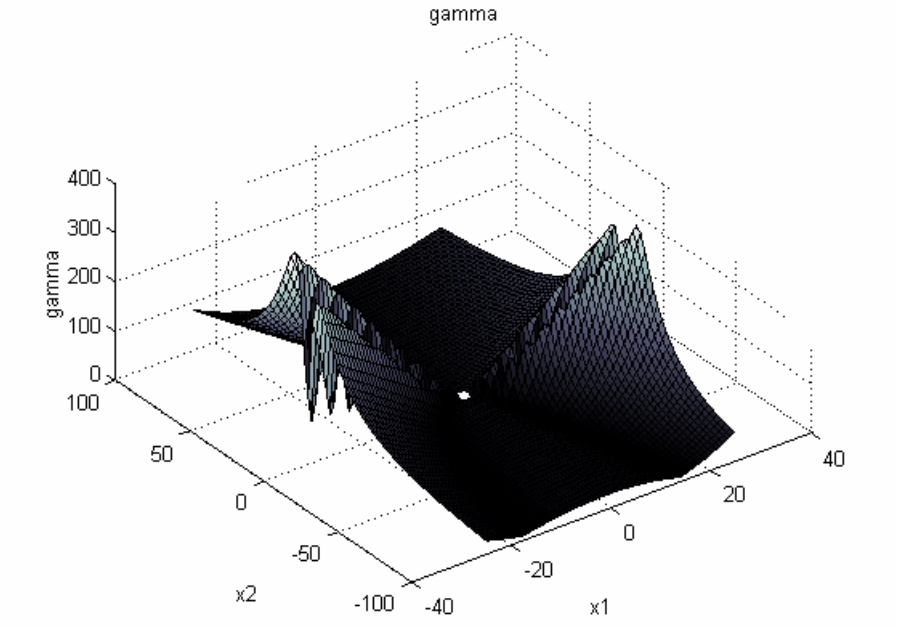

Figure 3-36: "Gamma" (γ) for NGAC on Artificial System (3.3) (W=[0 1])

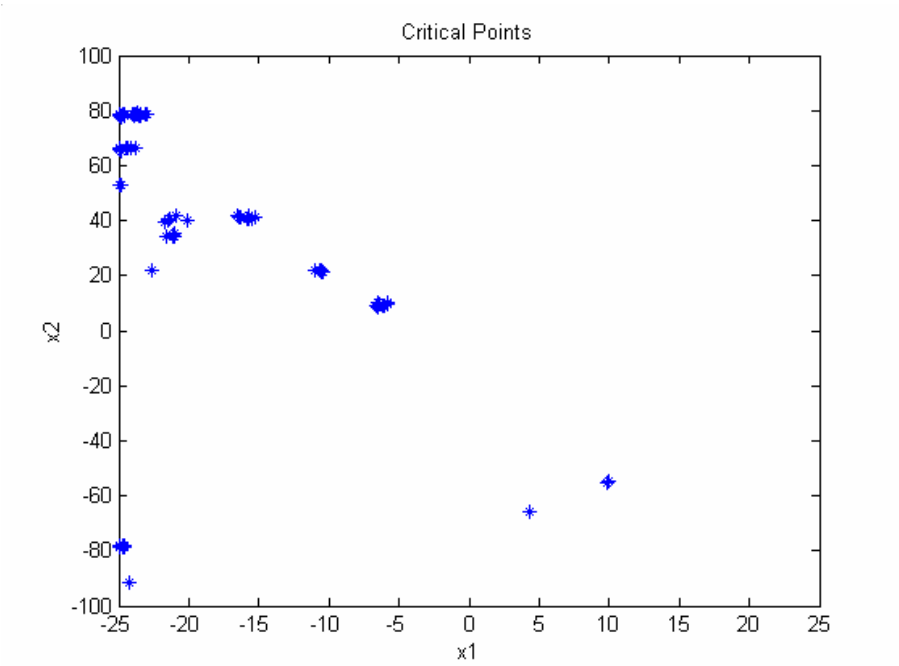

Figure 3-37: Critical Points for NGAC Population Using System (3.3) (W=[0 1])

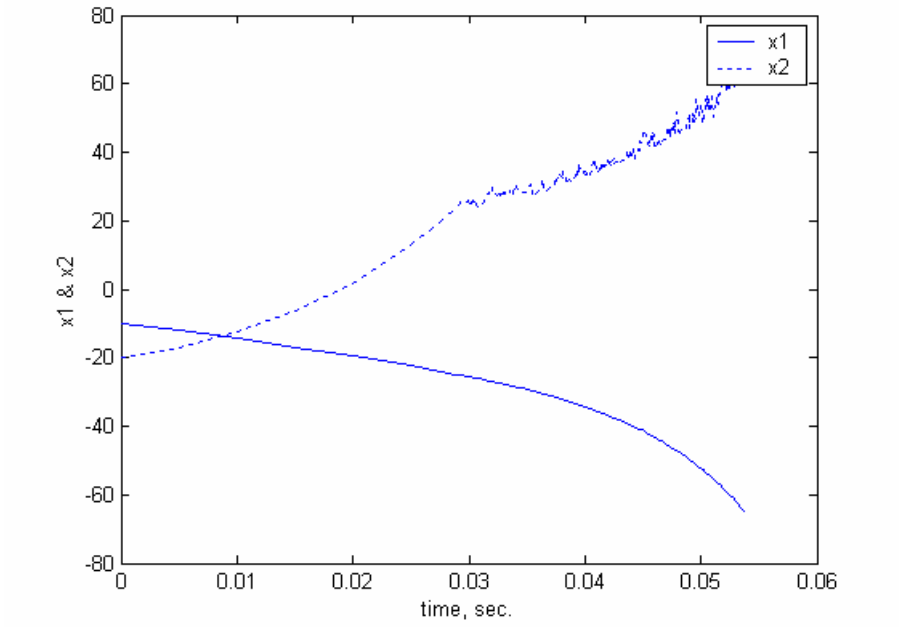

Figure 3-38: States of System (3.3) Using NGAC (W=[0 1])

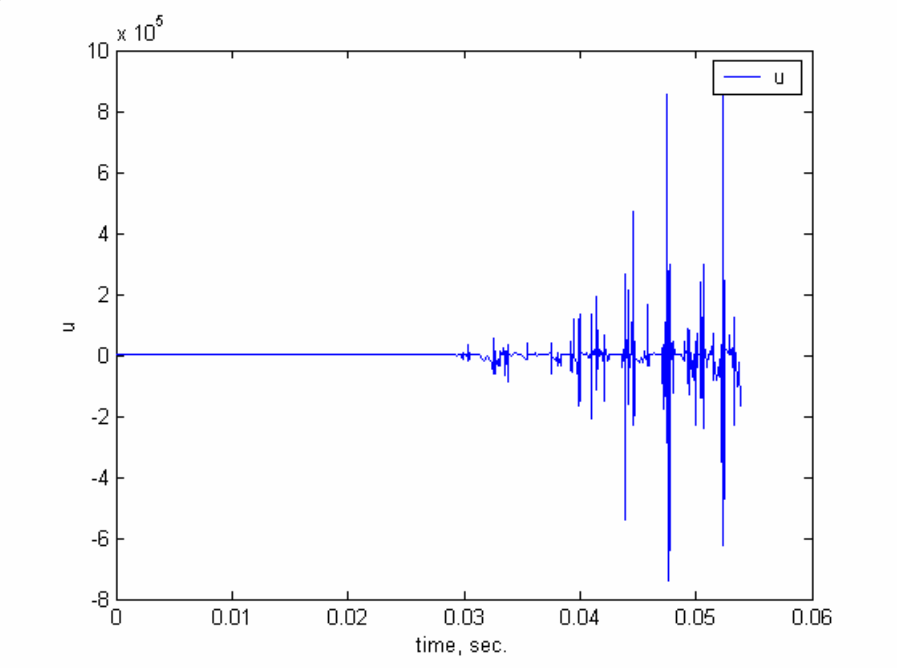

Figure 3-39: Control Signal of NGAC on System (3.3) (W=[0 1])

3.2.3 GAC Case #3: Denser Checking Sets

The genetic algorithm parameters for Case #3 are given in Table 3-13. We use the same GA parameters as Case #2 with nonlinear control, but the number of checking points in X_d^0 has been doubled. The intention is to catch the numerically ill-condition "crevice" in the γ plots of Case #2 and tune *P* such that the $\underline{\lambda}(P)/\overline{\lambda}(P)$ ratio is not too small, yet is not restricted by some predefined lower threshold.

System: Artificial (3.3)	Controller Type: nonlinear	Generations: 20	Population Size: 50	Checking points: 200
Critical Points: 100	Δ_E: 100	$K_{\max}$: 1000	c: 0	X^0 Range: $x_1 \in \lfloor -25\ \ 25 \rfloor$ $x_2 \in \lfloor -100\ \ 100 \rfloor$

Table 3-13: GA Parameters to Optimize NGAC for System (3.3) (High Density X_d^0)

Table 3-14 lists the results of the optimization of the nonlinear GAC for the artificial system (3.3) using the parameters from Table 3-13. Comparing Table 3-10 to Table 3-14 the doubling of the size of X_d^0 seems to have substantially affected the GA's output for the Weight Case 1.

Weight Case	Weights	*P*	*K*	*Q*
1	W=[1 0]	4976.2 339.00 339.00 36.100	-847.46 -90.348	$5.8939x10^5$ 19600 19600 1060
2	W=[1 1]	4287.3 376.90 376.90 45.800	-942.37 -114.517	$8.0232x10^5$ 60900 60900 5480
3	W=[0 1]	3.8772 3.5242 3.5242 3.5240	-8.8106 -8.8100	0.0829 0.0751 0.0751 0.0688

Table 3-14: Optimized Parameters of NGAC for System (3.3) (High Density X_d^0)

Table 3-15 displays the randomly sampled performance of the nonlinear GAC on the artificial system (3.3) using double the number of elements in X_d^0. Table 3-16 displays the randomly sampled performance of the nonlinear GAC on the linearized artificial system (3.4) using double the number of elements in X_d^0. The change in the performance of the nonlinear GAC for W=[1 0] between Case #2 and Case #3 is understandable given the substantial change in *P* and *K*. The change in $\gamma_{\min}$ of the nonlinear GAC for W=[0 1] between Case #2 and Case #3 was not expected because *P* and *K* are very similar between the two cases. A likely cause of the discrepancy is the small $\underline{\lambda}(P)/\overline{\lambda}(P) = 0.0238$ ratio where the true value of $\gamma_{\min}$ lies on a thin "escape manifold" as discussed in Case #2.

Weight Case	Weights	$\gamma_{\min}$	μ_{γ}	σ_{γ}	$\|u\|_{\max}$	$\mu_{\|u\|}$	$\sigma_{\|u\|}$
1	W=[1 0]	32.628	174.013	62.836	87234	14322	12010
2	W=[1 1]	33.280	234.056	101.40	73692	14871	11249
3	W=[0 1]	0.3079	53.8784	57.666	57181	1399.1	2042.3

Table 3-15: Performance of NGAC on System (3.3) (High Density X_d^0)

Weight Case	Weights	$\gamma_{\min}$	μ_{γ}	σ_{γ}	$\|u\|_{\max}$	$\mu_{\|u\|}$	$\sigma_{\|u\|}$
1	W=[1 0]	151.578	254.15	31.559	30036	11165	7067.5
2	W=[1 1]	128.115	361.94	75.626	34929	12711	8190.3
3	W=[0 1]	0.00120	20.715	3.6789	1090.8	447.79	268.42

Table 3-16: Performance of NGAC on Linearized System (3.4) (High Density X_d^0)

For Weight Cases 1 and 2, Figure 3-40 & Figure 3-45 display closed V contours, showing stable trajectories and well conditioned γ's (Figure 3-41 & Figure 3-46). Figure 3-51 still possesses the negative γ as in Case #2, however this is expected because the GA search ignores γ altogether. Hence, it is advisable to put some small weight on γ when searching for controllers that yield low $|u|_{\max}$. Comparing the state and control response of the nonlinear LARC in Figure 3-8 & Figure 3-9 to those of the nonlinear GAC with weight settings W=[1 0] and W=[1 1] in Figure 3-43, Figure 3-44, Figure 3-48, & Figure 3-49, we see that the nonlinear GAC converges about 4 times faster with about 20% less maximum control effort.

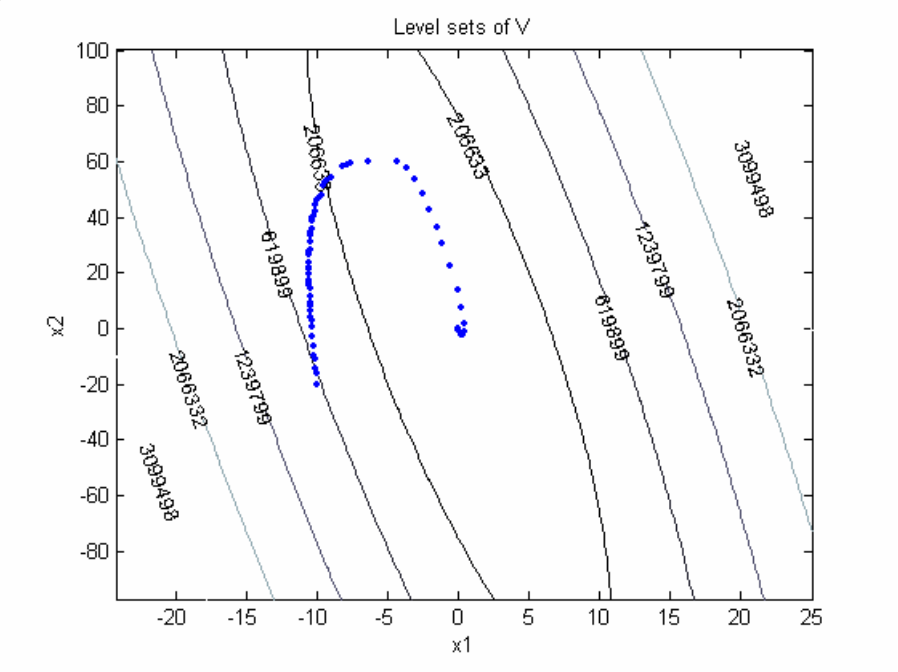

Figure 3-40: Level Sets of *V* for NGAC on System (3.3) (W=[1 0], High Density X_d^0)

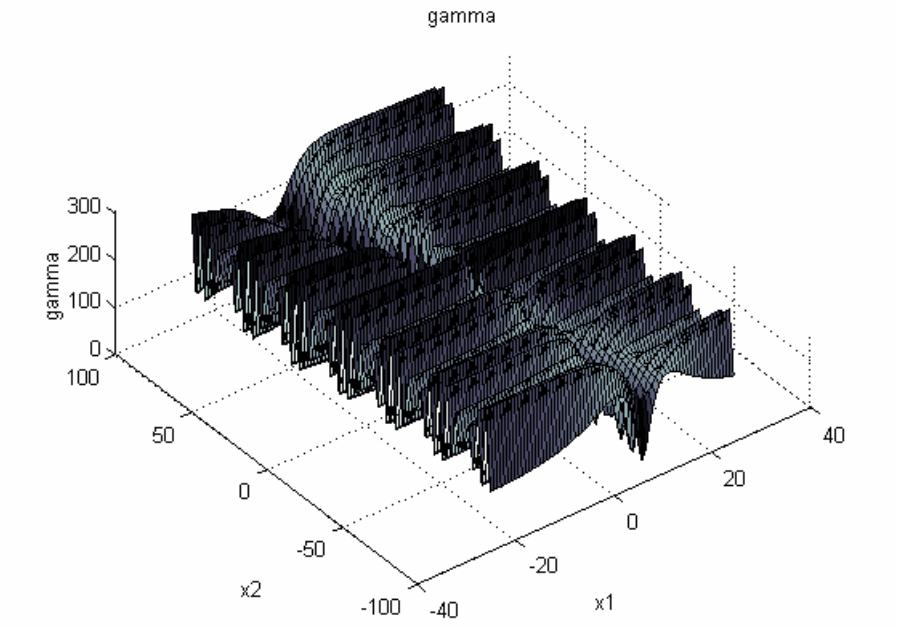

Figure 3-41: "Gamma" (γ) for NGAC on System (3.3) (W=[1 0], High Density X_d^0)

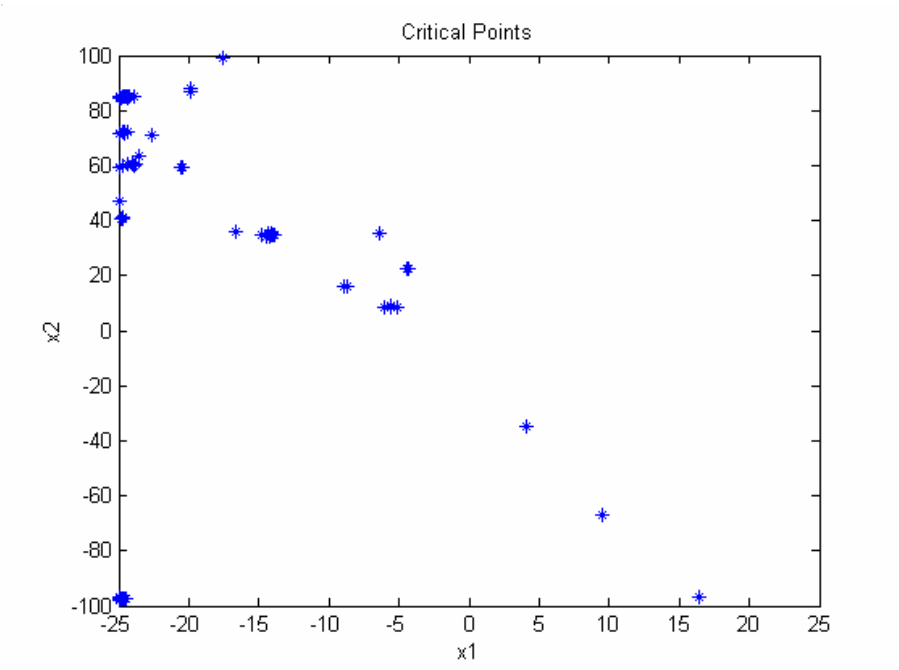

Figure 3-42: Critical Points for NGAC Using System (3.3) (W=[1 0], High Density X_d^0)

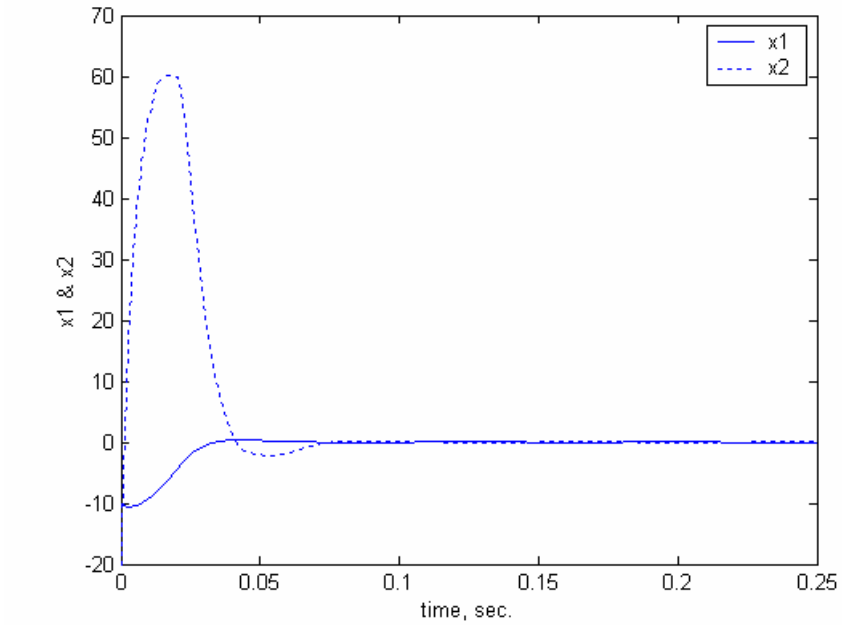

Figure 3-43: States of System (3.3) Using NGAC (W=[1 0], High Density X_d^0)

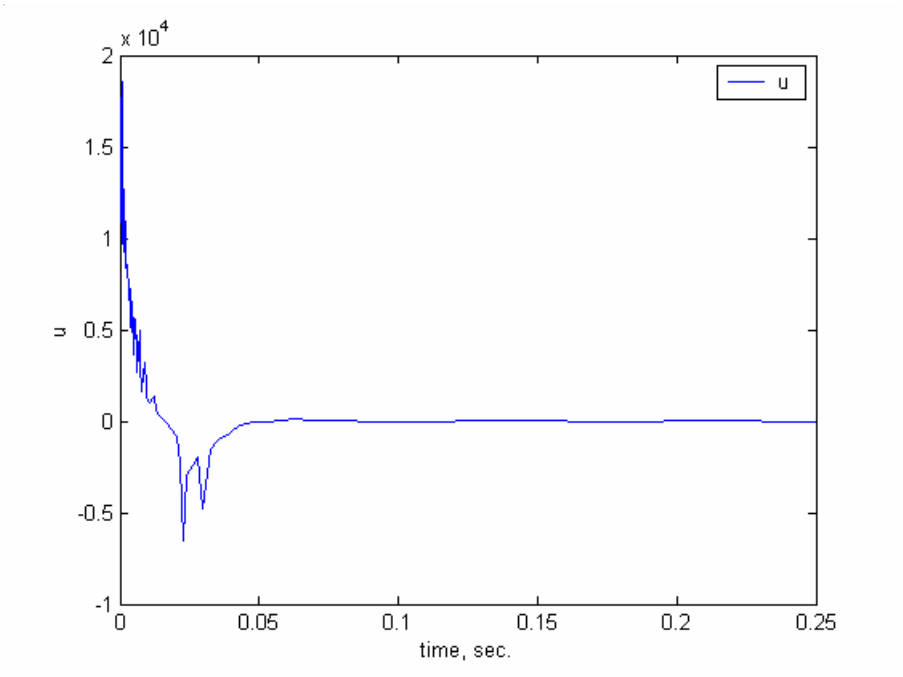

Figure 3-44: Control Signal of NGAC on System (3.3) (W=[1 0], High Density X_d^0)

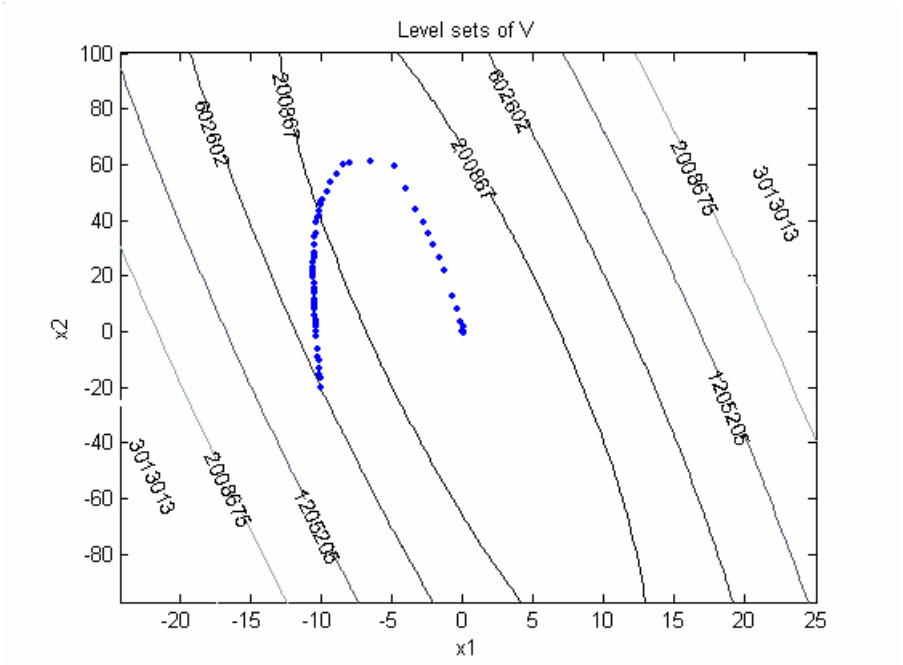

Figure 3-45: Level Sets of *V* for NGAC on System (3.3) (W=[1 1], High Density X_d^0)

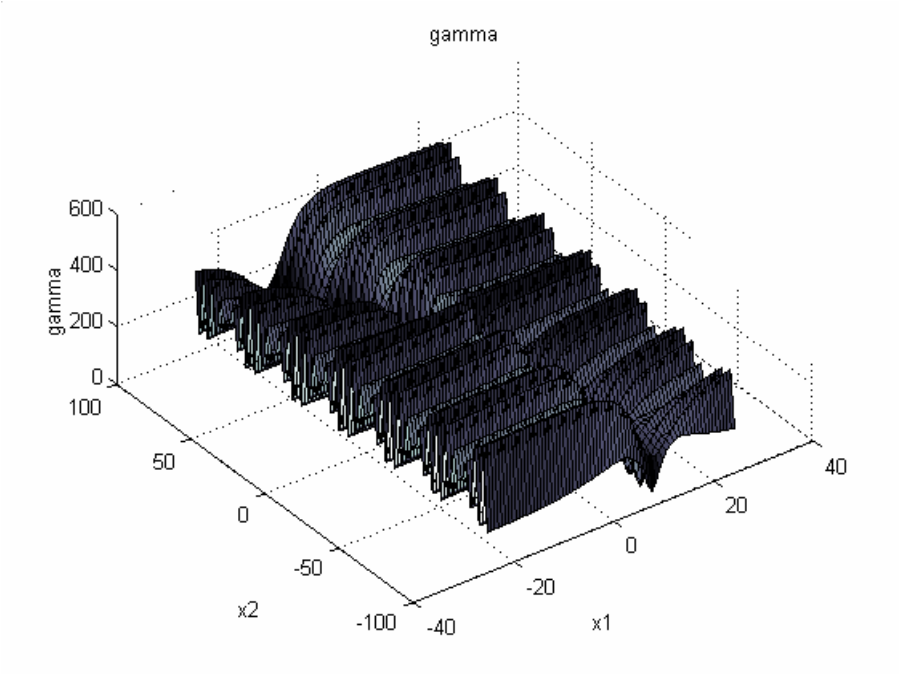

Figure 3-46: "Gamma" (γ) for NGAC on System (3.3) (W=[1 1], High Density X_d^0)

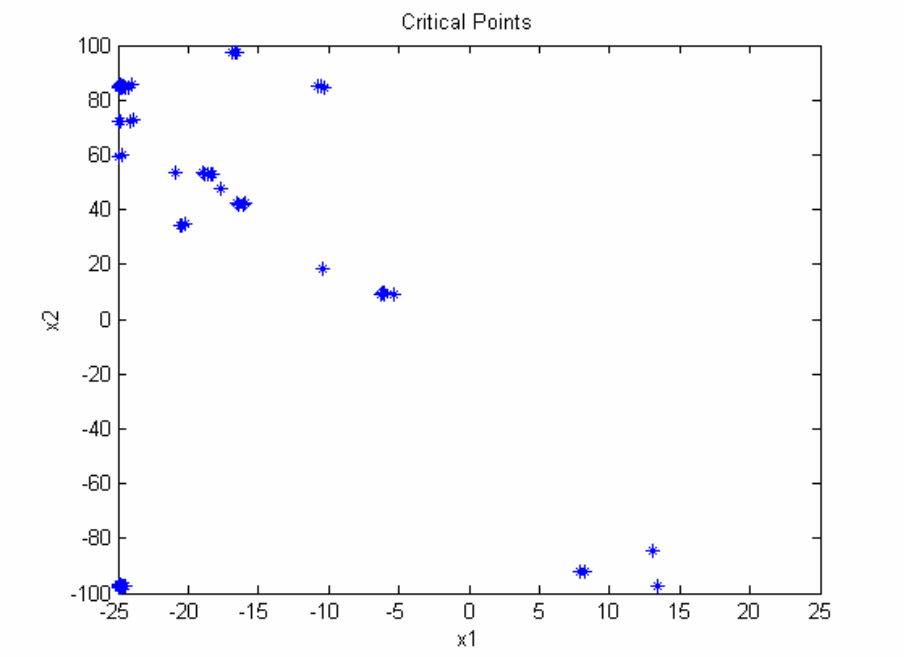

Figure 3-47: Critical Points for NGAC Using System (3.3) (W=[1 1], High Density X_d^0)

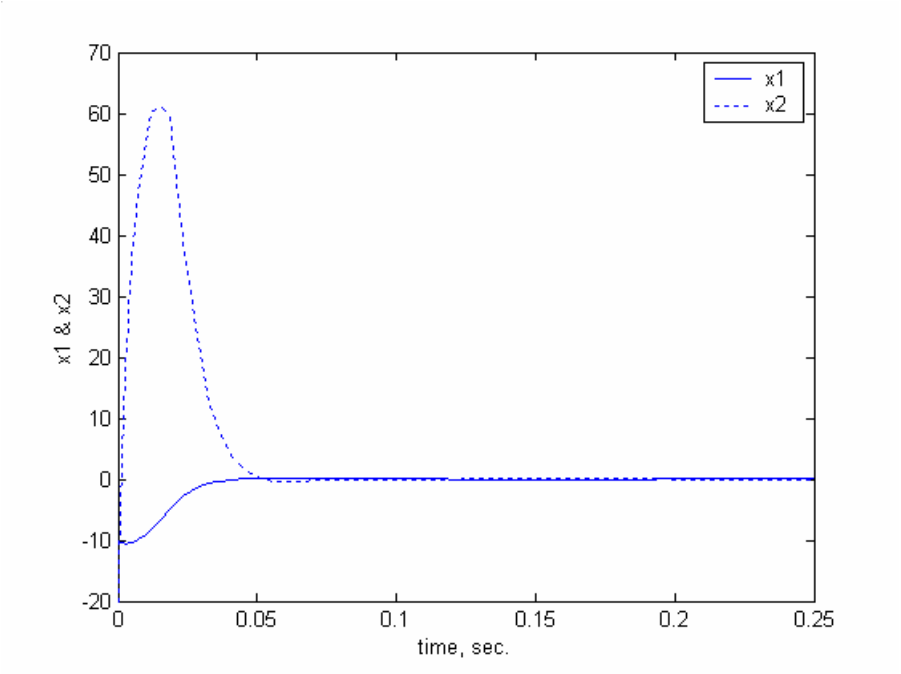

Figure 3-48: States of Artificial System (3.3) Using NGAC (W=[1 1], High Density X_d^0)

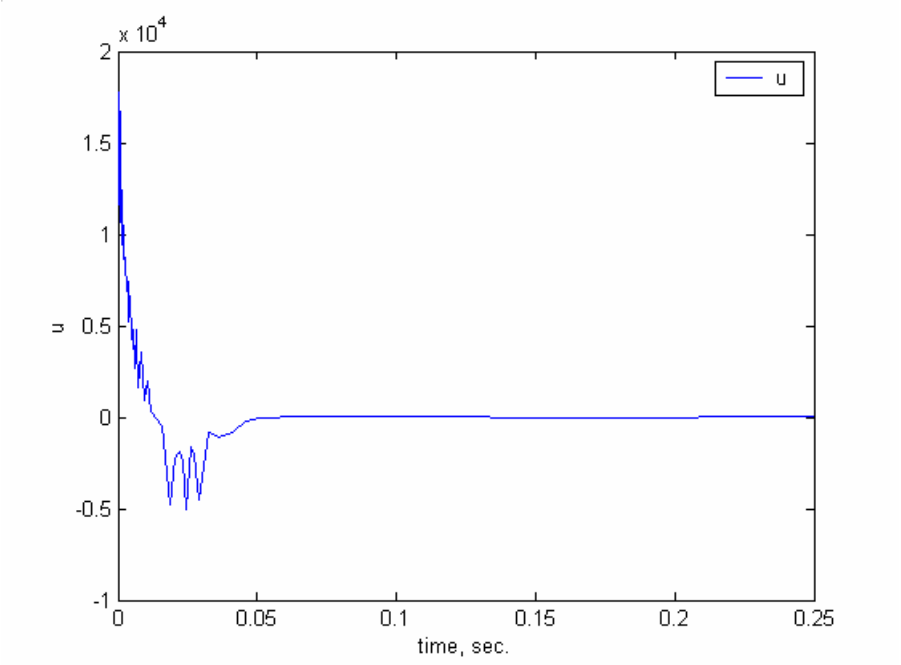

Figure 3-49: Control Signal of NGAC on System (3.3) (W=[1 1], High Density X_d^0)

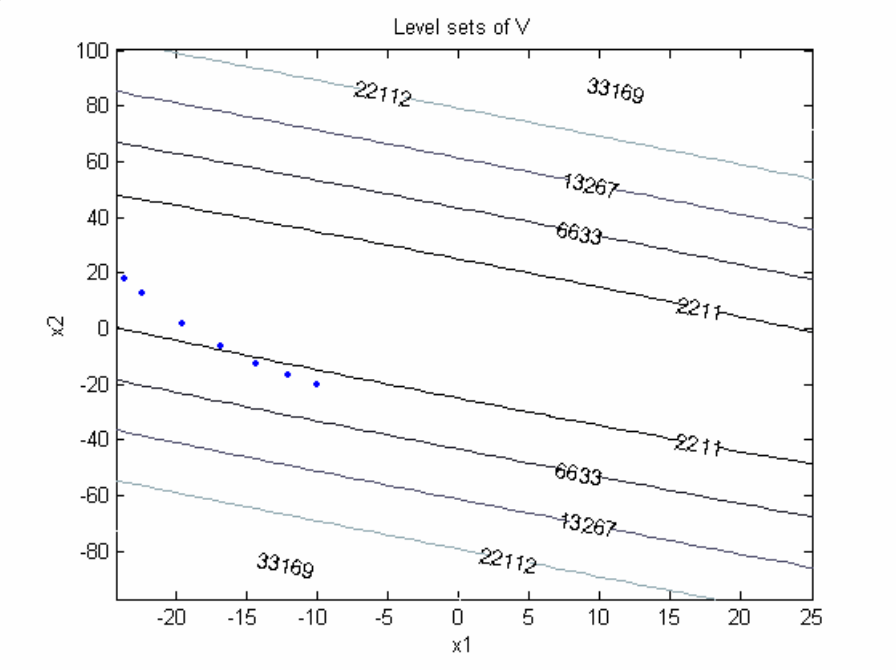

Figure 3-50: Level Sets of *V* for NGAC on System (3.3) (W=[0 1], High Density X_d^0)

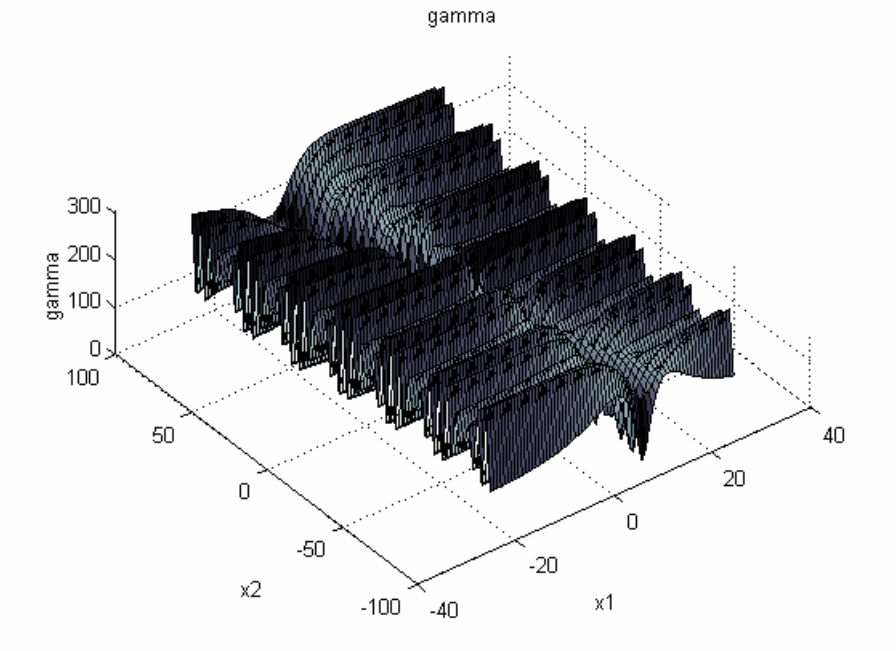

Figure 3-51: "Gamma" (γ) for NGAC on System (3.3) (W=[0 1], High Density X_d^0)

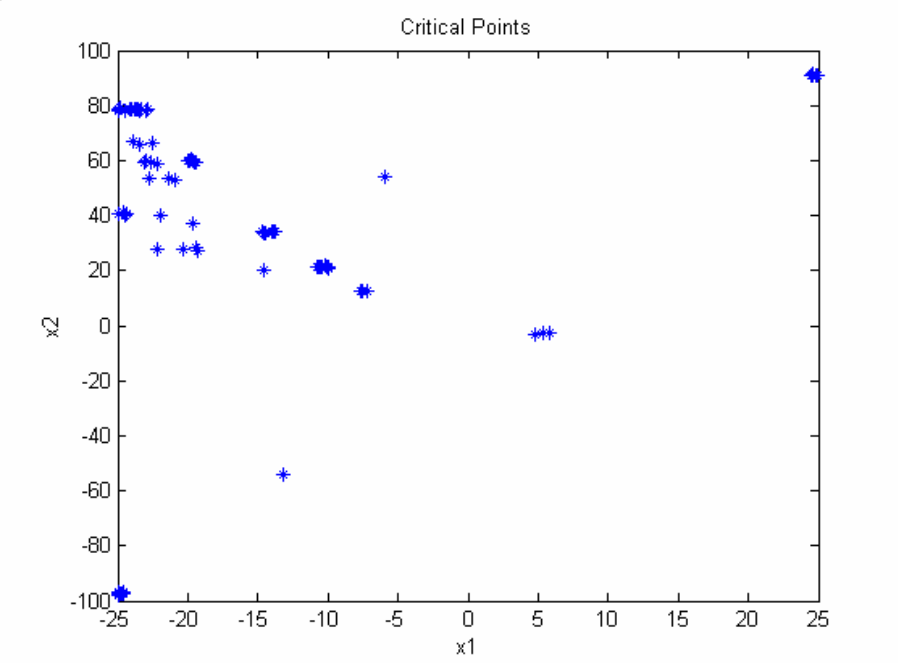

Figure 3-52: Critical Points for NGAC Using System (3.3) (W=[0 1], High Density X_d^0)

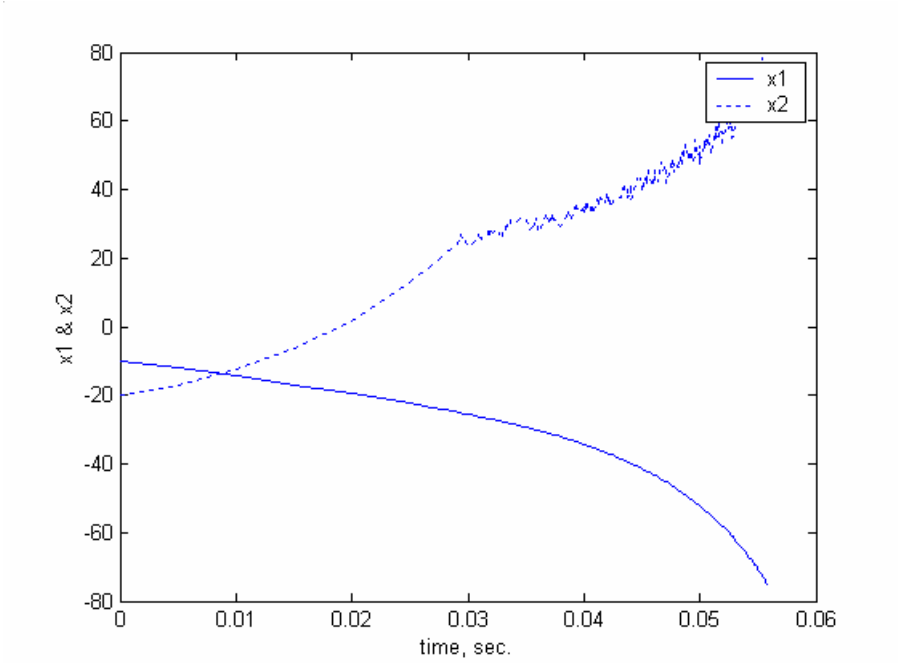

Figure 3-53: States of System (3.3) Using NGAC (W=[0 1], High Density X_d^0)

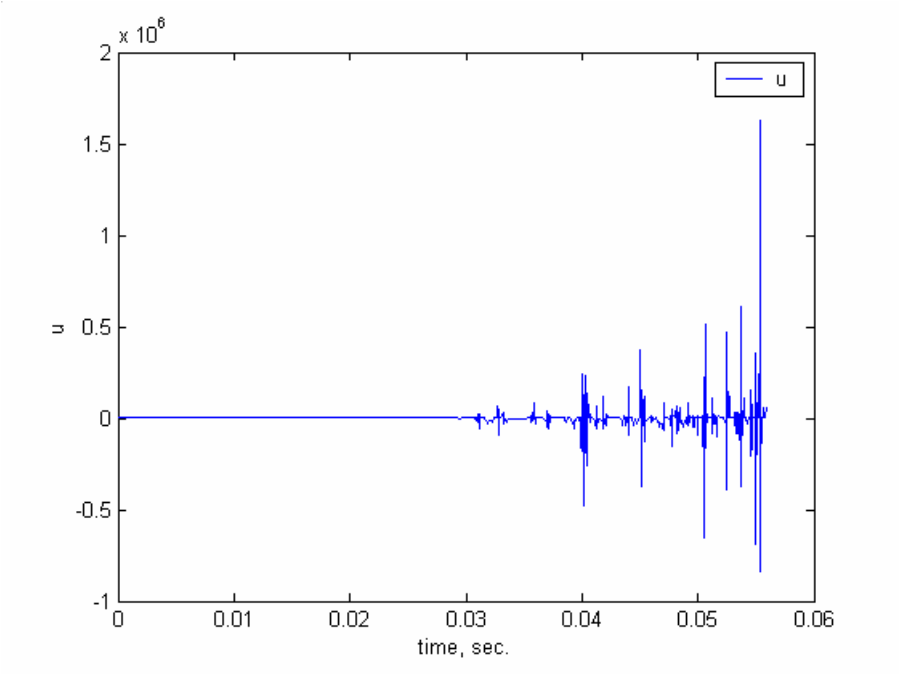

Figure 3-54: Control Signal of NGAC on System (3.3) (W=[0 1], High Density X_d^0)

In conclusion, we have seen in this example that while the LARC, by design, has a larger region of stability, the GAC can yield a better performance in terms of minimum rate of convergence and maximum control effort. The GAC also offers the ability to select the region of state space to perform the optimization. To answer question 3.1, checking the performance of the nonlinear system directly on X_d^0 using the CLF, $V = x^T Px$, does offer a clear advantage over the robust linear control theory based procedure of Ngamsom's LARC. It is important, however, to set both elements of W to a positive number and use a sufficient number of elements in X_d^0 (found via experimentation) to yield closed level sets for the CLF ($\underline{\lambda}(P)/\overline{\lambda}(P) >> 0$).

3.3 Example 2: Double Inverted Pendulum System

The system in Figure 3-55 was taken originally from Misawa (1995) where it is described in detail.

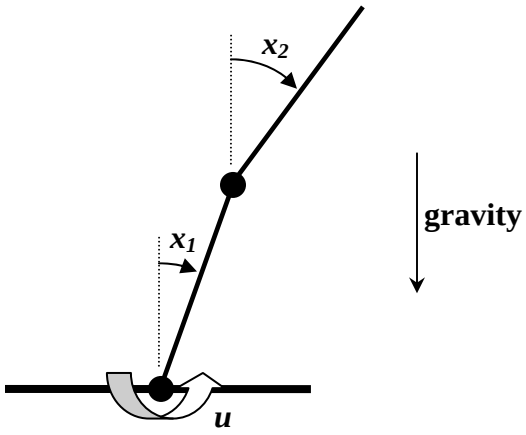

Figure 3-55: Double Inverted Pendulum System (Misawa et al, 1995)

The equations of motion are

$$\dot{x} = f(x) + g(x)u \tag{3.5}$$

where $x \in \Re^4$, $f, g : \Re^4 \to \Re^4$, $u \in \Re$, and

$$f_1 = x_3$$

$$f_2 = x_4$$

$$f_3 = \frac{\left\{\begin{array}{l}\left(\sin(x_1 - x_2)x_3^2 + 0.2824x_3 - 0.2824x_4 + 48.2776\sin(x_2)\right)\cos(x_1 - x_2) \\ + 0.9833x_3 + 1.1206\sin(x_1 - x_2)x_4^2 - 0.3165x_4 - 214.3082\sin(x_1)\end{array}\right\}}{-5.9809 + \cos^2(x_1 - x_2)}$$

$$f_4 = \frac{\left\{\begin{array}{l}\left(-0.8774x_3 - \sin(x_1 - x_2)x_4^2 + 0.2824x_4 + 191.2383\sin(x_2)\right)\cos(x_1 - x_2) \\ -5.3371\sin(x_1 - x_2)x_3^2 - 1.5071x_3 + 1.5071x_4 - 257.6614\sin(x_2)\end{array}\right\}}{-5.9809 + \cos^2(x_1 - x_2)}$$

$$g_1 = 0$$
$$g_2 = 0$$
$$g_3 = \frac{-565.1008}{-5.9809 + \cos^2(x_1 - x_2)}$$
$$g_4 = \frac{504.2688\cos(x_1 - x_2)}{-5.9809 + \cos^2(x_1 - x_2)}$$

The linearized dynamics about the upright position are:

$\dot{x} = Ax + Bu$, with

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 43.0260 & -9.6925 & -0.2541 & 0.1202 \\ 0 & 13.3356 & 0.4787 & -0.3593 \end{bmatrix} \tag{3.6}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ 113.4536 \\ -101.2405 \end{bmatrix}$$

3.3.1 LARC Results

Two ranges of the checking set X_d^0 were used for all controllers in this example. The small range, $x_i \in [-0.1 \quad 0.1], i = 1,2,3,4$, is used to capture the local slightly nonlinear behavior about the origin. The large range, $x_i \in [-1 \quad 1], i = 1,2,3,4$, is used to capture the highly nonlinear behavior away from the origin. Table 3-17 displays the LARC results for the double inverted pendulum system as reported in Ngamsom (2001).

P				*K*				*Q*			
0.3882	0.9764	0.1241	0.1412	0.1036	5.2008	0.3618	0.8021	2	0	0	0
0.9764	9.2923	1.0084	1.2328					0	2	0	0
0.1241	1.0084	0.1211	0.1429					0	0	2	0
0.1412	1.2328	0.1429	0.1759					0	0	0	2

Table 3-17: LLARC Parameters for System (3.5) (Ngamsom, 2001)

Table 3-18 lists the randomly sampled performance of the linear LARC on the double inverted pendulum system (3.5). For the small range, $\gamma_{\min}$ is negative (although Ngamsom reports the system to be stable). It turns out that the system is unstable with respect to the particular quadratic CLF dictated by *P* in Table 3-17 and not necessarily quadratically unstable. For the large range, Ngamsom (2001) reports the system to be unstable. The $\gamma_{\min}$ value found here is highly negative supporting Ngamsom's report.

X^0 Range	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
$x_i \in [-0.1 \quad 0.1]$ $i = 1,2,3,4$	-2.6472	0.0127	1.5402	0.6262	0.2628	0.1543
$x_i \in [-1 \quad 1]$ $i = 1,2,3,4$	-166.31	-30.482	36.980	6.4308	2.6432	1.5451

Table 3-18: Performance of LLARC on System (3.5)

Table 3-19 shows the randomly sampled performance of the nonlinear LARC on the double inverted pendulum system (3.5). Because the linear LARC performs poorly for the large range, we may attest that the system nonlinearities cause the nonlinear controller to use high control effort to keep the state trajectory within the level sets whose "angle" and "skewing" (if one can visualize in 4-D!) are optimized for the locally linear behavior of the system.

X^0 Range	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
$x_i \in [-0.1 \quad 0.1]$ $i = 1,2,3,4$	0.4378	10.837	4.3018	1.4729	0.5487	0.3318
$x_i \in [-1 \quad 1]$ $i = 1,2,3,4$	0.0914	11.313	4.5151	17982	12741	174.81

Table 3-19: Performance of NLARC on System (3.5)

Table 3-20 lists the randomly sampled performance of the linear LARC on the linearized double inverted pendulum system (3.6). The rate of convergence is negative

for the linear LARC on the nonlinear dynamics and relatively slow for the linear LARC on the linear dynamics which suggests that significant nonlinearities will destabilize the system.

X^0 Range	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
$x_i \in [-0.1 \quad 0.1]$ $i = 1,2,3,4$	0.0208	0.4217	1.4648	0.6187	0.2634	0.1546
$x_i \in [-1 \quad 1]$ $i = 1,2,3,4$	0.0207	0.4017	1.2807	6.2708	2.6458	1.5476

Table 3-20: Performance of LLARC on Linearized System (3.6)

Figure 3-56 & Figure 3-57 contain the system signal responses to the initial condition (0.5,0,0,0) for the linear LARC on the double inverted pendulum system (3.5). Figure 3-58 & Figure 3-59 contain the system signal responses to the initial condition (0.5,0,0,0) for the nonlinear LARC on the double inverted pendulum system (3.5). The nonlinear LARC response is interesting with the chattering of the control signal (Figure 3-59) hence the chattering of the state variables. Such is the nature of the nonlinear controller combined with a finite time stepping numerical simulation method. The infamous denominator term of the Sontag-like control law, $x^T Pg$, gets very close to zero and rapidly switches sign during the first second of the simulation. With progressively smaller time stepping, the switching is less severe, however the smallness of the time step becomes impractically small to produce a reasonably long enough simulation. The problem reflects an actual control law physical implementation issue: discrete-time sampling. The digital implementation of the control law will have the same problem that most likely will be even worse given the limitations of sampling rates in real-time digital control systems.

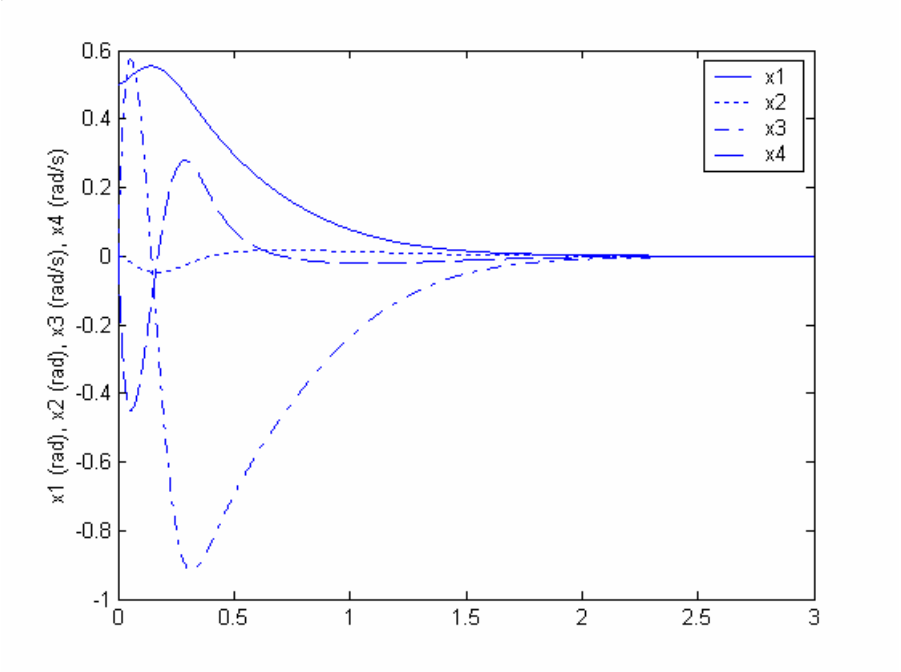

Figure 3-56: States of System (3.5) Using LLARC

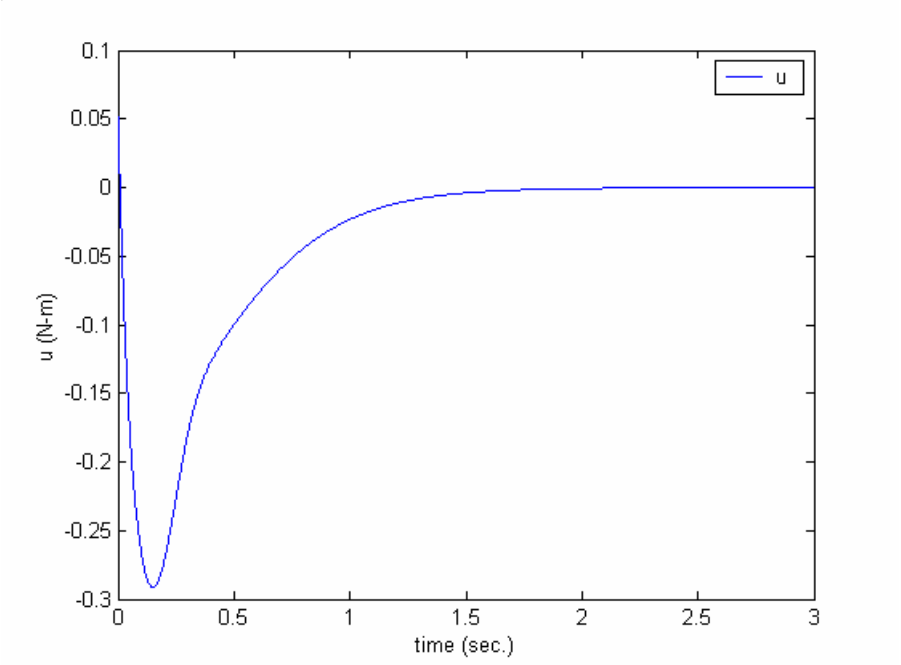

Figure 3-57: Control Signal of System (3.5) Using LLARC

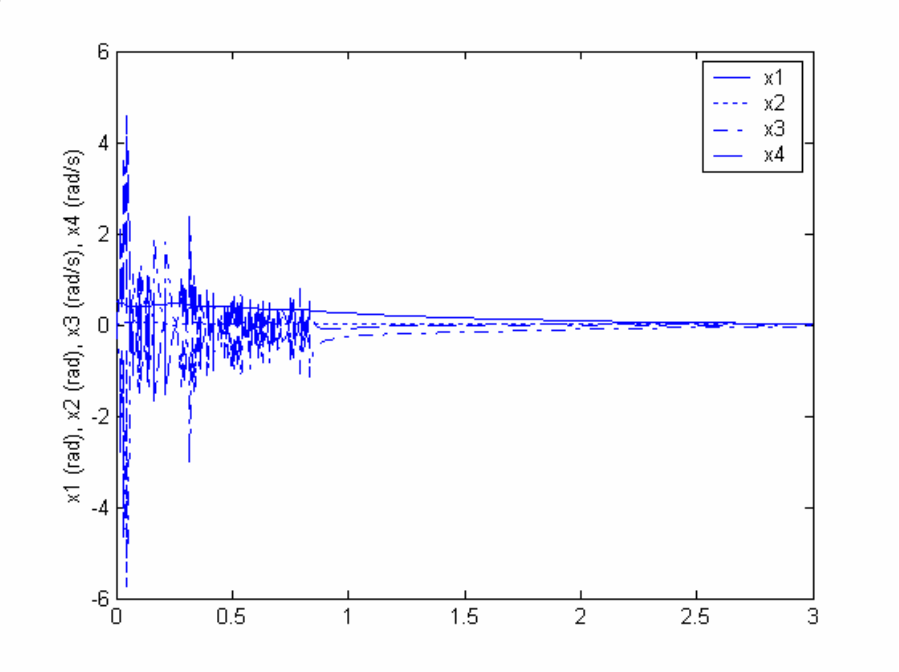

Figure 3-58: States of System (3.5) Using NLARC

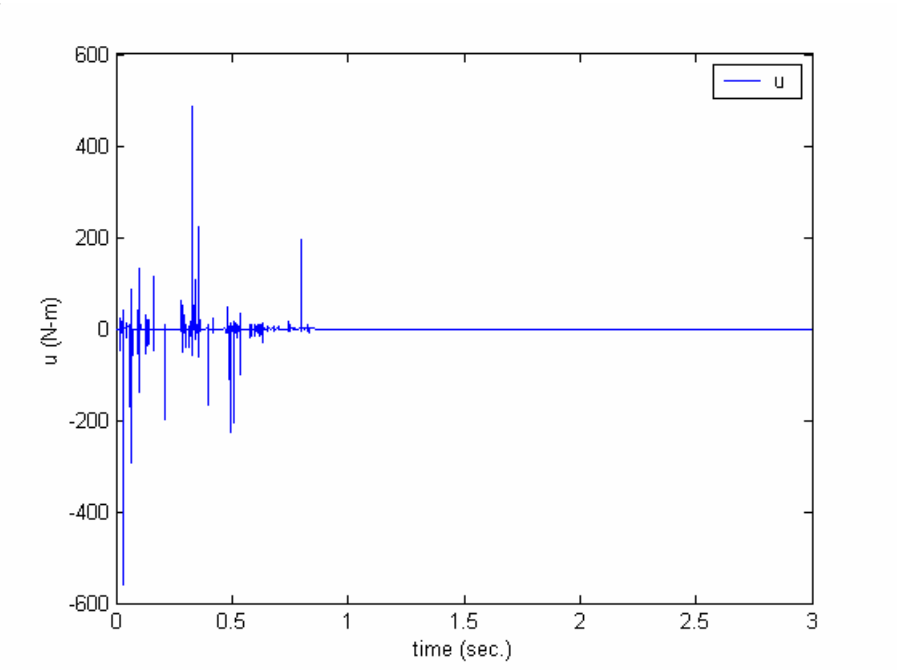

Figure 3-59: Control Signal of System (3.5) Using NLARC

3.3.2 Linear GAC with Small Checking Set Range

Table 3-21 lists the genetic algorithm parameters used to optimize the linear GAC for the double inverted pendulum system (3.5) using a small X_d^0 range. The checking set range is set low to minimize the effects of the system nonlinearities. The control gain upper limit, K_{max}, is set to 10.

System: Double Inverted Pendulum (3.5)	Controller Type: linear	Generations: 50	Population Size: 50	Checking points: 200
Critical Points: 50	Δ_E: 10	$K_{\max}$: 10	c: 0	X^0 Range: $x_i \in [-0.1 \quad 0.1]$ $i = 1,2,3,4$

Table 3-21: GA Parameters to Optimize LGAC for System (3.5) (Small X_d^0 Range)

Table 3-22 shows the genetic algorithm results of the optimization of the linear GAC for the double inverted pendulum system (3.5) using a small X_d^0 range. It is interesting that the optimal Q has a structure such that the first diagonal element is dominate and all other elements are almost negligible besides the terms involving the state x_1. It is also interesting that some terms are negative which suggests that an optimal controller will ensure that the states that are coupled to these terms are opposite in sign during most of the state trajectory. This type of solution is not obvious so would probably not have been selected by a human designer, yet it becomes a useful insight for the controller design.

Weight Case	Weights	P	K	Q
1	W=[1 0]	0.3010 0.6116 0.0781 0.0914 0.6116 2.0038 0.2208 0.2984 0.0781 0.2208 0.0258 0.0330 0.0914 0.2984 0.0330 0.0445	0.3922 5.1569 0.4182 0.7570	0.4513 0.0053 -0.0021 -0.0041 0.0053 0.0001 0.0000 0.0000 -0.0021 0.0000 0.0001 0.0001 -0.0041 0.0000 0.0001 0.0002
2	W=[1 1]	0.1179 0.2750 0.0367 0.0418 0.2750 1.2195 0.1267 0.1819 0.0367 0.1267 0.0144 0.0190 0.0418 0.1819 0.0190 0.0272	0.0676 4.0455 0.2904 0.5908	0.0559 0.0019 0.0010 -0.0004 0.0019 0.0001 0.0000 -0.0001 0.0010 0.0000 0.0000 0.0000 -0.0004 -0.0001 0.0000 0.0001
3	W=[0 1]	0.1300 0.2955 0.0392 0.0448 0.2955 1.2698 0.1327 0.1895 0.0392 0.1327 0.0151 0.0199 0.0448 0.1895 0.0199 0.0283	0.0901 4.1270 0.2997 0.6029	0.0765 -0.0021 -0.0007 -0.0010 -0.0021 0.0001 0.0000 0.0000 -0.0007 0.0000 0.0000 0.0000 -0.0010 0.0000 0.0000 0.0000

Table 3-22: Optimized Parameters of LGAC for System (3.5) (Small X_d^0 Range)

Table 3-23 displays the randomly sampled performance of the linear GAC on the double inverted pendulum system (3.5) using a small X_d^0 range. Unlike the linear LARC (Table 3-17), the linear GAC yields a positive minimum rate of convergence with a much higher average rate of convergence and without a significant difference between control effort or gains for all sets of fitness function weights.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	0.2212	12.6432	6.5883	0.6455	0.2597	0.1526
2	W=[1 1]	0.0568	12.6826	6.5130	0.4851	0.2049	0.1192
3	W=[0 1]	0.0739	12.5958	6.4883	0.5028	0.2090	0.1229

Table 3.22: Performance of LGAC on System (3.5) (Small X_d^0 Range)

Table 3-23 displays the randomly sampled performance of the linear GAC on the linearized double inverted pendulum system (3.6) using a small X_d^0 range. The rate of convergence is fairly close to that of Table 3-22 which suggests that the nonlinearities do not play as significant a role locally as they do with the linear LARC where the difference in $\gamma_{\min}$ between the nonlinear and linear dynamics is substantial.

Weight Case	Weights	$\gamma_{\min}$	μ_{γ}	σ_{γ}	$\lvert u \rvert_{\max}$	$\mu_{\lvert u \rvert}$	$\sigma_{\lvert u \rvert}$
1	W=[1 0]	0.1113	13.1476	6.9741	0.6540	0.2605	0.1539
2	W=[1 1]	0.0799	13.0182	6.8206	0.4877	0.2048	0.1198
3	W=[0 1]	0.0168	13.0161	6.8388	0.5043	0.2068	0.1218

Table 3-23: Performance of LGAC on Linearized System (3.6) (Small X_d^0 Range)

Figure 3-60 through Figure 3-65 contain the system signal responses to the initial condition (0.5,0,0,0) for the linear GAC on the double inverted pendulum system (3.5) using a small X_d^0 range. The weight setting *W*=[1 0] yields a much faster response but at the cost of more control effort than the linear LARC. The weight setting *W*=[1 1] does not yield a noticeably different response speed than the linear LARC, but the control effort is higher. The weight setting *W*=[0 1] yields a much faster response and with slightly more control effort than the linear LARC. The linear GAC for *W*=[0 1] would probably be selected over the linear LARC.

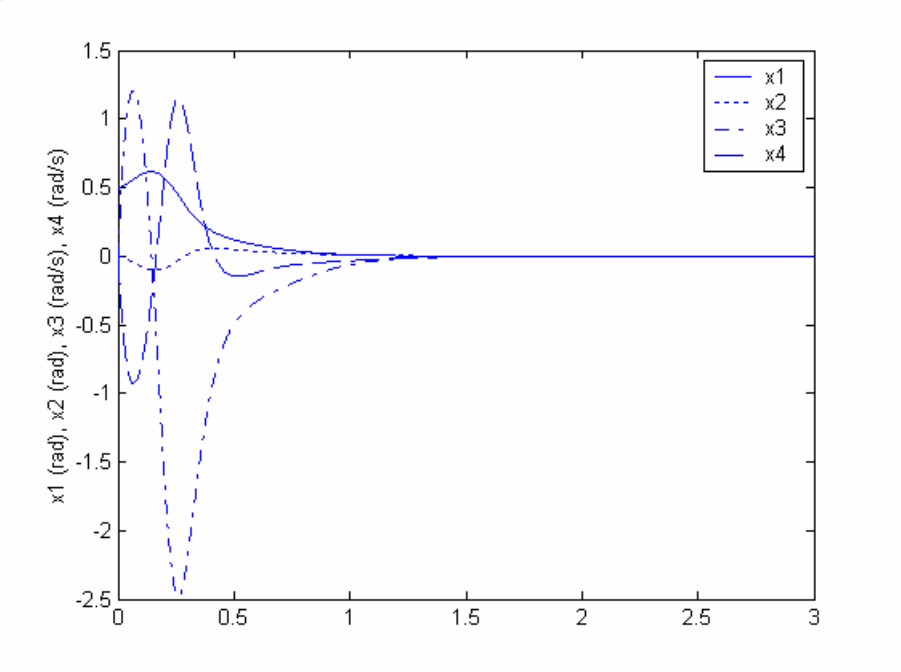

Figure 3-60: States of System (3.5) Using LGAC (W=[1 0], Small X_d^0 Range)

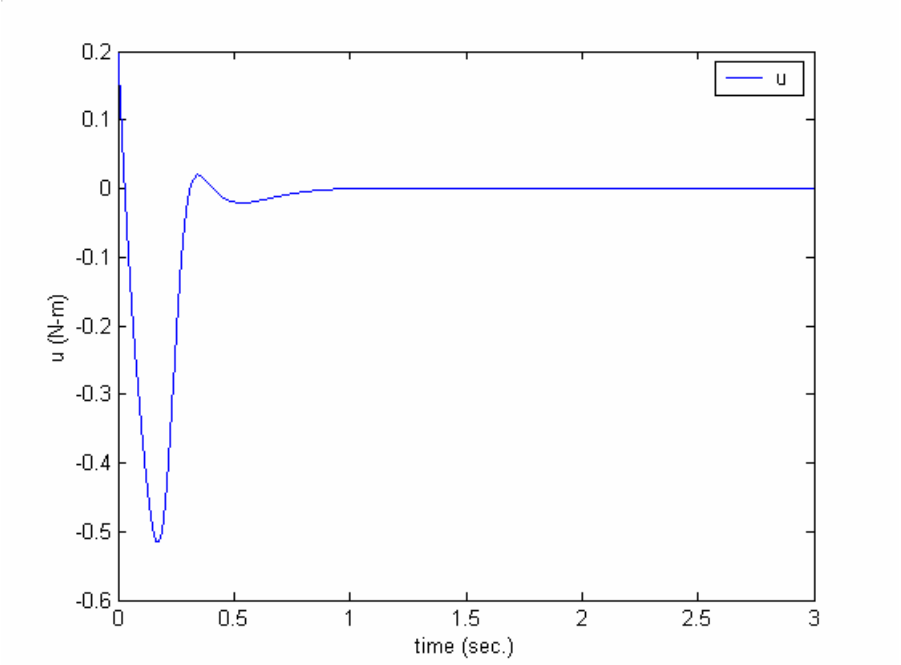

Figure 3-61: Control Signal of System (3.5) Using LGAC (W=[1 0], Small X_d^0 Range)

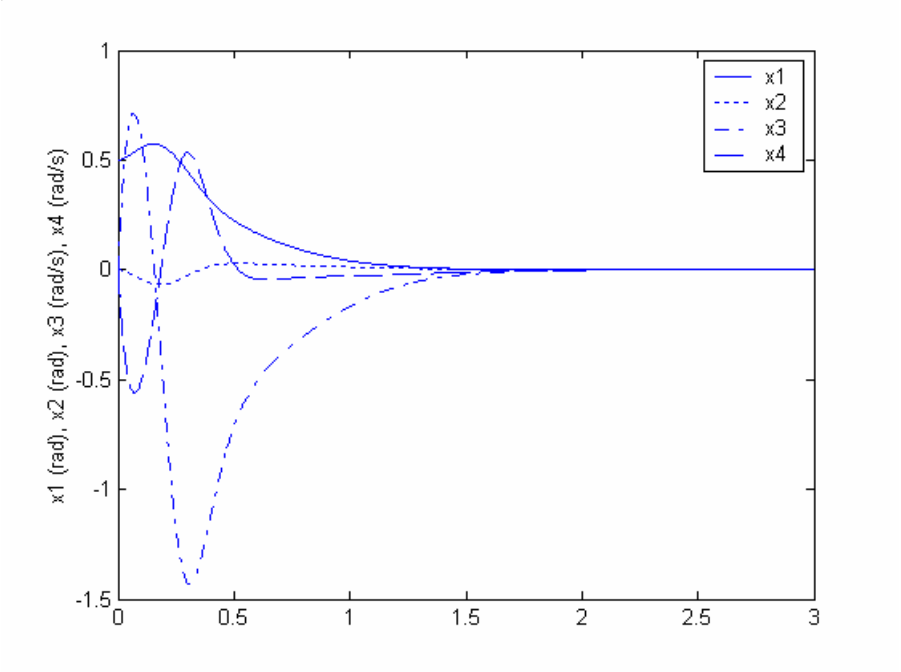

Figure 3-62: States of System (3.5) Using LGAC (W=[1 1], Small X_d^0 Range)

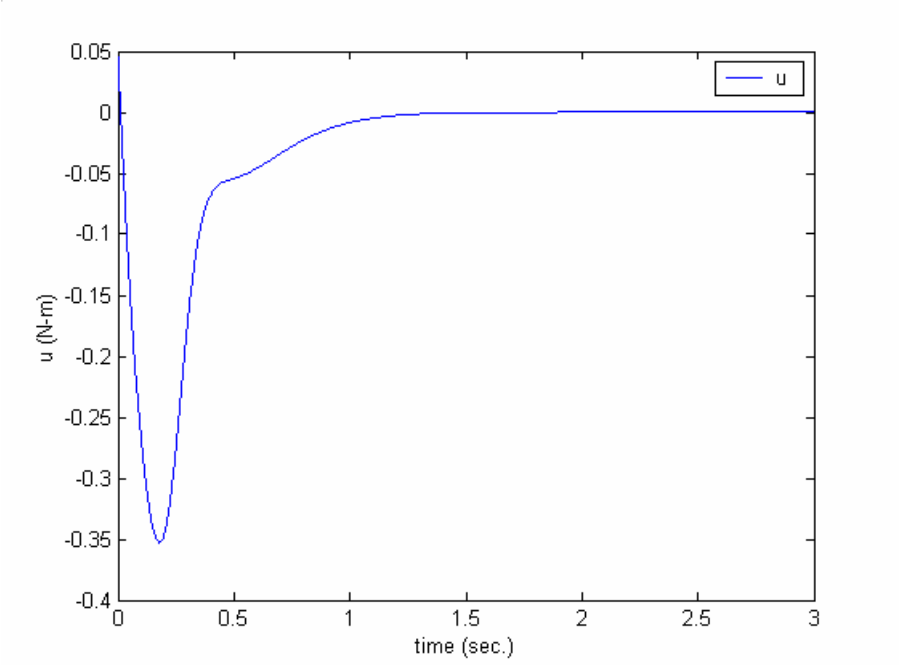

Figure 3-63: Control Signal of System (3.5) Using LGAC (W=[1 1], Small X_d^0 Range)

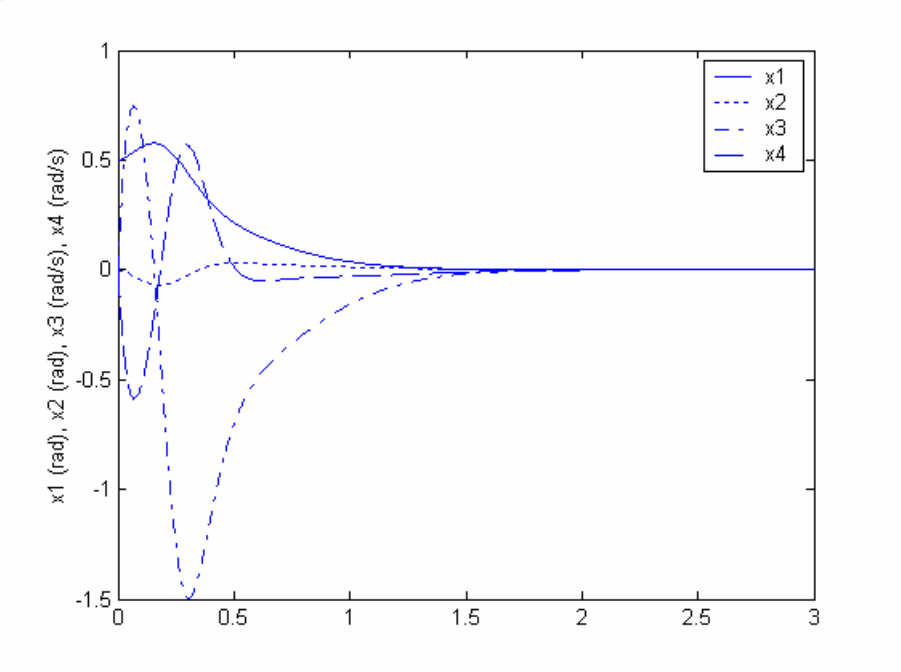

Figure 3-64: States of System (3.5) Using LGAC (W=[0 1], Small X_d^0 Range)

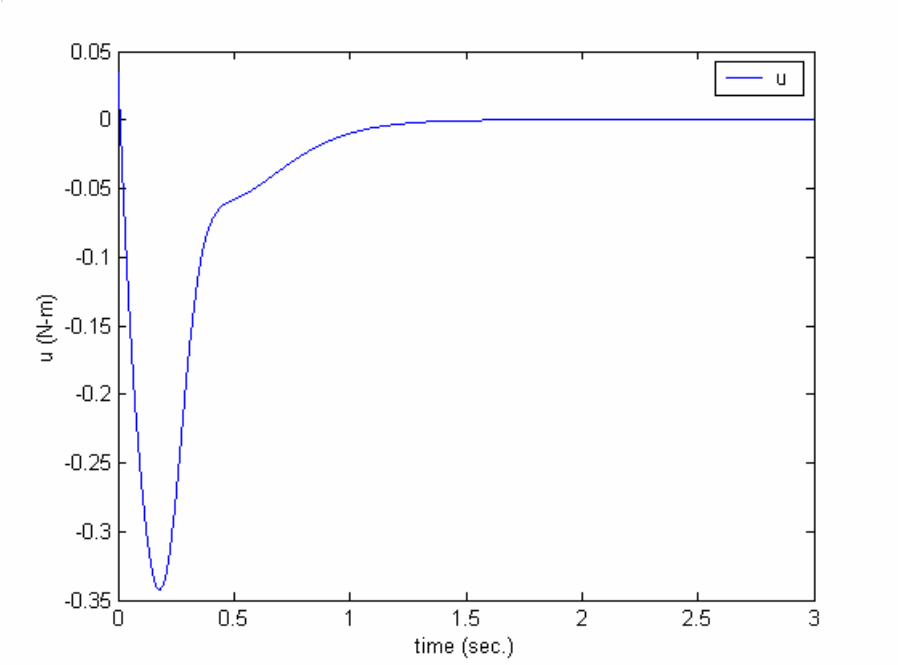

Figure 3-65: Control Signal of System (3.5) Using LGAC (W=[0 1], Small X_d^0 Range)

3.3.3 Linear GAC with Large Checking Set Range

Table 3-24 shows the genetic algorithm parameters used to optimize the linear GAC for the double inverted pendulum system (3.5) using a large X_d^0 range. The checking set range is set high to demonstrate the affects of the system nonlinearities. The control gain upper limit, K_{max}, is set to 10.

System: Double Inverted Pendulum (3.5)	Controller Type: linear	Generations: 50	Population Size: 50	Checking points: 200
Critical Points: 50	Δ_E: 10	$K_{\max}$: 10	c: 0	X^0 Range: $x_i \in [-1 \quad 1]$ $i = 1,2,3,4$

Table 3-24: GA Parameters to Optimize LGAC for System (3.5) (Large X_d^0 Range)

Table 3-25 displays the GA results of the optimization of the linear GAC for the double inverted pendulum system using a large X^0 range. The results are somewhat similar to those of the linear GAC run with a small X^0 range indicating low sensitivity to the X^0 range.

Weight Case	Weights	*P*	*K*	*Q*
1	W=[1 0]	0.1009 0.2370 0.0322 0.0363 0.2370 1.1231 0.1154 0.1678 0.0322 0.1154 0.0131 0.0174 0.0363 0.1678 0.0174 0.0251	0.0237 3.8883 0.2727 0.5673	0.0186 0.0019 -0.0002 0.0008 0.0019 0.0002 -0.0000 0.0000 - 0.0002 -0.0000 0.0000 0.0001 0.0008 0.0000 0.0001 0.0002
2	W=[1 1]	0.1724 0.3756 0.0496 0.0574 0.3756 1.4711 0.1574 0.2202 0.0496 0.1574 0.0182 0.0237 0.0574 0.2202 0.0237 0.0330	0.1804 4.4422 0.3357 0.6500	0.1694 -0.0010 0.0005 0.0025 -0.0010 0.0000 -0.0000 0.0000 0.0005 -0.0000 0.0000 0.0000 0.0025 -0.0000 0.0000 0.0001
3	W=[0 1]	0.1581 0.3498 0.0458 0.0528 0.3498 1.4030 0.1484 0.2091 0.0458 0.1484 0.0170 0.0223 0.0528 0.2091 0.0223 0.0312	0.1486 4.3311 0.3230 0.6336	0.1348 -0.0014 -0.0011 -0.0017 -0.0014 0.0001 -0.0000 -0.0001 -0.0011 -0.0000 0.0000 0.0000 -0.0017 -0.0001 0.0000 0.0002

Table 3-25: Optimized Parameters of LGAC for System (3.5) (Large X_d^0 Range)

Table 3-26 lists the randomly sampled performance of the linear GAC on the double inverted pendulum system (3.5) using a large X_d^0 range. Weight case 1 yielded a faster rate of convergence than the LARC, but a larger maximum control effort.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	0.1073	9.7520	3.4343	45061	11.004	332.63
2	W=[1 1]	-158.19	-17.641	35.302	5.4339	2.2438	1.3166
3	W=[0 1]	-148.99	-16.986	34.249	5.2611	2.1870	1.2875

Table 3-26: Performance of LGAC on System (3.5) (Large X_d^0 Range)

Table 3-27 displays the randomly sampled performance of the linear GAC on the linearized double inverted pendulum system (3.6) using a large X_d^0 range. Notice $\gamma_{\min}$ is negative for *W*=[0 1]. This is because the minimum eigenvalue for *P* is slightly negative ($\underline{\lambda}(P) = -1.4331x10^{-5}$) due to *Q* having an eigenvalue close to zero. The linear system is stable, with the closed loop poles for the linearized dynamics all in the left half plane closed loop poles = $\{-7.2235 \pm 2.0095i, -7.7300, -5.9369\}$). However it is unstable with respect to the CLF. Though it is expected when not checking the rate of convergence, this sort of problem is solved by setting *c>0* in Table 3-24 such that the smallest eigenvalue of *Q* is equal to *c*.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	0.0604	13.024	6.7999	4.6734	1.9640	1.1540
2	W=[1 1]	0.0235	13.001	6.7290	5.3853	2.2410	1.3147
3	W=[0 1]	-0.7703	12.994	6.8051	5.2696	2.1968	1.2867

Table 3-27: Performance of LGAC on Linearized System (3.6) (Large X_d^0 Range)

Figure 3-66 through Figure 3-71 contain the system signal responses to the initial condition (0.5,0,0,0) for the linear GAC on the double inverted pendulum system (3.5)

using a large X_d^0 range. The controller gains of this set of controllers are similar, hence the dynamic responses are similar; none of them being an improvement over the linear LARC. It is suspected that the number of checking points used is insufficient for the large X_d^0 range. The same number of points was used in this case as with the smaller X_d^0 range, meaning the estimates of $\gamma_{\min}$ and $|u|_{\max}$ were better for the small X_d^0 range, making the controllers better. The large range is 10 times larger than the small range. Given a 4^{th} order system, to duplicate the checking set density of the small range for the large range, the large range checking set size would have to be increased by a factor of 10,000! We now see the major drawback of direct point-wise CLF checking of performance. The random search of the critical points relaxes the checking set size requirements, however the sufficient number of checking points still becomes prohibitively large for higher dimensional systems. Future research should focus on reducing the scale rate of the required checking set size with the system dimension.

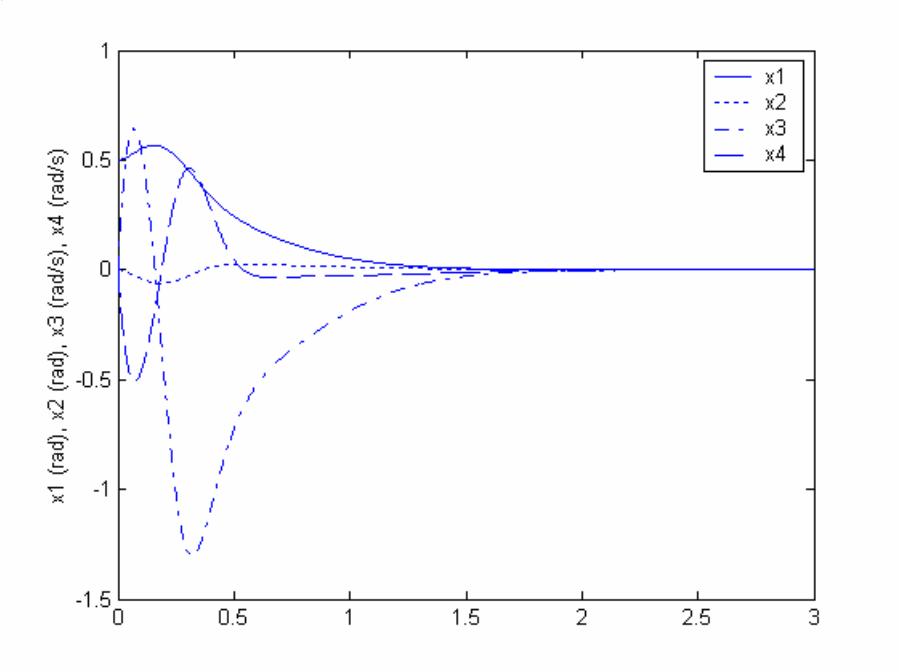

Figure 3-66: States of System (3.5) Using LGAC (W=[1 0], Large X_d^0 Range)

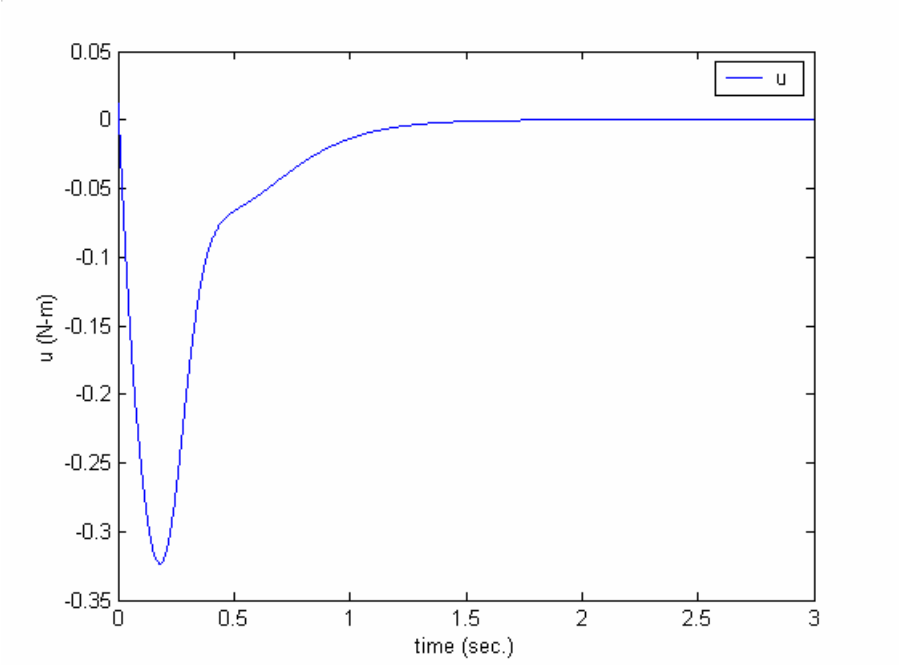

Figure 3-67: Control Signal of System (3.5) Using LGAC (W=[1 0], Large X_d^0 Range)

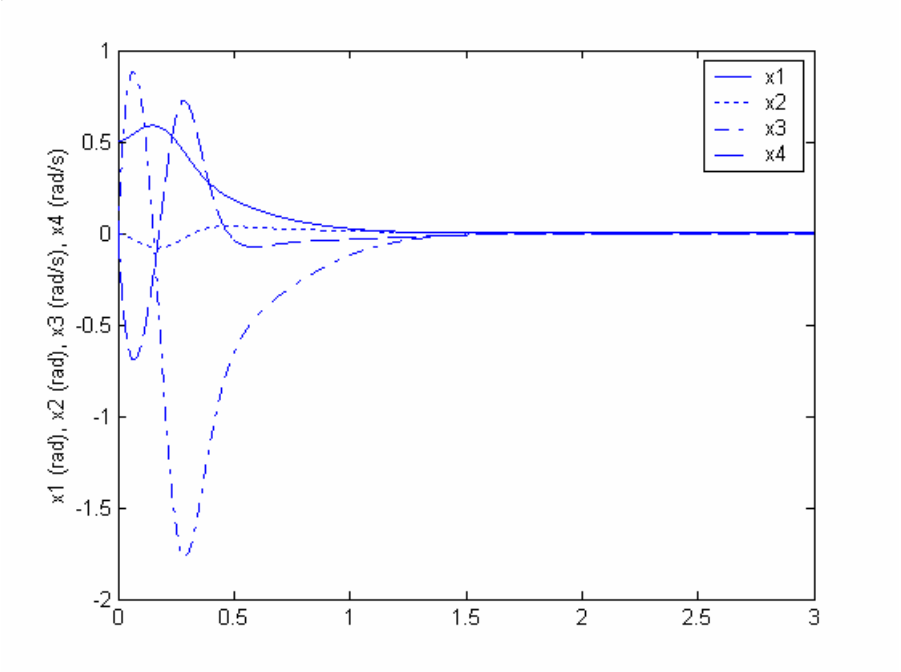

Figure 3-68: States of System (3.5) Using LGAC (W=[1 1], Large X_d^0 Range)

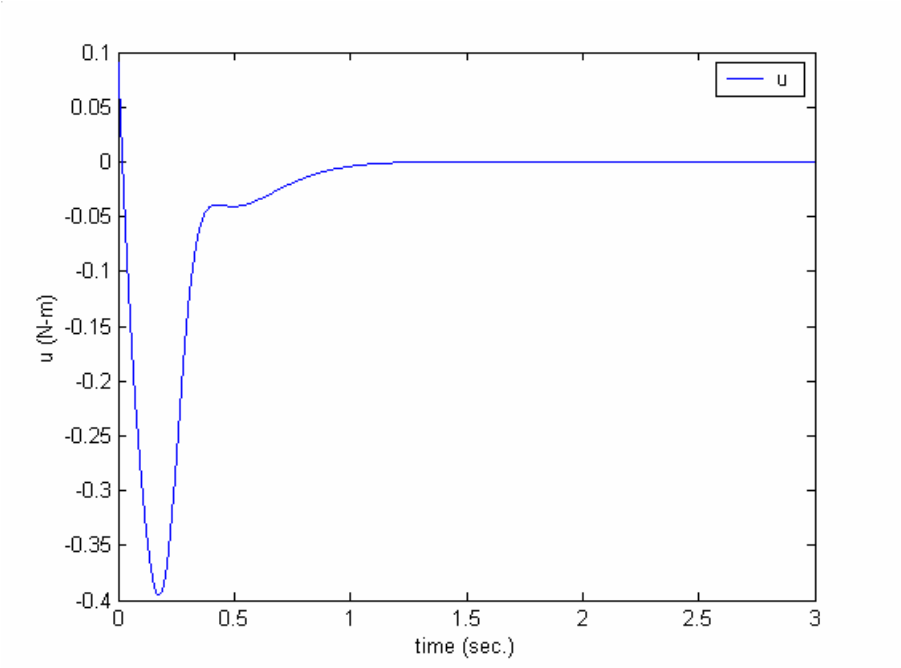

Figure 3-69: Control Signal of System (3.5) Using LGAC (W=[1 1], Large X_d^0 Range)

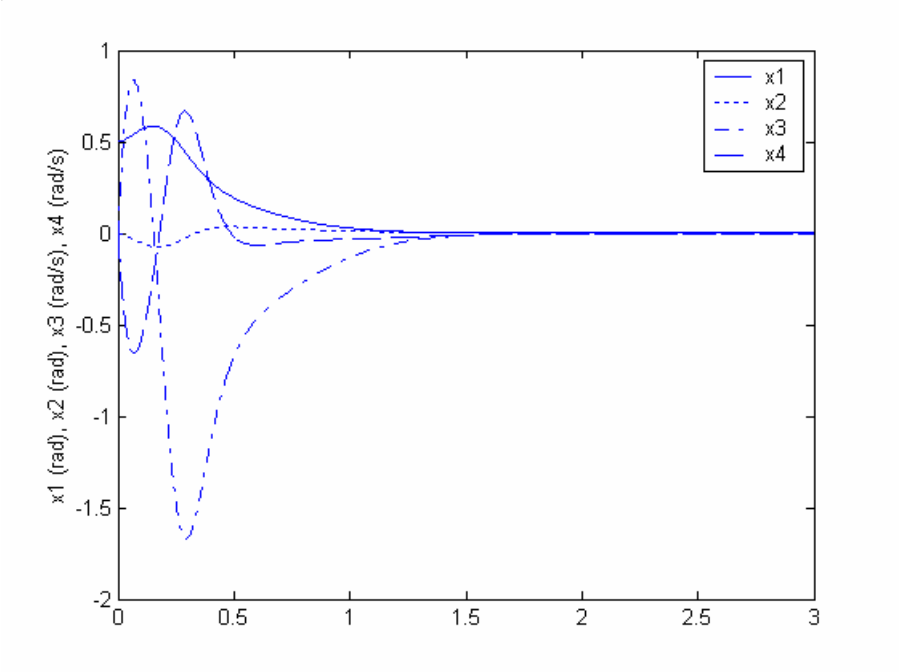

Figure 3-70: States of System (3.5) Using L GAC (W=[0 1], Large X_d^0 Range)

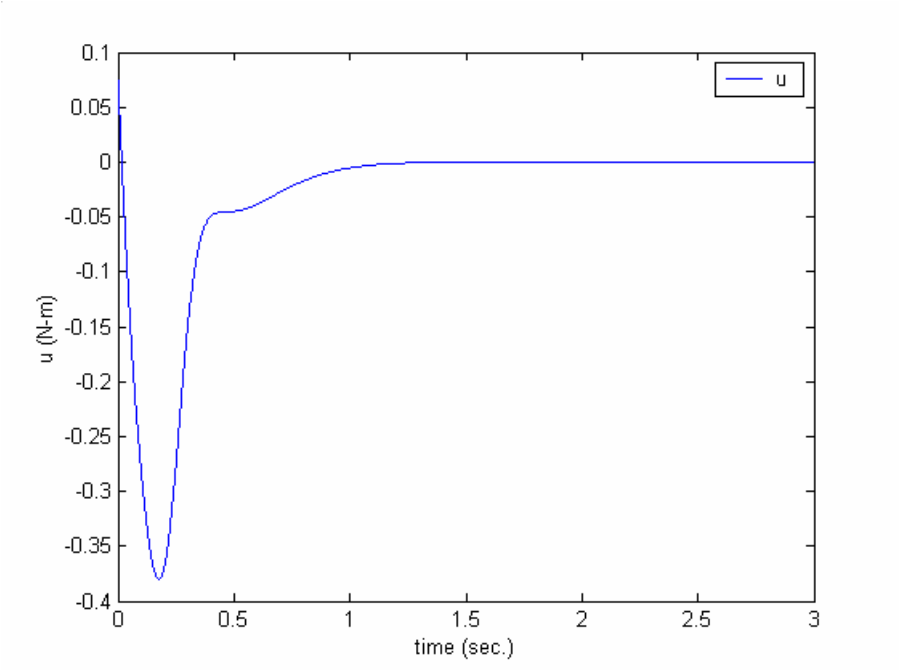

Figure 3-71: Control Signal of System (3.5) Using LGAC (W=[0 1], Large X_d^0 Range)

3.3.4 Nonlinear GAC with Small Checking Set Range

Table 3-28 shows the genetic algorithm parameters used to optimize the nonlinear GAC for the double inverted pendulum system (3.5) using a small X_d^0 range. The checking set range is set small to minimize the effects of the system nonlinearities. The control gain upper limit, K_{max}, is set to 10.

System: Double Inverted Pendulum (3.5)	Controller Type: nonlinear	Generations: 50	Population Size: 50	Checking points: 200
Critical Points: 50	Δ_E: 10	$K_{\max}$: 10	c: 0	X^0 Range: $x_i \in [-0.1 \quad 0.1]$ $i = 1,2,3,4$

Table 3-28: GA Parameters to Optimize NGAC for System (3.5) (Small X_d^0 Range)

Table 3-29 displays the GA results of the optimization of the nonlinear GAC for the double inverted pendulum system (3.5) using a small X^0 range. The results are similar to those of the linear GAC run.

Weight Case	Weights	P	K	Q
1	W=[1 0]	0.1536 0.3442 0.0453 0.0522 0.3442 1.3934 0.1474 0.2079 0.0453 0.1474 0.0169 0.022 0.0522 0.2079 0.0222 0.0311	0.1442 4.3183 0.3216 0.6321	0.1301 -0.0019 0.0011 -0.0000 -0.0019 0.0000 -0.0000 -0.0000 0.0011 -0.0000 0.0001 0.0002 -0.0000 -0.0000 0.0002 0.0006
2	W=[1 1]	0.1524 0.3407 0.0449 0.0518 0.3407 1.3835 0.1465 0.2067 0.0449 0.1465 0.0168 0.0220 0.0518 0.2067 0.0220 0.0309	0.1420 4.3067 0.3202 0.6298	0.1279 0.0021 0.0011 0.0007 0.0021 0.0002 0.0000 -0.0001 0.0011 0.0000 0.0001 0.0001 0.0007 -0.0001 0.0001 0.0001
3	W=[0 1]	0.1665 0.3652 0.0480 0.0554 0.3652 1.4418 0.1535 0.2154 0.0480 0.1535 0.0177 0.0231 0.0554 0.2154 0.0231 0.0322	0.1666 4.3942 0.3302 0.6428	0.1541 -0.0022 -0.0005 -0.0002 -0.0022 0.0001 0.0000 0.0000 -0.0005 0.0000 0.0000 0.0000 -0.0002 0.0000 0.0000 0.0000

Table 3-29: Optimized Parameters of NGAC for System (3.5) (Small X_d^0 Range)

Table 3-30 lists the randomly sampled performance of the nonlinear GAC on the double inverted pendulum system (3.5) using a small X_d^0 range. The nonlinear LARC's minimum rate of convergence is about twice as fast as the nonlinear GAC's, but the

nonlinear LARC's maximum control effort is over twice as much as the nonlinear GAC's.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	0.2230	12.9758	6.7768	0.5439	0.2201	0.1311
2	W=[1 1]	0.0416	12.9448	6.7075	0.5349	0.2216	0.1311
3	W=[0 1]	0.1084	12.9666	6.7272	0.5400	0.2256	0.1338

Table 3-30: Performance of NGAC on System (3.5) (Small X_d^0 Range)

Table 3-31 tabulates the randomly sampled performance of the nonlinear GAC on the linearized double inverted pendulum system (3.6) using a small X_d^0 range. As with the linear GAC, the linearized dynamics are close to the nonlinear dynamics locally decreasing the effect of the system nonlinearities.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	0.2890	13.0663	6.9025	0.5251	0.2170	0.1291
2	W=[1 1]	0.1984	12.9550	6.7552	0.5317	0.2176	0.1283
3	W=[0 1]	0.0433	12.9104	6.8273	0.5330	0.2215	0.1303

Table 3-31: Performance of NGAC on Linearized System (3.6) (Small X_d^0 Range)

Figure 3-72 through Figure 3-77 contain the system signal responses to the initial condition (0.5,0,0,0) for the nonlinear GAC on the double inverted pendulum system (3.5) using a small X_d^0 range. As with the nonlinear LARC, the nonlinear GAC for the first two weight settings yields a choppy control response that in turn causes the response be choppy. The weight setting *W*=[0 1] is ideal and is a major improvement over that of the nonlinear LARC.

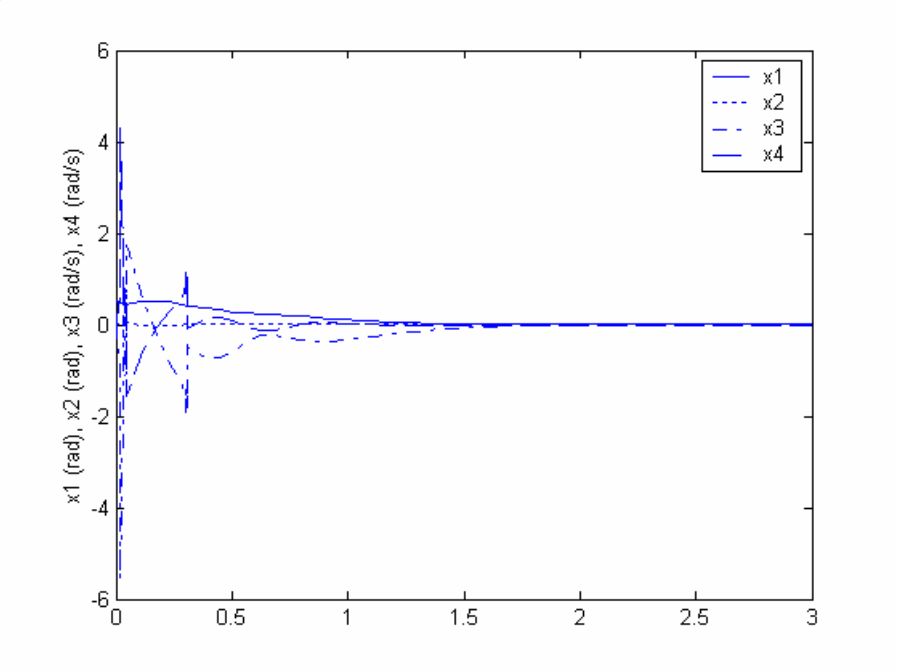

Figure 3-72: States of System (3.5) Using NGAC (W=[1 0], Small X_d^0 Range)

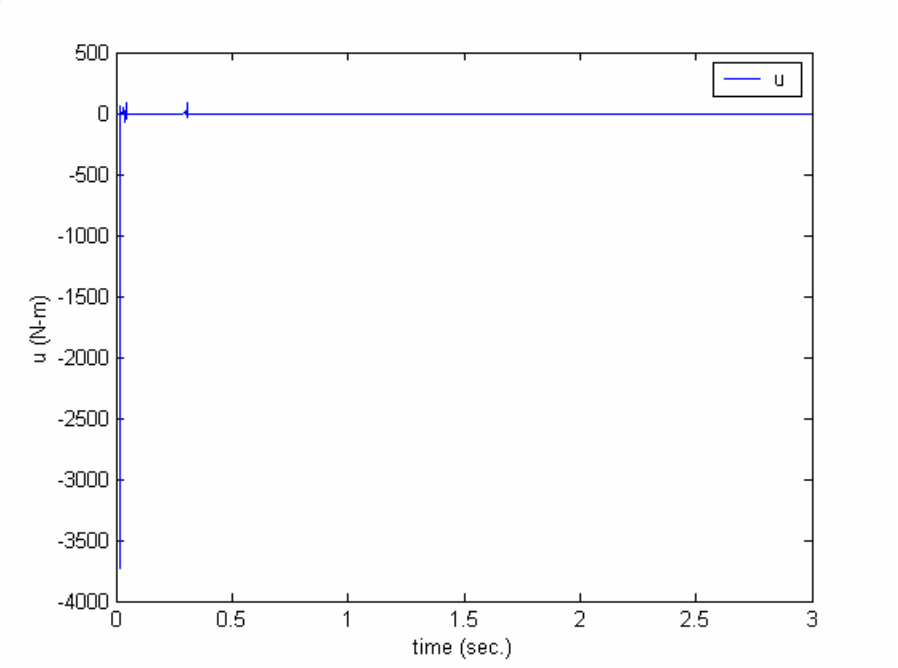

Figure 3-73: Control Signal of System (3.5) Using NGAC (W=[1 0], Small X_d^0 Range)

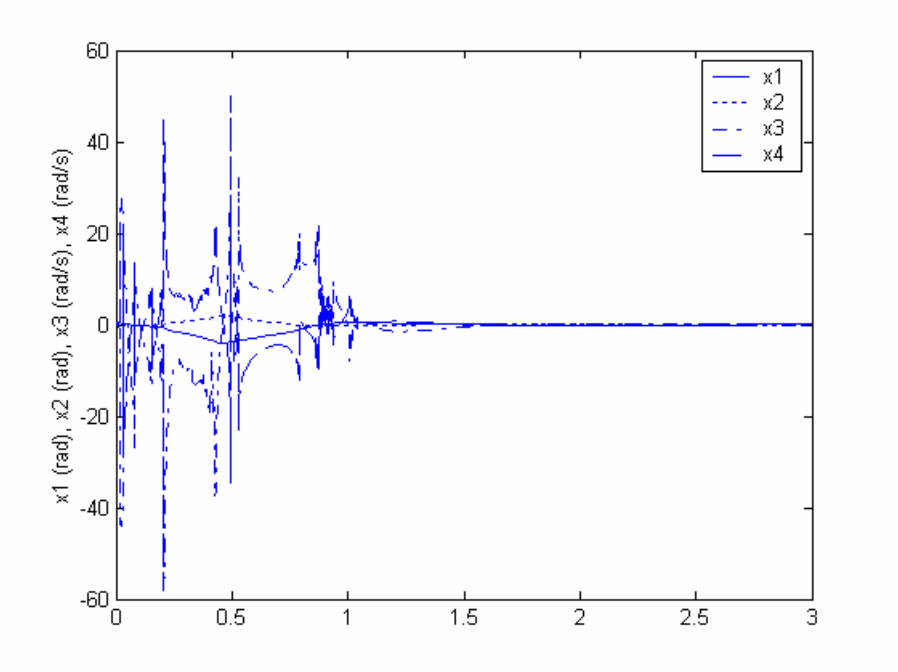

Figure 3-74: States of System (3.5) Using NGAC (W=[1 1], Small X_d^0 Range)

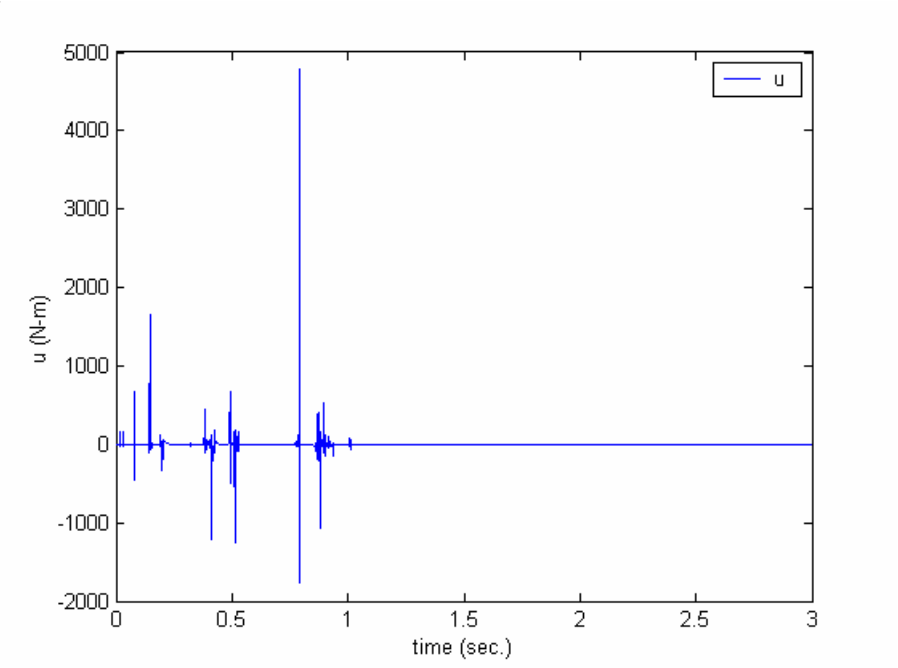

Figure 3-75: Control Signal of System (3.5) Using NGAC (W=[1 1], Small X_d^0 Range)

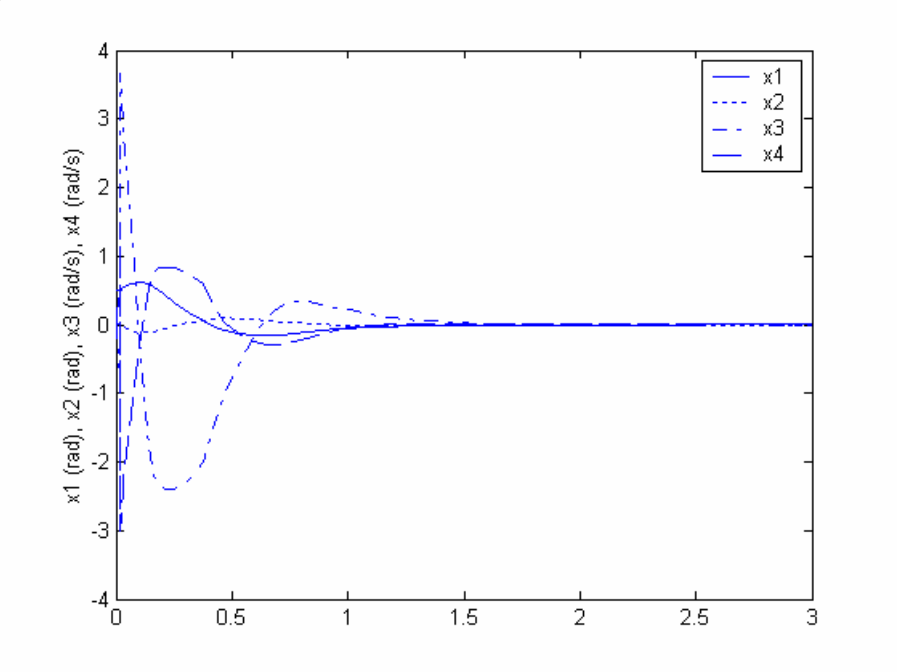

Figure 3-76: States of System (3.5) Using NGAC (W=[0 1], Small X_d^0 Range)

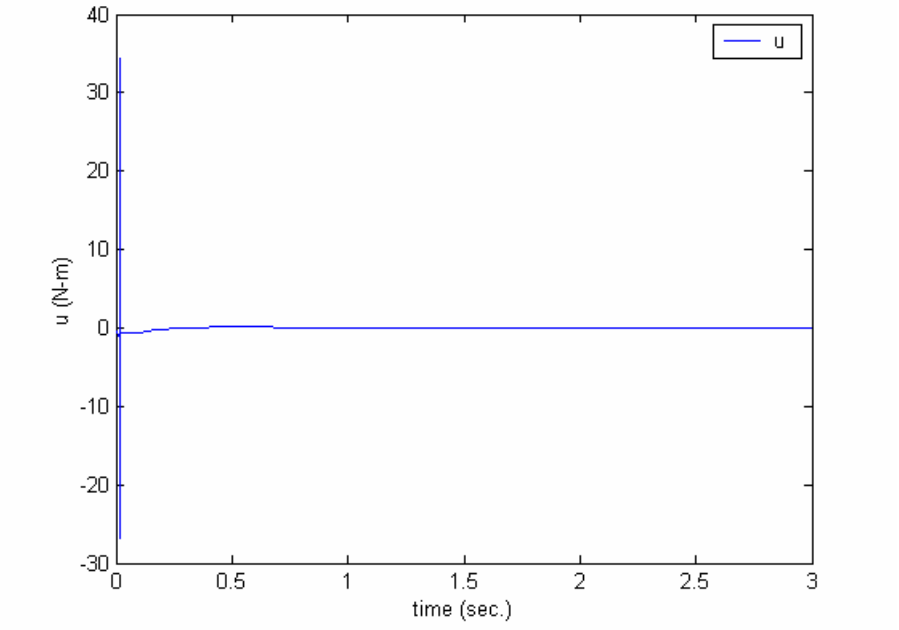

Figure 3-77: Control Signal of System (3.5) Using NGAC (W=[0 1], Small X_d^0 Range)

3.3.5 Nonlinear GAC with Large Checking Set Range

Table 3-32 shows the genetic algorithm parameters used to optimize the nonlinear GAC for the double inverted pendulum system (3.5) using a large X_d^0 range. The checking set range is set high to demonstrate the affects of the system nonlinearities. The control gain upper limit, K_{max}, is set to 10.

System: Double Inverted Pendulum (3.5)	Controller Type: nonlinear	Generations: 50	Population Size: 50	Checking points: 200
Critical Points: 50	Δ_E: 10	$K_{\max}$: 10	c: 0	X^0 Range: $x_i \in [-1 \quad 1]$ $i = 1,2,3,4$

Table 3-32: GA Parameters to Optimize NGAC for System (3.5) (Large X_d^0 Range)

Table 3-33 displays the GA results of the optimization of the nonlinear GAC for the double inverted pendulum system (3.5) using a large X^0 range.

Weight Case	Weights	P	K	Q
1	W=[1 0]	0.1009 0.2370 0.0322 0.0363 0.2370 1.1231 0.1154 0.1678 0.0322 0.1154 0.0131 0.0174 0.0363 0.1678 0.0174 0.0251	0.0237 3.8883 0.2727 0.5673	0.0186 0.0019 -0.0002 0.0008 0.0019 0.0002 -0.0000 0.0000 -0.0002 -0.0000 0.0000 0.0001 0.0008 0.0000 0.0001 0.0002
2	W=[1 1]	0.2060 0.4427 0.0577 0.0670 0.4427 1.6272 0.1759 0.2432 0.0577 0.1759 0.0204 0.0264 0.0670 0.2432 0.0264 0.0364	0.2423 4.6634 0.3609 0.6830	0.2426 -0.0081 0.0009 -0.0009 -0.0081 0.0003 -0.0000 0.0000 0.0009 -0.0000 0.0000 0.0000 -0.0009 0.0000 0.0000 0.0000
3	W=[0 1]	0.1272 0.2925 0.0390 0.0445 0.2925 1.3032 0.1371 0.1950 0.0390 0.1371 0.0158 0.0207 0.0445 0.1950 0.0207 0.0292	0.0807 4.1851 0.3027 0.6116	0.0677 -0.0005 0.0006 0.0001 -0.0005 0.0001 0.0002 0.0000 0.0006 0.0002 0.0015 0.0003 0.0001 0.0000 0.0003 0.0001

Table 3-33: Optimized Parameters of NGAC for System (3.5) (Large X_d^0 Range)

Table 3-34 lists the randomly sampled performance of the nonlinear GAC on the double inverted pendulum system (3.5) using a large X_d^0 range.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	0.1073	9.7520	3.4343	45061	11.004	332.63
2	W=[1 1]	0.1202	10.819	3.6796	9902.3	9.0307	107.42
3	W=[0 1]	0.0798	9.9874	3.2041	9128.9	8.2549	92.792

Table 3-34: Performance of NGAC on System (3.5) (Large X_d^0 Range)

Table 3-35 tabulates the randomly sampled performance of the nonlinear GAC on the linearized double inverted pendulum system (3.6) using a large X_d^0 range. In theory, the nonlinear GAC should yield a positive $\gamma_{\min}$ for the linearized double inverted pendulum system since the nonlinear GAC becomes the LQR for *Q* in Table 3-33. However, weight case 3 yielded a *P* with an eigenvalue of -0.00001313 due to *Q* having eigenvalues so close to zero, which caused $\gamma_{\min}$ to have a small negative value. Setting *c>0* forces the eigenvalues of *Q* to be no less than *c* and alleviates the problem.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	0.1505	12.9276	6.6813	5.0258	2.1126	1.2399
2	W=[1 1]	0.0933	13.1822	6.9588	5.9003	2.3706	1.3879
3	W=[0 1]	-0.0045	12.9970	6.8068	4.6618	1.9632	1.1494

Table 3-35: Performance of NGAC on System (3.6) (Large X_d^0 Range)

Figure 3-78 through Figure 3-83 contain the system signal responses to the initial condition (0.5,0,0,0) for the nonlinear GAC on the double inverted pendulum system (3.5) using a large X_d^0 range. All three sets of weights yield controller chattering. The cause is attributed to lack of sufficient number of checking points in X_d^0.

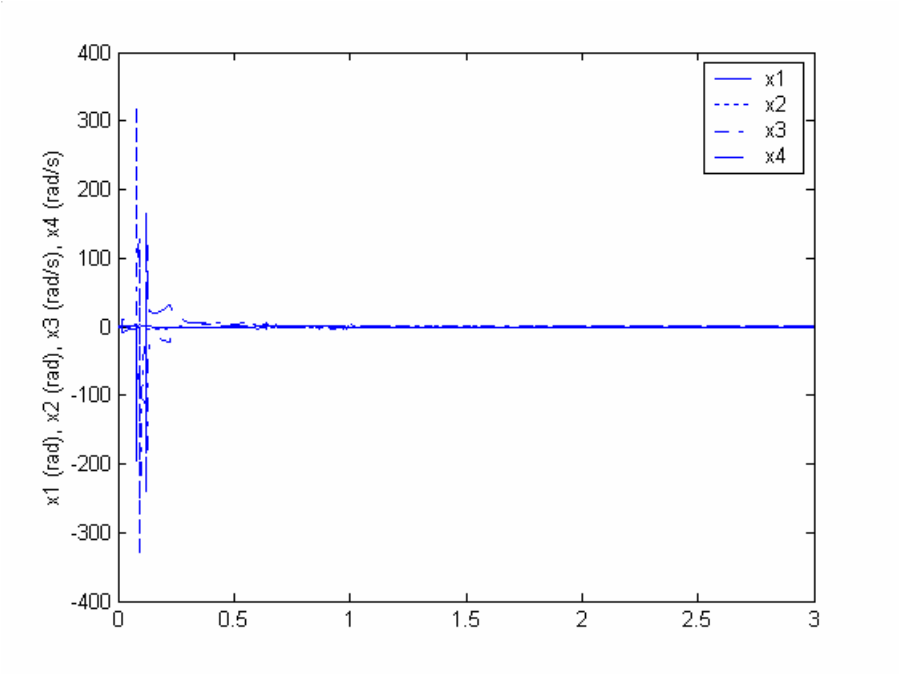

Figure 3-78: States of System (3.5) Using NGAC (W=[1 0], Large X_d^0 Range)

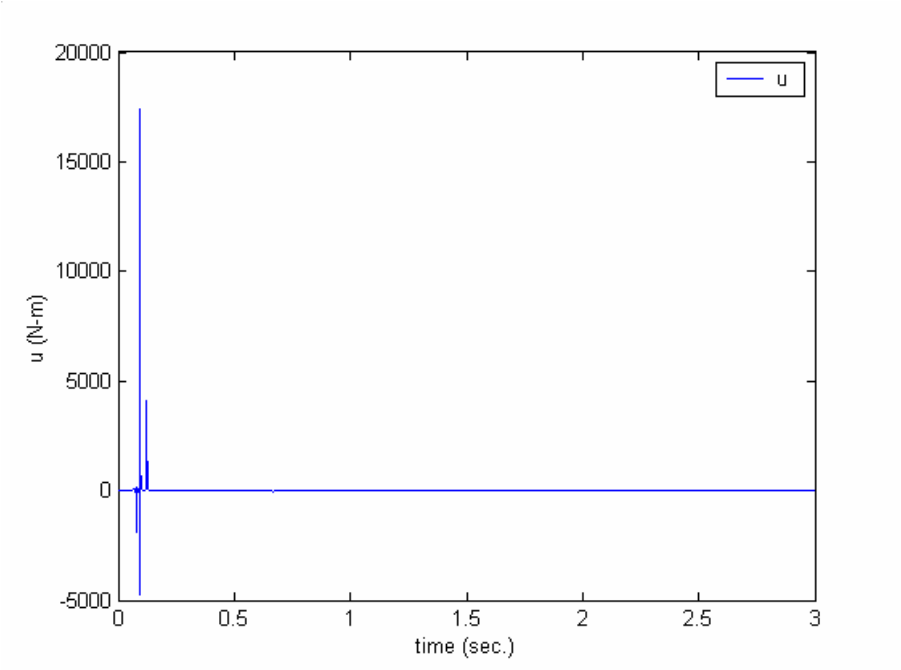

Figure 3-79: Control Signal of System (3.5) Using NGAC (W=[1 0], Large X_d^0 Range)

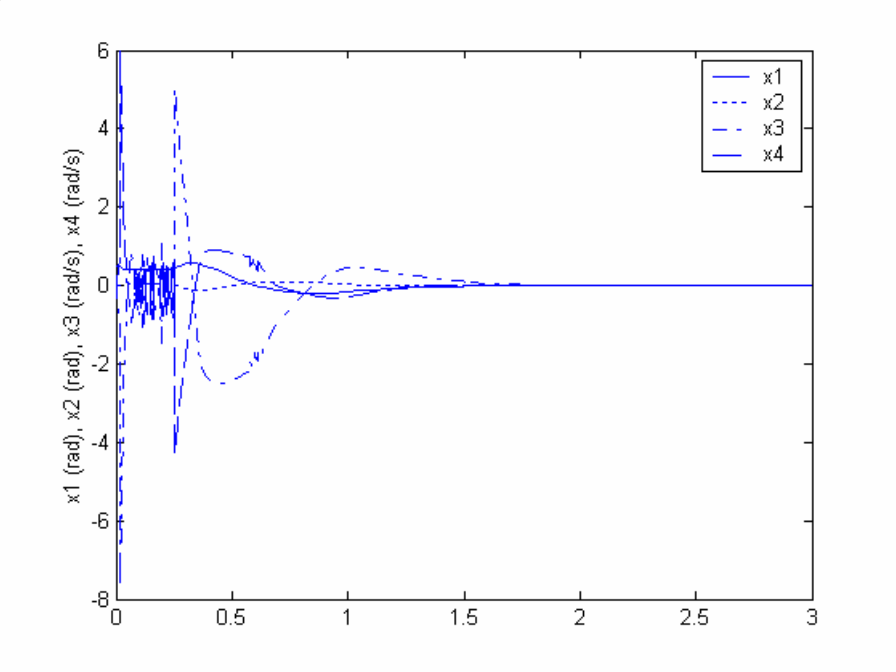

Figure 3-80: States of System (3.5) Using NGAC (W=[1 1], Large X_d^0 Range)

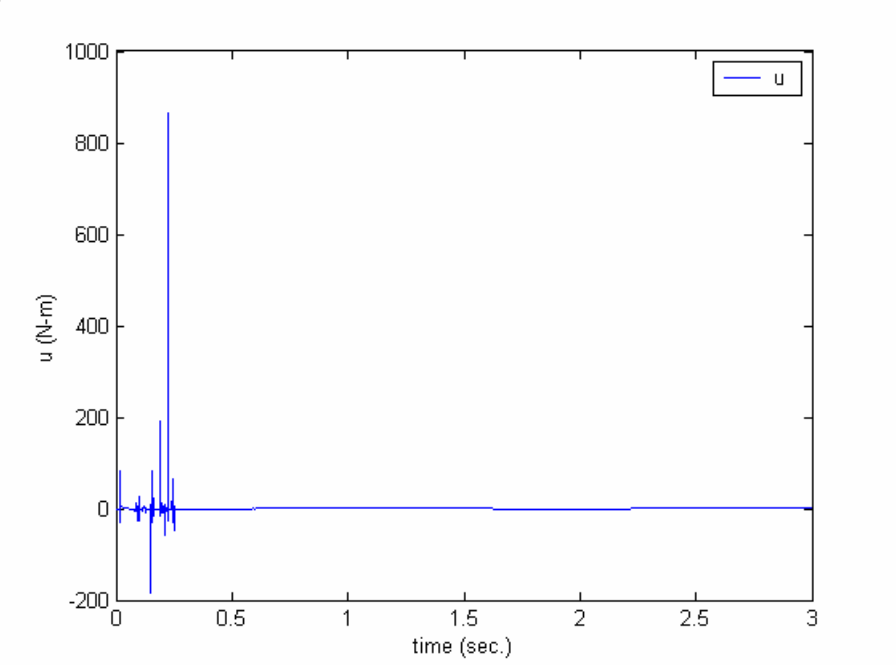

Figure 3-81: Control Signal of System (3.5) Using NGAC (W=[1 1], Large X_d^0 Range)

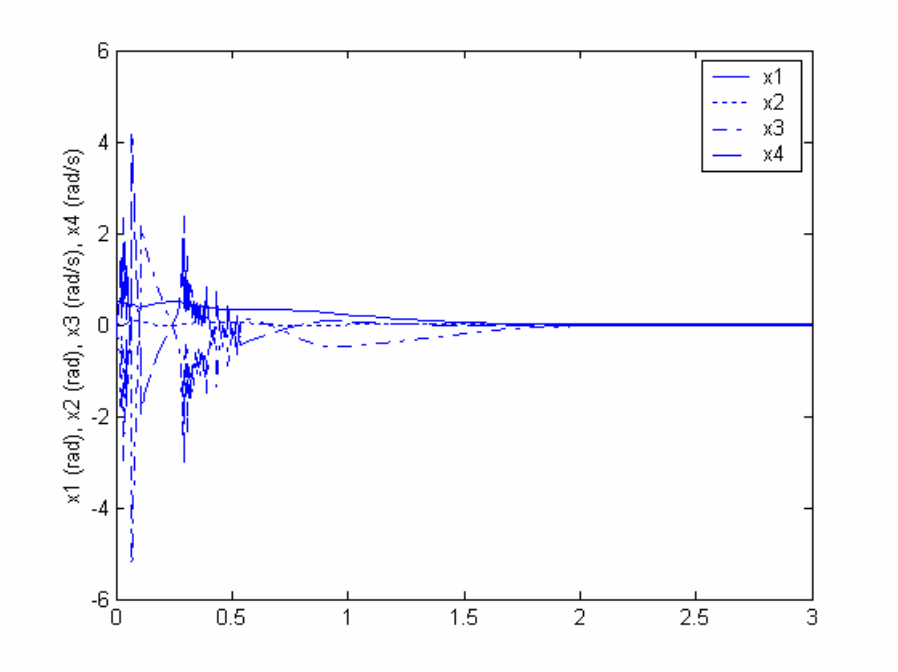

Figure 3-82: States of System (3.5) Using NGAC (W=[0 1], Large X_d^0 Range)

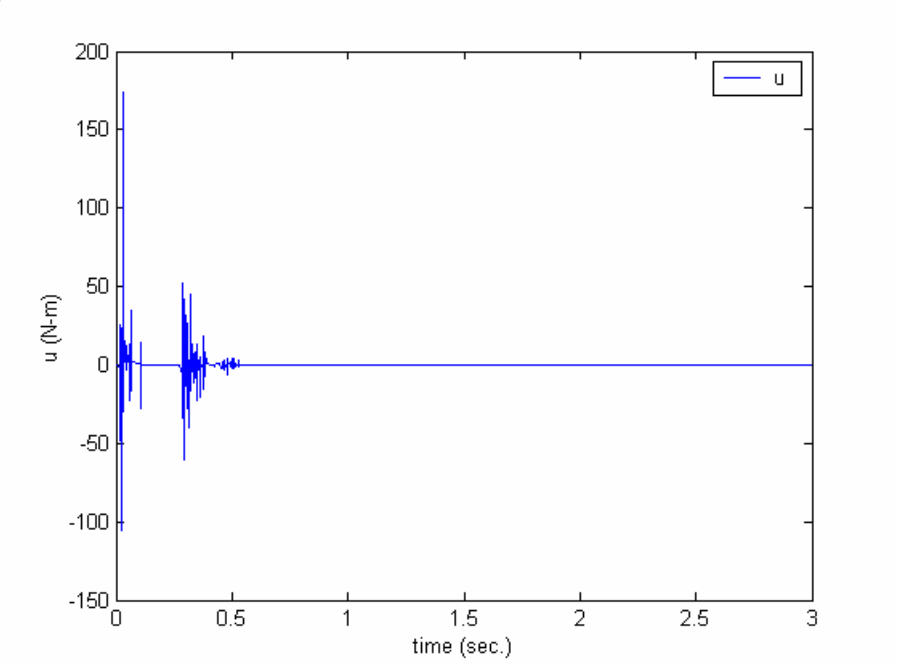

Figure 3-83: Control Signal of System (3.5) Using NGAC (W=[0 1], Large X_d^0 Range)

3.4 Example 3: Cart-and-Pole System

The cart-and-pole system in figure 3.82 was taken originally from Slotine and Li (1991) and Ogata (1997) where it is described in detail.

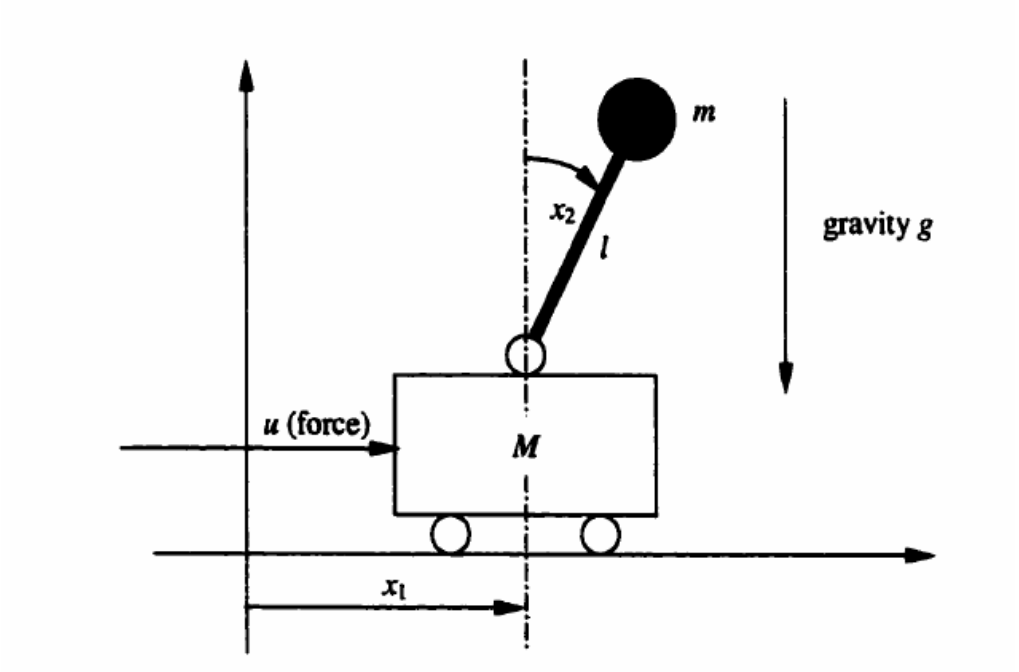

Figure 3.83: A Cart-and-Pole System (Slotine and Li, 1991),(Ogata, 1997)

The equations of motion are

$$\dot{x} = f(x) + g(x)u \tag{3.7}$$

where $x \in \Re^4$, $f, g : \Re^4 \to \Re^4$, $u \in \Re$, and

$$f_1 = x_3$$
$$f_2 = x_4$$
$$f_3 = \frac{ml\sin(x_2)x_4^2 - mg\sin(x_2)\cos(x_2)}{M + m\sin^2(x_2)}$$
$$f_4 = \frac{(M+m)g\sin(x_2)}{(M + m\sin^2(x_2))l} - \frac{m\sin(x_2)\cos(x_2)x_4^2}{M + m\sin^2(x_2)}$$

$$g_1 = 0$$
$$g_2 = 0$$
$$g_3 = \frac{1}{M + m\sin^2(x_2)}$$
$$g_4 = \frac{1/l}{M + m\sin^2(x_2)}$$

The linearized dynamics about the upright position using the parameter values *M = 2 kg, m = 0.1 kg, l= 0.5 m, and g = 9.81* $\frac{kg \cdot m}{s^2}$ are:

$\dot{x} = Ax + Bu$, with

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & -0.4905 & 0 & 0 \\ 0 & 20.6010 & 0 & 0 \end{bmatrix} \tag{3.8}$$

$$B = \begin{bmatrix} 0 \\ 0 \\ 0.5 \\ -1 \end{bmatrix}$$

3.4.1 LARC Results

As with the double inverted pendulum system example, two ranges of the checking set X_d^0 were used for all controllers in this example. The small range, $x_i \in [-0.1 \quad 0.1], i = 1,2,3,4$, is used to capture the local slightly nonlinear behavior about the origin. The large range, $x_i \in [-1 \quad 1], i = 1,2,3,4$, is used to capture the highly nonlinear behavior away from the origin. Table 3-36 displays the LARC results for the cart-and-pole system (3.7) as reported in Ngamsom (2001).

$Px10^{-4}$				K				Q			
0.3192	0.4116	0.1548	0.0819	22.3607	180.0604	35.6916	46.0217	2000	0	0	0
0.4116	2.0490	0.5752	0.3236					0	2000	0	0
0.1548	0.5752	0.2051	0.1097					0	0	2000	0
0.0819	0.3236	0.1097	0.0640					0	0	0	2000

Table 3-36: LLARC Parameters for System (3.7) (Ngamsom, 2001)

Table 3-37 lists the randomly sampled performance of the linear LARC on the cart-and-pole system (3.7). The minimum rate of convergence is negative and increases in magnitude as the range increases.

X^0 Range	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
$x_i \in [-0.1 \quad 0.1]$ $i = 1,2,3,4$	-0.1902	0.7238	1.3742	27.4543	9.3467	5.7675
$x_i \in [-1 \quad 1]$ $i = 1,2,3,4$	-27.6645	-6.5949	7.9197	265.8421	93.6272	57.7435

Table 3-37: Performance of LLARC on System (3.7)

Table 3-38 displays the randomly sampled performance of the nonlinear LARC on the cart-and-pole system (3.7). Although positive, the minimum rate of convergence decreases with increasing range while the average rate of convergence stays about the same. The maximum control effort becomes very large for large checking set range.

X^0 Range	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
$x_i \in [-0.1 \quad 0.1]$ $i = 1,2,3,4$	0.7839	5.7403	3.0464	56.152	19.247	12.0453
$x_i \in [-1 \quad 1]$ $i = 1,2,3,4$	0.1096	5.5442	3.3907	$2.8400x10^6$	896.46	$2.3882x10^4$

Table 3-38: Performance of NLARC on System (3.7)

Table 3-39 displays the randomly sampled performance of the linear LARC on the linearized cart-and-pole system (3.8).

X^0 Range	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
$x_i \in [-0.1 \quad 0.1]$ $i = 1,2,3,4$	0.0857	0.8288	1.5351	27.2465	9.3589	5.7781
$x_i \in [-1 \quad 1]$ $i = 1,2,3,4$	0.0856	0.8175	1.5193	280.6155	94.1132	57.9198

Table 3-39: Performance of LLARC on Linearized System (3.8)

Figure 3-84 & Figure 3-85 contain the system signal responses to the initial condition (0.5,0,0,0) for the linear LARC on the cart-and-pole system (3.7). Figure 3-86 & Figure 3-87 contain the system signal responses to the initial condition (0.5,0,0,0) for the nonlinear LARC on the cart-and-pole system (3.7). The responses are very similar however the nonlinear LARC has a higher initial control effort due to the strict requirement $\dot{V} = \sqrt{\left(x^T Pf\right)^2 + \left(x^T Qx\right)\left(x^T Pg\right)^2}$.

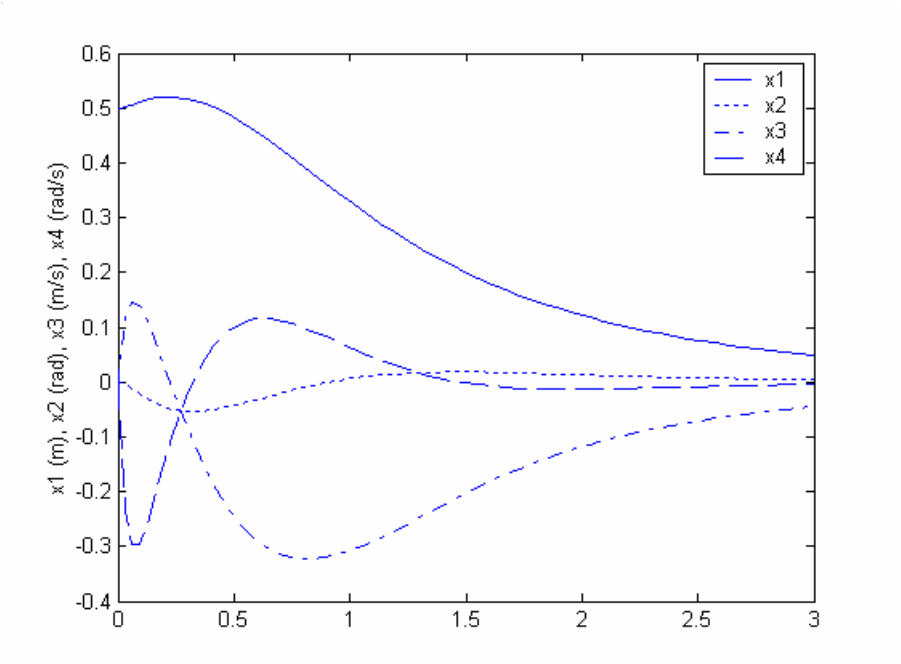

Figure 3-84: States of System (3.7) Using LLARC

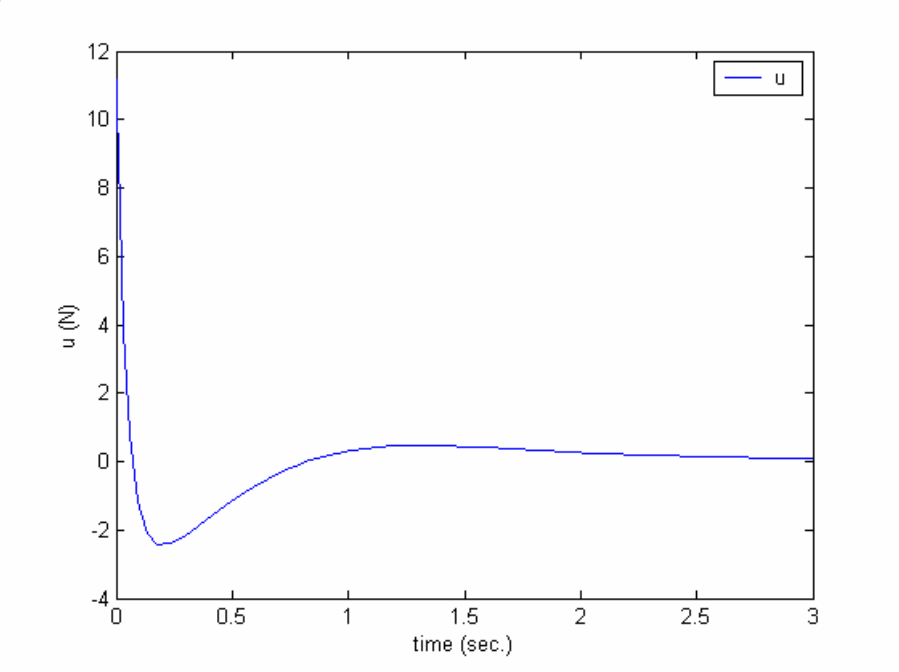

Figure 3-85: Control Signal of System (3.7) Using LLARC

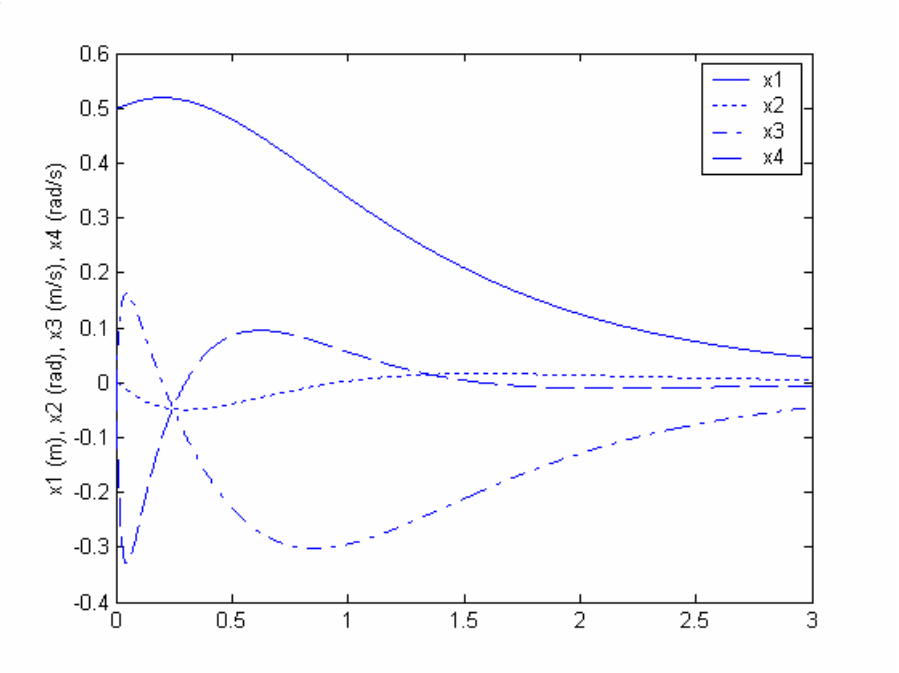

Figure 3-86: States of System (3.7) Using NLARC

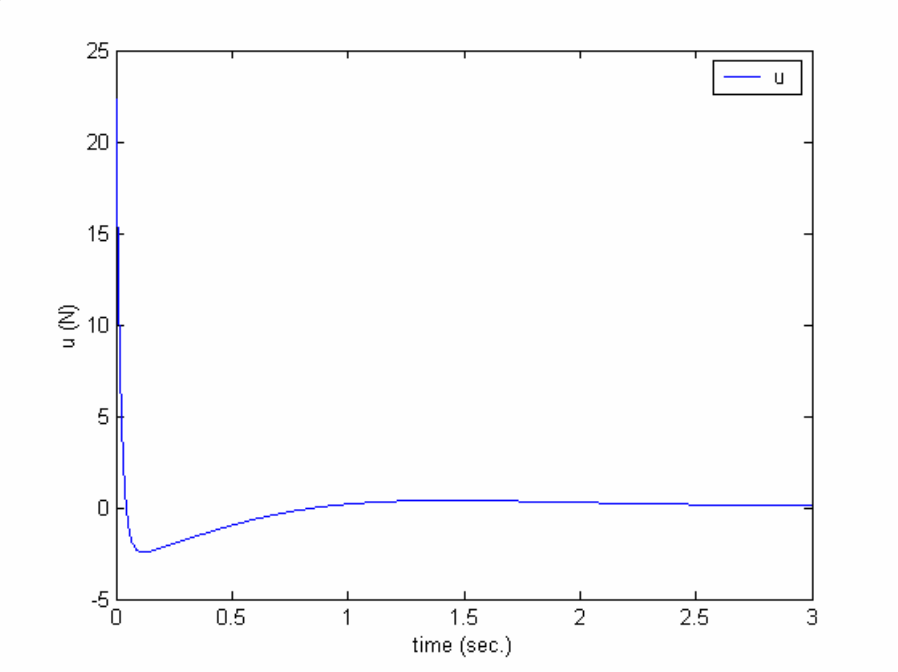

Figure 3-87: Control Signal of System (3.7) Using NLARC

3.4.2 Linear GAC with Small Checking Set Range

Table 3-40 tabulates the GA parameters used to optimize the linear GAC for the cart-and-pole system (3.7) using a small X_d^0 range.

System: Cart-and-Pole (3.7)	Controller Type: linear	Generations: 50	Population Size: 50	Checking points: 200
Critical Points: 50	Δ_E: 100	$K_{\max}$: 1000	c: 0	X^0 Range: $x_i \in [-0.1 \quad 0.1]$ $i = 1,2,3,4$

Table 3-40: GA Parameters to Optimize LGAC for System (3.7) (Small X_d^0 Range)

Table 3-41 shows the GA results of the optimization of the linear GAC for the cart-and-pole system (3.7) using a small checking set X_d^0 range. The first set of control gains are similar in magnitude to the LARC. The second set is about half the magnitude of the LARC gains while the last set is substantially smaller.

Weight Case	Weights	P	K	Q
1	W=[1 0]	2641.9 3145.2 1155.5 641.40 3145.2 5997.7 1782.9 1109.3 1155.5 1782.9 750.60 426.50 641.40 1109.3 426.50 261.80	63.60 217.84 51.17 48.54	4045.3 1215.4 612.90 -57.800 1215.4 3519.9 -410.80 -604.10 612.90 -410.80 307.60 59.800 -57.800 -604.10 59.800 137.80
2	W=[1 1]	228.92 315.43 112.01 71.407 315.43 949.23 236.53 210.02 112.01 236.53 77.047 53.585 71.407 210.02 53.585 47.538	15.40 91.75 15.06 20.74	216.03 -1.6800 2.7972 3.4266 -1.6800 0.3278 0.7261 1.3324 2.7972 0.7261 2.2937 3.6616 3.4266 1.3324 3.6616 8.3791
3	W=[0 1]	0.0011 0.0604 0.0142 0.0136 0.0604 193.63 1.5486 42.699 0.0142 1.5486 0.3615 0.3498 0.0136 42.699 0.3498 9.4160	.0065 41.924 0.169 9.241	10^{-3} *x* 0.0414 0.0022 -0.0256 -0.0148 0.0022 0.0597 -0.0477 0.2025 -0.0256 -0.0477 0.1143 -0.1535 -0.0148 0.2025 -0.1535 0.7241

Table 3-41: Optimized Parameters of LGAC for System (3.7) (Small X_d^0 Range)

Table 3-42 lists the randomly sampled performance of the linear GAC on the cart-and-pole system (3.7) using a small X_d^0 range. The minimum and average rate of convergence is better than the linear LARC for all three cases. The maximum and average control effort is also better only for the last two cases.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	1.7672	6.9593	2.8508	35.607	11.533	7.3452
2	W=[1 1]	0.0506	7.0643	2.8946	13.735	4.6982	2.9125
3	W=[0 1]	$9.0619x10^{-4}$	8.7534	1.1504	5.0841	2.1292	1.2615

Table 3-42: Performance of LGAC on System (3.7) (Small X_d^0 Range)

Table 3-43 shows the randomly sampled performance of the linear GAC on the linearized cart-and-pole system (3.8) using a small X_d^0 range. The minimum and average rate of convergence is better than the linear LARC for the first weight case but the maximum and average control effort is higher. The second weight case has a slightly better minimum rate of convergence, a much better average rate of convergence, and half the maximum and average control effort. The third weight case yields a CLF that is locally numerically ill-conditioned because $\gamma_{\min}$ was not checked during the optimization. The same result occurred in the previously discussed weight case 3 of Table 3-27.

Weight Case	Weights	$\gamma_{\min}$	μ_{γ}	σ_{γ}	$\lvert u \rvert_{\max}$	$\mu_{\lvert u \rvert}$	$\sigma_{\lvert u \rvert}$
1	W=[1 0]	1.7182	7.0822	2.8899	35.389	11.654	7.3918
2	W=[1 1]	0.0921	7.1671	2.9032	13.309	4.7735	2.9160
3	W=[0 1]	-0.0086	8.7801	1.1480	5.0915	2.1214	1.2660

Table 3-43: Performance of LGAC on Linearized System (3.8) (Small X_d^0 Range)

Figure 3-88 through Figure 3-93 contain the system signal responses to the initial condition (0.5,0,0,0) for the linear GAC on the cart-and-pole system (3.7) using a small X_d^0 range. *W*=[1 0] yields a much faster convergence than the linear LARC, but at a higher $\lvert u \rvert_{\max}$. *W*=[1 1] is the case where the convergence rate is faster and the control effort is smaller than the linear LARC. *W*=[0 1] yields a prohibitively slow response for an extremely small $\lvert u \rvert_{\max}$.

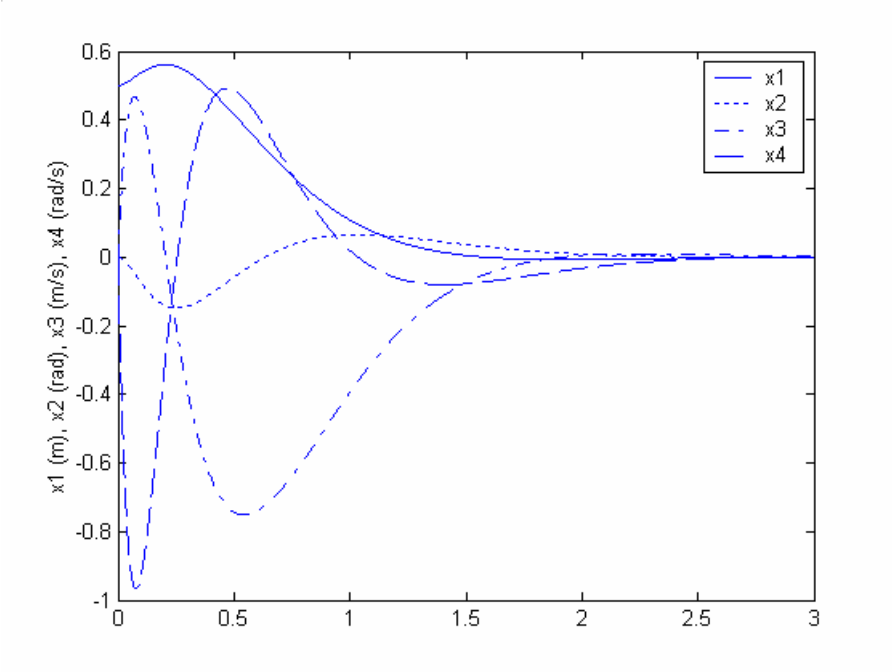

Figure 3-88: States of System (3.7) Using LGAC (W=[1 0], Small X_d^0 Range)

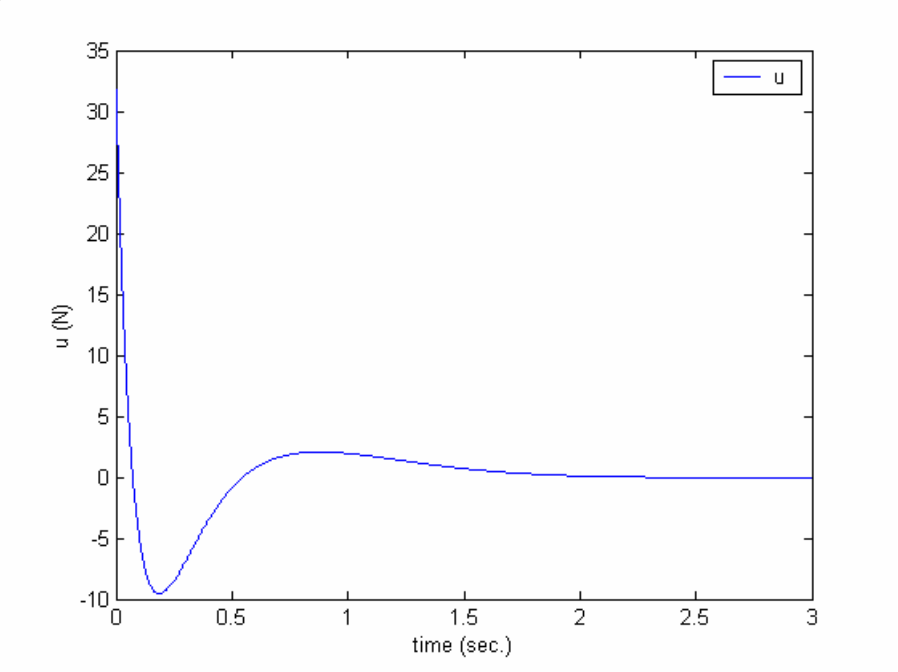

Figure 3-89: Control Signal of System (3.7) Using LGAC (W=[1 0], Small X_d^0 Range)

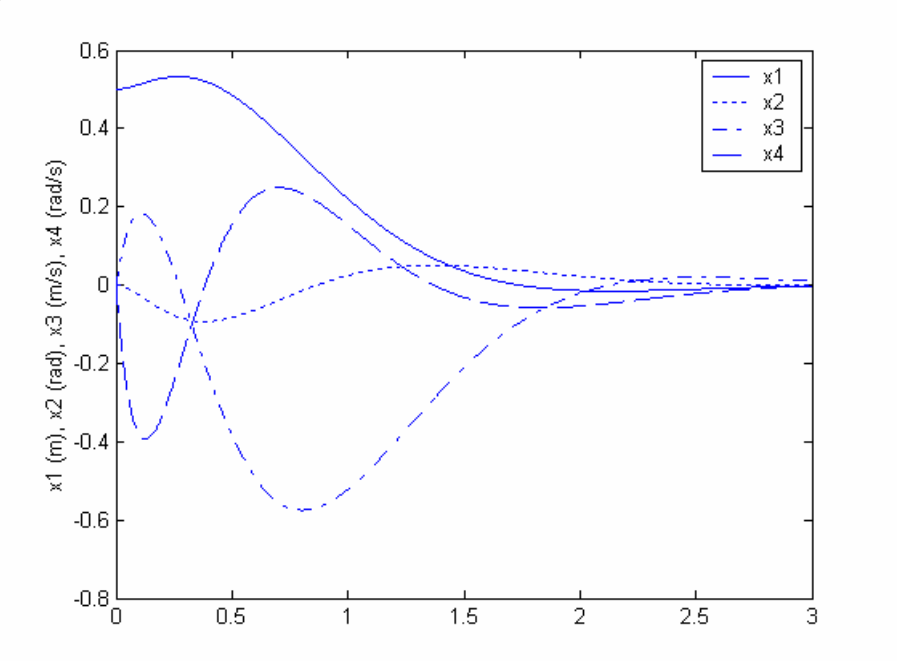

Figure 3-90: States of System (3.7) Using LGAC (W=[1 1], Small X_d^0 Range)

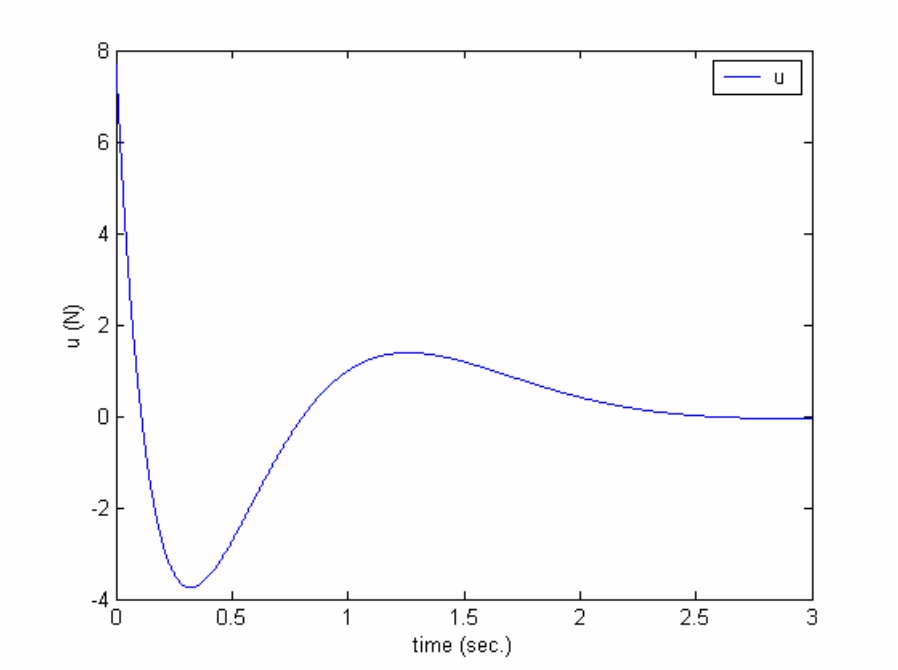

Figure 3-91: Control Signal of System (3.7) Using LGAC (W=[1 1], Small X_d^0 Range)

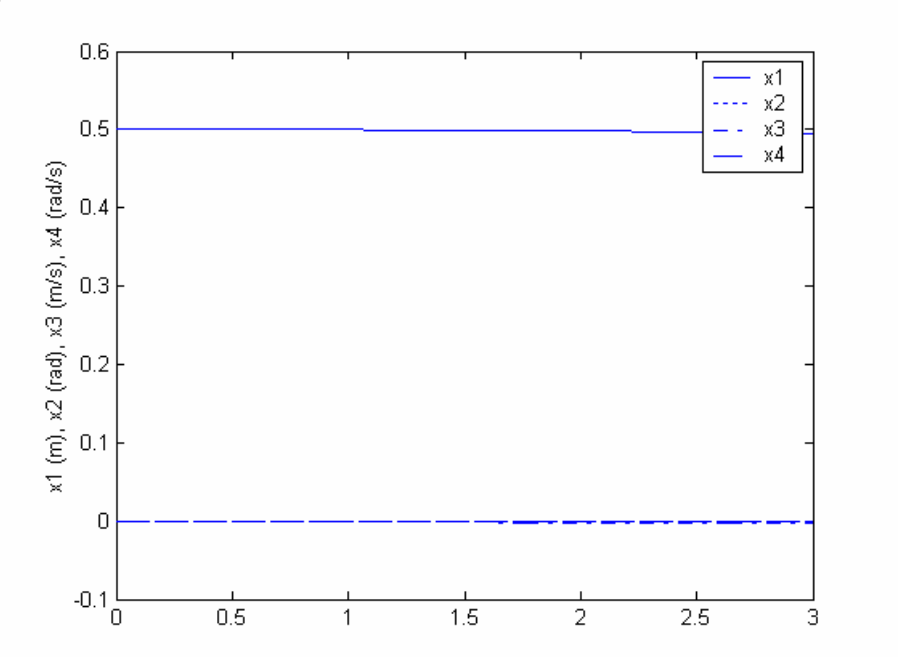

Figure 3-92: States of System (3.7) Using LGAC (W=[0 1], Small X_d^0 Range)

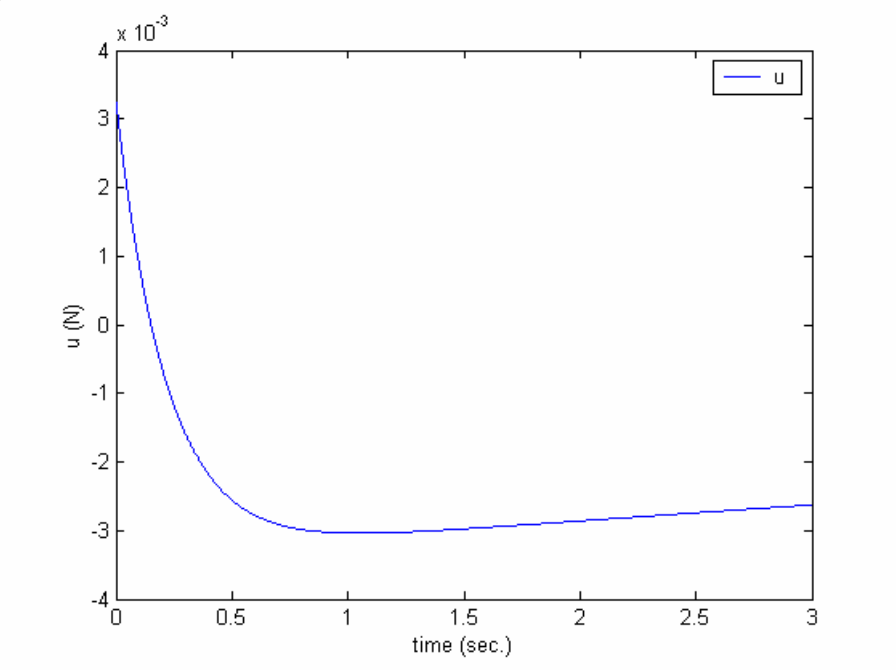

Figure 3-93: Control Signal of System (3.7) Using LGAC (W=[0 1], Small X_d^0 Range)

3.4.3 Linear GAC with Large Checking Set Range

Table 3-44 tabulates the GA parameters used to optimize the linear GAC for the cart-and-pole system (3.7) using a large X_d^0 range.

System: Cart-and-Pole (3.7)	Controller Type: linear	Generations: 50	Population Size: 50	Checking points: 200
Critical Points: 50	Δ_E: 100	$K_{\max}$: 1000	c: 0	X^0 Range: $x_i \in [-1 \quad 1]$ $i = 1,2,3,4$

Table 3-44: GA Parameters to Optimize LGAC for System (3.7) (Large X_d^0 Range)

Table 3-45 shows the GA results of the optimization of the linear GAC for the cart-and-pole system (3.7) using a large X_d^0 range. All three sets of control gains are very close and much smaller than those of the LARC.

Weight Case	Weights	P	K	Q
1	W=[1 0]	0.0250 0.4884 0.1181 0.1102 0.4884 206.11 4.5277 45.517 0.1181 4.5277 1.0732 1.0227 0.1102 45.517 1.0227 10.053	0.0511 43.252 0.486 9.541	0.0026 0.0002 -0.0001 -0.0004 0.0002 0.0000 0.0000 0.0001 -0.0001 0.0000 0.0000 0.0001 -0.0004 0.0001 0.0001 0.0004
2	W=[1 1]	0.1983 1.9905 0.5018 0.4498 1.9905 226.55 9.5821 50.135 0.5018 9.5821 2.3253 2.1645 0.4498 50.135 2.1645 11.096	0.1989 45.344 1.001 10.013	0.0396 0.0013 0.0009 0.0010 0.0013 0.0001 0.0000 0.0000 0.0009 0.0000 0.0001 0.0001 0.0010 0.0000 0.0001 0.0003
3	W=[0 1]	0.2022 2.0147 0.5068 0.4542 2.0147 226.75 9.6325 50.180 0.5068 9.6325 2.3375 2.1756 0.4542 50.180 2.1756 11.106	0.2008 45.364 1.007 10.018	0.0403 0.0029 -0.0000 -0.0028 0.0029 0.0016 -0.0003 -0.0002 -0.0000 -0.0003 0.0001 0.0000 -0.0028 -0.0002 0.0000 0.0003

Table 3-45: Optimized Parameters of LGAC for System (3.7) (Large X_d^0 Range)

Table 3-46 lists the randomly sampled performance of the linear GAC on the cart-and-pole system (3.7) using a large X_d^0 range. While still unstable, the rate of convergence is much less negative than the linear LARC. Unlike the linear LARC, the average rate of convergence is positive on X^0. Because of the smaller control gains, the maximum and average control effort is much smaller for the GAC cases.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	-0.1516	6.0398	2.2173	52.693	22.107	12.966
2	W=[1 1]	-0.0758	5.7345	2.3488	55.447	23.146	13.733
3	W=[0 1]	-0.0551	5.7263	2.3473	55.231	22.974	13.668

Table 3-46: Performance of LGAC on System (3.7) (Large X_d^0 Range)

Table 3-47 displays the randomly sampled performance of the linear GAC on the linearized cart-and-pole system (3.8) using a large X_d^0 range. The linear LARC minimum rate of convergence is better than all three linear GAC cases, however the average rate of convergence is better for the linear GAC cases.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	$7.3976x10^{-4}$	8.6143	1.4466	52.945	22.119	13.091
2	W=[1 1]	$7.1437x10^{-4}$	8.4254	1.7404	55.494	23.023	13.757
3	W=[0 1]	0.0022	8.4241	1.7325	55.849	23.024	13.745

Table 3-47: Performance of LGAC on Linearized System (3.8) (Large X_d^0 Range)

Figure 3-94 through Figure 3-99 contain the system signal responses to the initial condition (0.5,0,0,0) for the linear GAC on the cart-and-pole system (3.7) using a large X_d^0 range. All three controllers yield a very slow response with a very small control effort. Again, it is believed the most likely cause for this is an insufficient number of checking points.

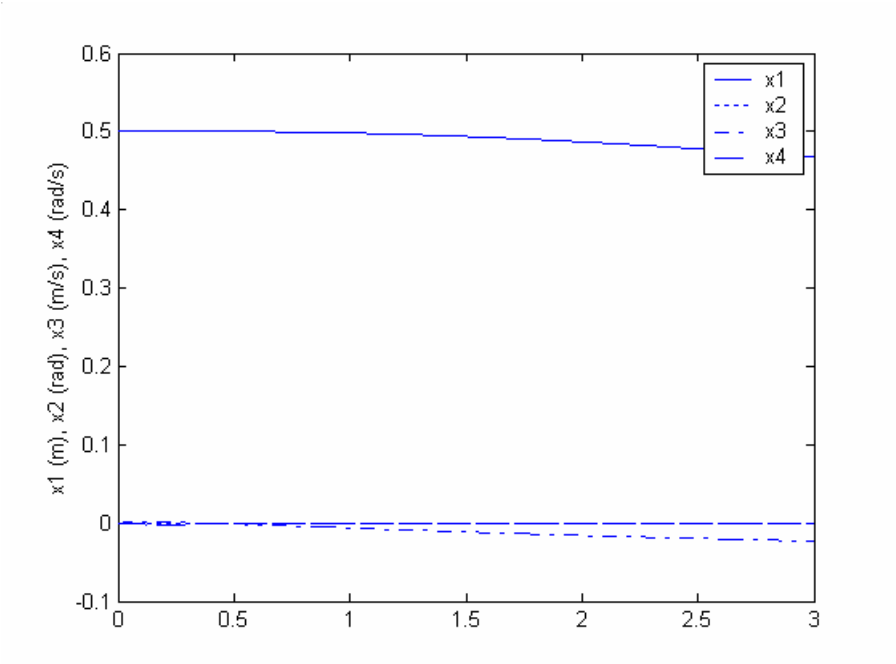

Figure 3-94: States of System (3.7) Using LGAC (W=[1 0], Large X_d^0 Range)

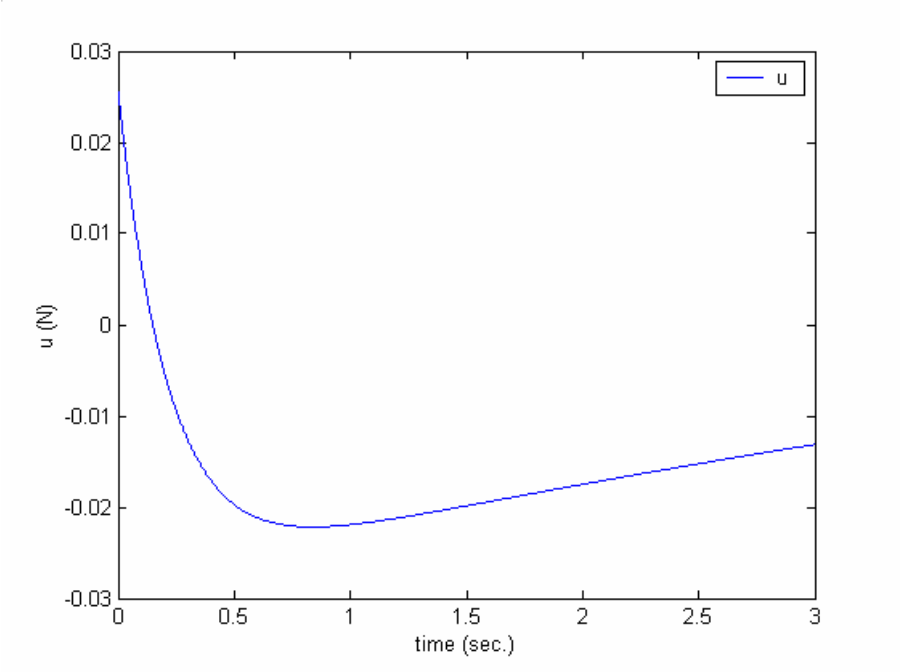

Figure 3-95: Control Signal of System (3.7) Using LGAC (W=[1 0], Large X_d^0 Range)

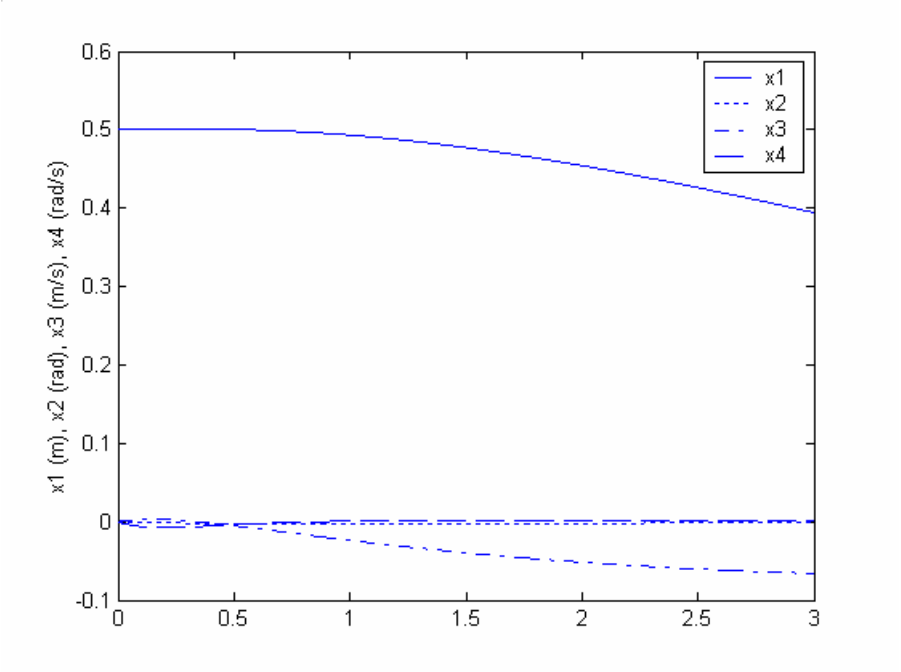

Figure 3-96: States of System (3.7) Using LGAC (W=[1 1], Large X_d^0 Range)

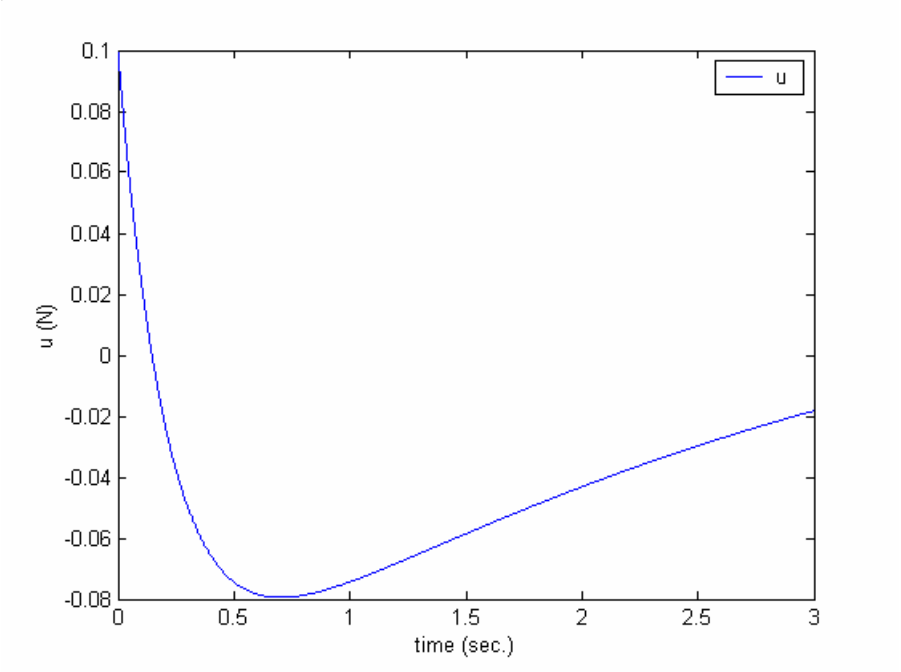

Figure 3-97: Control Signal of System (3.7) Using LGAC (W=[1 1], Large X_d^0 Range)

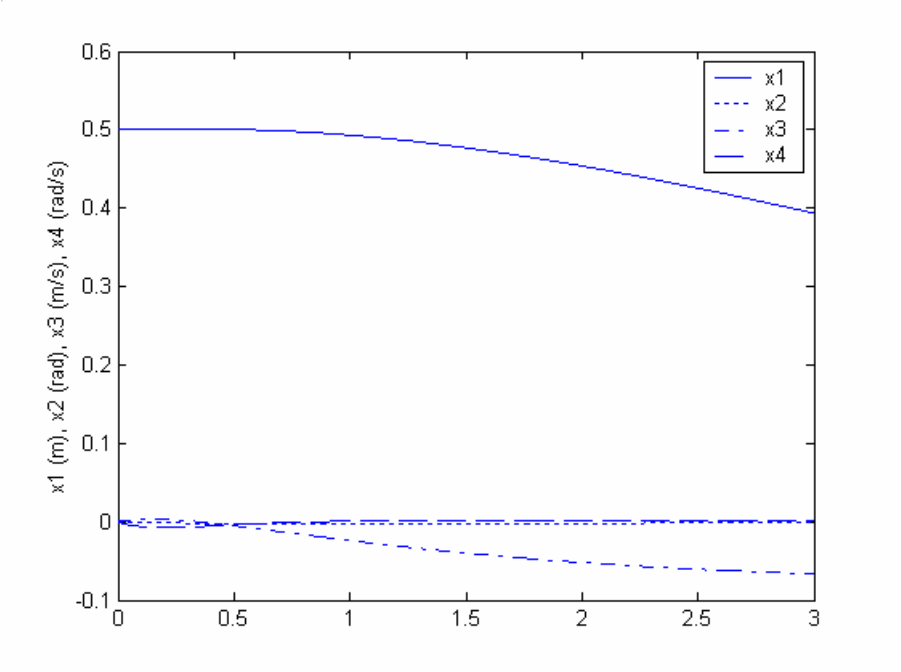

Figure 3-98: States of System (3.7) Using LGAC (W=[0 1], Large X_d^0 Range)

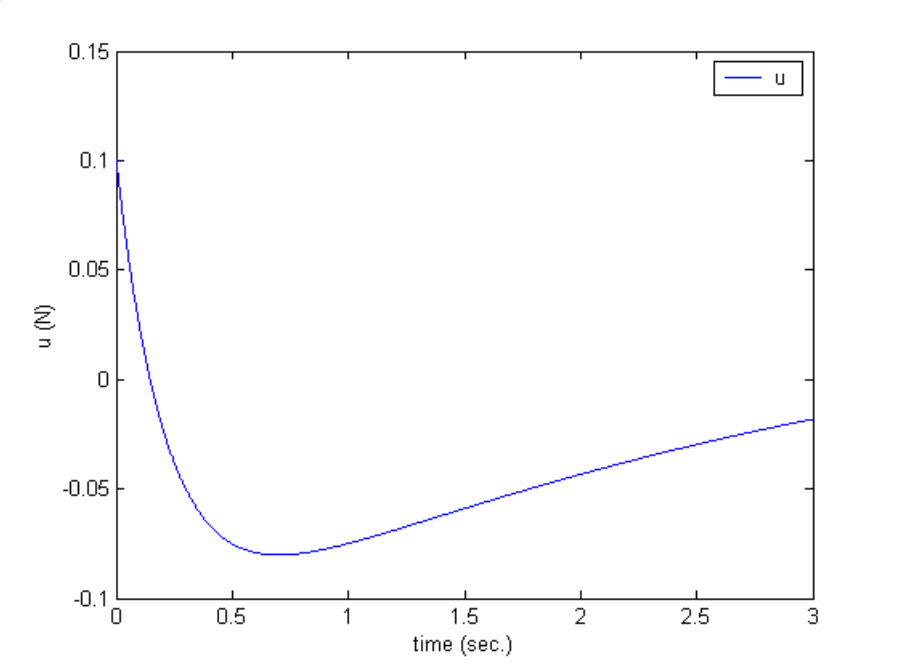

Figure 3-99: Control Signal of System (3.7) Using LGAC (W=[0 1], Large X_d^0 Range)

3.4.4 Nonlinear GAC with Small Checking Set Range

Table 3-48 displays the GA parameters used to optimize the nonlinear GAC for the cart-and-pole system (3.7) using a small X_d^0 range.

System: Cart-and-Pole (3.7)	Controller Type: nonlinear	Generations: 50	Population Size: 50	Checking points: 200
Critical Points: 50	Δ_E: 100	$K_{\max}$: 1000	c: 0	X^0 Range: $x_i \in [-.1 \quad .1]$ $i = 1,2,3,4$

Table 3-48: GA Parameters to Optimize NGAC for System (3.7) (Small X_d^0 Range)

Table 3-49 shows the GA results of the optimization of the nonlinear GAC for the cart-and-pole system using a small X_d^0 range. Unlike the linear GAC, the first two sets of gains are larger than those of the LARC for the nonlinear GAC. The last set of gains are smaller.

Weight Case	Weights	P	K	Q
1	W=[1 0]	29040 39800 16690 8570.0 39800 192950 40750 21310 16690 40750 13670 7040.0 8570.0 21310 7040.0 3820.0	222.773 933.871 202.77 304.469	49628 39780 16132 28030 39780 34576 11375 16121 16132 11375 7730.0 12420 28030 16121 12420 50086
2	W=[1 1]	59914 45370 19098 9885.0 45370 39161 16214 8699.0 19098 16214 7232.0 3815.0 9885.0 8699.0 3815.0 2040.0	335.88 592.155 199.519 132.529	100240 4700 5550 -890 4700 6720 -1810 -670 5550 -1810 1310 280 -890 -670 280 130.00
3	W=[0 1]	0.4961 3.7086 0.9600 0.8384 3.7086 241.93 13.510 53.609 0.9600 13.510 3.3321 3.0517 0.8384 53.609 3.0517 11.881	0.3584 46.8542 1.3857 10.3546	0.1285 -0.0022 0.0005 0.0024 -0.0022 0.0001 0.0000 0.0000 0.0005 0.0000 0.0001 0.0001 0.0024 0.0000 0.0001 0.0001

Table 3-49: Optimized Parameters of NGAC for System (3.7) (Small X_d^0 Range)

Table 3-50 tabulates the randomly sampled performance of the nonlinear GAC on the cart-and-pole system (3.7) using a small X_d^0 range. The first case yields worse minimum rate of convergence and maximum control effort for the nonlinear GAC than for the nonlinear LARC. The second case yields a much better minimum rate of convergence, but still possesses a higher maximum control effort. The last case has a slower rate of convergence than the nonlinear LARC, but possesses a tenfold decrease in maximum control effort.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	0.7616	5.4053	5.8597	179.86	52.375	34.966
2	W=[1 1]	2.3045	9.5942	8.2171	132.03	35.450	24.977
3	W=[0 1]	0.0017	8.3313	1.8583	5.7884	2.3860	1.4252

Table 3-50: Performance of NGAC on System (3.7) (Small X_d^0 Range)

Table 3-51 lists the randomly sampled performance of the nonlinear GAC on the linearized cart-and-pole system (3.8) using a small X_d^0 range. The first and second cases have a faster rate of convergence than the linear LARC on the linearized system but use more control effort. The last case has a slower minimum rate of convergence but uses less control effort than the linear LARC on the linear system.

Weights	$\gamma_{\min}$	μ_{γ}	σ_{γ}	$\lvert u \rvert_{\max}$	$\mu_{\lvert u \rvert}$	$\sigma_{\lvert u \rvert}$
W=[1 0]	0.6930	5.5483	6.9042	159.35	49.888	32.019
W=[1 1]	1.9508	9.6642	8.3087	122.07	34.542	23.704
W=[0 1]	0.0028	8.3309	1.8672	5.724	2.3808	1.4171

Table 3-51: Performance of NGAC on Linearized System (3.8) (Small X_d^0 Range)

Figure 3-100 through Figure 3-105 contain the system signal responses to the initial condition (0.5,0,0,0) for the nonlinear GAC on the cart-and-pole system (3.7) using a small X_d^0 range. The first controller yields a faster response than the nonlinear LARC, but the control effort is higher. The second controller yields a much faster response but the control effort is much higher. The third controller yields a much slower response but the control effort is also very small. A weight set that yields a controller with a faster rate of convergence and lower control effort than the nonlinear LARC may be somewhere in between the weight sets *W*=[1 1] and *W*=[0 1], but further investigation must be done to know for sure.

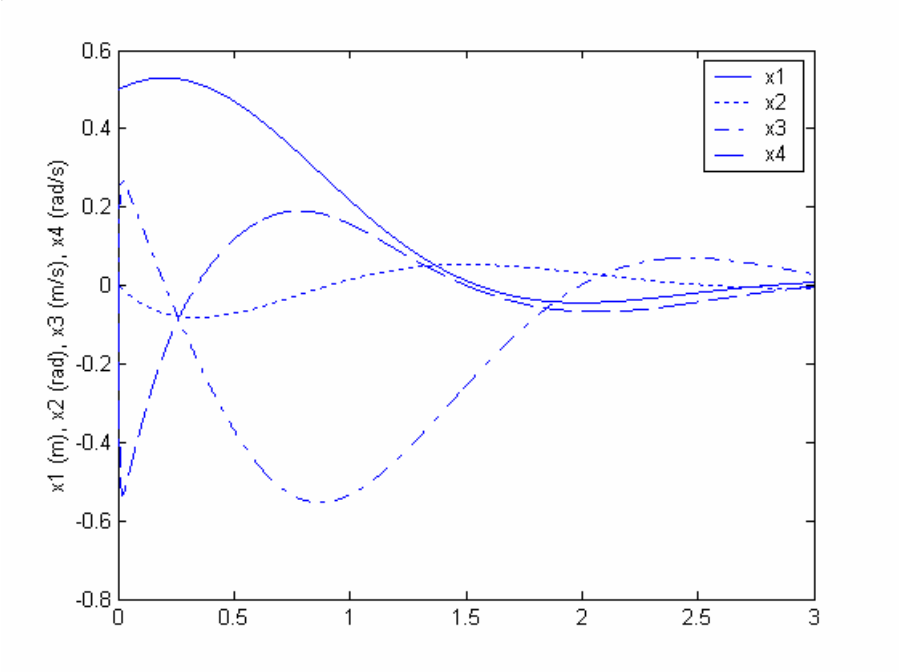

Figure 3-100: States of System (3.7) Using NGAC (W=[1 0], Small X_d^0 Range)

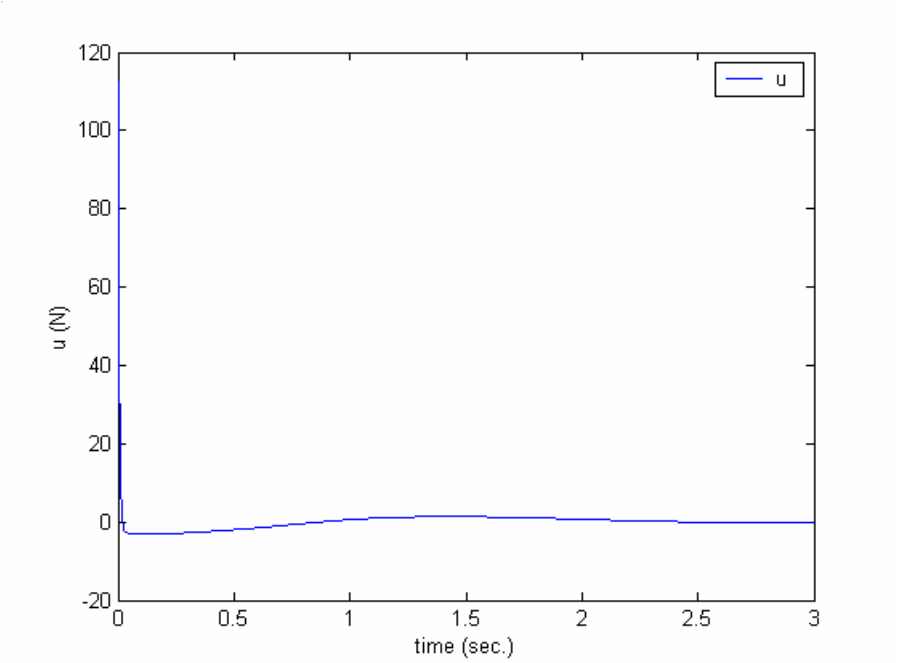

Figure 3-101: Control Signal of System (3.7) Using NGAC (W=[1 0], Small X_d^0 Range)

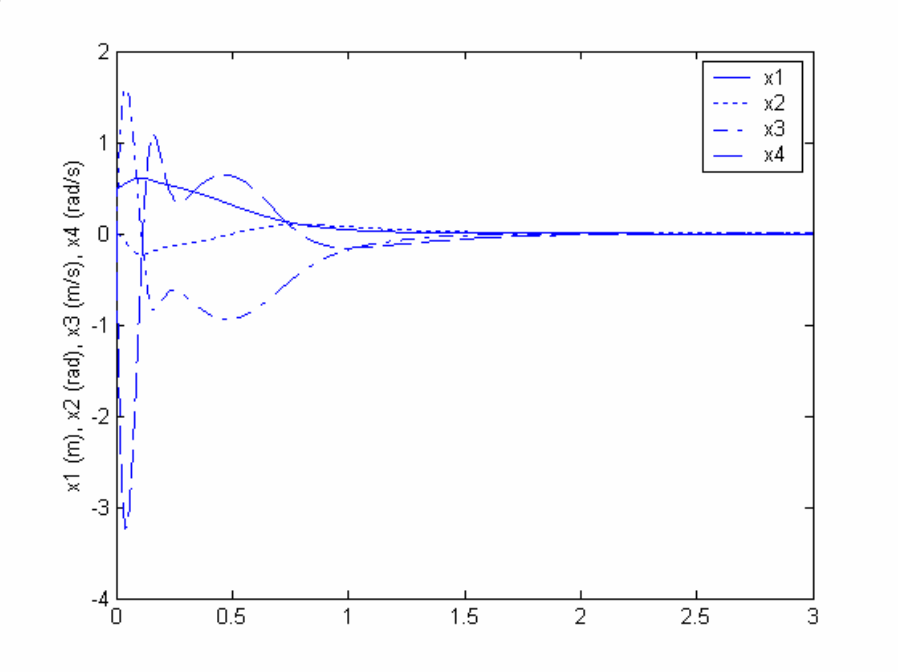

Figure 3-102: States of System (3.7) Using NGAC (W=[1 1], Small X_d^0 Range)

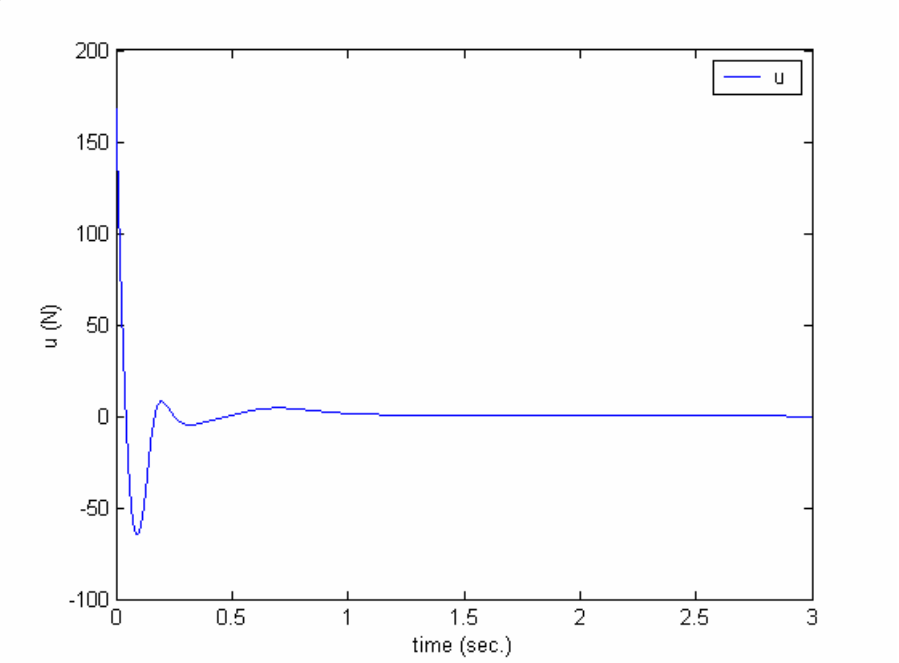

Figure 3-103: Control Signal of System (3.7) Using NGAC (W=[1 1], Small X_d^0 Range)

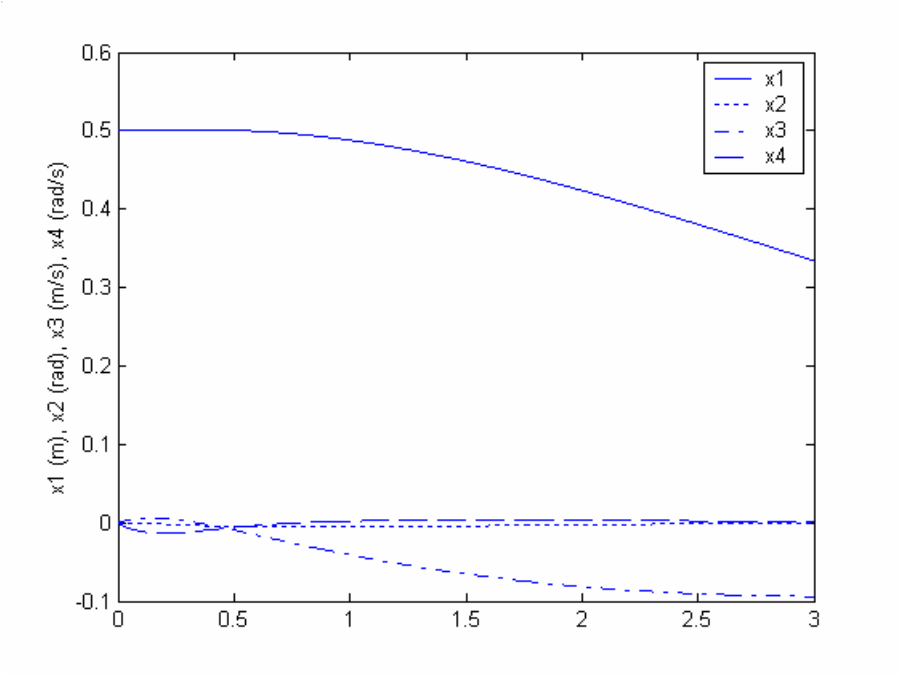

Figure 3-104: States of System (3.7) Using NGAC (W=[0 1], Small X_d^0 Range)

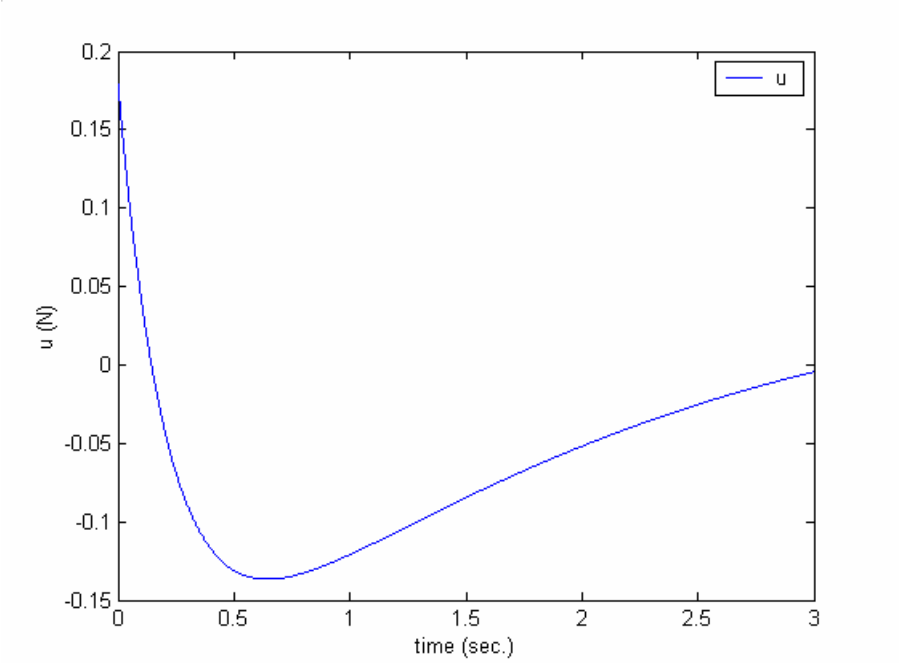

Figure 3-105: Control Signal of System (3.7) Using NGAC (W=[0 1], Small X_d^0 Range)

3.4.5 Nonlinear GAC with Large Checking Set Range

Table 3-52 shows the GA parameters used to optimize the nonlinear GAC for the cart-and-pole system (3.7) using a large X_d^0 range.

System: Cart-and-Pole (3.7)	Controller Type: nonlinear	Generations: 50	Population Size: 50	Checking points: 200
Critical Points: 50	Δ_E: 100	$K_{\max}$: 1000	c: 0	X^0 Range: $x_i \in [-1 \quad 1]$ $i = 1,2,3,4$

Table 3-52: GA Parameters to Optimize NGAC for System (3.7) (Large X_d^0 Range)

Table 3-53 tabulates the GA results of the optimization of the nonlinear GAC for the cart-and-pole system using a large X_d^0 range. The gains for the large range tend to be smaller than those for the small range.

Weight Case	Weights	$Px10^{-4}$	K	$Qx10^{-4}$
1	W=[1 0]	2.0718 1.5909 0.6450 0.3434 1.5909 2.0248 0.6508 0.3643 0.6450 0.6508 0.3453 0.1861 0.3434 0.3643 0.1861 0.1033	209.28 388.82 134.66 102.60	4.3030 1.2062 0.6749 0.5097 1.2062 0.6109 -0.0143 -0.0581 0.6749 -0.0143 0.4241 0.3138 0.5097 -0.0581 0.3138 0.2625
2	W=[1 1]	1.3437 1.6011 0.6749 0.3475 1.6011 6.0468 2.0370 1.0880 0.6749 2.0370 1.2199 0.6351 0.3475 1.0880 0.6351 0.3395	100.24 695.46 251.80 219.28	1.0048 0.1478 1.1804 0.5970 0.1478 5.5554 3.4311 2.5243 1.1804 3.4311 4.9908 3.1371 0.5970 2.5243 3.1371 2.6324
3	W=[0 1]	$10^{-4}x$ 0.0445 0.7230 0.1765 0.1632 0.7230 210.38 5.5668 46.482 0.1765 5.5668 1.3263 1.2574 0.1632 46.482 1.2574 10.271	0.0750 43.698 0.5943 9.6417	$10^{-4}x$ 0.0056 0.0007 0.0000 -0.0002 0.0007 0.0004 -0.0001 0.0001 0.0000 -0.0001 0.0001 -0.0000 -0.0002 0.0001 -0.0000 0.0000

Table 3-53: Optimized Parameters of NGAC for System (3.7) (Large X_d^0 Range)

Table 3-54 displays the randomly sampled performance of the nonlinear GAC on the cart-and-pole system (3.7) using a large X_d^0 range. For the first two cases, the minimum rate of convergence is smaller than that of the nonlinear LARC with less maximum control effort. For the last case, the minimum rate of convergence is much

slower than the nonlinear LARC but the maximum control effort is 5 orders of magnitude smaller than the nonlinear LARC.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	0.0895	7.0395	3.9501	$1.4010x10^{6}$	699.22	$1.3568x10^{4}$
2	W=[1 1]	0.0837	12.738	10.508	$1.7821x10^{6}$	723.97	$1.3329x10^{4}$
3	W=[0 1]	$3.3456x10^{-4}$	7.9549	1.4253	91.308	27.959	20.1231

Table 3-54: Performance of NGAC on System (3.7) (Large X_d^0 Range)

Table 3-55 lists the randomly sampled performance of the nonlinear GAC on the linearized cart-and-pole system (3.8) using a large X_d^0 range. The local behavior of the controller is much different than the behavior on X_d^0.

Weight Case	Weights	$\gamma_{\min}$	μ_γ	σ_γ	$\lvert u\rvert_{\max}$	$\mu_{\lvert u\rvert}$	$\sigma_{\lvert u\rvert}$
1	W=[1 0]	1.4295	6.2835	1.9826	765.33	225.85	151.81
2	W=[1 1]	0.7794	7.8585	3.8187	$1.1782x10^{3}$	375.43	243.90
3	W=[0 1]	$2.4382x10^{-4}$	8.5672	1.5073	53.385	22.117	13.185

Table 3-55: Performance of NGAC on Linearized System (3.8) (Large X_d^0 Range)

Figures 3.104 through 3.111 contain the system signal responses to the initial condition (0.5,0,0,0) for the nonlinear GAC on the cart-and-pole system (3.7) using a large X_d^0 range. The trends of the responses with the settings of the weights are similar to the previous small X_d^0 range case, however the set of weights that yield the target controller with a better response than the nonlinear LARC probably lies somewhere in between *W*=[1 0] and *W*=[1 1].

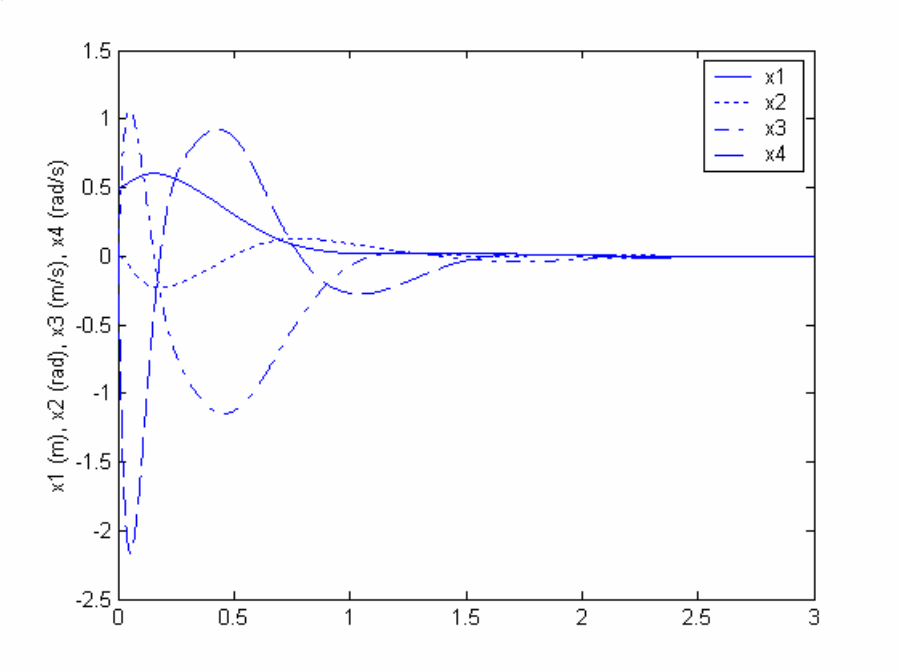

Figure 3-106: States of System (3.7) Using NGAC (W=[1 0], Large X_d^0 Range)

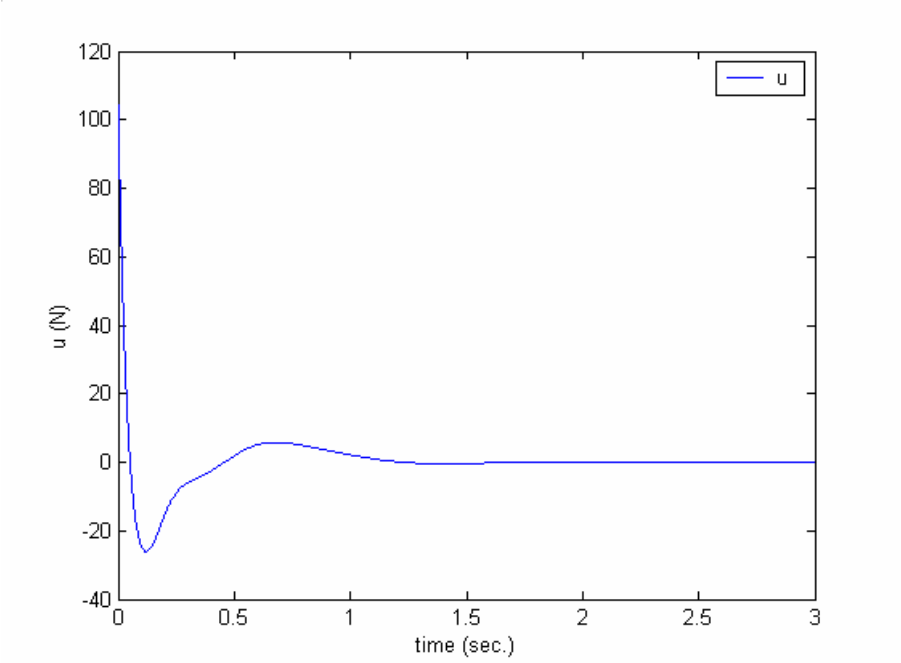

Figure 3-107: Control Signal of System (3.7) Using NGAC (W=[1 0], Large X_d^0 Range)

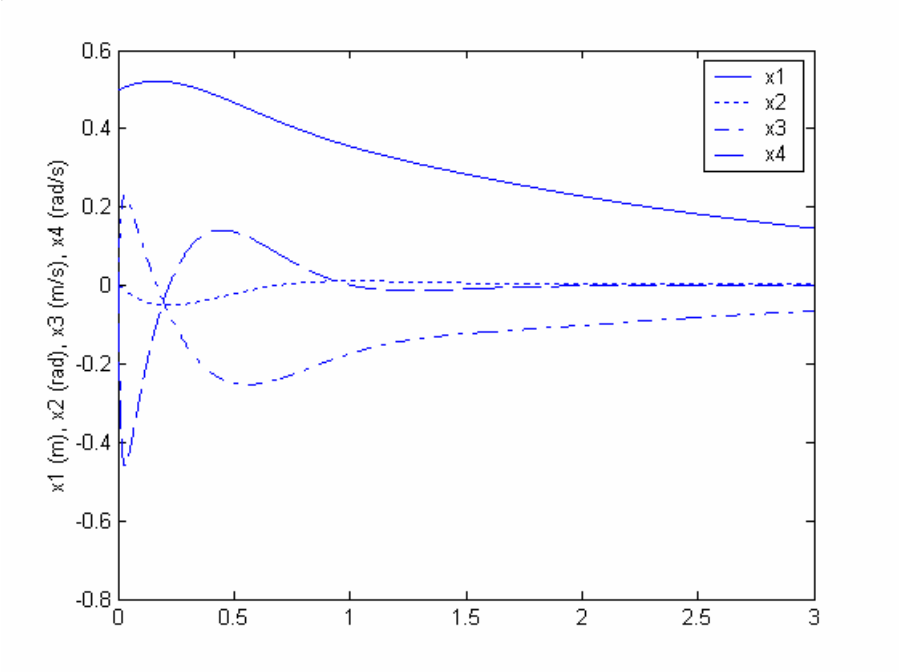

Figure 3-108: States of System (3.7) Using NGAC (W=[1 1], Large X_d^0 Range)

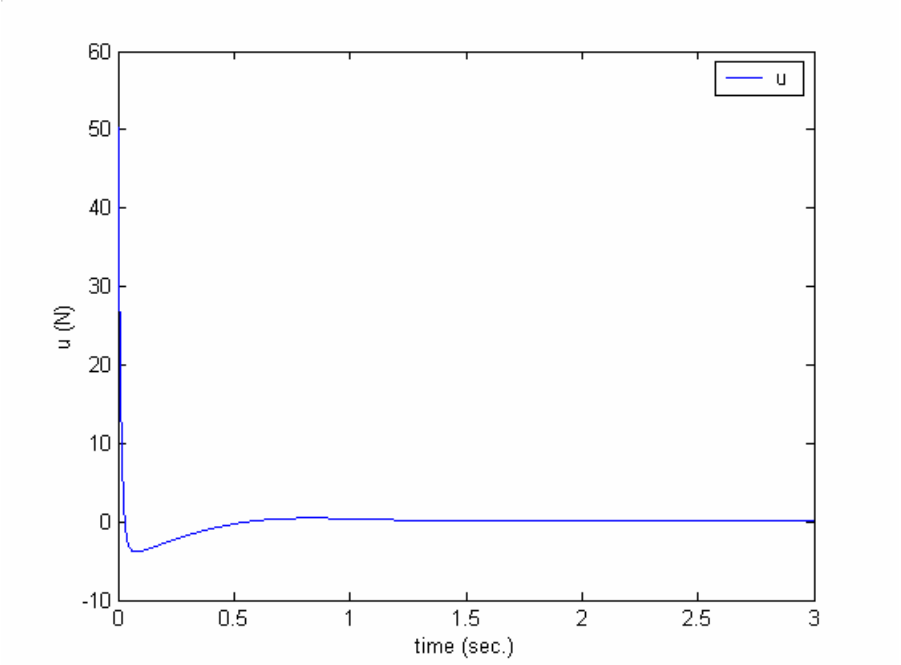

Figure 3-109: Control Signal of System (3.7) Using NGAC (W=[1 1], Large X_d^0 Range)

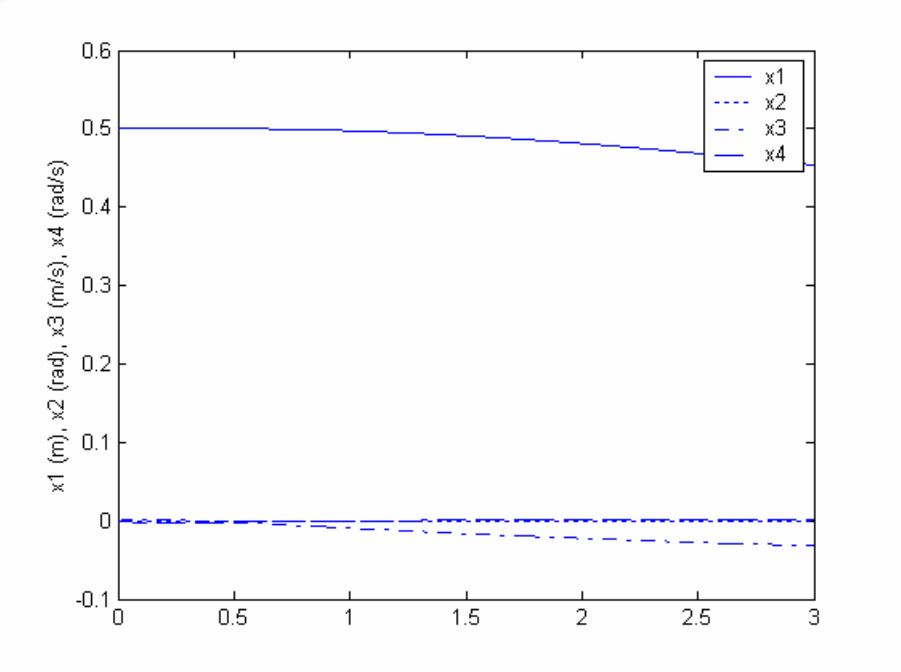

Figure 3-110: States of System (3.7) Using NGAC (W=[0 1], Large X_d^0 Range)

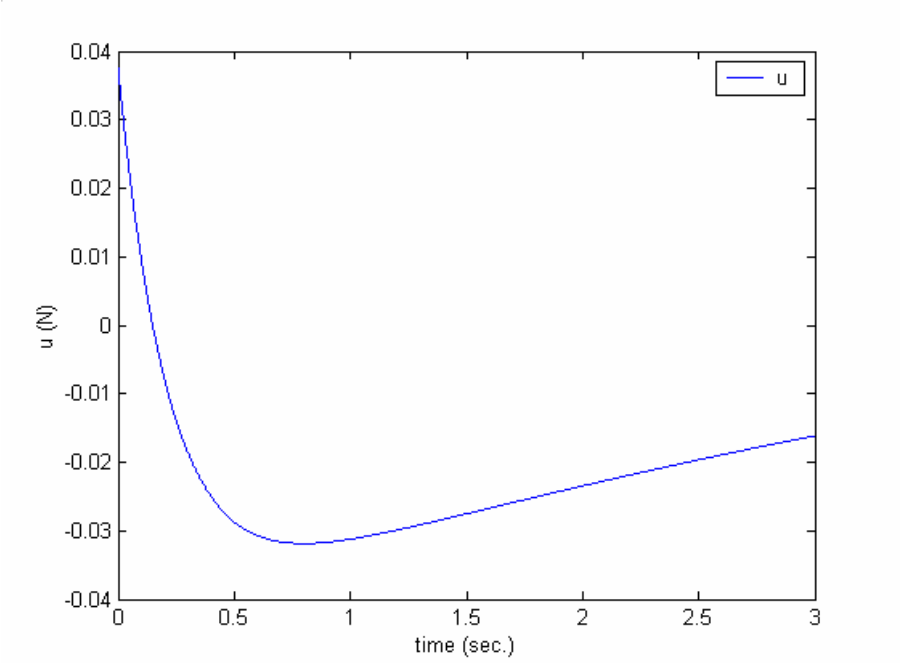

Figure 3-111: Control Signal of System (3.7) Using NGAC (W=[0 1], Large X_d^0 Range)

3.5 Discussion of Results

We list the following observations and discuss them below:

1. The fitness function is nonlinearly dependent on *W* and more importantly is non-monotic in *W*.
2. The linear GAC can achieve a higher $\gamma_{\min}$ on X_d^0 than the linear LARC, but $\gamma_{\min}$ is still negative in some cases.
3. Although at the cost of a large $|u|_{\max}$, both the nonlinear GAC and LARC yield $\gamma_{\min} > 0$ globally, but the nonlinear GAC yields a smaller $|u|_{\max}$.
4. GAC yields a non-intuitive *Q* matrix with non-zero diagonal terms that are sometimes negative.
5. Most of the time the GAC has a smaller local $\gamma_{\min}$ but yields a high overall $\gamma_{\min}$ on X_d^0.

Point 1:

One should expect that focusing solely on the rate of convergence should result in the controllers with a close to optimal value for $\gamma_{\min}$ on X_d^0. Likewise, we should expect that using only the maximum control effort in the fitness function should yield an optimal value for $|u|_{\max}$ on X_d^0. Therefore, when considering both $\gamma_{\min}$ and $|u|_{\max}$ in the fitness function, a trade-off in the optimal values should be made. This is not always the case, however. The search for both $\gamma_{\min}$ and $|u|_{\max}$ leads the GA search to regions of parameter space that would otherwise not be searched, hence better values of both $\gamma_{\min}$ and $|u|_{\max}$ are achieved. In other words, looking for a controller with a fast rate of convergence sometimes produces a controller that also yields a small control effort because it restricts

the search to regions of parameter space that yield good values for both sets of performances.

Point 2:

The GAC considers the actual nonlinearities rather than conservative estimates in the form of uncertainties. This yields a less conservative controller with a better performance on X_d^0. A negative rate of convergence is expected in situations where the system is not linearly stabilizable or a quadratic CLF does not exist.

Point 3:

The Sontag-like control law is both globally asymptotically stable and inverse optimal by design. It does however, yield control laws with high $|u|_{\max}$ because of the restriction that $\dot{V} = \sqrt{\left(\frac{\partial V}{\partial x} f\right)^2 + q(x)\left(\frac{\partial V}{\partial x} g\right)^2}$. The high control effort occurs when $\frac{\partial V}{\partial x} g << \frac{\partial V}{\partial x} f$. The GAC is able to find a CLF such that this inequality condition is less extreme.

Point 4:

An important feature to notice is that the GA found non-diagonal *Q* matrices to be the optimal solution for all cases. Selecting the relative weighting between the diagonal elements of *Q* is fairly straight-forward when observing the response of the system. RMS values of the states and control go down as their weighting factor goes up. However, there is no straightforward intuition for selecting the off-diagonal terms which highlights one of the advantages of using the GAC method.

Point 5:

The GAC uses the linear model to compute a CLF, but uses the nonlinear model to evaluate the CLF. Hence, the locally linear behavior is not as important as the entire behavior on X_d^0.

In summary, the GAC is advantageous because treating nonlinear systems as linear systems with uncertainties (structure and unstructured) as done in Ngamsom (2001) will yield a conservative controller, since the nonlinearity is rarely at its worst-case value all the time. However, a disadvantage of the GAC is that it only approximates the effects of the nonlinearities on the CLF because we don't know exactly where the critical points are located. Therefore the controller may not be conservative enough. The GAC also requires the tuning of the fitness function to achieve the desired performance and is not guaranteed to find a solution nor does it determine if one exists. Future work should involve better estimates of the critical point locations as well as an investigation of the required number of checking points and tuning of the fitness function weights to achieve the desire performance characteristics.

4 Future Directions and Conclusions

Numerical checking of CLF conditions is useful for control system design because it relaxes the need to analytically solve for the control. Instead, a desired performance level on a subset of state space may be achieved via automatic offline tuning of the CLF and controller. In practice, probabilistic performance checking in a desired region of state space may suffice. PID control is still the dominant controller type in industry even though it is usually not globally stable nor guaranteed in any theoretical sense for the actual nonlinear and uncertain dynamics. The transient and steady-state response of the controller to specified initial conditions and set points is the ultimate concern. The work proposed in this thesis is the beginning of the development of an alternative method for nonlinear system controller design. Four major directions are proposed below for future research in extending this work.

4.1 Generalized Control Lyapunov Functions

The modified Sontag universal control law is equivalent to using the gradient direction of V normalized to satisfy the HJB equation (Primbs, et al 2000). The control law forces the vector $\dot{x}$ to have a positive projection along the negative gradient direction of the CLF. Therefore, because the level sets of the CLF are closed, the trajectory must settle towards the origin, i.e. the state vector cannot escape the enclosure and continuously enters smaller and smaller enclosures. However, staying within elliptically shaped enclosures for a quadratic CLF may require a large amount of control effort,

because such trajectories may approach singularities (i.e. the "$\frac{\partial V}{\partial x} g$" term in the control law approaches zero). Using a CLF with level sets whose shape is close to the unforced stated trajectory is the key to lowering the control effort when using CLF-based control laws. Johansen (2000 a) showed that a Lyapunov function candidate of the form

$$V(x) = \frac{1}{2} x^T P(x) x \tag{4.1}$$

could approximate any Lyapunov function to arbitrary degree of accuracy, where the matrix valued function $P(x)$: $X^0 \rightarrow \Re^{nxn}$ is defined by the following linear parameterization:

$$P(x) = \sum_{i=1}^{N} P_i p_i(x) \tag{4.2}$$

where $p_i : X^0 \rightarrow \Re$ are smooth basis-functions (typically normalized Gaussians) and P_i are parameter matrices for all $i = 1,2,\ldots,N$. To ensure the CLF has closed level sets, the parameters are restricted such that $P_i > 0$ and $\sum_{i=1}^{N} p_i(x) = 1$. Although Johansen applies the use of such Lyapunov function candidates to quantify nonlinear system performance, the work herein did not explicitly address the use of such generalized functions as CLFs.

The Lyapunov function described by (4.1) and (4.2) is easily blended into the GAC framework proposed in this thesis. The parameter matrices P_i, the number of basis functions N, and any parameters related to $p_i(x)$ (e.g. mean and variance of Gaussian basis functions) could be included in the set of parameters to be tuned by the GA. The level sets of (4.1) may take on arbitrary closed shapes by restricting $P_i > 0$, such that the

state trajectories could be shaped to yield high convergence rates with small control effort using CLF-based controllers such as the modified Sontag control law.

4.2 Improved the Critical Point Computation

Theorem 1.1 implies that the major cause of inefficiency in point-wise numerical optimization of CLFs is the required checking set density. Theorem 1.2 helps relax the required number of checking points by allowing the use of the linearized dynamics to obtain a local guarantee of positive rate of convergence (e.g. by checking the ratio of the eigenvalues of A^TP+PA and P). This way, a small region about the origin is removed from the need to check the rate of convergence. However, for a wide spanning X^0, local stability is not a good assumption for the entire set being stable or having some positive minimum rate of convergence. Theorem 1.2 could be used at many points on X^0, and many locally linearized system models could be used to check the local minimum rate of convergence. Although the required total number of checking points would be reduced substantially, the points in the regions between the linearizations must also be checked (a problem similar to unstable switching in gain scheduling).

A few approaches to improving the search for the critical points are listed below:

1. Spend time converging towards the critical points rather than taking a single guess:

   The work herein assumed that $\gamma_{\min}$ and $|u|_{\max}$ occur at only one place each on X^0. These locations must be estimated and the rate of convergence and control effort need to be computed only at these points. Because the CLF changes with the adjustment of its parameters, the critical point locations change, making the required checking set, X_d^0, dynamic. The method proposed in Chapter 2 for adapting X_d^0 is basically a random walk using memory of the critical point

estimations of the controllers of past generations. A problem with such a search method is that the algorithm spends little time looking for the critical points; it merely uses a new randomly generated set of guesses along with the best guesses from the previous generations. Genetic algorithms could do a better job at finding the critical points. A method could be used similar to the method of Jamshidi et al, (2003), where linear optimal control, specifically, mixed H_2 and H_∞ optimization, is performed using a GA to tune the controllers. In the method of Jamshidi et al., an additional GA is used to find the worst case uncertainities and disturbances, along with finding the frequencies at which the H_∞ norm occurs. Such an approach would be better than using a single set of guess points because the GA would spend time converging towards the critical points so that the narrow escape manifolds seen in Chapter 3 are found during the controller optimization and not during the controller verification simulation.

2. Rather than use a GA, try a specially designed particle filter:

Another interesting avenue of research is the use of particle filters (Arulampalam et al, 2002) for searching for the critical point locations. Particle filters approximate the *pdf* of a variable by a discrete set of weighted points (particles). In this case, the variable would be the location of a critical point. Using a particle filter in an optimization application would be similar in operation to a GA that uses only *Copy* and *Mutation*. However, the difference would be that particle filters have a stronger theoretical basis in Bayesian statistics; therefore they may offer more resources for a analyzing the convergence of the algorithm towards the critical points. In addition, because particle filters are generalizations of Kalman

filters, the dynamics of the system may be used to guide the search towards the points in state space where the rate of convergence becomes negative (instability) for the CLF (i.e. the critical points). For example, the direction of the state time derivative at a particular location in state space could be used by the particle filter as a guess towards a better estimate of the critical point.

3. Create an empirical mapping of the relationship between critical points, and controller and CLF parameters.

With the current critical point searching method, no memorization occurs of the locations of the critical points for the given set of controller and CLF parameters. Therefore when a similar set of parameters is introduced to the GA population, the search effort for the critical points is duplicated (although sometimes reduced by the use of the "random walk with memory" approach introduced in Chapter 2). Neural networks with their excellent generalized mapping ability, could be used to learn the mapping between the controller and CLF parameters and the locations of the critical points. A clustering neural network, such as ARTMAP (Carpenter et al. 1991) could be trained to provide close initial guesses of locations of the estimated critical points for any set of controller and CLF parameters, rather than searched for blindly at each generation. The 2nd GA procedure proposed in Item 1 above would use the output of the neural network as the initial starting point to fine tune the critical point estimation, and then the neural network would be retuned with the better estimation point.

4.3 Output-feedback, Adaptive, and Robust Control

The work herein exclusively considered full state-feedback control. Output-feedback, adaptive and robust control all fit into the CLF framework (Kokotovic & Arcak, 2001). Output feedback requires the addition of observer states and the tuning of observer gains. The GAC method could be used to achieve similar control objectives to those in this thesis as well as to hinder the well known "peaking phenomenon" (estimation error growing very high during transients) by appropriate tuning of the CLF and observer gains. For adaptive control, the adapted controller parameters may be thought of as additional states of the system, and the GAC method could be used to tune the adaptation gains to carry out the performance objectives. Robust control, as with the critical point searching concept, involves finding a "worst-case" set of admissible structured and unstructured uncertainties and disturbances, while the controller gains are tuned to carry out the performance objective despite the effects of the uncertainties and disturbances. Genetic algorithms are proven solutions to such *dynamic game* problems (Jamshidi et al, 2003).

4.4 Discrete-Time Control

The control systems literature is dominated by continuous time analysis. Likewise, CLF theory for continuous time control is generally simpler than discrete time control. Often a continuous time control signal can be separated from the remaining terms in $\dot{V}$ and solved to satisfy the CLF condition $\dot{V} < -\alpha V, \alpha > 0$. For example, in control affine systems, the Sontag-like control laws exploit the fact that the control is multiplied by $\frac{\partial V}{\partial x} g(x)$ in $\dot{V}$ and division by $\frac{\partial V}{\partial x} g(x)$ of both sides of the $\dot{V}$ equation separates out the control. In addition to the difficulty of solving for the control analytically, tuning the

continuous time controller does not guarantee that the discrete time implementation will satisfy the performance requirements. It is desirable to use a more direct design method in the discrete time domain. The proposed GAC method is easily modified to ensure the discrete-time CLF condition $V_{k+1} < \alpha V_k, |\alpha| < 1$ (k is the time step index) is satisfied in place of the continuous time CLF condition $\dot{V} < -\alpha V, \alpha > 0$. Another attraction of tuning the discrete time CLF and controller is that a discrete time model based on system identification techniques, including system time delays, may be used to check and optimize the performance measures. In addition, given that the actual system can sustain the use of "bad" controllers during the GA search phase, the GA could bypass the system model altogether and tune the system directly using online system identification. Such a notion has been coined "genetic adaptive control" by Spooner et al. (2002).

4.5 Putting It All Together For Practical Control Design

System analysis is used to determine an appropriate controller and CLF structure. However, using a genetic algorithm to tune controllers and CLFs does not call for rigorous analysis of the system. The algorithm simply finds a set of parameters of the specified controller that meets all the desired requirements. In a sense, the GA makes some very complex controllers "practical" because it tunes them to work even if the system does not fall into the particular class of systems assumed during the controller synthesis procedure. Such "inappropriate" mixing of controllers and systems happens in practice more often than not. A common example is the tuning of PID controllers, a linear control method, for highly nonlinear industrial robot arms (Rocco, 1996). Another such mismatch between controller and system type is seen in a typical system identification procedure where a linear model is "fit" to input/output response to a known

input with the data taken most likely from a nonlinear system. The “best fit” model is then often used to design a robust linear controller.

With such a large number of research publications on adaptive, robust, and digital control, it is clear that a design procedure that implements all three concepts for a broad class of systems is very useful. However, such a procedure is difficult to find because of the level of difficulty involved in the analysis of discrete time systems. While a large host of control design philosophies exist in the literature, many are not used in practice because of their complexity and a general lack of understanding of their theoretical basis by typical control engineers with terminal B.S. degrees. For example, many control engineers do not recognize when a system is in a particular canonical feedback form (e.g. strict feedback form), so an appropriate back-stepping control design is not considered. On the other hand, many times the use of a practical control law may not be appropriate for a given system, but since it is a method well known to the particular control engineer, it is implemented. Such is the legacy of PID control. In this light, many times there is a reason to simply “guess” a suitable controller type, even if the actual system does not perfectly fit into the theoretical framework upon which the controller is based.

The genetic algorithm optimization procedure in this thesis is a way to may be a way to address these problems simultaneously, using discrete time CLFs subject to a specified set of states, uncertainties, and disturbances. A particular control type may be selected or guessed, and the algorithm tunes the parameters to satisfy a specified level of performance. This leads us to outline a general approach to controller design for a broad class of systems. The background and primary interest area of the author is the control of mechanical systems; therefore such systems shall be the primary focus. Many mechanical

systems fall into the control affine category, so restricting the analysis to such form still yields broad applicability. Let the system of interest have the following form, where the meaning of the terms are explained in the paragraph following (4.3) and (4.4):

$$\begin{aligned} \dot{x} &= f(x,\theta,\delta) + g(x,\theta,\delta)u(y,\hat{x},\hat{\theta},\hat{\delta}) \\ y &= h(x,\theta,\delta) \\ \dot{\theta} &= F_{\theta}(\theta,t) \\ \dot{\delta} &= F_{\delta}(\delta,t) \end{aligned} \tag{4.3}$$

where

$$\begin{aligned} &f, g : \Re^{n_x} \times \Re^{n_\theta} \times \Re^{n_\delta} \to \Re^{n_x}, \\ &u : \Re^{n_x} \times \Re^{n_\theta} \times \Re^{n_\delta} \times \Re^{n_y} \to \Re^{n_u}, \\ &h : \Re^{n_x} \times \Re^{n_\theta} \times \Re^{n_\delta} \to \Re^{n_y}, \\ &x \in X \subset \Re^{n_x}, \theta \in \Theta \subset \Re^{n_\theta}, \delta \in \Delta \subset \Re^{n_\delta}, y \in \Re^{n_y} \end{aligned}$$

Let the observer of the system of interest have a similar form

$$\begin{aligned} \dot{\hat{x}} &= \hat{f}(\hat{x},\hat{\theta},\hat{\delta}) + \hat{g}(\hat{x},\hat{\theta},\hat{\delta})u(y,\hat{x},\hat{\theta},\hat{\delta}) + L(e) \\ \hat{y} &= \hat{h}(\hat{x},\hat{\theta},\hat{\delta}) \\ \dot{\hat{\theta}} &= F_{\hat{\theta}}(\hat{x},\hat{\theta},\hat{\delta},e,t) \\ \dot{\hat{\delta}} &= F_{\hat{\delta}}(\hat{x},\hat{\theta},\hat{\delta},e,t) \\ e &= y - \hat{y} \end{aligned} \tag{4.4}$$

where

$$\begin{aligned} &\hat{f}, \hat{g} : \Re^{n_{\hat{x}}} \times \Re^{n_{\hat{\theta}}} \times \Re^{n_{\hat{\delta}}} \to \Re^{n_{\hat{x}}}, \\ &L : \Re^{n_y} \to \Re^{n_{\hat{x}}}, \\ &\hat{h} : \Re^{n_{\hat{x}}} \times \Re^{n_{\hat{\theta}}} \times \Re^{n_{\hat{\delta}}} \to \Re^{n_y}, \\ &\hat{x} \in \hat{X} \subset \Re^{n_{\hat{x}}}, \hat{\theta} \in \hat{\Theta} \subset \Re^{n_{\hat{\theta}}}, \hat{\delta} \in \hat{\Delta} \subset \Re^{n_{\hat{\delta}}}, \hat{y} \in \Re^{n_y} \end{aligned}$$

with the signals of interest being defined exclusively on the set $\Psi = X \times \Theta \times \Delta \times \hat{X} \times \hat{\Theta} \times \hat{\Delta}$.

The system dynamics are dependent on the states x, parameters θ, and disturbances δ, with all three signals existing on closed bounded real sets X, Θ, and Δ, respectively. These sets along with the sets where the estimated signals $\hat{x}$, $\hat{\theta}$, and $\hat{\delta}$

exist, $\hat{X}$, $\hat{\Theta}$, and $\hat{\Delta}$, respectively, are specified by the controller designer. These signals also exist on closed bounded real sets $\hat{X}$, $\hat{\Theta}$, and $\hat{\Delta}$, respectively, and may be the same as X, Θ, and Δ, when the signal estimates, $\hat{x}$, $\hat{\theta}$, and $\hat{\delta}$, are estimates of the real signals, x, θ, and δ. The control u, is assumed to be dependent on the system's measurable output(s) y, and estimates of the states $\hat{x}$, parameters $\hat{\theta}$, and disturbances $\hat{\delta}$. As with the state derivatives, the output(s) *y*, is dependent on x, θ, and δ. The system parameters and disturbances are assumed to be time dependent and autoregressive. Note that the set of observer equations and parameters are not necessarily the same type and order as the system equations and parameters. That is $\hat{f}, \hat{g}, \hat{h}, F_{\hat{\theta}}, \text{ and } F_{\hat{\delta}}$ may not have the same form as $f, g, h, F_{\theta}, \text{ and } F_{\delta}$. For example, the system may be a 4th order double inverted pendulum with a sinusoidal torque disturbance, while the observer is a 6th order linear parameter time varying system, or a 6th order nonlinear system with neural network functions of the states with many network weights. This selection is made to preserve the control system generality. The structure of the observer is similar to the actual system in that it is also control affine. However, an additional term is added that acts as the observer correction function, *L(e)* in (4.4). In a Luenberger-type observer, this function has the linear form "$L \cdot e$". The estimates of the system parameters and disturbances are assumed to be time dependent and autoregressive, but also dependent on the estimates of each other, the states, and the tracking error.

Ultimately the control law will likely be digitally implemented such that discretization of the controller must be performed. Because of the inclusion of unstructured uncertainty terms in the proposed framework, discretization errors may be

attenuated by proper tuning of the control law. To maintain the theme of controller practicality, very simple discretization methods may be used. Euler's approximation is the simplest approximation to use for the time derivatives of the continuous time control law and CLF, such that it is a prime candidate for our purposes. The discretized CLF is checked directly to ensure it monotonically decreases with time, decreasing and any quantization error effects are assumed to act as disturbances. Since the critical point searching algorithm will consider the full range of admissible disturbance values, the quantization effects will be accounted for in the controller and CLF tuning. Another feature of the proposed method is the elimination of the need to compute $\dot{V}$, which can be very laborious and computational intensive when using many basis functions in (4.2). Computing $\dot{V}$ in (4.18) can be very lengthy due to the lengthy $\dot{p}_j$ terms:

$$\dot{V} = \dot{\tilde{\psi}}\left(\sum_{j=1}^{N} p_j(\tilde{\psi})P_j\right)\tilde{\psi}^T + \frac{1}{2}\tilde{\psi}\left(\sum_{j=1}^{N} \dot{p}_j(\tilde{\psi})P_j\right)\tilde{\psi}^T \tag{4.18}$$

where ψ is the vector of all signals of interest (e.g. x, θ, and δ) and $\tilde{\psi}$ is the error between these signals and their estimates, $\tilde{\psi} = \psi - \hat{\psi}$. Rather than computing $\dot{V}$, the performance checking may instead include checking the change in value of *V* between current time step and the next time step. To clarify, let "$[\bullet]_k$" denote a signal at time step *k*. Then *V* for the next time sample is expressed as

$$[V]_{k+1} = \left[\frac{1}{2}\tilde{\psi}\left(\sum_{j=1}^{N} p_j(\tilde{\psi})P_j\right)\tilde{\psi}^T\right]_{k+1} \tag{4.20}$$

where

$$[\tilde{\psi}]_{k+1} = \left[\tilde{\psi} + \Delta t \cdot \dot{\tilde{\psi}}\right]_k \tag{4.21}$$

Under this regime of approximation, the time stepping of all time dependent signals involved in the terms of (4.3) and (4.4) is approximated by the form (4.21). The rate of convergence for the discrete time case becomes $[V]_{k+1}/[V]_k$ rather than $\left[V + \Delta t \cdot \dot{V}\right]_k /[V]_k$, with the latter expression being much more complicated because $\dot{V}$ must be computed. In fact, the former quantity is a more accurate estimate of the rate of convergence, because $\dot{V}$ does not dictate the performance of the system, only $\dot{\psi}$.

The idea behind the proposed routine is for the optimization method to adjust the parameters of the controller and CLF to maximize the performance objective, while at the same time search for a combination of $x, \hat{x}, \theta, \hat{\theta}, \delta, and\ \hat{\delta}$ (i.e. the critical points) on Ψ that minimizes the performance objective for a given set of controller and CLF parameters. The controller and CLF parameters are then evaluated using the estimated critical points. Finally, the controller and CLF parameters are updated. The process is repeated until a desired performance level is achieved or progress has stagnated.

To demonstrate the generalized approach outlined above, consider a particular nonlinear system approximated as a linear parameter time varying system and use a robust adaptive controller for linear systems with a very fast parameter adaptation rate to compensate for the parameter changes due to the nonlinearities. The adaptive controller may be synthesized in the continuous time domain, but it must be discretized as above by approximating the time stepping of all signals. The uncertainty caused by the discretization is assumed to be attenutable by proper tuning of the control law parameters by the GA. We may address the time variability of the linear system coefficients by using Krstic et al.'s (1995, Chapter 10) adaptive back-stepping controller with tuning functions for linear systems, denoted ABC. We shall assume that along with the controller's

inherent robustness to parameter perturbations, the parameter adaptation rate may be tuned fast enough to account for the coefficient variability. The Laplace transform model of the linear system may be expressed as

$$y(s) = \frac{b_m s^m + \cdots + b_1 s + b_0}{s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0} u(s), \qquad m < n \tag{4.5}$$

For ease of analysis, the state space model is used and expressed in observer canonical form:

$$\begin{aligned}
\dot{x}_1 &= x_2 - a_{n-1} y \\
&\vdots \\
\dot{x}_{\rho-1} &= x_\rho - a_{m+1} y \\
\dot{x}_\rho &= x_{\rho+1} - a_m y + b_m u \\
&\vdots \\
\dot{x}_{n-1} &= x_n - a_1 y + b_1 u \\
\dot{x}_n &= -a_0 y + b_0 u \\
y &= x_1
\end{aligned} \tag{4.6}$$

Equation (4.6) may also be expressed in a more compact and convenient form:

$$\begin{aligned}
\dot{x} &= Ax - ya + \begin{bmatrix} 0_{(\rho-1)x1} \\ b \end{bmatrix} u \\
y &= e_1^T x
\end{aligned} \tag{4.7}$$

where

$A = \begin{bmatrix} 0_{(n-1)x1} & I_{n-1} \\ 0 & 0_{1x(n-1)} \end{bmatrix}$, $a = \begin{bmatrix} a_{n-1} \\ \vdots \\ a_0 \end{bmatrix}$, $b = \begin{bmatrix} b_m \\ \vdots \\ b_0 \end{bmatrix}$, and e_j is the unit vector whose jth element is equal to 1.

The set of *K-filters* below is used to reconstruct the state vector via the relationship $x = \xi + \Omega^T \theta$:

$$\begin{aligned}\dot{\eta} &= A_0\eta + e_n y \\ \dot{\lambda} &= A_0\lambda + e_n u\end{aligned} \tag{4.8}$$

where $A_0 = A - ke_1^T$ with *k* being the Luenberger observer gains.

$$\begin{aligned}\Xi &= -\left[A_0^{n-1}\eta, \quad \ldots, \quad A_0\eta \quad \eta\right] \\ \xi &= -A_0^n\eta \\ v_j &= A_0^j\lambda \\ \Omega^T &= \left[v_m, \quad \ldots, \quad v_1 \quad v_0 \quad \Xi\right]\end{aligned} \tag{4.9}$$

The rationale behind using such a filter structure lies in it's minimal order (Krstic et al., 1995).

Next is the structure of the control law. The variable *z* represents the tracking error and it's successive derivatives:

$$\begin{aligned}z_1 &= y - y_r \\ z_i &= v_{m,i} - \hat{\sigma}y_r^{(i-1)} - \alpha_{i-1}, \qquad i = 2,\ldots,\rho\end{aligned} \tag{4.10}$$

where $y_r^{(i-1)}$ is the $(i-1)^{th}$ derivative of the reference output trajectory and $\hat{\sigma}$ is the estimate of the inverse if the high frequency gain b_m, that is $1/b_m$ ($1/\hat{b}_m$ is not used because of the possibility of division by zero during controller operation). The variable α represents the well known *virtual control* concept of the back-stepping procedure and is defined in (4.11):

$$
\begin{aligned}
\alpha_1 &= \hat{\sigma}\overline{\alpha}_1 \\
\overline{\alpha}_1 &= -(c_1 + d_1)z_1 + \xi_2 - \overline{\omega}^T\hat{\theta} \\
\alpha_2 &= -\hat{b}_m z_1 - \left[c_2 + d_2\left(\frac{\partial \alpha_1}{\partial y}\right)^2\right]z_2 + \beta_2 + \frac{\partial \alpha_1}{\partial \hat{\theta}}\Gamma\tau_2 \\
\alpha_i &= -z_{i-1} - \left[c_i + d_i\left(\frac{\partial \alpha_{i-1}}{\partial y}\right)^2\right]z_i + \beta_i + \frac{\partial \alpha_{i-1}}{\partial \hat{\theta}}\Gamma\tau_i \\
&\quad - \sum_{j=2}^{i-1}\frac{\partial \alpha_{i-1}}{\partial \hat{\theta}}\Gamma\frac{\partial \alpha_{i-1}}{\partial y}z_j, \qquad j = 3,\ldots,\rho
\end{aligned}
\tag{4.11}
$$

where c_i and d_i are CLF parameters, Γ is the adaptation rate matrix, $\hat{\theta}$ is the estimate of the linear system coefficients from (4.5), $\hat{\theta} = \begin{bmatrix}\hat{b} & \hat{a}\end{bmatrix}^T$, and finally

$$
\begin{aligned}
\beta_i &= \frac{\partial \alpha_{i-1}}{\partial y}\left(\xi_2 + \omega^T\hat{\theta}\right) + \frac{\partial \alpha_{i-1}}{\partial \eta}\left(A_0\eta + e_n y\right) + \sum_{j=1}^{i-1}\frac{\partial \alpha_{i-1}}{\partial y_r^{(j-1)}}y_r^{(j)} + k_i v_{m,1} \\
&+ \sum_{j=1}^{m+i-1}\frac{\partial \alpha_{i-1}}{\partial \lambda_j}\left(-k_j\lambda_1 + \lambda_{j+1}\right) + \left(y_r^{(i-1)} + \frac{\partial \alpha_{i-1}}{\partial \hat{\sigma}}\right)\dot{\hat{\sigma}}
\end{aligned}
\tag{4.12}.
$$

One of the unique concepts of ABC is the use of the so-called tuning functions τ. Tuning functions are used to update the parameter estimates without causing bad transients. For the current controller design, the tuning functions are defined as

$$
\begin{aligned}
\tau_1 &= \left(\omega - \hat{\sigma}\left(\dot{y}_r + \overline{\alpha}_1\right)e_1\right)z_1 \\
\tau_i &= \tau_{i-1} - \frac{\partial \alpha_{i-1}}{\partial y}\omega z_i, \qquad i = 2,\ldots,\rho
\end{aligned}
\tag{4.13}
$$

Finally the control and parameter update laws are

$$
u = \alpha_\rho - v_{m,\rho+1} + \hat{\sigma}y_r^{(\rho)} \tag{4.14}
$$

$$
\begin{aligned}
\dot{\hat{\theta}} &= \Gamma\tau_\rho \\
\dot{\hat{\sigma}} &= -\gamma\,\mathrm{sgn}(b_m)\left(\dot{y}_r + \overline{\alpha}_1\right)z_1
\end{aligned}
\tag{4.15}
$$

Note that the sign of the high frequency gain, $\text{sgn}(b_m)$, is assumed to be known for the adaptation law of σ.

The structure and parameters of the controller are the result of an analytical process that is so procedural that symbolic math procedures exist to automate the controller synthesis process and output the controller code (Rios-Bolivar & Zinober, 1997). However, the number of parameters to tune becomes very high with high system order, such that tuning the controller is still an *ad hoc* procedure.

As with the relatively simple controllers presented in this thesis, the same GA tuning algorithm may be used for the more complex ABC controller. The ABC controller synthesis process is based on constructing the following CLF:

$$V = \frac{1}{2}\sum_{k=1}^{\rho}\left(z_k^2 + \frac{1}{2d_k}\varepsilon^T P\varepsilon \right) + \frac{1}{2}\tilde{\theta}^T\Gamma^{-1}\tilde{\theta} + \frac{|b_m|}{2\gamma}\tilde{\sigma}^2 \tag{4.16}$$

where $\varepsilon = x - \hat{x}$, $\tilde{\theta} = \theta - \hat{\theta}$, and $\tilde{\sigma} = \sigma - \hat{\sigma}$. Because the CLF is the result of human analysis, it is relatively simple in that it is quadratic with no cross-coupling of the various error signal types. For the proposed generalized control design procedure, one would use the same error signals in (4.16) with the proposed generalized CLF of (4.1) & (4.2). Thus the generalized CLF would have the form:

$$V = \frac{1}{2}\left[z,\varepsilon,\tilde{\theta},\tilde{\sigma}\right]\left(\sum_{j=1}^{M} p_j(z,\varepsilon,\tilde{\theta},\tilde{\sigma})P_j \right)\left[z,\varepsilon,\tilde{\theta},\tilde{\sigma}\right]^T \tag{4.17}$$

The idea is that a simple CLF (4.15) is used to tractably derive a control law and key error signals (e.g. $z,\varepsilon,\tilde{\theta}$, & $\tilde{\sigma}$) while a more general yet more complex CLF (4.17) that varies with the error signals and exploits cross-coupled terms is used to tune the controller parameters and measure the performance. *Based on the results of Chapter 3,*

there is reason to believe tuning the controller parameters using the CLF of (4.17) could achieve a superior performance over that of using (4.16).

To demonstrate the usefulness of checking the change in V between time steps rather than $\dot{V}$ for the generalized CLF in (4.17), consider the structure of $\dot{V}$:

$$\begin{aligned}\dot{V} = &\left[\dot{z},\dot{\varepsilon},\dot{\tilde{\theta}},\dot{\tilde{\sigma}}\right]\left(\sum_{j=1}^{M} p_j(z,\varepsilon,\tilde{\theta},\tilde{\sigma})P_j\right)\left[z,\varepsilon,\tilde{\theta},\tilde{\sigma}\right]^T \\ &+\frac{1}{2}\left[z,\varepsilon,\tilde{\theta},\tilde{\sigma}\right]\left(\sum_{j=1}^{M}\nabla p_j\left[\dot{z},\dot{\varepsilon},\dot{\tilde{\theta}},\dot{\tilde{\sigma}}\right]^T P_j\right)\left[z,\varepsilon,\tilde{\theta},\tilde{\sigma}\right]^T\end{aligned} \tag{4.18}$$

where

$$\begin{aligned}
\dot{z}_1 &= -c_1 z_1 - d_1 z_1 + \hat{b}_m z_2 + \varepsilon_2 + \left(\omega - \hat{\sigma}\left(\dot{y}_r + \overline{\alpha}_1\right)e_1\right)^T\tilde{\theta} - b_m\left(\dot{y}_r + \overline{\alpha}_1\right)\tilde{\sigma} \\
\dot{z}_2 &= -c_2 z_2 - d_2\left(\frac{\partial\alpha_1}{\partial y}\right)^2 z_2 - \hat{b}_m z_1 + z_3 - \frac{\partial\alpha_1}{\partial\hat{\theta}}\left(\dot{\hat{\theta}} - \Gamma\tau_2\right) - \frac{\partial\alpha_1}{\partial y}\varepsilon_2 - \frac{\partial\alpha_1}{\partial y}\omega^T\tilde{\theta} \\
\dot{z}_i &= -c_i z_i - d_i\left(\frac{\partial\alpha_{i-1}}{\partial y}\right)^2 z_i - z_{i-1} + z_{i+1} - \sum_{j=2}^{i-1}\frac{\partial\alpha_{j-1}}{\partial\hat{\theta}}\Gamma\frac{\partial\alpha_{j-1}}{\partial y}z_j - \frac{\partial\alpha_{j-1}}{\partial\hat{\theta}}\left(\dot{\hat{\theta}} - \Gamma\tau_i\right) \\
&\quad - \frac{\partial\alpha_{i-1}}{\partial y}\varepsilon_2 - \frac{\partial\alpha_{i-1}}{\partial y}\omega^T\tilde{\theta}, \qquad i = 3,\ldots,\rho \\
\dot{\varepsilon} &= \dot{x} - \dot{\hat{x}} = f(x) + g(x)u - \dot{\xi} - \dot{\Omega}^T\hat{\theta} - \Omega^T\dot{\hat{\theta}} \quad \text{with } \hat{x} = \xi + \Omega^T\hat{\theta}
\end{aligned} \tag{4.19}$$

The alternative method of performance checking that uses the Euler approximation for the signal time derivatives is expressed as

$$\left[V\right]_{k+1} = \left[\frac{1}{2}\left[z,\varepsilon,\tilde{\theta},\tilde{\sigma}\right]\left(\sum_{j=1}^{M} p_j(z,\varepsilon,\tilde{\theta},\tilde{\sigma})P_j\right)\left[z,\varepsilon,\tilde{\theta},\tilde{\sigma}\right]^T\right]_{k+1} \tag{4.20}$$

where

$$\begin{aligned}
\left[z\right]_{k+1} &= \left[z + \Delta t\cdot\dot{z}\right]_k \\
\left[\varepsilon\right]_{k+1} &= \left[x - \hat{x} + \Delta t\cdot\left(f(x) + g(x)u - \dot{\xi} - \dot{\Omega}^T\hat{\theta} - \Omega^T\dot{\hat{\theta}}\right)\right]_k \\
\left[\tilde{\theta}\right]_{k+1} &= \left[\theta - \hat{\theta} + \Delta t\cdot\left(F_\theta - \Gamma\tau_\rho\right)\right]_k \\
\left[\tilde{\sigma}\right]_{k+1} &= \left[\sigma - \hat{\sigma} + \Delta t\cdot\left(F_\sigma - \gamma\,\mathrm{sgn}\left(b_m\right)\left(\dot{y}_r + \overline{\alpha}_1\right)z_1\right)\right]_k
\end{aligned} \tag{4.21}$$

The expression in (4.20) is much simpler than (4.18), especially when using many basis functions in the CLF and when the signal space is very large.

While the use of an adaptive controller is analogous to online system identification, it is not a "complete" online system identification procedure because the model order must be determined by the controller designer. The GA can readily handle this problem by selecting the order of the state and zero dynamics (e.g. *m* and *n* of equation (4.5)). Xian et al. (2003) has shown how the linear system back-stepping controller using tuning functions can not only adaptively stabilize an uncertain system, but also employ the *internal model principle* by tuning the "additional dynamics" (the part of the system that represents the external signal dynamics) to match the dynamics of disturbances. In other words, both the system and it's external time varying disturbances can be approximated by a sufficiently high order model. The GA could tune both the controller parameters and the system order, or the "observer" order for the general framework outlined in this chapter, to robustly achieve the performance objectives in the face of disturbances and uncertainties.

The following list of theoretical work must accompany the proposed controller design method:

1. A method must be developed that guarantees that the critical point searching algorithm checks only the interior of the level sets of the CLF that are completely contained in Ψ. Points not enclosed by level sets contained by Ψ have no guarantee to stay in Ψ.
2. Johansen's proof of the generalized Lyapunov function represented by equations (4.1) and (4.2) must be extended to the discrete time case.

3. Inter-sample behavior of the discretized dynamics must be addressed.
4. A probabilistic framework must be developed for evaluating and guaranteeing the operation of the optimization method (e.g. genetic algorithm, particle filter, etc. convergence analysis).

With a generalized method for automated controller tuning, the use of complex control laws will be made more practical, hence more widely used in real-world applications. In addition, a step is made towards the endeavor of truly automated systems with automated controller selection and tuning.

4.6 Conclusion

In conclusion, we make the following assertions on the contributions of this thesis:

1. A genetic algorithm for tuning CLF controllers using arbitrary fitness functions including hard constraints on controller gains, control effort, and rate of convergence is presented. The genetic algorithm is novel in that it is based on point-wise CLF minimum rate of convergence and maximum control effort estimates rather than the use of simulations of the system responses to tune the controller parameters.
2. A procedure for designing full state-feedback linear controllers for nonlinear systems using combined local and non-local information of the system dynamics is presented. The controllers are locally inverse optimal and CLF-based.
3. A procedure for designing full-state feedback Sontag-like controllers with minimal control effort for nonlinear systems is presented. As with

the linear controllers, they are locally inverse optimal and by definition are CLF-based.

4. A general framework has been outlined for future research on a constructive control procedure for optimized output feed-back, nonlinear, adaptive, robust, and discrete time control. The framework will allow for practical implementation and tuning of complex control algorithms and may lead to the development of truly automated systems that have the ability to select and tune their own control systems.

References

Arulampalam, M., S. Maskell, N. Gordon, & T. Clapp, "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking", IEEE Trans. On Signal Processing, Vol. 50, No. 2, Feb. 2002.

Carpenter, G.A., S. Grossberg, and J. Reynolds, "ARTMAP: Supervised real-time learning and classification of non-stationary data by a self-organizing neural network," *Neural Networks*, vol. 4, pp. 565-588, 1991.

El Ghaoui, L.& V. Balakrishnan, "Synthesis of fixed-structure controllers via numerical optimization", Proceedings of the 33rd IEEE Conference on Decision and Control, Vol.3, Iss., 14-16 Dec. 1994.

Flemming, P.J. & R.C. Purshouse, "Evolutionary algorithms in control systems engineering: a survey", Control Engineering Practice 10 (2002), pp. 1223-1241.

Goldberg, David E., *The Design of Innovation: Lessons from and for Competent Genetic Algorithms*, Kluwer Academic Publishers, 2002.

Jamshidi, M., L Coelho , P Fleming , and R Krohling, *Robust Control Systems with Genetic Algorithms*, CRC Press, 2003.

[a] Johansen, T.A., "Computation of Lyapunov functions for smooth nonlinear systems using convex optimization", *Automatica*, vol. 36, 2000.

[b] Johansen, T. A., "Computational Performance Analysis of Nonlinear Dynamic Systems using Semi-infinite Programming", 39th Conf. on Decision and Control, IEEE, Sydney, Australia, Dec. 2000.

Khalil, Hassan K., *Nonlinear Systems, 2nd Edition*, Prentice Hall, 1996.

Kokotovic, Petar and Murat Arcak, "Constructive nonlinear control: a historical perspective", Automatica, Volume 37, Issue 5, May 2001, Pages 637-662

Koza, John R., Forrest H Bennett III, David Andre, & Martin A. Keane, Genetic Programming III: Darwinian Invention and Problem Solving, Morgan Kaufmann Publishers, 1999.

Krstic, Miroslav, Ioannis Kanellakopoulos, & Peter Kokotovic, *Nonlinear and Adaptive Control Design*, John Wiley & Sons, Inc., 1995.

Ngamsom, P., "Enlarged Local Stabilizations of a Class of Uncertain Nonlinear Systems Using Lyapunov Attractive Region Control", Ph.D Dissertation, Oklahoma State University, Stillwater, OK, 2001.

Ogata, K., *Modern Control Engineering: 3rd Edition*, Upper Saddle River, NJ: Prentice Hall, 1997.

Primbs, James A., Vesna Nevistic, and John C. Doyle, "A receding horizon generalization of pointwise min-norm controllers", IEEE Trans. On Automatic Control, Vol. 45, No. 5, May 2000.

Resnick, Sidney I., *A Probability Path,* Birkhauser, 1999.

Rios-Bolivar, M., A.S.I. Zinober, **"**Symbolic algebra toolbox for the design of dynamical adaptive backstepping controllers**",** IEE Colloquium on Robust Control: Theory, Software and Applications (Digest No: 1997/380), Iss., 17 Oct 1997, Pages:9/1-9/3

Rocco, Paolo, "Stability of PID Control for Industrial Robot Arms", IEEE Trans. On Robotics and Automation, Vol. 12, No. 4, August, 1996.

Sontag, E. D. "A 'universal' construction of Arstein's theorem on nonlinear stabilization", Sys. Contr. Lett., vol 13, 1989, pp. 117-123.

Stark, Henry and John W. Woods, *Probability, Random Processes, and Estimation Theory for Engineers (2nd Edition)*, Prentice-Hall, Inc., 1994.

Spooner, Jeffrey T., Manfredi Maggiore, Raul Ordenez, and Kevin M. Passino, *Stable Adaptive Control and Estimation for Nonlinear Systems*, Wiley, 2002.

Slotine, J.-J. E., and Li. W., *Applied Nonlinear Control*, Englewood Cliffs, NJ: Prentice Hall, 1991.

Xian, B. , N. Jalili, D. Dawson, and Y. Fang, "Adaptive Tracking Control of Linear Uncertain Mechanical Systems in Presence of Unknown Sinusoidal Disturbances", Transactions of the ASME: Journal of Dynamic Systems, Measurements and Control, Vol. 125, No. 1, pp. 129-134, 2003

Zhou ,Kemin and John C. Doyle, *Essentials of Robust Control*, Prentice-Hall, Inc., 1997

Appendix: Matlab Code Listing and How to Use It

The following is the recommended controller design process using the algorithms proposed in the thesis. The relevant Matlab files (files with ".m" extension) and variables are referenced in the process steps to aid in the use of the listed code that follows.

1. Select control law → change equations in "f.m" and "uc.m"
2. Select the region in state space to optimize the controller's performance (i.e. span of X^0) → set "range" in "compLyaK.m"
3. Select the maximum magnitude of the controller gains → set "maxK" in "compLyaK.m"
4. Select GA population size, GA maximum number of generations, and the number of checking points (i.e. number of elements in X_d^0) → set "popsize", "maxgen", and "numPoints" in "compLyaK.m"
5. Run GA → execute "compLyaK.m"
6. Compute controller performance statistics → "getgamma.m"
7. Simulate controller's performance over time against various initial conditions, disturbances and inputs of interest → a user constructed Simulink model is suggested
8. If controller performance is satisfactory, implement on real system, otherwise repeat process from step 1.

The following is the listing of the code used to perform the optimization and performance evaluation of the controllers in the thesis. Note that for the code to work properly, the programs must be run in the same Matlab workspace because they share some of the same variables. Below, the program titles are listed in quotations followed by a brief description of the program's functionality and along with the code. The code is commented (text following the "%" symbol) in the more crucial areas. All other areas are assumed to be straightforward to readers familiar with basic Matlab programming.

"compLyaK.m"

This is the initialization and main loop of the program. Variables are initialized including the settings of the three example problems in chapter 3. After initialization, the genetic algorithm loops a number of times specified by the variable "maxgen" (i.e. the maximum number of generations).

code:

```
clear;%clear workspace
tic;%start timer
%Initialization
%set range of X0 (performance checking set)
if 1 %set to 0 for 2nd order system example; set to 1 for 4th order examples
   range(1)=0.1;
   range(2)=range(1);
   range(3)=range(1);
   range(4)=range(1);
else
   range=[25 100];
end
numPoints=200;%number of points in X0
maxgen=50;%number of GA generations
popsize=50;%size of GA population
pr=0;%probability of reproduction
pc=0.5;%probability of crossover
%probability of mutation = 1-pr-pc
```

```
%sys = system number
%sys=1 --> 2nd order nonlinear artificial system
%sys=2 --> 2nd order linearized artificial system
%sys=3 --> 4th order nonlinear double inverted pendulum system
%sys=4 --> 4th order linearized double inverted pendulum system
%sys=5 --> 4th order nonlinear cart-and-pole system
%sys=6 --> 4th order linearized cart-and-pole system
sys=5;

%ctype = controller type: 1-->linear, 2-->nonlinear
ctype=2;
c=1;%Q=E'*E+cI from equation 2.4
maxK=1000;%upper controller gain magnitude limit
sigmaE=100;%variance of E mutation steps
W=[10 1];%fitess function weights
W=W/norm(W);%normalize weight vector to length=1
%Linearized System Matrices
if (sys==1)|(sys==2)
   A=[10 10;0 1];
   B=[0 2.5]';
   dimf=2;%dimension of states
   bestp=[4160.8 125.6;125.6 54.4];%set best P of population to Ngamsom's
   bestk=-[157.487 68.244];%set best K of population to Ngamsom's
end
if (sys==3)|(sys==4)
   B=[0 0 113.4536 -101.2405]';
   A=[0 0 1 0;0 0 0 1;43.0258 -9.6925 -.2541 .1202;-38.3942 51.7297 .4787 -.3593];
   dimf=4;%dimension of states
   %set best P of population to Ngamsom's
   bestp(1,1:4)=[3.8822 9.7644 1.2414 1.4116];
   bestp(2,2:4)=[92.9231 10.0839 12.3278];
   bestp(3,3:4)=[1.2109 1.4285];
   bestp(4,4)=1.7592;
   bestp(2:4,1)=bestp(1,2:4)';
   bestp(3:4,2)=bestp(2,3:4)';
   bestp(4,3)=bestp(3,4);
   bestp=.1*bestp;
   bestk=[.1036 5.2008 .3618 .8021];%set best K of population to Ngamsom's
end
if (sys==5)|(sys==6)
   B=[0 0 .5 -1]';
   A=[0 0 1 0;0 0 0 1;0 -.495 0 0;0 20.6 0 0];
   dimf=4;%dimension of states
   %set best P of population to Ngamsom's
   bestp(1,1:4)=[3.1924 4.1163 1.5478 .8186];
```

```
   bestp(2,2:4)=[20.4904 5.7517 3.236];
   bestp(3,3:4)=[2.0509 1.0969];
   bestp(4,4)=.6405;
   bestp(2:4,1)=bestp(1,2:4)';
   bestp(3:4,2)=bestp(2,3:4)';
   bestp(4,3)=bestp(3,4);
   bestp=1000*bestp;
   bestk=[22.3607 180.0604 35.6916 46.0217];%set best K of population to Ngamsom's
end
E=.5*randn(dimf,dimf,popsize);%generate initial set of E matrices
P=zeros(size(E));%allocate set of P matrices
bestspecies=1;%index of best E
fitness(1:popsize)=0;%vector of fitness values for set of E matrices
V(1:numPoints,1:popsize)=0;%matrix of CLF values
Vdot(1:numPoints,1:popsize)=0;%matrix of CLF rate of change values
gamma(1:numPoints,1:popsize)=0;%matrix of gamma values
u(1:numPoints,1:popsize)=0;%matrix of control effort values

%genetic algorithm main loop
for gen=1:maxgen %perform fixed number of iterations (generations)
   %randomly distribute search points
   for ii=1:dimf
      X(:,ii)=2*range(ii)*(rand(numPoints,1)-.5);
   end
   %set some of the search points equal to the estimated critical points
   if gen>1
      numPoints2=min(round(numPoints/3),size(problemXg,1));
      X(1:numPoints2,1:dimf)=problemXg(1:numPoints2,1:dimf);
      X(numPoints2+1:2*numPoints2,1:dimf)=problemXu(1:numPoints2,1:dimf);
   end
   operatePK;%genetic operations on P and K (actually E)
   evalPK;%evaluate P and K (actually E)
end
toc;
```

“operatePK.m”

Implicitly performs genetic operations on “P” and “K” by manipulating “E”.

code:

```
%Genetic Operations
if gen>1
   %create a sorted list of species from highest to lowest fitness
   [sortedfitness sortedindices]=sort(-fitness);
   %tempE holds the population for the next generation
   tempE(:,:,1)=bestE;%auto copy best from last generation
```

```
   for specimen=2:popsize
      %select two species
      first=round(rand^2*(popsize-1))+1;%s: rand^s (from chapter 2)
      second=round(rand^2*(popsize-1))+1;
      first=sortedindices(first);
      second=sortedindices(second);
      %select operation based on 'decision' variable
      decision=rand;
      if decision<=pr %reproduce
         tempE(:,:,specimen)=E(:,:,first);
      end
      if and(decision>pr,decision<=pr+pc) %crossover
         for state1=1:dimf
            for state2=1:dimf
               betaE=2*rand;
               tempE(state1,state2,specimen)=betaE*E(state1,state2,first)+…
(1-betaE)*E(state1,state2,second);
            end
         end
      end
      if and(decision>pr+pc,decision<=1)%mutate
         tempE(:,:,specimen)=E(:,:,first)+sigmaE*randn(dimf);
      end
   end
   %replace old population
   E=tempE;
end
for specimen=1:popsize
   % compute P from continuous-time algebraic ricatti equation solver
   P(:,:,specimen)=care(A,B,E(:,:,specimen)'*E(:,:,specimen)+c*eye(dimf));
   %compute K
   K(:,:,specimen)=-(B'*P(:,:,specimen))';
end
```

“evalPK.m”

Implicitly evaluates “E” by evaluating “P” and “K”. “P” and “K” are evaluated by their effect on the estimated minimum rate of convergence “mingamma” and the estimated maximum control effort “maxu”.

code:

```
   %Fitness Calculation
   for specimen=1:popsize %cycle through entire population
```

```
      for point=1:numPoints
         %compute V for 'specimen' at 'point' on the checking set
         V(point,specimen)=X(point,:)*P(:,:,specimen)*X(point,:)';
         %compute Vdot for 'specimen' at 'point' on the checking set
Vdot(point,specimen)=f(X(point,:),K(:,:,specimen),P(:,:,specimen),sys,ctype)*…
(P(:,:,specimen)'+P(:,:,specimen))*X(point,:)';
         %compute rate of convergence for 'specimen' at 'point' on the checking set
         gamma(point,specimen)=-Vdot(point,specimen)/(V(point,specimen)+10^(-8));
         %compute control effort for 'specimen' at 'point' on the checking set
         u(point,specimen)=uc(X(point,:),K(:,:,specimen),P(:,:,specimen),sys,ctype);
      end
      if numPoints>0
         %among all the checking points find the minimum gamma and its index
         [mingamma(specimen) badindex(specimen)]=min(gamma(:,specimen));
%estimated critical point for gamma among all the checking points (plus a random
%perturbation)
problemXg(specimen,1:dimf)=X(badindex(specimen),:)+.01*min(range)*randn(1,dimf);
         if norm(problemXg(specimen,1:dimf))>min(range)
            for ii=1:dimf
               if abs(problemXg(specimen,ii))>range(ii)
                  problemXg(specimen,ii)=sign(problemXg(specimen,ii))*range(ii);
               end
            end
         end
         [maxu(specimen) badindex(specimen)]=max(abs(u(:,specimen)));
 %estimated critical point for u among all the checking points (plus a random
%perturbation)
problemXu(specimen,1:dimf)=X(badindex(specimen),:)+.01*min(range)*randn(1,dimf);
         if norm(problemXu(specimen,1:dimf))>min(range)
            for ii=1:dimf
               if abs(problemXu(specimen,ii))>range(ii)
                  problemXu(specimen,ii)=sign(problemXu(specimen,ii))*range(ii);
               end
            end
         end
      else
         mingamma(specimen)=0;
         maxu(specimen)=0;
      end
      maxKval(specimen)=max(abs(K(:,:,specimen)));
   end%end fitness calculation

%compute fitness
   L1max=max(mingamma);
   L1min=min(mingamma);
   L2max=max(maxu);
```

```
    L2min=min(maxu);
    if numPoints==0
      L1max=1;
      L2max=1;
    end
    if L1max==L1min
       L1min=0;
    end
     if L2max==L2min
       L2min=0;
    end
    for specimen=1:popsize
       f1(specimen)=(mingamma(specimen)-L1min)/(L1max-L1min);
       f2(specimen)=(1-(maxu(specimen)-L2min)/(L2max-L2min));
       if (maxKval(specimen)<=maxK)
          fitness(specimen)=W(1)*f1(specimen)+W(2)*f2(specimen);
       else
          fitness(specimen)=0;
          E(specimen)=.9*E(specimen);%reduce magnitude of E to reduce magnitude of K
       end
    end

     %get best species
     [fit bestspecies]=max(fitness);
     bestfitgen(gen)=fit;
     disp('-------------------------------------------')
     disp(strcat('generation #',num2str(gen)))
     disp('  mingamma     maxu')
     disp([mingamma(bestspecies) maxu(bestspecies)])
     disp('subfitnesses')
     disp([f1(bestspecies) f2(bestspecies)])
     plot(bestfitgen);
     title(strcat('Best Fitness of Generation = ',num2str(bestfitgen(gen))));
     xlabel('Generation');
     ylabel('Best Fitness');
     pause(.01)

disp('best P')
disp(P(:,:,bestspecies))
disp('best K')
disp(K(:,:,bestspecies)')
bestE=E(:,:,bestspecies);
disp('best Q')
disp(bestE'*bestE+c*eye(dimf))
bestp=P(:,:,bestspecies);
bestk=K(:,:,bestspecies);
```

save c:\matlabr12\work\bestE bestE bestp bestk;

“f.m”

This is the function for the time derivatives of the states. The following arguments are used: the state vector “x”, the controller and CLF parameters “k” and “p”, respectively, the system number “sys” (out of 6 choices), and the controller type “ctype” (linear or nonlinear).

code:

```
function y=f(x,k,p,sys,ctype);
if sys==1
   %artificial system
   y(1)=10*x(2)+10*x(1)+10*(sin(x(1))^2)*sin(x(2))-x(1)^2;
   y(2)=5*x(1)^2+sin(x(2))+(cos(x(2))+1.5)*uc(x,k,p,sys,ctype);
end
if sys==2
   %linearized artificial system
   y(1)=10*x(2)+10*x(1);
   y(2)=x(2)+2.5*uc(x,k,p,sys,ctype);
end
if sys==3
   %double inverted pendulum
   a(1)=x(3);
   b(1)=0;
   a(2)=x(4);
   b(2)=0;
   a(3)=sin(x(1)-x(2))*x(3)^2+.2824*x(3)-.2824*x(4)+48.2776*sin(x(2));
   a(3)=a(3)*cos(x(1)-x(2))+.9833*x(3)+1.1206*sin(x(1)-x(2))*x(4)^2;
   a(3)=a(3)-.3165*x(4)-214.3082*sin(x(1));
   a(3)=a(3)/(-5.9809+cos(x(1)-x(2))^2);
   b(3)=-565.1008/(-5.9809+cos(x(1)-x(2))^2);
   a(4)=-.8774*x(3)-sin(x(1)-x(2))*x(4)^2+.2824*x(4)+191.2383*sin(x(1));
   a(4)=a(4)*cos(x(1)-x(2))-5.3371*sin(x(1)-x(2))*x(3)^2-1.5071*x(3);
   a(4)=a(4)+1.5071*x(4)-257.6614*sin(x(2));
   a(4)=a(4)/(-5.9809+cos(x(1)-x(2))^2);
   b(4)=504.2688*cos(x(1)-x(2))/(-5.9809+cos(x(1)-x(2))^2);
   y=a+b*uc(x,k,p,sys,ctype);
end
if sys==4
   %linearized double inverted pendulum
   a(1)=x(3);
```

```
    b(1)=0;
    a(2)=x(4);
    b(2)=0;
    a(3)=[43.0258   -9.6925   -0.2541    0.1202]*[x(1) x(2) x(3) x(4)]';
    b(3)=113.4536;
    a(4)=[-38.3942   51.7297    0.4787   -0.3593]*[x(1) x(2) x(3) x(4)]';
    b(4)=-101.2405;
    y=a+b*uc(x,k,p,sys,ctype);
end
if sys==5
    %cart and pole
    a(1)=x(3);
    a(2)=x(4);
    a(3)=(.05*sin(x(2))*x(4)^2-.981*sin(x(2))*cos(x(2)))/(2+.1*sin(x(2))^2);
    a(4)=41.2*sin(x(2))/(2+.1*sin(x(2))^2)-…
.1*(cos(x(2))*sin(x(2))*x(4)^2)/(2+.1*sin(x(2))^2);
    b(1)=0;
    b(2)=0;
    b(3)=1/(2+.1*sin(x(2))^2);
    b(4)=-2*cos(x(2))/(2+.1*sin(x(2))^2);
    y=a+b*uc(x,k,p,sys,ctype);
end
if sys==6
    %linearized cart and pole
    a(1)=x(3);
    a(2)=x(4);
    a(3)=-0.4950*x(2);
    a(4)=20.6000*x(2);
    b(1)=0;
    b(2)=0;
    b(3)=.5;
    b(4)=-1;
    y=a+b*uc(x,k,p,sys,ctype);
end
```

“uc.m”

This is the function for the controller. The arguments are the same as “f.m”: the state vector “x”, the controller and CLF parameters “k” and “p”, respectively, the system number “sys” (out of the 6 choices), and the controller type “ctype” (linear or nonlinear).

code:

```
function u=uc(x,k,p,sys,ctype)
```

```
if (sys==1)|(sys==2)
   if ctype==1
      u=k(1)*x(1)+k(2)*x(2);
   end
   if ctype==2
      a(1)=10*x(2)+10*x(1)+10*(sin(x(1))^2)*sin(x(2))-x(1)^2;
      b(1)=0;
      a(2)=5*x(1)^2+sin(x(2));
      b(2)=cos(x(2))+1.5;
      dVdx=2*x*p;
      dVdxg=dVdx*b';
      if abs(dVdxg)>0
         dVdxf=dVdx*a';
         A=[10 10;0 1];
         B=[0 2.5]';
         Q=(-A'*p-p*A+p*B*B'*p);
         q=x*Q*x';
         u=-(dVdxf+sqrt(dVdxf^2+q*dVdxg^2))/dVdxg;
      else
         u=0;
      end
   end
end
if (sys==3)|(sys==4)
   if ctype==1
      u=k(1)*x(1)+k(2)*x(2)+k(3)*x(3)+k(4)*x(4);
   end
   if ctype==2
      a(1)=x(3);
      b(1)=0;
      a(2)=x(4);
      b(2)=0;
      a(3)=sin(x(1)-x(2))*x(3)^2+.2824*x(3)-.2824*x(4)+48.2776*sin(x(2));
      a(3)=a(3)*cos(x(1)-x(2))+.9833*x(3)+1.1206*sin(x(1)-x(2))*x(4)^2;
      a(3)=a(3)-.3165*x(4)-214.3082*sin(x(1));
      a(3)=a(3)/(-5.9809+cos(x(1)-x(2))^2);
      b(3)=-565.1008/(-5.9809+cos(x(1)-x(2))^2);
      a(4)=-.8774*x(3)-sin(x(1)-x(2))*x(4)^2+.2824*x(4)+191.2383*sin(x(1));
      a(4)=a(4)*cos(x(1)-x(2))-5.3371*sin(x(1)-x(2))*x(3)^2-1.5071*x(3);
      a(4)=a(4)+1.5071*x(4)-257.6614*sin(x(2));
      a(4)=a(4)/(-5.9809+cos(x(1)-x(2))^2);
      b(4)=504.2688*cos(x(1)-x(2))/(-5.9809+cos(x(1)-x(2))^2);
      dVdx=2*x*p;
      dVdxg=dVdx*b';
      if abs(dVdxg)>0
         dVdxf=dVdx*a';
```

```
        B=[0 0 113.4536 -101.2405]';
        A=[0 0 1 0;0 0 0 1;43.0258 -9.6925 -.2541 .1202;-38.3942 51.7297 .4787 -.3593];
        Q=(-A'*p-p*A+p*B*B'*p);
        q=x*Q*x';
        u=-(dVdxf+sqrt(dVdxf^2+q*dVdxg^2))/dVdxg;
      else
        u=0;
      end
   end
end
if (sys==5)|(sys==6)
   if ctype==1
      u=k(1)*x(1)+k(2)*x(2)+k(3)*x(3)+k(4)*x(4);
   end
   if ctype==2
      a(1)=x(3);
      a(2)=x(4);
      a(3)=(.05*sin(x(2))*x(4)^2-.981*sin(x(2))*cos(x(2)))/(2+.1*sin(x(2))^2);
      a(4)=41.2*sin(x(2))/(2+.1*sin(x(2))^2)-…
.1*(cos(x(2))*sin(x(2))*x(4)^2)/(2+.1*sin(x(2))^2);
      b(1)=0;
      b(2)=0;
      b(3)=1/(2+.1*sin(x(2))^2);
      b(4)=-2*cos(x(2))/(2+.1*sin(x(2))^2);
      dVdx=2*x*p;
      dVdxg=dVdx*b';
      if abs(dVdxg)>0
        dVdxf=dVdx*a';
        B=[0 0 .5 -1]';
        A=[0 0 1 0;0 0 0 1;0 -.495 0 0;0 20.6 0 0];
        Q=(-A'*p-p*A+p*B*B'*p);
        q=x*Q*x';
        u=-(dVdxf+sqrt(dVdxf^2+q*dVdxg^2))/dVdxg;
      else
        u=0;
      end
   end
end
```

getgamma.m

Randomly sample “gamma” and “u” to compute statistical quantities: “gammamin” ($\gamma_{\min}$), “gammamu” (μ_{γ}), “gammasig” (σ_{γ}), “umax” ($|u|_{\max}$), “ummu” ($\mu_{|u|}$), “usig” ($\sigma_{|u|}$).

code:

```
clear X V Vdot u gammarec;
samplesize=20000;
gammamin=10000;
Vdotmax=0;
gammarec(1:samplesize)=0;
X(1:samplesize,1:dimf)=0;
V(1:samplesize)=0;
Vdot(1:samplesize)=0;
u(1:samplesize)=0;
for sample=1:samplesize
   X(sample,:)=range.*(2*rand(1,dimf)-1);
    if norm(X(sample,:))<.01
       X(sample,:)=0;
    end
   V(sample)=X(sample,:)*bestp*X(sample,:)';
   Vdot(sample)=f(X(sample,:),bestk,bestp,sys,ctype)*(bestp'+bestp)*X(sample,:)';
   u(sample)=uc(X(sample,:),bestk,bestp,sys,ctype);
   if and(not(Vdot(sample)==0),not(V(sample)==0))
      gammamin=min(-Vdot(sample)/V(sample),gammamin);
      gammarec(sample)=-Vdot(sample)/V(sample);
   end
       Vdotmax=max(Vdot(sample),Vdotmax);
end
 plot(u);
%display quantities
disp( range)
disp(gammamin)
gammamu=mean(gammarec)
gammasig=std(gammarec)
umax=max(abs(u))
umu=mean(abs(u))
usig=std(abs(u))
Pgammamin=.01;% for 99% confidence
eps=gammamu/2;
```

```
%"residual" is the difference between terms in left and right hand side of equation 2.24.
%If it is positive, then the number of sample points is sufficient to the specified
%confidence level (1-"Pgammamin")
residual=samplesize^2-(gammasig^2+eps*abs(mean(gammarec-
gammamu)))/((Pgammamin)*eps^2)
```

VITA

Brian K. Hargrave

Candidate for the Degree of

Master of Science

Thesis: USING GENETIC ALGORITHMS TO OPTIMIZE CONTROL LYAPUNOV FUNCTIONS

Major Field: Mechanical Engineering

Biographical:

Education:

Bachelor of Science in Mechanical Engineering, Oklahoma State University, Stillwater, Oklahoma, May, 1999.

Completed the requirements for the Master of Science in Mechanical Engineering at Oklahoma State University, Stillwater, Oklahoma in July, 2008.

Experience:

Mr. Hargrave has worked professionally as an engineer for 9 years, developing advanced robotic, automation, and control systems for the Department of Defense and NASA.

Name: Brian K. Hargrave Date of Degree: July, 2008

Institution: Oklahoma State University Location: Stillwater, Oklahoma

Title of Study: USING GENETIC ALGORITHMS TO OPTIMIZE CONTROL LYAPUNOV FUNCTIONS

Pages in Study: 179 Candidate for the Degree of Master of Science

Major Field: Mechanical Engineering

Scope and Method of Study:

Findings and Conclusions:

The problem of simultaneously tuning the control law and the control Lyapunov function (CLF) for nonlinear systems is considered, and a genetic algorithm optimization method is proposed as a general solution. It is shown that direct sampling of the nonlinear effects on the CLF offers an advantage over robust linear control methods that assume a linearized system with the nonlinearities treated as uncertainties. While the genetic algorithm approach is not guaranteed to find a solution, experimentation suggests that it does possess a high likelihood of finding "good" solutions to some nontrivial problems. The control law and the local CLF are tuned simultaneously to maximize the rate of convergence and minimize the control effort of nonlinear systems. Two control laws are tested: 1. an LQR full-state feedback controller, and 2. a Sontag-like nonlinear full-state feedback controller. It is assumed that a quadratic Lyapunov function is a local CLF for the nonlinear systems considered. The proposed optimization method does not offer a strict guarantee on controller performance. However, it is suggested that with enough randomized performance sampling, the controller will achieve the estimated performance level with sufficiently high confidence, making the proposed method a practical solution for real-world controller design.

ADVISER'S APPROVAL: Lawrence L. Hoberock