# Robustness Improvement by Dynamic State Feedback Stabilization

Submitted for the partial fulfillment of the requirement for the degree of

Master of Technology (Research)

in

Electrical Engineering

by

DUSHMANTA KUMAR DAS

Under the guidance of

Prof. SANDIP GHOSH

DEPT. OF ELECTRICAL ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY
ROURKELA-769008, INDIA.

NOVEMBER 2010

# CERTIFICATE OF APPROVAL

This is to certify that Mr. Dushmanta Kumar Das has successfully defended his M. Tech (Research) via-voce examination on the thesis entitled "Robustness Improvement by Dynamic State Feedback Stabilization" today.

Prof.B. Subudhi
(MSC Member)

Prof.K. C. Pati
(MSC Member)

Prof. S. Maity
(MSC Member)

Prof.S.Ghosh
(Supervisor)

Prof. D. R. Parhi
(External Examiner)

Prof.B.D.Subudhi
(Chairman)

# CERTIFICATE

This is to certify that the thesis entitled "Robustness Improvement by Dynamic State Feedback Stabilization" by Mr. Dushmanta Kumar Das, submitted to the National Institute of Technology, Rourkela for the award of Master of Technology (Research) in Electrical Engineering, is a record of bonafide research work carried out by him in the Department of Electrical Engineering, National Institute of Technology, Rourkela under my supervision. I believe that this thesis fulfills part of the requirements for the award of degree of Master of Technology (Research). The results embodied in the thesis have not been submitted for the award of any other degree elsewhere.

*Prof. Sandip Ghosh*
*Department of Electrical Engineering*
*National Institute of Technology*
*Rourkela- 769008*

# Acknowledgement

"The mediocre teacher tells. The good teacher explains. The superior teacher demonstrates. The great teacher inspires."

Prof. Sandip Ghosh truly epitomizes these words. He is one of the most influential people in my life. I have gained immense knowledge and experience from him, which is invaluable. Without his expertise in the field of control system, this thesis wouldn't have been possible. I sincerely thank him for demonstrating confidence in me and providing an opportunity to work under his able guidance.

I thank all my teachers Prof. B. D. Subudhi, Prof. J. K. Satpathy, Prof. (Mrs.) S. Das, for their contribution in my studies and research work. They have been great sources of inspiration to me and I thank them from the core of my heart.

I take this opportunity to thank my beloved parents, wife, brother and my daughter for believing my ability and providing constant encouragement and support. Lastly, I thank GOD, the almighty for providing me with strength and His choicest blessings,

Dushmanta Kumar Das

# Contents

# Abstract

A general perception is that dynamic state feedback controllers may have superior capabilities compared to the static feedback ones. Although, for nominal systems, such superiority does not exist, it may hold true for systems other than the nominal one. This thesis investigates such capabilities of dynamic state feedback controllers.

First, the case of systems with time-invariant uncertainties is considered. It appears that only a limited number of example systems exist for which dynamic feedback controller has superior ability to improve tolerable uncertainty bounds. This thesis shows that not only the available ones but there exist a class of systems for which such improvement may occur. This claim has been verified by presenting more numerical examples in the category.

Second, this thesis considers the class of systems with feedback delays, more specifically, systems having both input and output delays. The superiority of dynamic feedback controllers for such systems appears to be not investigated so far in literature. However, it is observed, for the first time, that if one considers a dynamic state feedback controller but with an artificial delay in its state then the tolerable delay margin improves considerably. The performance of such a controller is investigated thoroughly for scalar systems using a continuous pole placement technique for delay systems.

# List of Symbols

| | |
|---|---|
| $u(t)$ | Control input to the System |
| $K$ | Static Controller gain |
| $x(t)$ | States of the system |
| $y(t)$ | Output of the system |
| $z(t)$ | Dynamic Controller states |
| $A$ | System matrix |
| $B$ | Control Input matrix |
| $\Delta A$ | Parametric Uncertainty in the system matrix |
| $\Delta B$ | Parametric Uncertainty in the Control input matrix |
| $r$ | Uncertainty Factor |
| $\lambda_i$ | Eigenvalues |
| $\tau_a$ | Actuation Delay |
| $\tau_s$ | Sensor Delay |
| $\tau_c$ | Controller Delay |
| $\tau_{total}$ | Total Delay in the Feedback loop |
| $v_i$ | Eigen Vectors |
| $n(v_i)$ | Normalizing Function for corresponding eigenvector |
| $S$ | Sensitivity matrix |
| $\Delta K$ | Change in Controller gain |
| $\Delta\Lambda^d$ | Desired Small displacement in eigenvalues |
| $S^\dagger$ | Moore-Penrose inverse or pseudo inverse of sensitivity matrix |
| $R(\lambda)$ | Real part of the eigenvalue |
| $H_\infty$ | The supremum norm of output to disturbance input |

# List of Figures

**Page No.**

# Chapter 1

# Introduction

## 1.1 Introduction

Feedback concept arises when the output of a plant is fed back to the plant input in order to attain certain desired performance automatically. A well-known performance measure is the robustness of the system in the sense that the system remains stable in spite of plant parameter variations. One may get improved robustness by using suitable feedback control.

Often, it is required first to stabilize an unstable plant by using feedback. Such a problem is known as stabilization problem. This thesis considers such stabilization problems in broad. A block diagram representation of such feedback stabilization problem using state feedback is shown in Fig.1.1. Moreover, problems on robust stabilization of different classes of systems are considered in this thesis.

## 1.2 State Feedback Stabilization

For illustration, consider a nominal Linear Time Invariant (LTI) system, the state space representation of which is given by

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{1.1}$$

where $x(t) \in R^n$ is the state of the system, $u(t)$ is the control input, $A$ is the system matrix and $B$ is the control input matrix.

Note that, in Fig 1.1, the controller may be of any type. The most basic one is the static state feedback controller in which one considers

$$u(t) = Kx(t)\,, \tag{1.2}$$

where $K$ is a constant gain matrix.

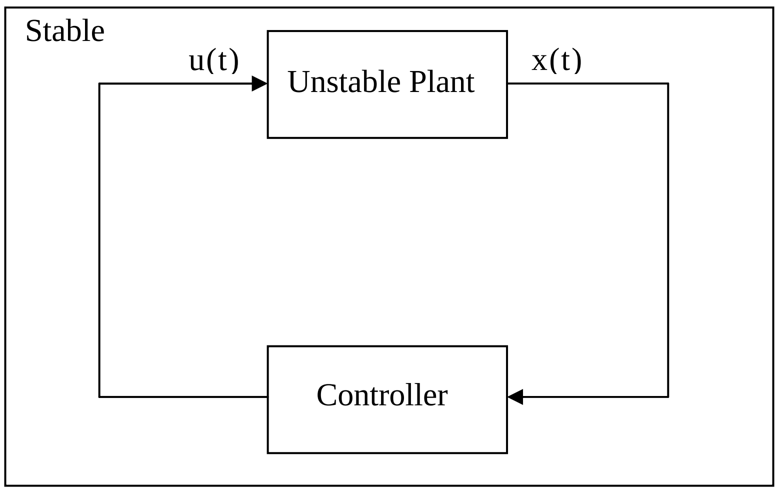


Fig.1.1: State Feedback Stabilization

On the other hand, controllers involving some internal dynamics into then are called dynamic feedback controllers. For example, for system (1.1), a linear dynamic state feedback controller may be of the form

$$\begin{aligned} \dot{z}(t) &= A_c z(t) + B_c x(t) \\ u(t) &= C_c z(t) + D_c x(t) \end{aligned} \tag{1.3}$$

where $z \in R^{n_c}$ is the state of the controller, and $A_c, B_c, C_c$ and $D_c$ are matrices of appropriate dimension.

Note in this regard that the controller (1.3) is complex compared to the controller (1.2) since more controller parameters are required to be designed as well as requires integrator(s) for implementation. However, for the nominal system (1.1), if the pair of $A$ and $B$ is controllable then there exists a static state feedback stabilizing controller ($K$) [24 and 35]. Obviously then the controllers of the type (1.3) is desired to be avoided. However, such dynamic controllers may be used to improve the robustness of the system other than the nominal one. This thesis investigates such possibilities. For the purpose, first we present a review on stabilization of an uncertain system for which there exists results that dynamic controllers may improve tolerable robustness bounds compared to the static feedback one. Next, a review on systems with feedback delay is also presented in which case it appears that superiority of dynamic feedback controllers has not been investigated so far.

### 1.3 A Review on State Feedback Stabilization of Uncertain Systems

One of the main objectives of feedback control is to achieve optimal plant performance in presence of uncertainties. There are several different causes that contribute to uncertainty in systems. One among these is the modeling error. Modeling error itself may arise out of two reasons (i) parameter perturbations originated by uncertainty in physical parameters, (ii) unmodeled dynamics arises out of the neglected delays and high frequency dynamics due to the model order reduction [1, 18, 21, 19 and 36]. In general, the order of a real plant is infinite dimensional due to the presence of time-delays and nonlinearities that is approximated to a finite dimensional one in order to have computationally convenient analysis, e.g., in case of a multi-loop control, the non-diagonal terms are often disregarded and each loop is individually tuned. In addition to these, there may also be unmeasurable perturbations that produce output deviations. With a robust controller in place, the achieved deviations must satisfy a user-defined bound.

The classical way of ensuring robustness is by designing a controller by guaranteeing relative stability in terms of gain and phase margin [17 and 39]. However, guaranteeing these margins ensures robustness of the system w.r.t. the loop gain and phase. In case of parametric uncertainty present in the system and at least their range of variations are known then such designs become conservative [13, 6 and 14]. Lyapunov based approaches have been used for determining the robustness bounds or robust stabilization in [5, 10 and 2]. Moreover, ensuring such frequency domain specifications are computationally heavy for Multi-Input-Multi-Output (MIMO) systems.

For systems having parametric uncertainties with known bounds of them, one may use the well-known Kharitonov's theorem to check system's stability for SISO systems [3].The methods based on Inverse Nyquist Array, Zero Inclusion Principle and Sixteen Plant Theorem may also be used to check the same [3]. However, some of these methods are based on graphical techniques and hence computationally heavy even for analysis. Controller design to obtain optimal robustness become more difficult and may be obtained by trial and error method for a certain finite range of the controller.

On the other hand, using the state-space models, where one does not have to bother about number of inputs/outputs of the system, the well-known approach for stability analysis is based on Lyapunov theory [12]. In this, one searches for the existence of a positive energy function over the states and tries to ensure that it decays with time for a particular system. The same approach may also be used for robust analysis of the system using the quadratic stability approach [23 and 24]. However, in this approach, one may obtain conservative results in the sense that this yields only sufficient conditions. In fact, it appears that given an uncertain system model, obtaining the stable parameter space in the case of the system having several uncertain parameters is difficult. However, for systems with a few uncertain parameters, it may be possible to obtain the parameter region for which the system remains stable. For systems either in state-space form or in transfer function form with a single uncertain parameter, criteria for obtaining tolerable parameter bound have been obtained in [8].

Apart from analysis, one may like to stabilize an uncertain system in such a way that its robustness is improved, in the sense that it can tolerate larger parameter variations. In comparison of the linear static state feedback and linear dynamic state feedback controllers, it is well known that using the quadratic stability approach (Lyapunov approach using a quadratic energy function), one does not get a better dynamic controller than the static one that can stabilize the system with larger uncertainty bound [24].

However, showing the limitations of the linear quadratic regulators in enhancing the open-loop gain-margin and phase-margin criterion, it has been shown in [39] that if a dynamic state feedback controller is used then the robustness can be improved considerably. Although the design result reported therein is erroneous as it has been pointed out in [17] and has further been rectified. More categorically, for systems with time-invariant uncertainties, it has been shown that there exist example cases of systems and dynamic controller for it that improves the robustness performance of the system [33]. That is, a dynamic feedback controller is superior to the static one at least for some stabilization problems. Note that, although no systematic design procedure of such controllers is available, one may use the stability results of [8] to check stability of the system. Also, one may possibly use parameter-dependent [16] or non-quadratic [40] Lyapunov function for analyzing stability of closed loop system in the case when controller parameters are given.

### 1.4 A Review on State Feedback Stabilization of Systems with Feedback Delay

Time-delays are common in real-world systems, e.g., it is often encountered in chemical processes and biological systems. In such cases, the future evolution of the state variables $x(t)$ depends on their past values, say $x(\xi), t-r \le \xi \le t$. Design of controllers for such time-delay systems are difficult compared to systems not having it. In fact, a closed loop control system may be unstable or may exhibit unacceptable transient response characteristics if the time delay used in the system model for controller design does not match with the actual delay present in the system [9]. Also, ignorance of the computation delay during analysis and design of digital control systems may lead to unpredictable and unsatisfactory system performance.

On the other hand, in perspective of the state feedback stabilization problem described in § 1.1, it was assumed that the control input is generated using the present states information only. However, in reality, this present states may not be available for the purpose due to (i) the time taken in measuring the states or to estimate the states from output measurement, (ii) computation time taken by the controller and (iii) time taken by the actuating process. Due to these reasons, the actual state used for the control input is basically the delayed state and the corresponding delay in the feedback loop is known as the feedback delay.

Based on the appearance of time-delays in the system model, there are two types of time-delay: input delay and state delay. Input delay is caused by the transmission of a control signal over a long-distance, or the delay is sometimes built into a system deliberately for control purposes [12, 34 and 37] whereas, state delay arises due to transmission or transport delay among interacting elements in a dynamic system [12].

In case of a closed loop control systems, time delay between output and input in the feedback loop is a common phenomena. They are introduced by transportation and communication lag

[12]. Time-delays in a control loop generally having a destabilizing influence and a make design of satisfactory controller is difficult to achieve [4]. When delays are not appropriately taken into account, a severe degradation of performance and even instability may occur [4 and 9]. The importance of study of time delays in control is now well recognized in a wide range of applications (transportation systems, communication networks, tele-operation systems, etc.). In last two decades, probably due to the new emerging applications in engineering (such as network controlled systems) well-supported by new theoretical results, several open problems have been solved (decoupling problems, stabilization, robustness, etc.) and several attempts have been made to obtain better and better results [12].

Several authors have considered stabilization of systems having both input and output delays in the feedback loop. In [11], a new approach is developed for sampled-data state feedback stabilization to the $H_\infty$ control by considering the input and output delay in the closed loop system. The approach consists of lifting technique in which the problem is transformed into an equivalent finite-dimensional discrete $H_\infty$ control problem and the solution is obtained in terms of differential Riccati equations. [11]. In case of such feedback delays, the stabilization of the closed loop system depends on the total tolerable delay in the feedback path that is the sum of the delays from the plant output to the controller and the controller to the plant input [15].

The stability of time-delay systems has been widely investigated in the last two decades. It is well known that there are several methods for studying stability of a time-delay system. Some of them are: Lyapunov-Krasovskii approach [12], Lyapunov-Razumikhin approach [12], and spectral radius method [26]. However, designing a stable controller is more difficult problem to solve as compared to the analysis one, since not only one has to guarantee the stability but also has to find a controller at the same time. For this, the above Lyapunov approaches yields only sufficient results.

The stabilization, i.e., to design a controller ensuring stability, for linear time-delay systems has been studied extensively in literature. Since Lyapunov approaches yields conservative results for stability analysis, they yield the same for stabilization problems as well. If the delay values are known exactly, then, in [29 and 30], an algorithm has been proposed based on numerical techniques. It is called as a *continuous pole placement* technique since it is an extension of the classical pole placement technique for linear systems. The approach makes use of the BIFTOOL toolbox [29] for computing the right most eigenvalues of the system and iteratively tries to place them in the left half plane by changing the controller parameters.

## 1.5 Scope of the Present Work

Based on the review carried out, the following problems on stabilization of uncertain systems and time-delay systems are seen to be not answered in literature.

- Although there exist few second order and third order example systems in literature for which dynamic feedback controllers may improve tolerable uncertainty bound, it is not clear whether there exist for which class of systems such improvements may be possible.
- Several attempts have been made to design both static and dynamic state feedback controllers for systems with delays. However, no attempt has been made so far to investigate whether a dynamic feedback controller can outperform static feedback controllers or not for systems with feedback delays in terms of the tolerable delay.

This thesis work attempts to answer the above issues.

## 1.6 Organization of the Thesis

This thesis has four chapters. As seen so far, the first chapter, a brief study on the stabilization of system with uncertainty and also a review on the stabilization of systems with feedback delay has been done. This chapter also discusses the scope of the present work.

In the second chapter, dynamic feedback stabilization of uncertain systems is considered. There, an algorithm is developed to check whether tolerable uncertainty bound for a given uncertain system can be enhanced by using a dynamic controller than that of a static one or not. The same algorithm is also used to develop some new example systems for which dynamic state feedback controller may enhance the tolerable uncertainty bound.

The third chapter considers the problem of stabilizing systems with feedback delay, a dynamic state feedback controller with a state delay is proposed to enhance the tolerable delay margin in the feedback loop, at least, for first order systems. The abilities of such controllers are thoroughly studied using continuous pole placement technique.

The final chapter presents a conclusion of the work and also proposes some future works.

At the end, three appendices are provided each on Frequency Sweeping Test (FST), BIFTOOL and MATLAB programs to demonstrate the uses of these computational tools used to obtain results of the third chapter.

# Chapter 2

# Stabilization of an Uncertain System

## 2.1 Introduction

It is well known that for systems with time-invariant uncertainties the robust margin in the sense that the tolerable uncertainty bound may be improved by using a dynamic state feedback controller compared to the static feedback one [33 and 24]. However, such improvements have been studied for some example cases [33] and it is not clear whether there exist some more examples or certain class of systems for which the improvement holds. In this paper, we investigate whether systems may be characterized in such a way that one may determine the tolerable uncertainty bound for static feedback case and compare with that obtainable by a suitable dynamic feedback one.

For the purpose, one first requires to obtain an approximate tolerable uncertainty bound achievable by static state feedback one. This has been computed by using a method of contradiction in [33]. Based on this method of contradiction, we attempted to characterize the class of second order systems for which this applies. Subsequently, we have developed an algorithm to test and verify whether a dynamic feedback controller can improve the uncertainty bound or not. Using this proposed algorithm, we have generated some new examples and verified that there exists a dynamic feedback controller which indeed improves the uncertainty bound for these cases.

Uncertainty in power system state estimation is mainly due to measurement inaccuracy and the network mathematical model used. For instance, meter inaccuracies and communication errors are major sources of measurement uncertainty [1]. In practice, most physical systems are generally nonlinear and include parametric uncertainty [38 and 41]. These nonlinearities and uncertainty may be caused by saturation of actuators, friction forces, backlashes, aging of components, changes in environmental conditions, or calibration errors [1].

### 2.2 Stabilization of Systems with Input Matrix Uncertainty

Consider a linear system of the form

$$\dot{x}(t) = Ax(t) + (B + \Delta B(r))u(t) \tag{2.1}$$

where $x(t) \in R^n$ is the state, $u(t) \in R^m$ is the control input, $r$ is the uncertain parameter in the control input matrix.

The static state feedback controller of the form

$$u(t) = Kx(t) \tag{2.2}$$

which can stabilize the system with an appropriate value of controller gain ( $K$ ).

Or dynamic state feedback controller of the form

$$\begin{aligned} \dot{z}(t) &= A_c z(t) + B_c x(t) \\ u(t) &= C_c z(t) + D_c x(t) \end{aligned} \tag{2.3}$$

where $z(t) \in R^{n_c}$ is the state of the controller, $A_c, B_c, C_c,$ $D_c$ are constant matrices in appropriate dimension.

If a system is controllable, all the poles of the closed loop system can be placed anywhere in the left half plane, obviously there is nothing more to attain by using dynamic state feedback controller. If the uncertainty parameter is time varying then the stability bound of the system cannot be enhanced by dynamic state feedback controller [24]. If the uncertainty parameter is time invariant then the stability bound of the system can be enhanced by dynamic state feedback controller than static state feedback controller which has been shown in the paper [33]

## 2.3 A Characterization of a Class of Second Order Uncertain System

Considering a second order system with time-invariant uncertain input matrix may be described as

$$\dot{x}(t) = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} x(t) + \begin{bmatrix} \bar{b}_1 \\ b_2 \end{bmatrix} u(t) \tag{2.4}$$

where $x(t) \in R^2$ is the state vector and $u(t) \in R$ is the control input, $A \in R^{2\times 2}$, $B \in R^{2\times 1}$ and $\bar{b}_1 = b_1 + r$ .

To categorize this class of systems whose stability bound can be enhanced by using a dynamic state feedback controller.

For this system (2.4), consider a static state feedback controller of the form

$$u(t) = \begin{bmatrix} k_1 & k_2 \end{bmatrix} x(t) \tag{2.5}$$

The closed loop equation is then given by $\dot{x}(t) = \begin{bmatrix} a_1 + \bar{b}_1 k_1 & a_2 + \bar{b}_1 k_2 \\ a_3 + b_2 k_1 & a_4 + b_2 k_2 \end{bmatrix} x(t)$ (2.6)

The eqn. (2.6) is in the form of, $\dot{x}(t) = \overline{A}x(t)$. The characteristic equation can be represented by $\left|sI - \overline{A}\right| = 0$, which yields

$$s^2 - s\left\{\overline{b}_1 k_1 + b_2 k_2 + (a_1 + a_4)\right\} + \left\{(a_4\overline{b}_1 - a_2 b_2)k_1 + (a_1 b_2 - a_3\overline{b}_1)k_2 + (a_1 a_4 - a_2 a_3)\right\} = 0 \qquad (2.7)$$

From the above characteristic equation (2.7), one may derive the following conditions to satisfy the stability criteria of the system.

$$\left\{\overline{b}_1 k_1 + b_2 k_2 + (a_1 + a_4)\right\} < 0 \qquad (2.8)$$

$$\left\{(a_4\overline{b}_1 - a_2 b_2)k_1 + (a_1 b_2 - a_3\overline{b}_1)k_2 + (a_1 a_4 - a_2 a_3)\right\} > 0 \qquad (2.9)$$

The above equations (2.8) and (2.9) will contradict each other, when coefficients of the equations are equal with opposite sign of $r$ (uncertainty factor) in both the sides to give us the range of $r$ for which the system will be stable.

Now, to satisfy the method of contradiction, one obtains

$$b_1 + r = (b_1 - r)a_4 - a_2 b_2 \qquad (2.10)$$

$$b_2 = a_1 b_2 - a_3(b_1 - r) \qquad (2.11)$$

$$a_1 + a_4 = a_1 a_4 - a_2 a_3 \qquad (2.12)$$

From these above conditions, we derive a set of conditions such that a given system may be checked whether a dynamic state feedback controller may yield better stability performance compared to the static one or not. These are presented in next section.

### 2.3.1 When this Proof of Contradiction Applies?

**Algorithm 2.1**

The following are the conditions for which the method of contradiction applies.

1: $Trace(A) = Det(A)$

2: $a_4 = +1$, (This is not that restrictive as it appears since one can always interchange the state definitions, i.e., can interchange $a_1$ and $a_4$).

3: $a_2 \neq \dfrac{a_4 + 1}{a_4 - 1 - a_3(a_4 + 1)}$.

4: Finally, if $a_1 = \dfrac{a_4 + a_2 a_3}{a_4 - 1}$ and $b_2 = \dfrac{2a_3 b_1}{(a_1 - 1)(a_4 + 1) - a_2 a_3}$.

Note that, if the system is in controllable canonical form, this enhancement may not be possible using a dynamic state feedback controller.

**Remark 2.1:** If the system satisfies the above condition then one may obtain a sufficient tolerable uncertainty bounds for any static state feedback controller by following the method of contradiction as

$$|r| < \frac{b_1(a_4 - 1) - a_2 b_2}{1 + a_4}$$

Following the above conditions, several examples may be developed for which the tolerable uncertainty bound by using static state feedback controller can be obtained.

For the same examples, one may also search for a dynamic feedback controller that improves the tolerable uncertainty bound. It is seen that the dynamic feedback controllers are able to improve the tolerable bound. Some numerical examples are developed to validate the above proposed algorithm.

Next, we present a lemma that will be used to determine the tolerable uncertainty bounds for a given controller case.

**Lemma 2.1** [8]: Given two appropriate dimensional matrices $M_0$ and $M_1$ with $M_0$ is Hurwitz. Then any matrix belonging to the set of this matrices

$$M = \{M_r = M_0 + rM_1, r \in [r_{\min}, r_{\max}]\} \tag{2.13}$$

remains stable for any $r \in \left(r_{\min} \quad r_{\max}\right)$, where

$$r_{\min} = \frac{1}{\lambda_{\min}(-(M_0 \oplus M_0)^{-1}(M_1 \oplus M_1))} \tag{2.14}$$

$$r_{\max} = \frac{1}{\lambda_{\max}(-(M_0 \oplus M_0)^{-1}(M_1 \oplus M_1))} \tag{2.15}$$

where $\lambda_{\max}(X)$ will denote the maximum positive real eigenvalue of $X$ (if $X$ has no positive eigenvalue then $\lambda_{\max}(X) = 0^+$) and $\lambda_{\min}(X)$ denotes the minimum real eigenvalue of $X$ (if $X$ has no negative eigenvalue then $\lambda_{\min}(X) = 0^-$).

## 2.4 Numerical Examples

In this section, we demonstrate two new examples for which the method of contradiction applies and there exists at least a dynamic state feedback controller that improves the tolerable uncertainty bound.

**Example 2.1:** Considering a second order system with constant uncertainty with input control matrix as

$$\dot{x}(t)=\begin{bmatrix}1 & -1\\ 1 & 2\end{bmatrix}x(t)+\begin{bmatrix}-1+r\\ -2\end{bmatrix}u(t) \tag{2.16}$$

Appling a static state feedback controller of the form $u(t)=k_1x_1(t)+k_2x_2(t)$, the closed loop equation may be written as

$$\begin{aligned}\dot{x}(t)&=\begin{bmatrix}1 & -1\\ 1 & 2\end{bmatrix}x(t)+\begin{bmatrix}-1+r\\ -2\end{bmatrix}\begin{bmatrix}k_1 & k_2\end{bmatrix}\\ &=\begin{bmatrix}1+(-1+r)k_1 & -1+(-1+r)k_2\\ 1-2k_1 & 2-2k_2\end{bmatrix}x(t)\end{aligned} \tag{2.17}$$

The characteristic equation corresponding to system (2.17) is

$$s^2+s\left\{k_1(1-r)+2k_2-3\right\}-\left\{k_1(4-2r)+k_2(1+r)-3\right\}=0 \tag{2.18}$$

For stability of the system, it requires that all the coefficient of $s^i, i=0,1,2$ must be positive, then one obtains

$$\left\{k_1(1-r)+2k_2-3\right\}>0 \tag{2.19}$$

and
$$\left\{k_1(4-2r)+k_2(1+r)-3\right\}<0 \tag{2.20}$$

At $r=-1$, from eqn. (2.19), one obtains $\left\{2k_1+2k_2-3\right\}>0$ (2.21)

Similarly, from eqn. (2.20), one obtains $\left\{6k_1-3\right\}<0$ (2.22)

Again, at $r=1$, from eqn. (2.19), it requires that $\left\{2k_1-3\right\}>0$ (2.23)

and from eqn. (2.20) $\left\{2k_1+2k_2-3\right\}<0$ (2.24)

Note from the above stability requirements that the equations (2.21) and (2.24) are contradicting each other so the system can be stabilized by a static feedback controller only when $|r| < 1$.The above system (2.16) has been simulated at different uncertain cases and it has been observed from the result that system is stable at $r = 0, 0.9$ and $1$ but unstable for a chosen $r = 1.1$ which validates the computed result. The simulation results are shown in the Fig.2.1.

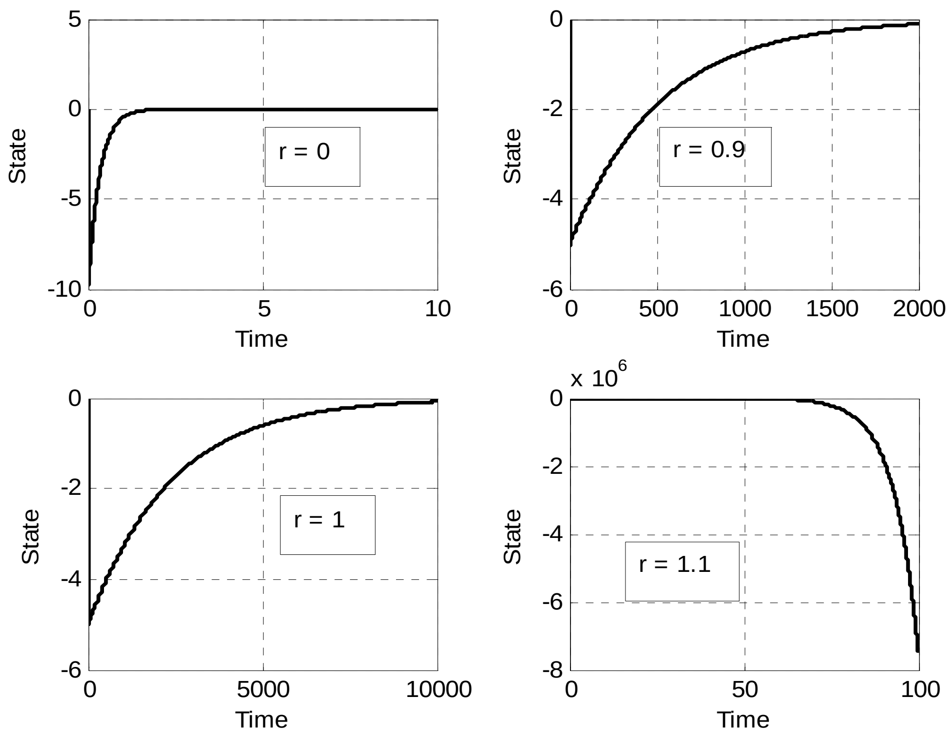


Fig 2.1: Simulation for the Stability of System (2.16) Using Static State Feedback Controller

Next, consider a dynamic state feedback controller for system (2.25), whose parameters are computed by using *fminsearch* program of MATLAB. The controller parameters are searched so that the most right hand eigenvalue of the system for $\mathrm{r} = 0, 1$ and $-1$ get placed to LHS of the complex plane.

By this, we obtain a dynamic feedback controller as,

$$\begin{aligned} \dot{z}(t) &= -122.5501x_1(t) + 12.06766x_2(t) - 2.3592z(t) \\ u(t) &= -40.8416x_1(t) + 51.1517x_2(t) + 0.8173z(t) \end{aligned} \tag{2.25}$$

The closed loop system of (2.16) using dynamic feedback controller (2.25) may then be represented as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 1 & -1 & 0 \\ 1 & 2 & 0 \\ -122.5501 & 12.0676 & -2.3592 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ z \end{bmatrix} + \begin{bmatrix} -1+r \\ -2 \\ 0 \end{bmatrix} \begin{bmatrix} -40.8416 & 51.1517 & 0.8173 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ z \end{bmatrix}$$

$$= \begin{bmatrix} 1 & -1 & 0 \\ 1 & 2 & 0 \\ -122.5501 & 12.0676 & -2.3592 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ z \end{bmatrix} + \begin{bmatrix} -(-1+r)*40.8416 & (-1+r)*51.1517 & (-1+r)*0.8173 \\ -40.8416*(-2) & 51.1517*(-2) & 0.8173*(-2) \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ z \end{bmatrix}$$

$$= \begin{bmatrix} 1 & -1 & 0 \\ 1 & 2 & 0 \\ -122.5501 & 12.0676 & -2.3592 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ z \end{bmatrix} + \begin{bmatrix} 40.8416 & -51.1517 & -0.8173 \\ 81.6832 & -102.3034 & -1.6346 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ z \end{bmatrix} + r \begin{bmatrix} -40.8416 & 51.1517 & 0.8173 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ z \end{bmatrix}$$

Then, the closed loop equation may be written as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 41.8416 & -52.1517 & -0.8173 \\ 82.6832 & -100.3033 & -1.6346 \\ -122.5501 & 12.0676 & -2.3592 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ z \end{bmatrix} + r \begin{bmatrix} -40.8416 & 51.1517 & 0.8173 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ z \end{bmatrix} \quad (2.26)$$

Now, the range of the tolerable uncertainty $r \in \left( r_{\min} \quad r_{\max} \right)$ may be determined using Lemma 2.1. For the purpose, we express the above closed loop system (2.26) as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{z} \end{bmatrix} = M \begin{bmatrix} x_1 \\ x_2 \\ z \end{bmatrix}$$

where $M$ is a set of perturbed matrices and can be expressed as

$$M = M_0 + rM_1 \tag{2.27}$$

where $M_0$ and $M_1$ are $n \times n$ matrices with $M_0$ strictly stable . The maximal range of $r$ for $M$ to be strictly stable using Lemma 2.1 is obtained as $r_{\min} = -1.1709$, $r_{\max} = 1.2655$ and $|r| < 1.1709$.Then we have simulated the closed loop system (2.26) at different uncertain cases and it has been observed that the system is stable at $r = -1.17$ to $1.17$ which has been shown in the Fig.2.2.

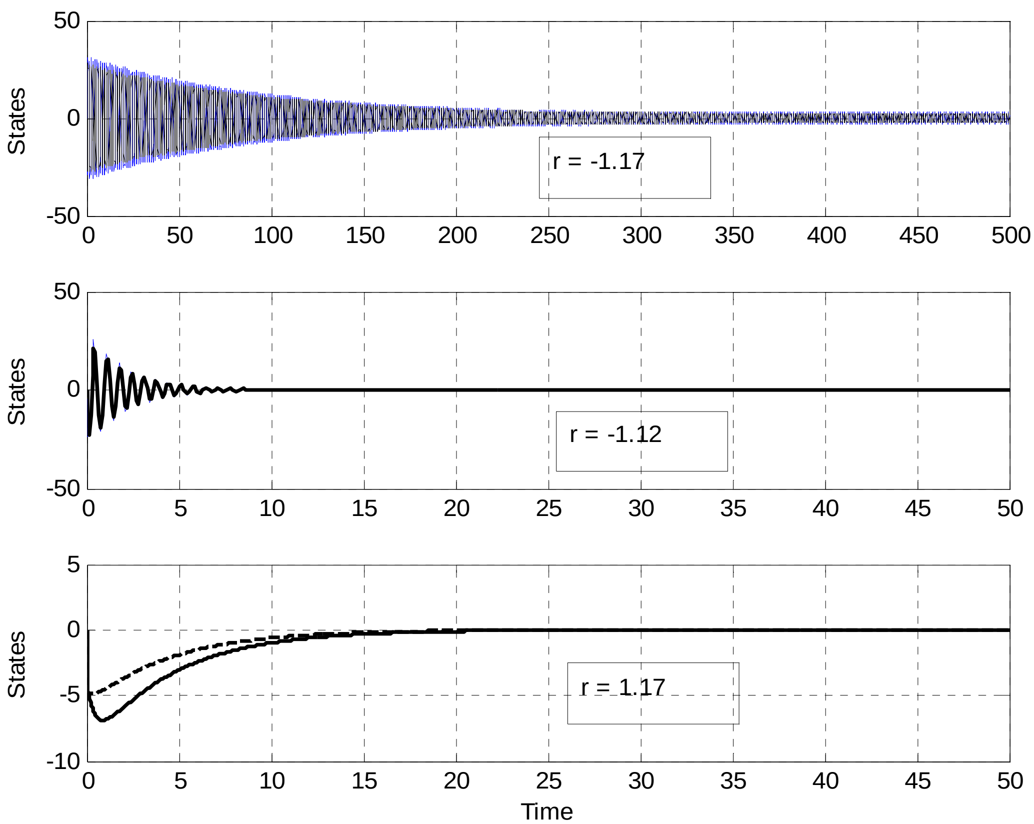


Fig 2.2: Simulation for the Stability of System (2.16) Using dynamic feedback controller (2.25)

Hence, the dynamic controller enhances the stability bound than that of any static state feedback controller.

**Example 2.2:** We have considered another example to justify more appropriately that by referring the above algorithm, one can develop new examples,

$$\dot{x}(t) = \begin{bmatrix} 1 & -1 \\ 1 & 3 \end{bmatrix} x(t) + \begin{bmatrix} 1+r \\ 2 \end{bmatrix} u(t) \tag{2.28}$$

The above system obeys the method of contradiction for $|r| \leq 1$. Hence, the system cannot be stabilized by a static state feedback controller for $|r| > 1$. Then, the closed loop system of system (2.28) using static state feedback controller is simulated at different cases and observed that the system is stable at $r = 0, 0.9$ and $1$ and unstable at $r = 1.1$.

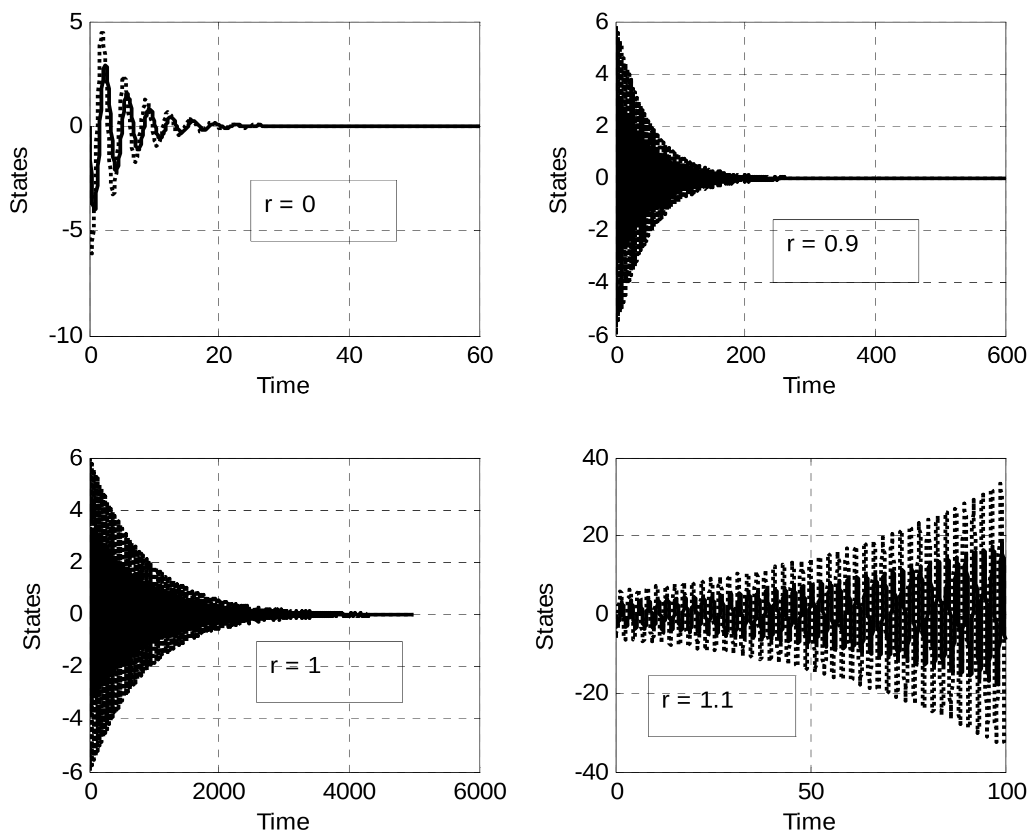


Fig 2.3: Simulation for the stability of system (2.28) using static feedback controller

However, consider a dynamic state feedback controller obtained using fminsearch program as

$$\begin{aligned} \dot{z}(t) &= -1.4294x_1(t) + 9.9279x_2(t) - 4.5957z(t) \\ u(t) &= 25.0799x_1(t) - 29.8277x_2(t) + 5.9040z(t) \end{aligned} \tag{2.29}$$

The closed loop equation may be represented in the form (2.27) as:

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 26.0799 & -30.8277 & 5.9040 \\ 51.1597 & -56.6554 & 11.8081 \\ -1.4294 & 9.9279 & -4.5957 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ z \end{bmatrix} + r \begin{bmatrix} 25.0799 & -29.8277 & 5.9040 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ z \end{bmatrix} \tag{2.30}$$

the tolerable uncertainty bound using Lemma 2.1 is obtained as: $r_{\min} = -1.1882$, $r_{\max} = 1.1187$ and $|r| < 1.1187$. We have simulated the closed loop system (2.30) and observed that the above system is stable at $r = -1.11$ to $1.11$. The simulated results are shown in the Fig.2.4.

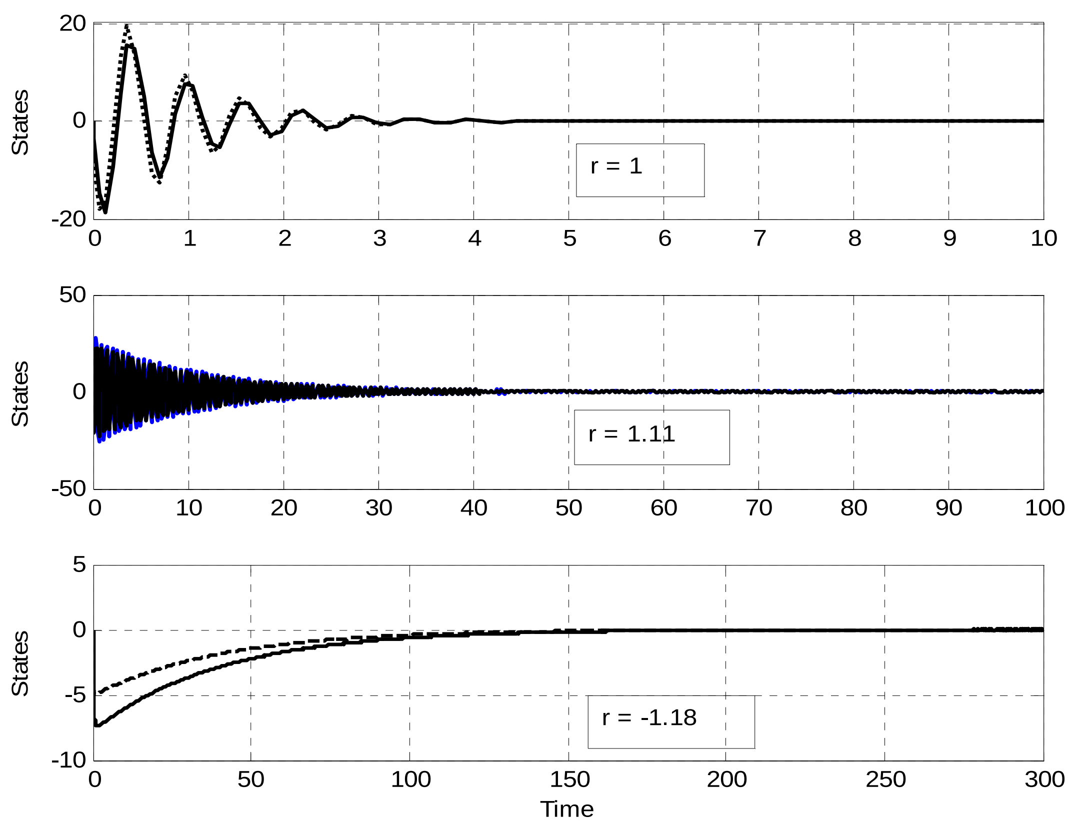


Fig 2.4: Simulation for the stability of system (2.28) using dynamic feedback controller (2.29)

Clearly this controller improves the tolerable uncertainty bound compared to that tolerable by using any static feedback one.

Corresponding to the above second order system, one may develop a third order system with uncertainty in the system matrix for which the enhancement in tolerable uncertainty bound is possible. To define it more appropriately, we have presented an example case of this class of third order system.

**Example 2.3**: Consider a third order system with uncertainty in the system matrix as

$$\dot{x}(t) = \begin{bmatrix} 1 & -1 & -1+r \\ 1 & 2 & -2 \\ 0 & 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u(t) \tag{2.31}$$

where $r$ is an uncertain parameter.

Appling a static state feedback controller of the form $u(t) = k_1 x_1(t) + k_2 x_2(t) + k_3 x_3(t)$, it is seen that the system cannot be stabilized by a static feedback controller for $|r| \geq 1$. The system (2.31) is simulated using static state feedback controller gain values for different uncertain cases. The corresponding simulation results are shown in Fig.2.5. From this, one may see that the system is stable at $r = 0, 0.9$ and $1$ but unstable for a chosen $r \geq 1$.

However, considering a dynamic state feedback controller of the form

$$\begin{aligned} \dot{z}(t) &= -1.3562 x_1(t) + 0.8641 x_2(t) + 0.8279 x_3(t) + 0.6120 z(t) \\ u(t) &= 1.5051 x_1(t) + 4.5800 x_2(t) - 3.7494 x_3(t) + 0.3555 z(t) \end{aligned} \tag{2.32}$$

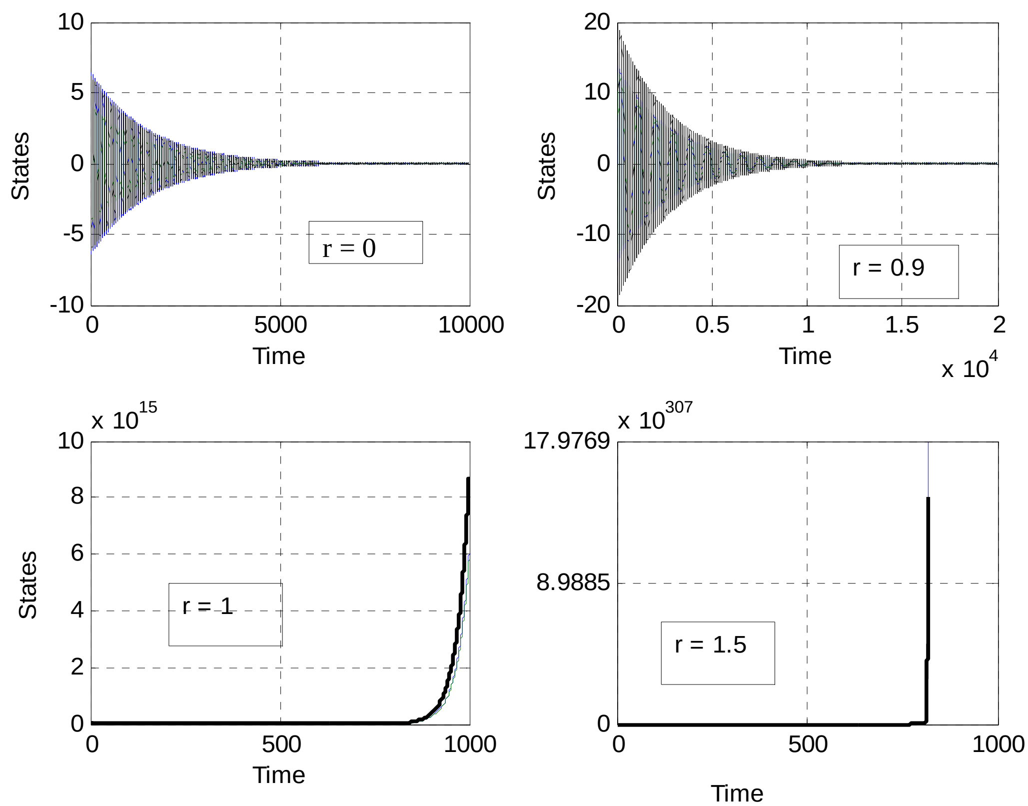


Fig 2.5: Simulation for the stability of system (2.31) using static feedback controller

the closed loop system of (2.31) using (2.32) may be written in the augmented form as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 1 & -1 & -1+r & 0 \\ 1 & 2 & -2 & 0 \\ 0 & 0 & 0 & 0 \\ -1.3562 & 0.8641 & 0.8279 & 0.6120 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ z \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1.5051 & 4.5800 & -3.7494 & 0.3555 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ z \end{bmatrix}$$

$$= \begin{bmatrix} 1 & -1 & -1+r & 0 \\ 1 & 2 & -2 & 0 \\ 0 & 0 & 0 & 0 \\ -1.3562 & 0.8641 & 0.8279 & 0.6120 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ z \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1.5051 & 4.5800 & -3.7494 & 0.3555 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ z \end{bmatrix}$$

The closed loop system may be represented into the desired form as

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 1 & -1 & -1 & 0 \\ 1 & 2 & -2 & 0 \\ 1.5051 & 4.5800 & -3.7494 & 0.3555 \\ -1.3562 & 0.8641 & 0.8279 & 0.6120 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ z \end{bmatrix} + r \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ z \end{bmatrix}$$

For this case, the tolerable uncertainty bound using Lemma 2.1 is obtained as: $r_{\min} = -21.3666$, $r_{\max} = 1.6698$ and $|r| < 1.6698$. The system (2.31) is simulated using dynamic state feedback controller (2.32) for different uncertain cases. This simulation results are shown in Fig.2.6. From this simulation one may observe that the system is stable at $r = -1.11, 1.11$ and $-21.3$.

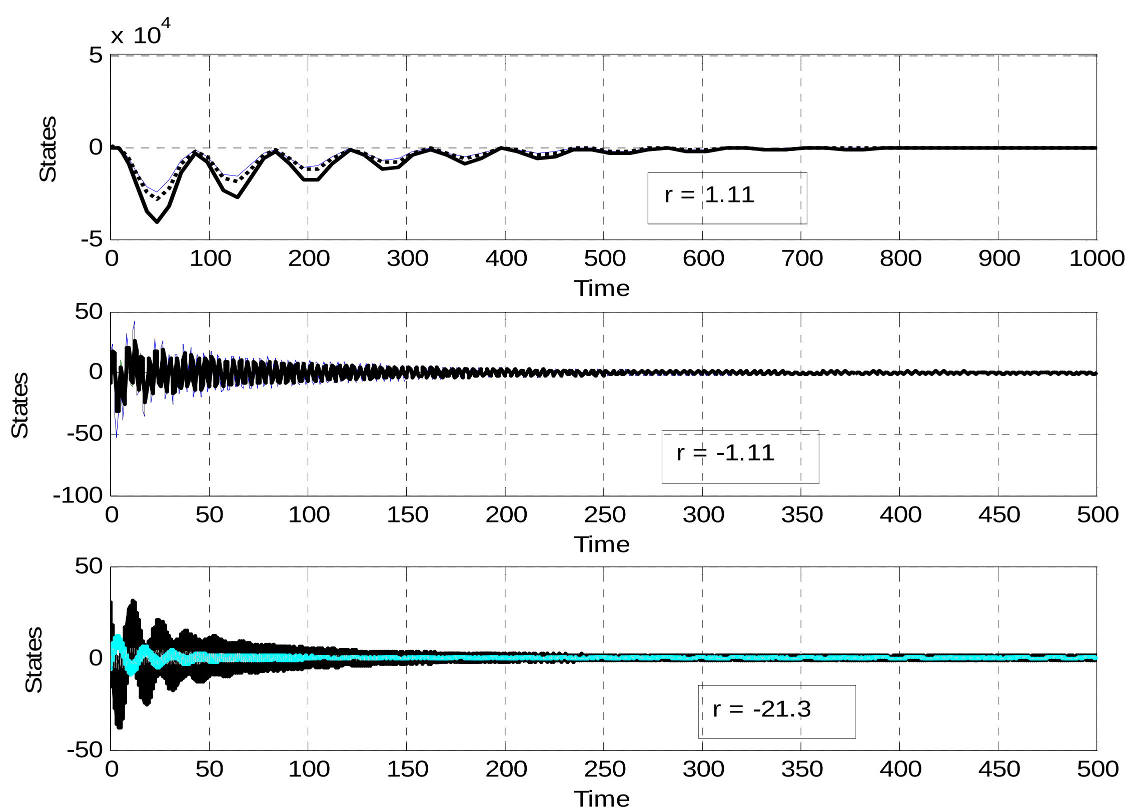


Fig 2.6: Simulation for the stability of system (2.31) using dynamic feedback controller (2.32)

Note that, the tolerable uncertainty bounds obtained using dynamic feedback controller are not the optimal one since one may design a different controller which may yield larger bounds than the present one.

**2.5 Conclusion**

The class of second order uncertain systems with time-invariant uncertainty in the input has been identified for which tolerable uncertainty bound may be improved by using suitable dynamic feedback controller. Two new examples of second order system have been demonstrated for which it has been seen that dynamic feedback controller indeed enhances the tolerable uncertainty bound. Correspondingly, a new example of third order system has been developed for which a dynamic state feedback controller may improve the tolerable uncertainty bound compared to the static one. However, searching the parameters of dynamic state feedback controller by *fminsearch* program of MATLAB is a heuristic approach because there is no systematic procedure is available to obtain the parameters of such a dynamic state feedback controller.

# Chapter 3

# Stabilization of an Input Delayed System

## 3.1 Introduction

Input delay often arises in engineering systems due to time-taken in communicating signal from one to other place. In feedback control systems, delay appear due to time taken in measuring the output signals, called as measurement delay or sensor delay, and activating the actuator, called as actuation delay. There may also be computational delay due to the time taken in computing the control law. It is well known that the existence of time-delay degrades the controller performance and even instability of the closed loop system [4]. In this chapter, the problem of stabilizing systems while maximizing such delays in the feedback loops is considered. To proceed further, the system under consideration is described in the next section.

## 3.2 Delays in the Feedback Loop

Due to finite time taken in sending (i) measured outputs from the sensor to the controller and (ii) from the controller to the actuator of the plant, the feedback loop may involve time-delay [11 and 22]. These delays may cause instability of the system [7 and 29]. A closed-loop control system with such delays in the feedback loop is shown in Fig. 3.1. The time-delay involved in measuring and thereby sending an output signal to the controller is represented by $\tau_s$, whereas $\tau_a$ represents the time taken to send the control signal from the controller to the actuator of the plant. The former one may be referred to as the sensing (measurement) delay and the latter one as the actuation delay.

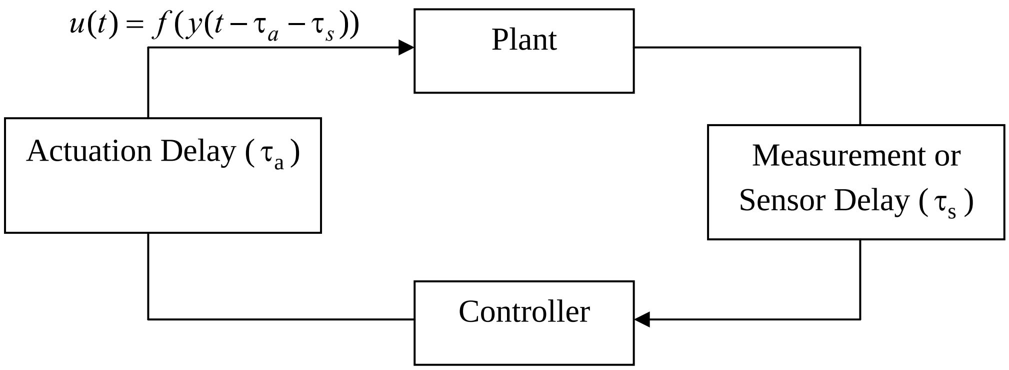


Fig 3.1: Feedback Control System with Transmission Delays

The total delay in the feedback path for static feedback case is the sum of these two delays, i.e., $\tau_{total} = \tau_a + \tau_s$ [15].

Now, consider the first order plant dynamics as:

$$\dot{x}(t) = ax(t) + bu(t - \tau_a) \tag{3.1}$$

where $x(t)$ is the state and $u(t)$ is the control input with $\tau_a$ being the actuation delay.

### 3.3 An Observation

In this section, a simple test is presented that leads to further work on the stabilization problem considered in this chapter.

#### 3.3.1 Methodology for the Test

To obtain an optimal controller in the sense that the tolerable delay bound is maximum, the controller parameters are searched using the *fminsearch* program of MATLAB® while computing the delay values using Frequency Sweeping Test (FST) [12].

To start with, first, note that, with known values of the controller parameters irrespective of static or dynamic, the closed loop system may be represented as

$$\dot{x} = Ax + A_d x(t-\tau) \tag{3.2}$$

where $A$ and $A_d$ are known matrices of appropriate dimension. Next, the static and dynamic feedback cases are treated individually.

### 3.3.2 The Case of Static State Feedback Controller

Considering a static state feedback controller with sensing delay $\tau_s$ as

$$u(t) = kx(t-\tau_s) \tag{3.3}$$

with $k$ is the controller gain, the closed loop dynamics becomes

$$\dot{x}(t) = ax(t) + bkx(t-\tau_a-\tau_s) \tag{3.4}$$

where $A_d = bk$ .

Note that, for stability of the closed loop system (3.2), it is required to be stable at $\tau = 0$, i.e., $A + A_d$ is stable. Now, if one increases $\tau$ from zero value then the system will remain stable up to a maximum value of the delay ( $\tau_{\max}$ ). Beyond $\tau_{\max}$ , the system will be unstable [12].

Next, we consider investigating stability of such delay systems.

The characteristic equation of (3.2) is

$$\det(\lambda I - A - A_d e^{-\lambda\tau}) = 0 \tag{3.5}$$

Note that due to the presence of the exponential term, the above equation has infinite number of roots, i.e., the system has infinite number of poles. Clearly, it is difficult to analyze stability considering all the poles at a time by finite computations [12, 27 and 28]. However, several methods are available for analyzing stability of time-delay systems by using computational algorithms or by approximating the infinite dimensional system to a finite dimensional one, e.g., using PADE approximation [31]. In this paper, we will use a FST for computing the tolerable delay margin for the time-delay systems. The procedure to compute the maximum tolerable delay using frequency sweeping test is presented in the Appendix-A.

**Example 3.1:** For static state feedback case, the closed-loop dynamics will be $\dot{x} = ax + b_k x(t-\tau)$ .For $a = 2$ and $b_k = -2.1$, the tolerable value of delay using FST is obtained as $\tau = 0.4839$ .In this case, the well known result is that the system is stabilizable with static state feedback for $\tau < 1/a$ [29]. Hence, in this case, the system is stabilizable for $\tau < 0.5$ .

### 3.3.3 The Cases of Dynamic State Feedback Controller

Now, consider a first order dynamic feedback controller of the form

$$\dot{x}_c(t) = a_c x_c(t) + b_c x(t-\tau_s) \tag{3.6}$$

If a dynamic feedback controller is used, the following two cases may be studied.

**Case I:** The case when, $\tau_s = 0$ and $\tau_a = \tau$ :

The closed loop equation may be represented by

$$\begin{aligned} \dot{x}(t) &= ax(t) + bx_c(t-\tau) \\ \dot{x}_c(t) &= a_c x_c(t) + b_c x(t) \end{aligned} \tag{3.7}$$

The augmented form of this system may be

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} a & 0 \\ b_c & a_c \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & b \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ x_c(t-\tau) \end{bmatrix} \tag{3.8}$$

**Example 3.2:** For the case, when $a = 2$ and $b = 1$, the $b_c$ and $a_c$ are searched by *fminsearch* method and tolerable delay is computed by FST. The system may be represented as

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 1e+004*(-2.4262) & 1e+004*(-4.8525) \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ x_c(t-\tau) \end{bmatrix}$$

The value of $\tau$ is found to be 0.5. So, the total tolerable delay ( $\tau_{total} = \tau_s + \tau_a$ ) is 0.5.

**Case II:** The case when, $\tau_s = \tau_a = \tau \neq 0$ :

The closed loop equation may be represented by

$$\begin{aligned} \dot{x}(t) &= ax(t) + bx_c(t-\tau) \\ \dot{x}_c(t) &= a_c x_c(t) + b_c x(t-\tau) \end{aligned} \tag{3.9}$$

The augmented form of this system may be

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & a_c \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & b \\ b_c & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ x_c(t-\tau) \end{bmatrix} \tag{3.10}$$

**Example 3.3:** For this case,

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 1.0e+004*(-2.6063) \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 1.0e+004*(-1.3029) & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ x_c(t-\tau) \end{bmatrix}$$

The value of $\tau_s$ and $\tau_a$ are found to be 0.25.So, the total delay ( $\tau_{total}$ ) is approximately 0.5.

It is observed from the above experimentation that neither the static feedback nor the conventional dynamic feedback controller is able to stabilize the system for $\tau > 0.5$. However, we have shown in the following that if we introduce an artificial delay in the state of the dynamic feedback controller then the tolerable delay margin is comprehensively improved.

### 3.3.4 A New Dynamic Feedback Controller with an Artificial Delay

If one considers a dynamic feedback controller with a state delay as

$$\begin{aligned} \dot{x}_c(t) &= a_c x_c(t-\tau_c) + b_c x(t-\tau_s) \\ u(t) &= x_c(t-\tau_a) \end{aligned} \tag{3.11}$$

and again search for the controller parameters $a_c$ and $b_c$. A representative block-diagram of the closed loop system with all the delays in the feedback network is shown in the Fig.3.2.

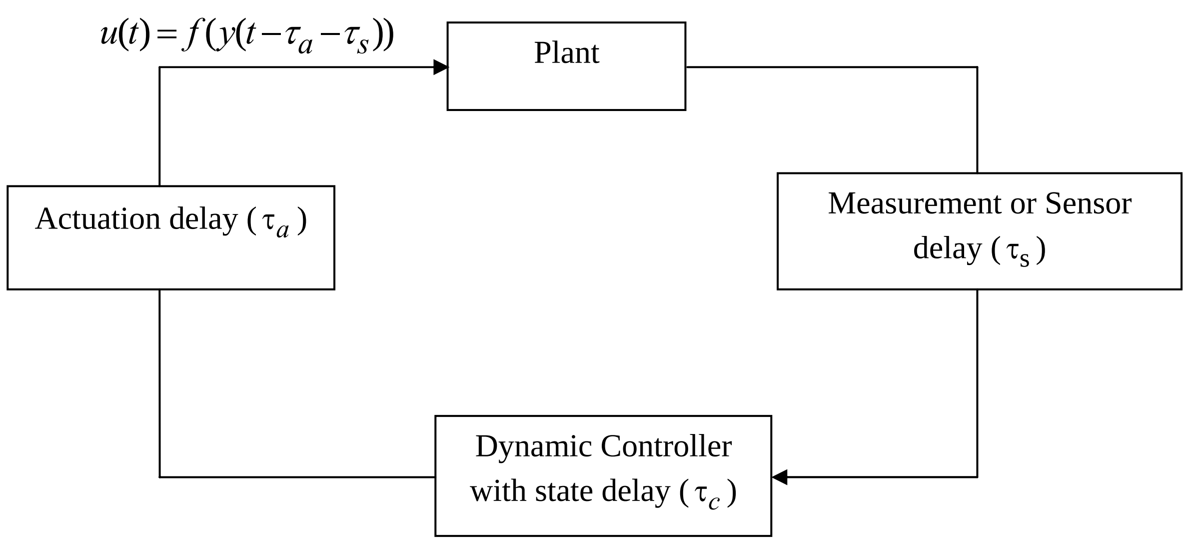


Fig 3.2: Feedback Control System with an Artificial Delay in the Controller.

Considering the controller (3.11) with $\tau_s = \tau_a = \tau_c = \tau \neq 0$ the dynamics of the closed loop system may be represented as

$$\begin{aligned} \dot{x}(t) &= ax(t) + bx_c(t-\tau) \\ \dot{x}_c(t) &= a_c x_c(t-\tau) + b_c x(t-\tau) \end{aligned} \tag{3.12}$$

In the matrix form, the system along with the dynamic controller may be represented as:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & b \\ b_c & a_c \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ x_c(t-\tau) \end{bmatrix} \tag{3.13}$$

Again considering the previous case for $a = 2$, we may check whether there is an enhancement in the total tolerable delay margin in the network due to our proposed controller or not. To compute the total tolerable delay, we have used the FST and to compute the controller parameter *fminsearch* program in MATLAB® is used.

**Example 3.4:** Considering $a = 2$ and $b = 1$.The controller parameters are searched to be -6.9719 and -13.9439 by *fminsearch* method. Then, the closed system may be

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ -6.9719 & -13.9439 \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ x_c(t-\tau) \end{bmatrix} \tag{3.14}$$

For this case, the value of $\tau$ is found to be 0.3565. Hence, the total tolerable delay ($\tau_{total}$) is 0.7130.

Note that, the total tolerable delay has been enhanced to 0.7130 by using our proposed dynamic state feedback controller, which is nearly 40% enhancement than the existing result.

**Remark 3.1:** For case-I, the eigenvalues of $A_d$ matrix are present at the origin. For case-II, all the eigenvalues of the matrix $A_d$ are on the imaginary plane that means the closed loop system is stable because of matrix $A$ but in case of our proposed controller the eigenvalues of $A_d$ matrix are present in left half plane of the imaginary axis that's why these eigenvalues have more influencing effect on $A$.

Several questions arise from the above study. These are (i) how the tolerable delay margin varies with $\tau_c$ and (ii) what happens if the sensor and actuation delays are not equal and what should be the choice of the $\tau_c$ in that case? Next, we attempt to answer these questions. For the purpose, note that, the above approach FST to compute the maximum tolerable delay margin along with the *fminsearch* program to obtain the controller parameters is heuristic. So, we next use the continuous pole placement method which is used to design controller parameters algorithmically.

### 3.4 A Continuous Pole Placement Method for Time-Delay Systems [29]

This is a numerical stabilization method for delay differential equation which is related to the classical pole-placement method for ordinary differential equations to stabilize a system, all the eigenvalues must be placed in the left half of the imaginary axis by bringing small changes in the feedback gain. In case of a delay system there are infinite number of poles, if we can check the location of the right most eigenvalue, then we can define the stability of the system. An unstable time-delay system may be stabilized by changing the feedback gain, bringing the right most eigenvalues to the left half of the imaginary axis and at the same time the movement of other uncontrolled eigenvalues must be taken care. This is the method by means of the stabilization of a linear: finite dimensional system in the presence of an actuation delay (input delay), sensor delay (output delay) and delay in the feedback controller itself. The unstable poles are controlled by procedural steps, which are shown below.

**Algorithm 3.1**

Step 1: Set how many eigenvalues to be controlled.

Step 2: Initialize the gain value, eigenvector.

Step 3: Calculate the right most eigenvalues at nominal value of $\tau$ .

Step 4: Calculate the corresponding eigenvector, ( $v_i$ ).

Step 5: Calculate the normalization function, ( $n(v_i)$ ).

Step6: Calculate the sensitivity function.

Step 7: Compute the desirable change in the controller parameters and update the gain.

Step 8: Check the real part of the right most eigenvalue.

Step 9: If the real part of the rightmost eigenvalue is greater than zero then go to Step 3.

Step 10: Else stop

**Flow Diagram for Continuous Pole Placement Technique:**

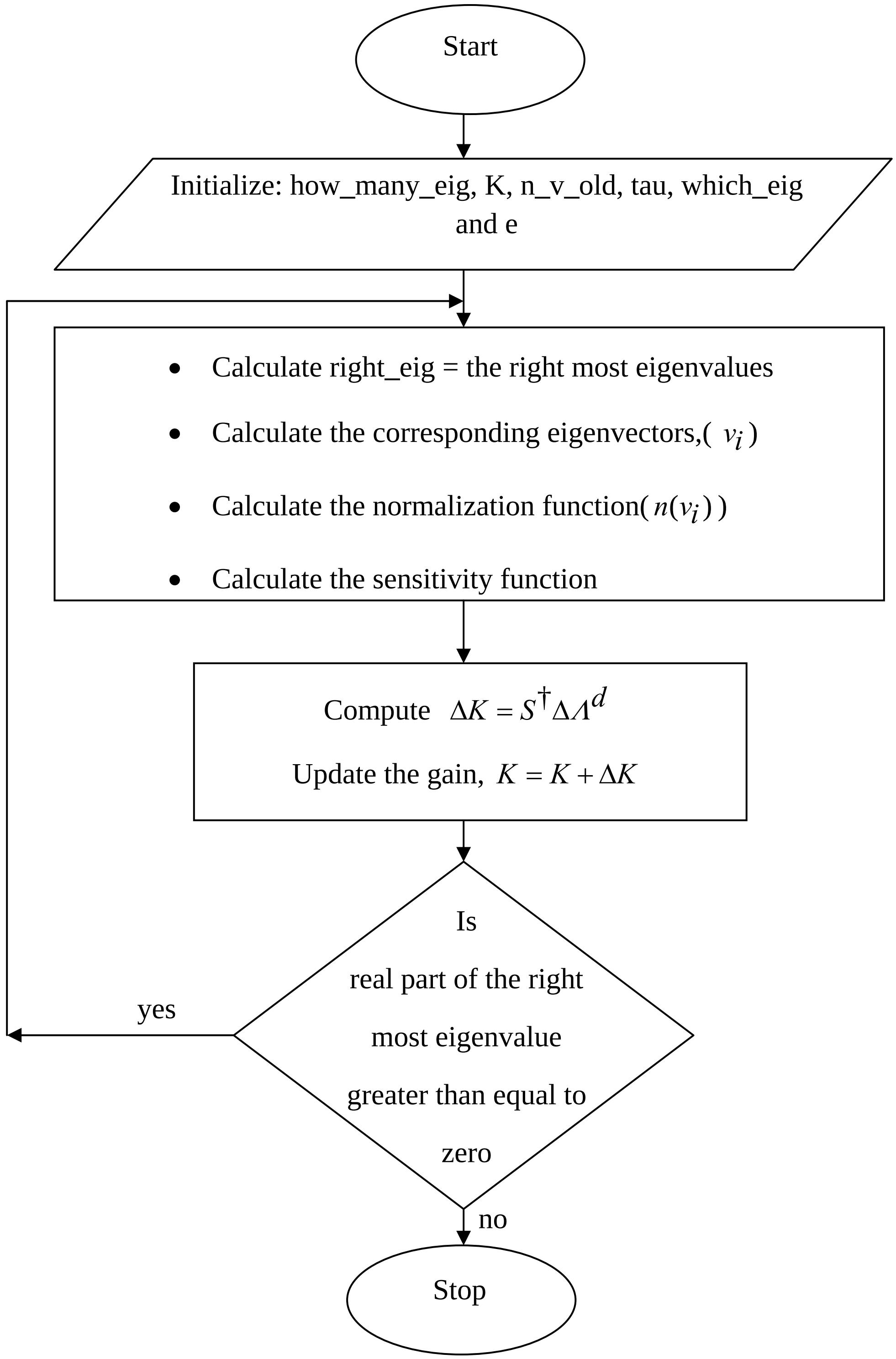


Fig 3.3: Flow Diagram for Continuous Pole Placement Technique.

Note: how_many_eig=No. of eigenvalue to be controlled.

K=gain of the controller.

right_eig=Right most eigen value.

n_v_old=old eigen vector.

n=no. of row the gain vector.

m=No. of column of the gain vector.

e=unity vector.

**Remark 3.2:** Note that, as per our experience, for faster convergence of the algorithm, the desired displacement of the controlled poles may be chosen to be dynamic and proportional to the real part of the rightmost pole. It indicates that when the poles are far away then convergence will be faster and it will converge slowly once it comes closer to the origin.

### 3.4.1 Computation of the Right Most Eigenvalues using BIFTOOL

Engelborghs and Roose proposed a method which automatically computes the rightmost eigenvalues of the characteristic equation of a Time-Delay system [29]. First, a discretization is obtained of the time integration operator of the linear or linearized system of Delay Differential Equations, whose eigenvalues are exponential transforms of the roots of the characteristic equation. Then, selected eigenvalues of the resulting large matrix are computed. A step length heuristic is used to ensure that all eigenvalues of interest are accurately approximated by the discretization. Accuracy can be increased by employing Newton iteration on the characteristic equation using the approximate eigenvalues as starting values. This method has been implemented in the Matlab package DDE-BIFTOOL, proposed by K. Engelborghs.

DDE-BIFFTOOL v.2.00 package is a collection of matlab routines for bifurcation analysis of a delay system. We have used this tool to locate the right most eigenvalues of the closed loop system with feedback delays. From the location of the rightmost poles of the closed loop system, one can analyze the stability of the system, if the rightmost pole is in the left-hand side of the imaginary axis of the s-plane then the system is stable. A brief idea for using BIFFTOOL is given in Appendix-B.

## 3.5 Results and Analysis

### 3.5.1 The Case of Equal Delays ($\tau_a = \tau_s = \tau_c$)

The system with dynamic feedback controller, when all the delays in the closed loop system, measurement delay or sensor delay ($\tau_s$), actuation delay ($\tau_a$) and controller delay ($\tau_c$) are equal. The closed loop system (3.12) may alternatively be represented as:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & b \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ x_c(t-\tau) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} k_1 & k_2 \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ x_c(t-\tau) \end{bmatrix} \tag{3.15}$$

Considering $a = 2$ and $b = 1$, the closed loop system (3.15) may be represented as

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ x_c(t-\tau) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ k_1 & k_2 \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ x_c(t-\tau) \end{bmatrix} \tag{3.16}$$

For computational approach, we have defined the system (3.16) as

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ k_1 & k_2 \end{bmatrix} \begin{bmatrix} x(t-\tau) \\ x_c(t-\tau) \end{bmatrix} \tag{3.17}$$

where $A = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}$ and $BK = \begin{bmatrix} 0 & 1 \\ k_1 & k_2 \end{bmatrix}$

Using BIFTOOL the right most poles of the system (3.17) are located at a particular value of delay ($\tau$). To check the convergence of our algorithm, we have set the value of $\tau = 0.3565$.

Initially, when there is no gain adjustment; one pole is unstable, which is present in the right half of the imaginary axis. The unstable rightmost pole is shown in the Fig.3.4 below. By following the above algorithm 3.1, the gain parameters are adjusted by computing the sensitivity of the eigenvalues. The Computational procedure for sensitivity is given in the next section below.

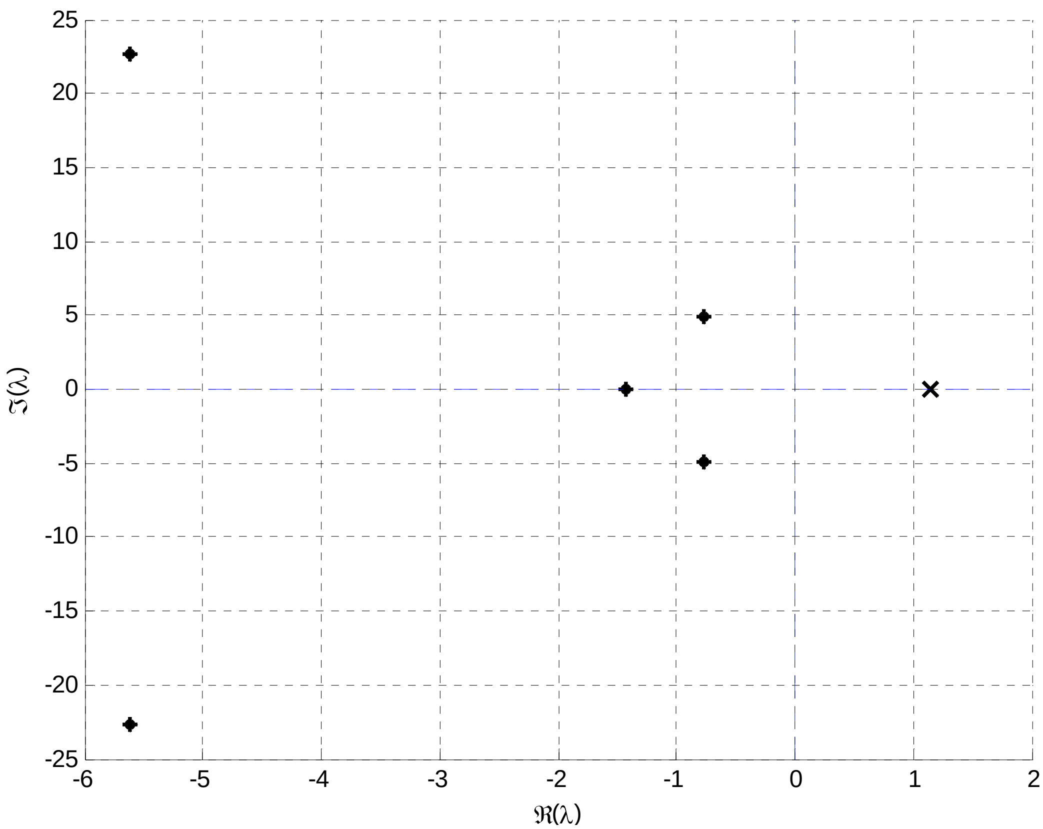


Fig.3.4:The Lacation of Pole when there is no Gain Adjustment

### 3.5.1.1 Computing Sensitivity of Eigenvalues w.r.t. the Feedback Gain Parameters

By making small changes in feedback gain, we may control the right most eigenvalue, but there is a possibility for other eigenvalues to be uncontrolled. So, we have to compute the sensitivity of the eigenvalues w.r.t. the changes in the feedback gain.

From the system (3.15), one may have two different conditions:

$$(\lambda_i I - A - BK^T e^{-\lambda_i \tau})v_i = 0, \tag{3.18}$$

$$n(v_i) = 0, \tag{3.19}$$

where $\lambda_i$ is a solution of the characteristic equation $, v_i e^{\lambda\theta}, \theta \in [-\tau, 0]$ is the corresponding eigen function and $n(v_i)$ is a normalizing condition. To know the variation of eigenvalues and eigenvector w.r.t. change in feedback gain, we can differentiate (3.18 and 3.19) w.r.t. feedback gain. We can obtain $\partial\lambda_i / \partial k_j$ and $\partial v_i / \partial k_j$ :

$$\begin{bmatrix} \lambda_i I - A - BK^T e^{-\lambda_i \tau} & (I + BK^T \tau e^{-\lambda_i \tau})v_i \\ dn^T / dv_i & 0 \end{bmatrix} \begin{bmatrix} \partial v_i / \partial k_j \\ \partial \lambda_i / \partial k_j \end{bmatrix} = \begin{bmatrix} Bv_i^T e_j e^{-\lambda_i \tau} \\ 0 \end{bmatrix} \tag{3.20}$$

with $e_j \in R^{n \times 1}$ the $j^{th}$ unity vector.

The sensitivity can be computed from the above equation (3.20) as $S = \partial\lambda_i / \partial k_j$ .

After computing the above sensitivity function, one may have the value of small change in gain in each iteration by following the computational procedure presented in the next section.

### 3.5.1.2 Computing the Required Change in the Feedback Gain( K)

After checking the location of rightmost eigenvalue and corresponding eigenvector of our defined system, if the system is not stable then we have to change the gain by adding the previous with small change in gain ( $\Delta K$ ). This small change in gain can be calculated by

$$S.\Delta K = \Delta \Lambda^d \qquad (3.21)$$

where $\Delta \Lambda^d = [\Delta\lambda_1^d, \Delta\lambda_2^d, ......., \Delta\lambda_m^d]$ is the desired small displacement of the controlled eigenvalues, $m$ is the eigenvalue of the system matrix. Then

$$\Delta K = S^{\dagger} \Delta \Lambda^d \qquad (3.22)$$

where $S^{\dagger}$ is the Moore-Penrose inverse of sensitivity matrix ( $S$ )

The change in gain can be computed using the above equation (3.22), after some changes in the gain, the unstable poles becomes stable but two stable poles become unstable ,the poles are shown in the Fig.3.5. Later on with some more gain adjustment (more iteration), all the poles become stable which has been shown in the Fig.3.6.

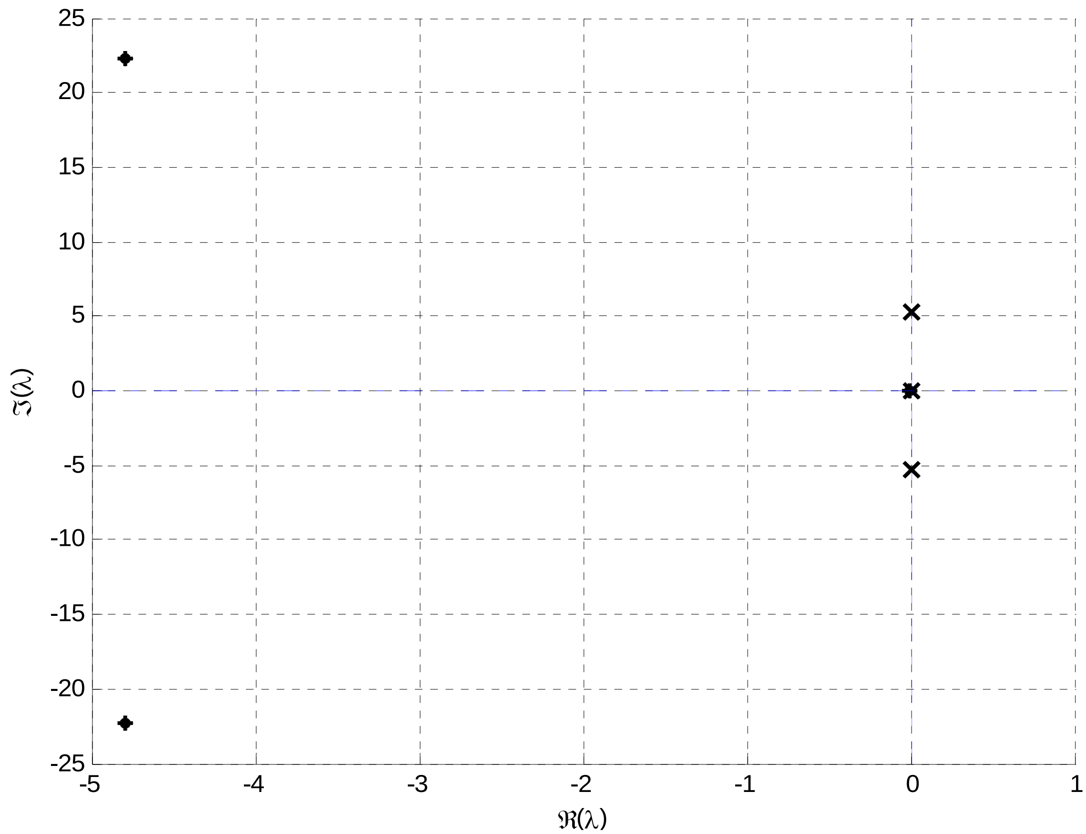


Fig.3.5: The Location of Stable and Unstable Poles After Some Adjustment in Gain

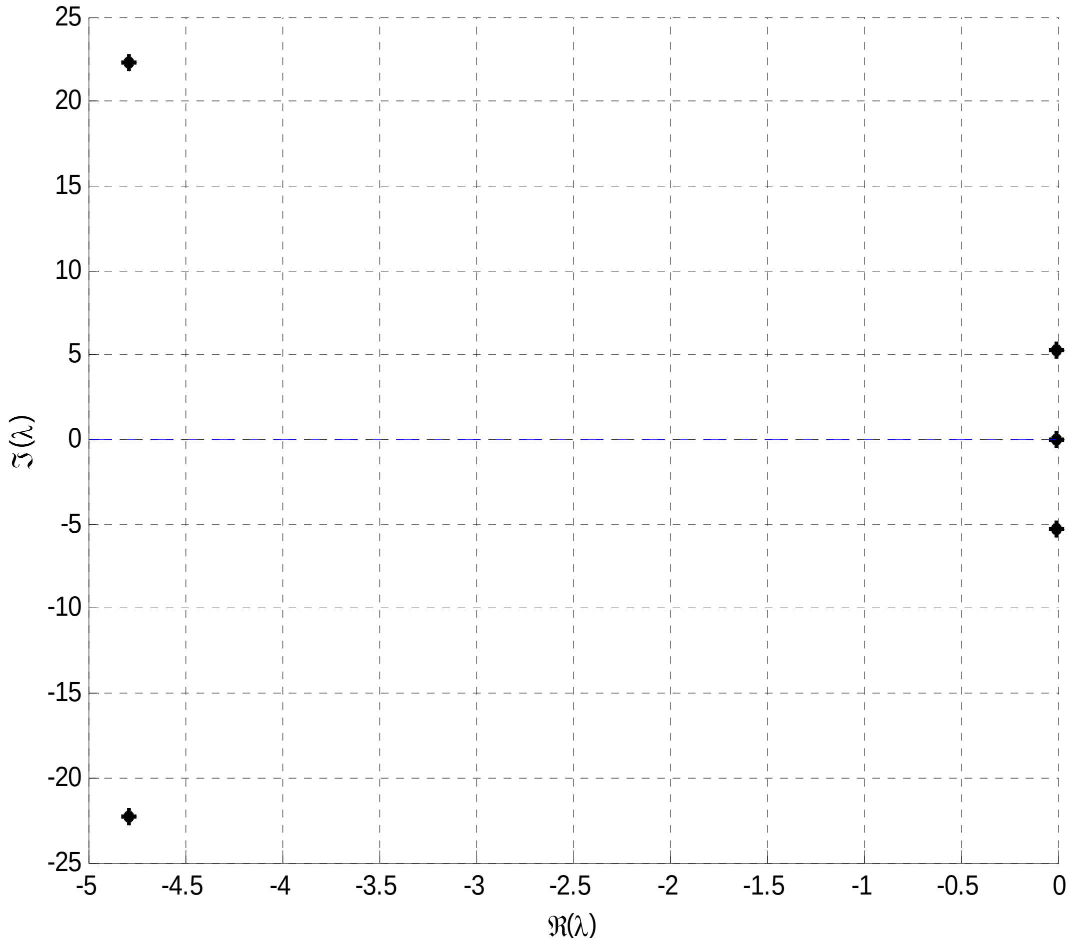


Fig.3.6: The Location of All the Stable Poles

The convergence of three rightmost eigenvalues are shown in the Fig.3.7. Simultaneously, the parameters of the controller have been computed at the convergence of the all the poles,it has been observed that the controller gains are $K = \left[-6.9719 \quad -13.9439\right]$.The controller parameters are found to be same, as we have computed using *fminsearch* and FST method,which validates this pole placement algorithm.The convergence of three rightmost poles have been shown in the Fig.3.7. From the Fig.3.7 below, one may observe the variation of three rightmost poles w.r.t. change in gain(iteration)

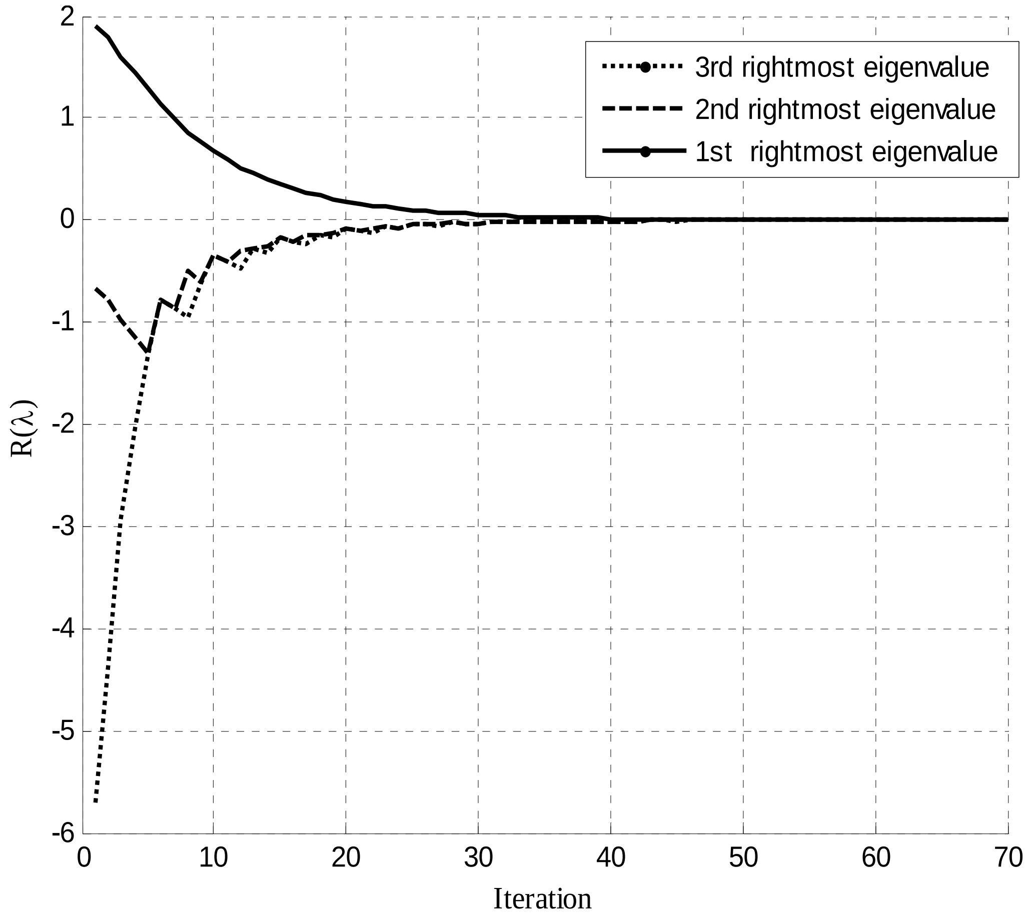


Fig 3.7: Plot Between $R(\lambda)$ and Iteration when All the Delays are Considered to be Equal

To check the stability of the states of both the system and the controller, the closed loop system is simulated using SIMULINK®. The corresponding simulation result is shown in the Fig.3.8. One may observe the states of the system and the controller to be stable at total tolerable delay, $\tau_{total} = 0.7130$.

It may be a question that the above total tolerability is the maximum one or not. For that reason, one must have a test of maximum tolerability.The procedure to have a test maximum tolerability is given in the next section.

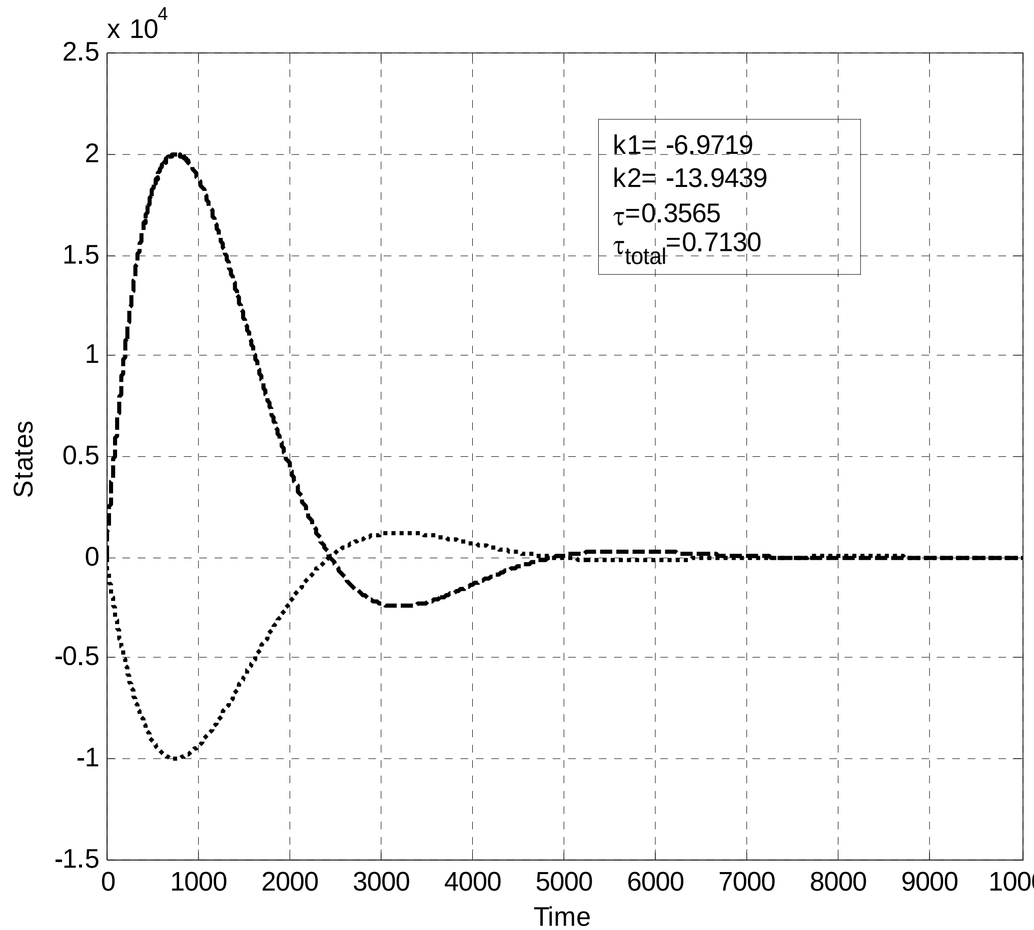

Fig 3.8: Simulation Result for the System (3.17)

### 3.5.1.3 Obtaining the Maximum Tolerable Delay

Using this pole placement algorithm the maximum tolerable delay value is obtained using the following steps:

1. Start with a small value of delay and check whether the algorithm converges for that.
2. If the algorithm converges then increase the delay value by a small amount and check whether for that convergence is there.
3. If it converges in the last step then repeat the same as Step 2 till the algorithm diverges. The last value up to which the algorithm converges will be the maximum tolerable delay value.

Maximum tolerability for the above system (3.17) is checked starting from a nominal value of delay ($\tau$) =0 to 0.3566. Till $\tau = 0.3565$, it is seen that the algorithm successfully yields controller parameters for which all the poles are placed in the left-half plane. Variations of the real parts of the rightmost eigenvalues of the closed-loop system w.r.t iterations for $\tau = 0.3566$ are shown in Fig.3.9. From the figure, it can easily be seen that initially the poles are converging towards zero but after some iteration they are diverging. Hence, we conclude that tolerable delay limit is $0.3565$ and the total tolerable delay ($\tau_{total}$) is $0.7130$.

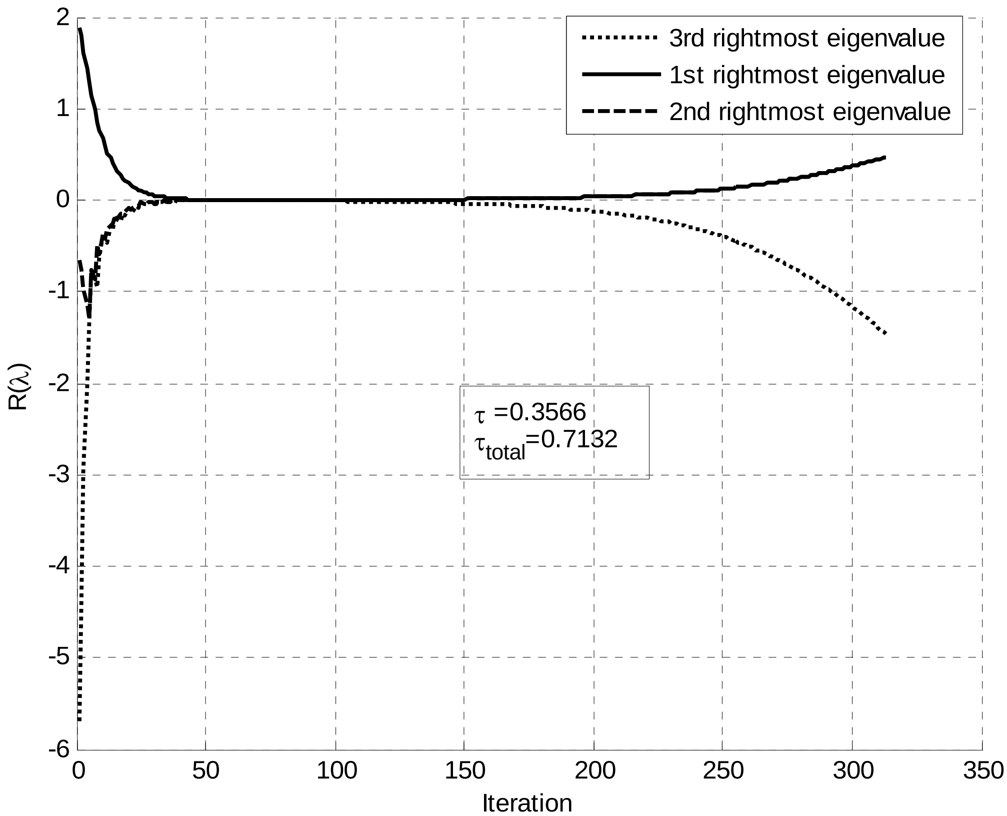


Fig.3.9: The Locus of Three Rightmost Eigenvalues at $\tau = 0.3566$

Next, the variation of maximum attainable $\tau$ has been studied w.r.t. variation in system parameter $a$ from 1 to 3.5, which is shown in Fig.3.10. The maximum attainable $\tau$ using the existing static feedback controller has also been plotted in the same figure. The maximum tolerable delay ($\tau_{total}$) in the feedback loop is 0.7130,when the system parameter ($a$) is 1 and its values decreases near about 0.2 when $a$ is equal to 3.5 but using the existing static feedback controller, the maximum tolerable delay is 0.5, when $a$ is equal to 1 and decreases near to 0.15,when $a$ is equal to 3.5.A comparison of these two characteristics shows that the proposed controller has better performance than the existing one.

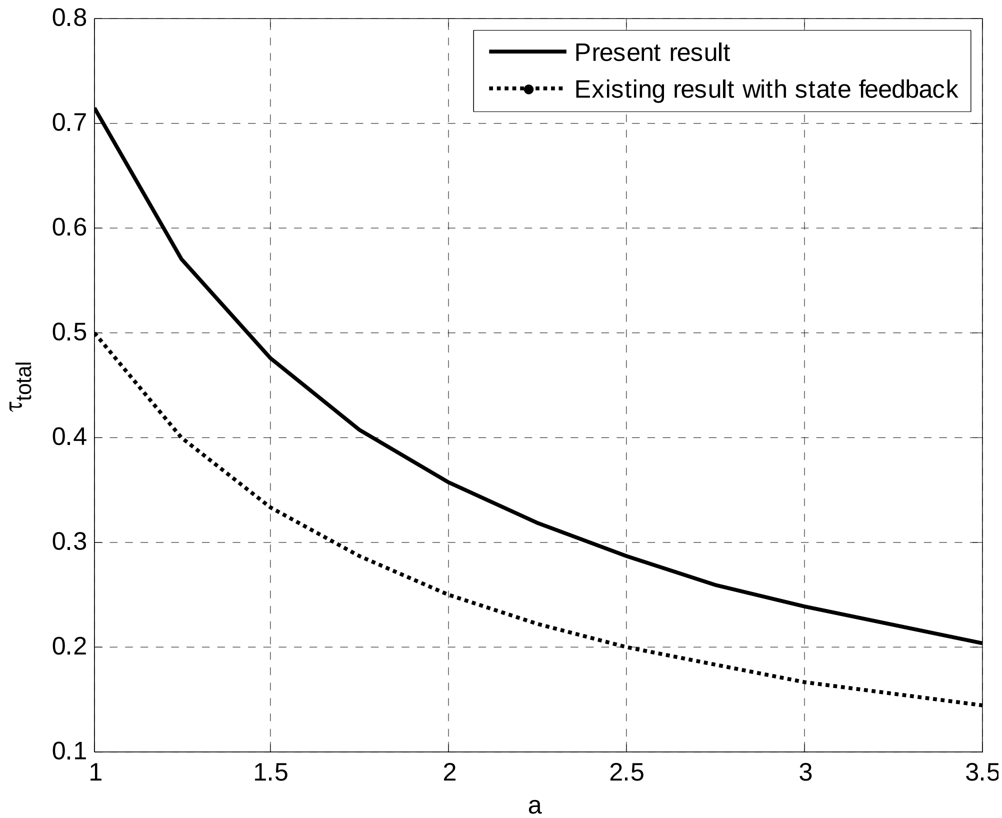


Fig 3.10: Plot Between $a$ and $\tau_{\max}$ when All the Delays are Considered to be Equal

### 3.5.2 The Case when $\tau_a \neq \tau_s$

In this section, we consider the case when $\tau_a \neq \tau_s$. In this regard, note that, $\tau_c$ is to be chosen by the designer and hence one may select $\tau_c = \tau_s$. Then, considering $\tau_a = \tau_1$ and $\tau_s = \tau_c = \tau_2$, the dynamic controller becomes $\dot{x}_c(t) = a_c x_c(t-\tau_2) + b_c u_c(t-\tau_2)$ and the closed loop system is

$$\begin{aligned}\dot{x}(t) &= ax(t) + bx_c(t-\tau_1) \\ \dot{x}_c(t) &= a_c x_c(t-\tau_2) + b_c x(t-\tau_2)\end{aligned} \tag{3.23}$$

The augmented form of the closed loop system may be represented as,

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & b \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau_1) \\ x_c(t-\tau_1) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ b_c & a_c \end{bmatrix} \begin{bmatrix} x(t-\tau_2) \\ x_c(t-\tau_2) \end{bmatrix} \tag{3.24}$$

For computational approach, we have defined the system (3.24) as

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & b \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau_1) \\ x_c(t-\tau_1) \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} \begin{bmatrix} k_1 & k_2 \end{bmatrix} \begin{bmatrix} x(t-\tau_2) \\ x_c(t-\tau_2) \end{bmatrix} \tag{3.25}$$

where $b_c = k_1$ and $a_c = k_2$

The characteristics equation of the closed loop system (3.25) may be represented,

$$\{\lambda_i I - A - A_d \exp(-\tau_1 \lambda_i) - BK \exp(-\tau_2 \lambda_i)\} = 0 \tag{3.26}$$

where $A = \begin{bmatrix} a & 0 \\ 0 & 0 \end{bmatrix}$, $A_d = \begin{bmatrix} 0 & b \\ 0 & 0 \end{bmatrix}$, $BK = B * K$, $B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$ and $K = \begin{bmatrix} k_1 & k_2 \end{bmatrix}$

where $\lambda_i$ is a solution of the characteristic equation and $v_i \exp(\lambda\theta), \theta \in [-\tau, 0]$ is its corresponding eigenfunction.

The procedure for computing the sensitivity of the eigenvalues for this special case is presented in the next section.

#### 3.5.2.1 Computation of Sensitivity of Eigenvalues w.r.t. the Feedback Gain

To compute the sensitivity of the eigenvalues w.r.t. the changes in the feedback gain for the case when $\tau_a = \tau_1$ and $\tau_s = \tau_c = \tau_2$. To compute the sensitivity function, we have

$$\{\lambda_i I - A - A_d \exp(-\lambda_i \tau_1) - BK \exp(-\lambda_i \tau_2)\}v_i = 0 \tag{3.27}$$

$$n(v_i) = 0, \tag{3.28}$$

where $\lambda_i$ is a solution of the characteristic equation , $v_i e^{\lambda\theta}, \theta \in [-\tau, 0]$ is the corresponding eigen function and $n(v_i)$ is a normalizing condition. To know the variation of eigenvalues and eigen vector w.r.t. change in feedback gain, we can differentiate (3.27) and (3.28) w.r.t. feedback gain to obtain $\partial\lambda_i / \partial k_j$ and $\partial v_i / \partial k_j$ :

$$\begin{bmatrix} \theta_1 & \theta_2 \\ \dfrac{dn^T}{dv_i} & 0 \end{bmatrix} \begin{bmatrix} \dfrac{\partial v_i}{\partial k_j} \\ \dfrac{\partial \lambda_i}{\partial k_j} \end{bmatrix} = \begin{bmatrix} B v_i^T e_j \exp(-\lambda_i \tau_2) \\ 0 \end{bmatrix} \tag{3.29}$$

where $\theta_1 = \{\lambda_i I - A - A_d \exp(-\lambda_i \tau_1) - BK \exp(-\lambda_i \tau_2)\}$ and

$$\theta_2 = \{I + A_d \tau_1 \exp(-\lambda_i \tau_1) + BK \tau_2 \exp(-\lambda_i \tau_2)\}v_i$$

with $e_j \in R^{n\times 1}$ the $j^{th}$ unity vector.

After computing the value of $S = \partial\lambda_i / \partial k_j$ , we may follow the §3.5.1.2 to obtain the change of controller gain ( $\Delta K$ ).

In this case, the pole placement algorithm is again used to study the variation of maximum tolerable $\tau_a$ w.r.t. variation of $\tau_s$ . This characteristic is shown in Fig.3.11. In this characteristic, one may observe that considering $\tau_c = \tau_s$, the maximum tolerability of $\tau_s$ has been enhanced to 0.7240 when $\tau_a$ is equal to 0.3480. It may be a case that one may choose $\tau_c = \tau_a$ and the plot of $\tau_s$ versus $\tau_a$ would be the same. If the $\tau_c$ is chosen to be equal to any one of the other delays ($\tau_s$ or $\tau_a$) then the tolerability of the other delay will enhanced to 0.7240 and the total tolerable delay bound ($\tau_{total}$) may even be enhanced to 1.0640 (see the case when $\tau_s = 0.7240$).

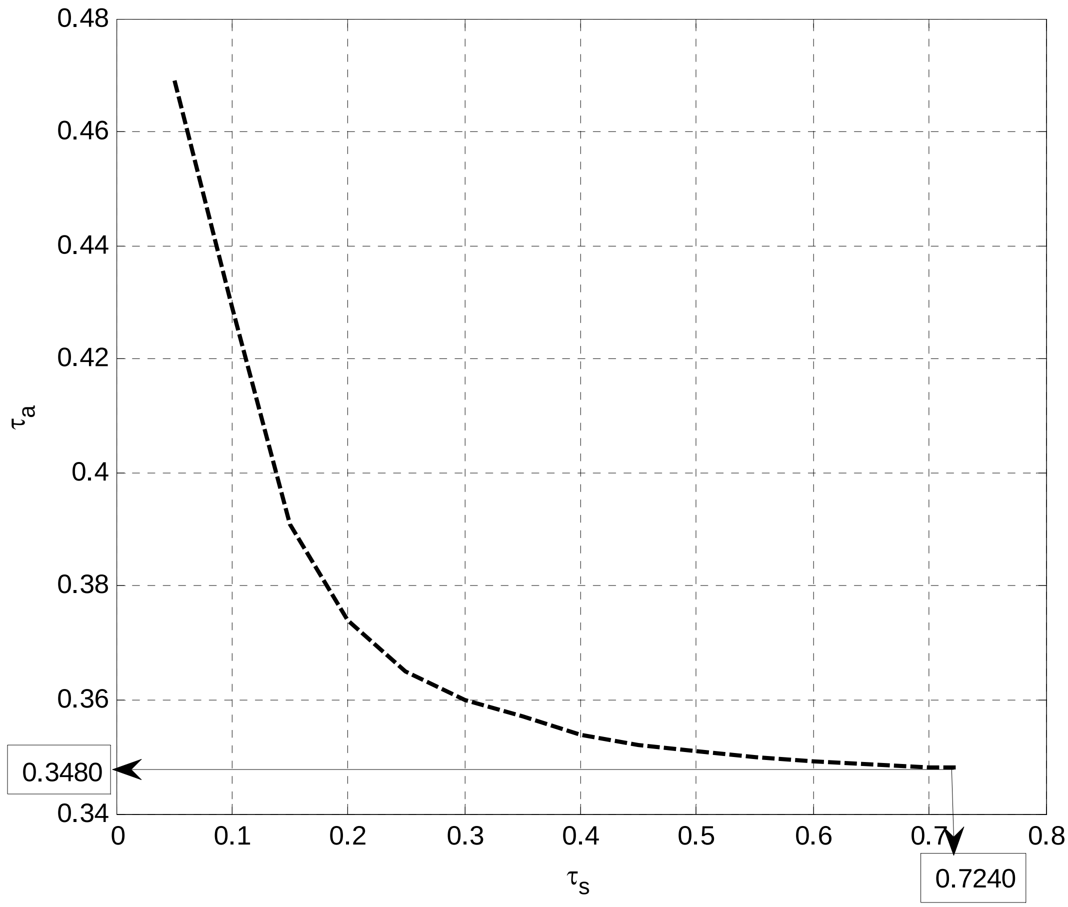


Fig.3.11: Variation of $\tau_a$ w.r.t. $\tau_s$

Note that, this tolerable delay is almost twice of that achievable by any static feedback controller. For this case, the convergence of the three right most eigenvalues is shown in the

Fig.3.12. And to check the stability of the above system (3.25) at the maximal tolerable value of delay, we have simulated the system, which is shown in the Fig.3.13.

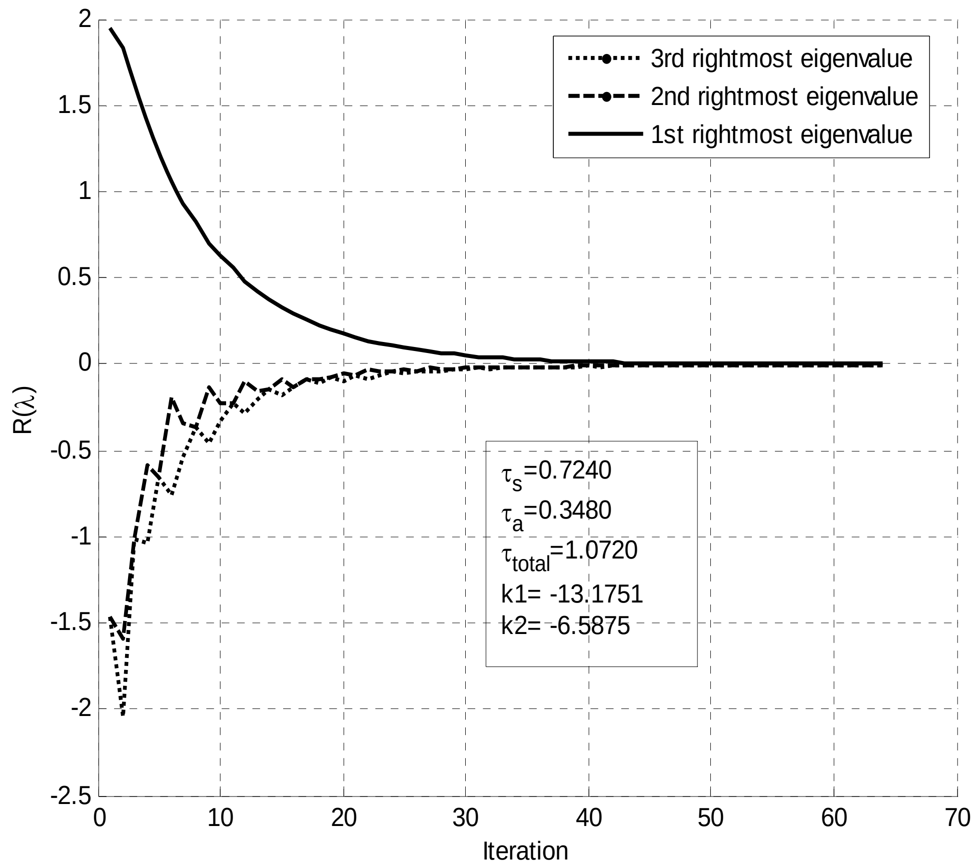


Fig.3.12: The Convergence of Rightmost Eigenvalues at $\tau_{total} = 1.0720$

From Fig.3.12, it is clear that the rightmost poles are on the left-half plane after gain adjustment in near about 64 iterations at maximal value of delay. From the simulation result shown in Fig.3.13, one may observe that the closed loop system is stable at the maximal tolerable delay value of $1.0640$.

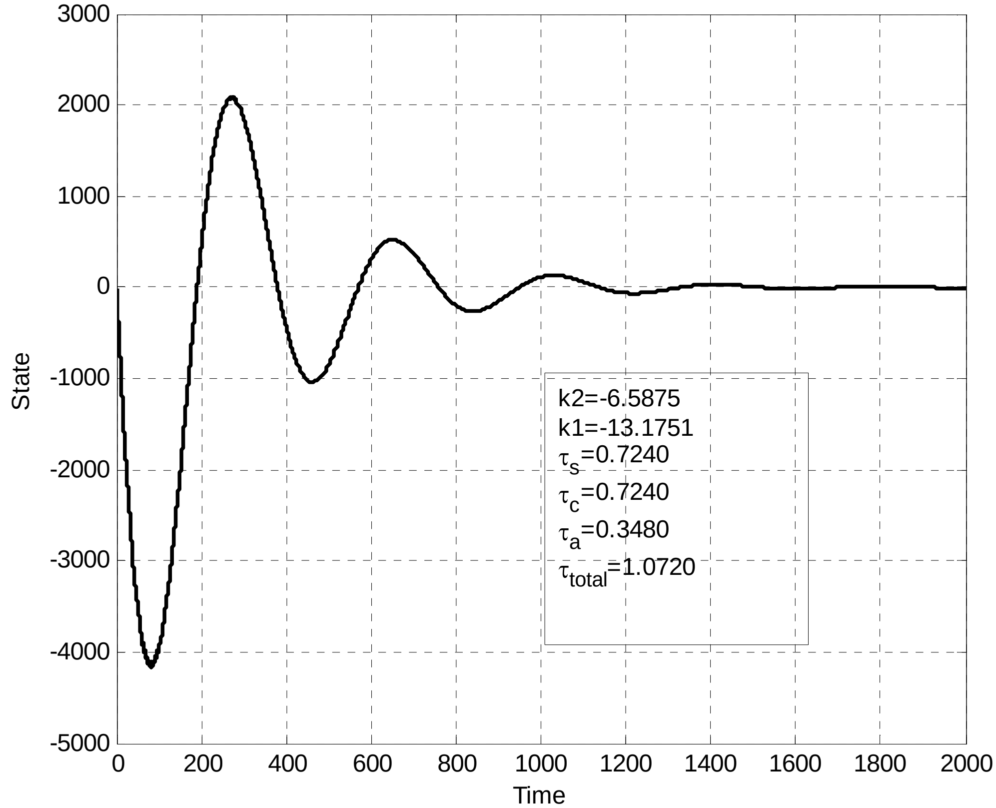


Fig.3.13: The Simulation of System (3.25) at Maximal Condition when $\tau_{total} = 1.0720$

If the controller delay is arbitrarily chosen then all the delays in the feedback path will be unequal.

### 3.5.3 The Case of $\tau_s \neq \tau_c \neq \tau_a$

When the controller delay ($\tau_c$) is arbitrarily chosen, then all the delays may not be equal. For this case, the controller may be chosen as $\dot{x}_c(t) = a_c x_c(t - \tau_c) + b_c u_c(t - \tau_s)$. The closed loop system may be defined as

$$\begin{aligned} \dot{x}(t) &= ax(t) + bx_c(t - \tau_a) \\ \dot{x}_c(t) &= a_c x_c(t - \tau_c) + b_c x(t - \tau_s) \end{aligned} \tag{3.30}$$

The augmented form of the closed loop system may be represented as:

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & b \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau_a) \\ x_c(t-\tau_a) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & a_c \end{bmatrix} \begin{bmatrix} x(t-\tau_c) \\ x_c(t-\tau_c) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ b_c & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau_s) \\ x_c(t-\tau_s) \end{bmatrix}$$

(3.31)

The characteristics equation of the closed loop system (3.17) may be represented,

$$\{\lambda_i I - A - B\exp(-\tau_a \lambda_i) - A_c \exp(-\tau_c \lambda_i) - B_c \exp(-\tau_s \lambda_i)\} = 0 \qquad (3.32)$$

Note that, when $\lambda_i$ is a solution of (3.32) then $v_i \exp(\lambda\theta), \theta \in [-\tau, 0]$ is its eigen function. The computation of the corresponding sensitivity function for this case is given in the next section.

**3.5.3.1 Computation of Sensitivity of Eigenvalues w.r.t. the Feedback Gain**

For the case, $\tau_s \neq \tau_c \neq \tau_a$, we may compute the sensitivity of the eigenvalues w.r.t. the changes in the feedback gain by the following procedure.

To compute the sensitivities, the following conditions are taken into account:

$$\{\lambda_i I - A - B\exp(-\tau_a \lambda_i) - A_c \exp(-\tau_c \lambda_i) - B_c \exp(-\tau_s \lambda_i)\} v_i = 0 \qquad (3.33)$$

$$n(v_i) = 0, \qquad (3.34)$$

where $\lambda_i$ is a solution of the characteristic equation , $v_i e^{\lambda\theta}, \theta \in [-\tau, 0]$ is the corresponding eigenfunction and $n(v_i)$ is a normalizing condition. To know the variation of eigenvalues and eigenvector w.r.t. change in feedback gain, we can differentiate (3.33 and 3.34) w.r.t. feedback gain. We can obtain $\partial\lambda_i / \partial k_j$ and $\partial v_i / \partial k_j$ from the equation below.

$$\begin{bmatrix} \theta_3 & \theta_4 \\ \dfrac{dn^T}{dv_i} & 0 \end{bmatrix} \begin{bmatrix} \dfrac{\partial v_i}{\partial k_j} \\ \dfrac{\partial \lambda_i}{\partial k_j} \end{bmatrix} = \begin{bmatrix} v(1) \\ 0 \end{bmatrix} \tag{3.35}$$

where $\theta_3 = \{\lambda_i I - A - B\exp(-\tau_a\lambda_i) - A_c\exp(-\tau_c\lambda_i) - B_c\exp(-\tau_s\lambda_i)\}$ and

$\theta_4 = \{I + B\tau_a\exp(-\tau_a\lambda_i) + A_c\tau_c\exp(-\tau_c\lambda_i) + B_c\tau_s\exp(-\tau_s\lambda_i)\}v_i$

with $e_j \in R^{n\times 1}$ the $j^{th}$ unity vector.

After computing the value of $S = \partial\lambda_i / \partial k_j$ , we may follow the §3.5.1.2 to obtain the change of controller gain

In this case, the pole placement algorithm is used to study on variation of $\tau_c$ is considered. Maximum tolerable $\tau_a$ w.r.t. variation of $\tau_c$ is obtained, keeping $\tau_s$ fixed. This study is shown in Fig.3.14, in which it may be observed that with increase in $\tau_c$, tolerable $\tau_a$ also increases up to a certain limit, here, approximately $\tau_c = 0.49$.Beyond that tolerable $\tau_a$ first gradually and then suddenly decreases.

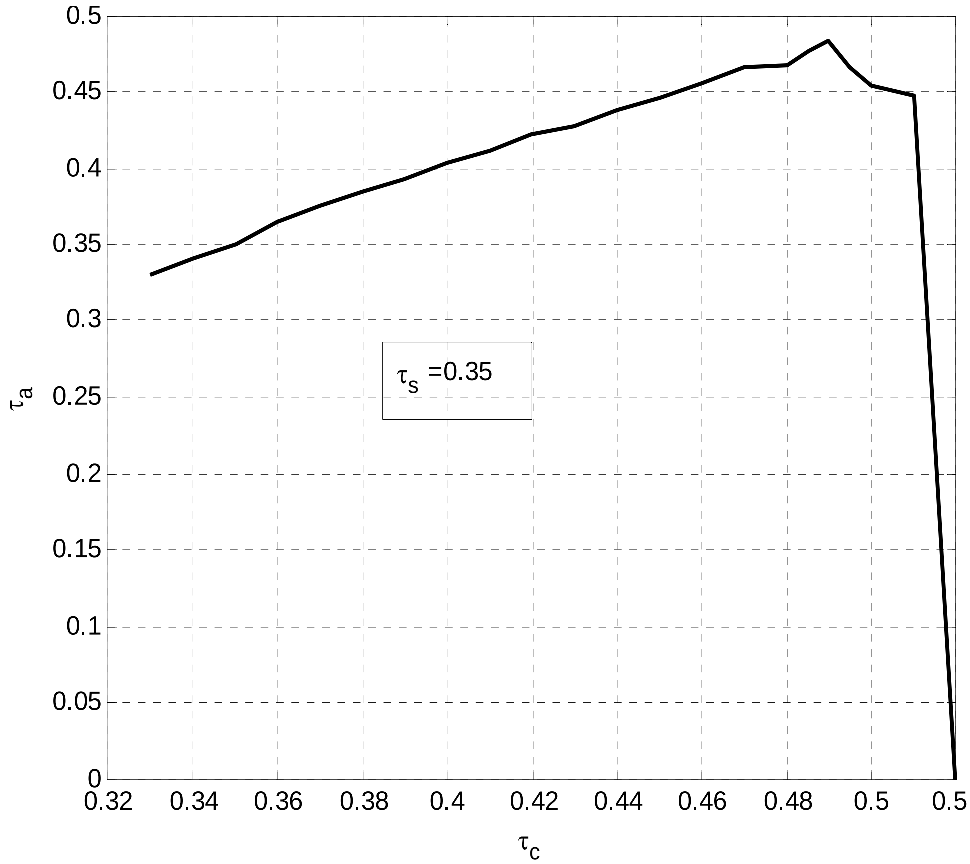


Fig.3.14: Variation of $\tau_a$ w.r.t. $\tau_c$ keeping $\tau_s$ constant.

It shows that $\tau_c$ cannot be chosen arbitrarily rather it should be chosen judicially to obtain improvement in tolerable delay margin.

To check the convergence of this algorithm developed for unequal delays in the feedback loop, we have taken a case for $a = 2, b = 1$, $\tau_a = 0.30$, $\tau_c = 0.34$ and $\tau_a = 0.25$.For this case, convergence of the algorithm is shown in the Fig.3.15 and its simulation result is shown to be stable in the Fig.3.16.

**Example 3.6:** We have considered the above case,

$$\begin{bmatrix} \dot{x}(t) \\ \dot{x}_c(t) \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t) \\ x_c(t) \end{bmatrix} + \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau_a) \\ x_c(t-\tau_a) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & k_1 \end{bmatrix} \begin{bmatrix} x(t-\tau_c) \\ x_c(t-\tau_c) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ k_2 & 0 \end{bmatrix} \begin{bmatrix} x(t-\tau_s) \\ x_c(t-\tau_s) \end{bmatrix}$$

(3.36)

From the Fig.3.15, it is clear that three rightmost eigenvalues are converging after an iteration of 70 and simultaneously the controller gain is found to be [-10.4469 -5.2229]. From the figure, one may observe that the system is stable at $\tau_a = 0.30$, $\tau_c = 0.34$ and $\tau_a = 0.25$.

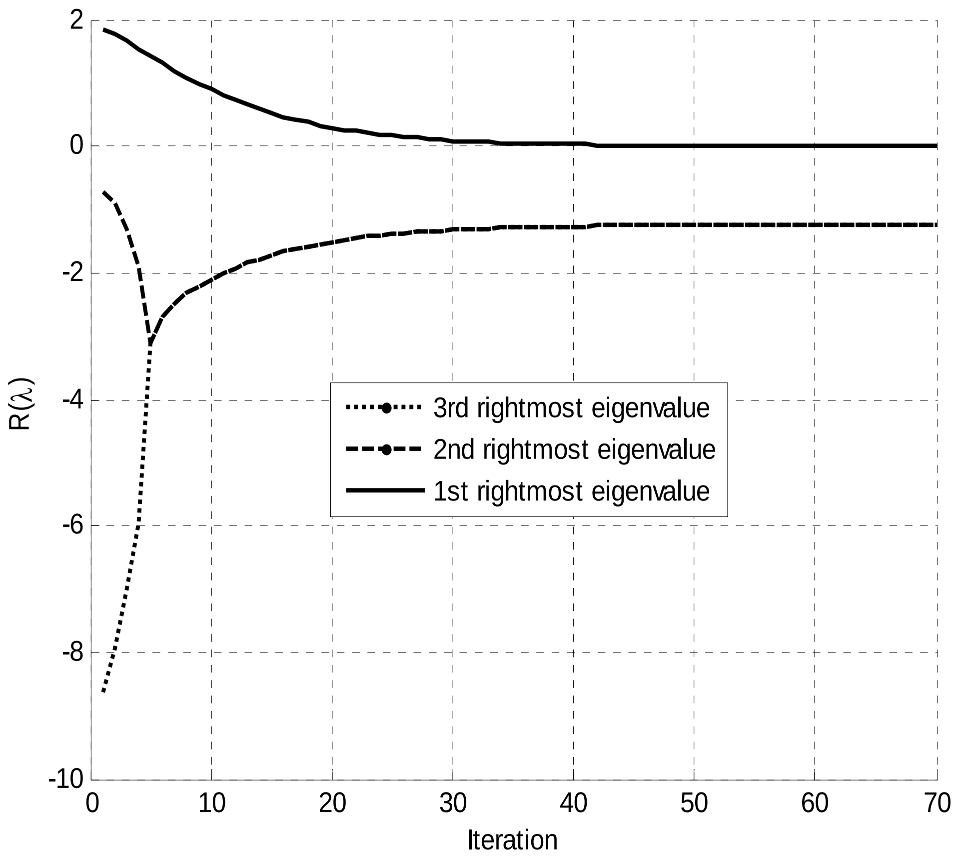


Fig 3.15: The Convergence of Three Right Most Eigenvalues of System (3.36)

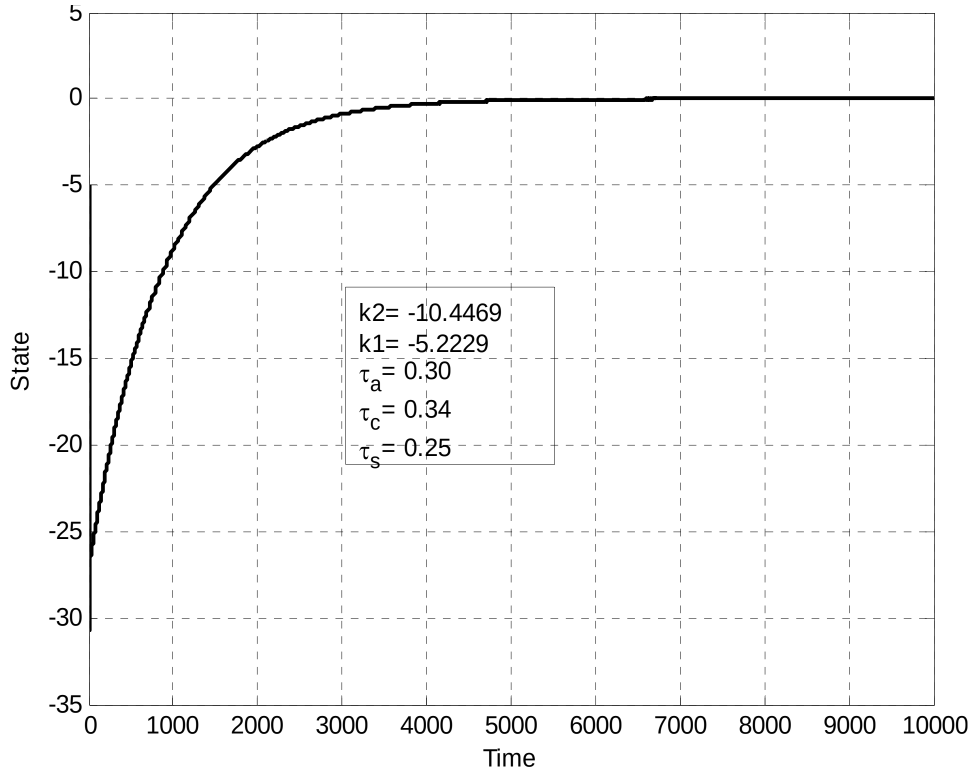


Fig 3.16: Simulation of System (3.36)

## 3.6 Conclusion

Tolerable delay margin in the feedback path of a closed loop system is enhanced to even twice that of existing results by a proposed dynamic state feedback controller with an artificial delay in its state. For designing such dynamic feedback controllers, a continuous pole placement algorithm is used. The study carried out in this chapter shows that the proposed controller is uniform in improving tolerable delay margin in respect to different system parameters and delay values.

# Chapter 4

# Conclusion

This thesis investigates benefits of using dynamic state feedback controllers. Firstly, the case of tolerable robustness improvement for systems with parametric uncertainty by using dynamic state feedback controllers is investigated. In this regard, some new examples have been developed for which dynamic feedback controller can improve the tolerable uncertainty bounds. Next, a dynamic state feedback controller with an artificial delay in its states has been proposed that enhances the tolerable delay margin for systems having delays in both the input and output. The design and characteristic of such a new type of dynamic feedback controller have been studied by implementing the continuous pole-placement algorithm of [29].

## 4.1 Contributions of the Thesis

The following are the salient contributions of the thesis.

- For the class of second order systems with constant uncertainty in the input matrix, an algorithm has been developed to identify the systems for which a sufficient tolerable uncertainty bound by using static feedback may be computed. The same algorithm has been used to develop several new examples and it has been tested for those that dynamic state feedback controller improves tolerable uncertainty bounds for such systems compared to the static state feedback controller.
- The same second order systems have been used to construct third order systems with system matrix uncertainty and verified that improvement by dynamic state feedback controller still holds.
- For systems with both input and output delays in the feedback loop, a dynamic feedback controller with an artificial delay in its states to improve the tolerable delay bound in the feedback loop. It is shown that such a controller improves the tolerable delay bounds considerably, at least, for nominal systems.

- The existence of such dynamic feedback controller that can improve tolerable delay bound has been computed by continuous pole placement algorithm of [29] and different characteristics of such controllers w.r.t. variations in system parameters have been studied. It is seen that the in improving the tolerable delay bound the ability of such controllers is consistent with respect of variations in different parameters. Such verification also establishes consistency in convergences of continuous pole placement algorithm.

## 4.2 Future Scope of Work

The present work not only presents some new results and but it also helps in defining several problems to be investigated in future. These are:

- So far no systematic methodology exists to design a stabilizing dynamic state feedback controller that may be used to stabilize a system with maximum possible tolerable uncertainty bound. One may implement and study some computational algorithm to address this issue.
- For systems with feedback delays, a scalar system has only been considered. It will be interesting to investigate the same problem considered here but for higher order systems.
- Constant delays in the feedback loop have been considered in this thesis. These delays may be time-varying. One may investigate the performance of the proposed controllers in that case.
- One may check whether the proposed controller for systems with feedback delays can able to improve the tolerable delay in the feedback loop, if the system is having uncertainty with time varying delay in the feedback loop.

# References


1. Albertos P. and Sala A., *Multivariable control system: An Engineering Approach*, Springer-Verlag London Limited, 2004.
2. Abootalebi A., Nia S. H. and Sheikholeslam F., “Reliable $H_\infty$ control systems with uncertainties in all system matrices: An LMI approach”, *Proc. of the 2005 IEEE Conf. on Control Applications Toronto, Canada,* pp.251-255, Aug. 2005.
3. Barmish B. R., “New tools for Robustness of linear Systems”, Pearson Higher Education, 1993.
4. Chen J., Gu G. and Nett C. N., “A New Method for Computing Delay Margin for stability of Linear Delay System”, *Proc. of the 33rd Conf. on Decision and Control*, Lake Buena vista, FL, pp.433-437, Dec.1994.
5. de Souza C. E., Barbosa K. A. and Trofino A., “Robust Filtering for Linear System with Convex-Bounded Uncertain Time-Varying Parameters”, *IEEE Trans. Automat. Contr.*, vol. 52, pp.1132-1138, Jun. 2007.
6. Doyle J. C., “Structured Uncertainty in Control System Design”, *Proc. of 24th Conf. on Decision and Control Ft. Lauderdale, FL*, pp.260-265, Dec. 1985.
7. Engelborghs K., Dambrine M. and Roose D., “Limitation of class of Stabilization Methods for Delay Systems”, *IEEE Trans. Automat. Contr.*, vol. 46, pp.336-339, Feb. 2001.
8. Fu M. and Barmish B. R., “Maximal unidirectional perturbation bounds for stability of polynomials and matrices”, *Systems & Control Letters*, vol.11, pp.173-179, 1988.
9. Fu M., Olbrot A. W. and Polis M. P., “Robust Stability for Time Delay Systems: The Edge Theorem and Graphical Tests”, *IEEE Trans. Automat. Contr.*, vol. 34, pp.813-820, Aug.1989.
10. Fuh C. C. and Tung P. C., “Robust Stability Bounds For LUR’e Systems with Parametric Uncertainty, *Journal of Marine Science and Technology*, Vol. 7, pp. 73-78, 1999.

11. Fridman E., shaked U. and Suplin V., “Input/output delay approach to robust sampled-data $H_\infty$ control”, *Systems & Control Letters*, vol.54, pp.271-282, 2005.

12. Gu K., Kharitonov V. L. and Chen J., *Stability of time-delay Systems*, Boston: Birkhauser, 2003.

13. Gelso E. R., Castillo S. M., and Armengol J., “Fault diagnosis by refining the parameter uncertainty space of nonlinear dynamic systems”, *American Control Conf. Westin Seattle Hotel, Seattle, Washington, USA*, pp. 4304-4308, Jun., 2008.

14. Gu K., “ $H_\infty$ Control of Systems under Norm Bounded Uncertainties in all System Matrices”, *IEEE Trans. Automat. Contr.*, vol. 39, pp. 1320-1322, Jun. 1994.

15. Guo H., Chen T. and Lam J., “A new delay system approach to network based control”, *Automatica*, vol.44, pp. 39-52, 2008.

16. Galunet P., Apkarian P. and Chilali M., “Affine Parameter Dependent Lyapunov Functions and Real Parametric Uncertainty”, *IEEE Trans. Automat. Contr.*, vol. 41, pp.436-442, Mar. 1996.

17. Holmberg U., “Comments on ‘A Revisit to the Gain and Phase Margins of Linear Quadratic Regulators’ ”, *IEEE Trans. Automat. Contr.*, vol. 46, pp.1509-1509, Sept. 2001.

18. Hall B. D., “Propagating Uncertainty in Instrumentation Systems”, *IEEE Trans. on Instrumentation and Measurement*, vol. 54, pp. 2376-2380, Dec. 2005.

19. Iwasaki T. and Shibata G., “LPV System Analysis via Quadratic Separator for Uncertain Implicit Systems”, *IEEE Trans. Automat. Contr.*, vol. 46, pp.1195-1208, Aug. 2001.

20. Jankovic M., “Control Lyapunov-Razumikhin Functions and Robust Stabilization of the Time Delay systems”, *IEEE Trans. Automat. Contr.*, vol. 46, pp.1048-1060, Jul. 2001.

21. Johansson A. and Norlander T., “Parametric Uncertainty in Sensor Fault Detection for a Turbofan Jet Engine”, *Proc. of the 42nd IEEE Conf. on Decision and Control Maui, Hawaii USA*, pp. 1950-1955, Dec. 2003.

22. Khattak A. A. and Iqbal N., “Design of a Robust Dynamic State Feedback Controller for System with Time-Varying State Delays”, *INMIC*, pp.451-455, Dec.2006.

23. Khargonekar P. P. and Rotea M. A., "Stabilization of Uncertain Systems with Norm Bounded Uncertainty Using Control Lyapunov Functions", *Proc. of the 27th Conf. on Decision and Control*, Austin, Texas, pp.503-508, Dec.1988.

24. Khargonekar P. P., Petersen I. R. and Zhou K., "Robust stabilization of Uncertain Linear Systems: Quadratic stabilizability and $H_\infty$ control theory", *IEEE Trans. Automat. Contr.*, vol. 35, pp.356-361, Mar. 1990.

25. Khattak A. A. and Iqbal N., "On robust Dynamic State Feedback Stabilization of Variable Time-Delay Uncertain Systems", *Proc. of Int. Bhurban Conf. on Applied Sciences and Technology, Islamabad, Pakistan*, pp. 147-154, Jan.2007.

26. Lee C. H., "D-Stability of Continuous Time-Delay Systems Subjected to a Class of Highly Structured Perturbations", *IEEE Trans. Automat. Contr.*, vol. 40, pp.1803-1807, Oct. 1995.

27. Liu J., "Design of Stable Closed Loop Systems with Uncertainties and Estimate Performance Variations", *Proc. of the 1996 IEEE Int. Conf. on Control Applications Dearborn, MI*, pp.173-178, Sept. 1996.

28. Liu P. L., "On the Stability of Neutral-Type Uncertain Systems with Multiple Time Delays", *Int. J. Appl. Math. Computer. Sci*., vol. 15, pp. 221–229, 2005.

29. Michiels W., Engelborghs K., Vansevenant P., Roose D., "Continuous pole placement for delay equations", *Automatica*, vol.38, pp. 747–761, 2002.

30. Michiels W. and Roose D., "Limitations of delayed state feedback: a numerical study", *Int. Journal of Bifurcation and Chaos in Applied Sciences and Engg.*, pp.1309-1320, Jun.2002.

31. Partington J. R., "Some frequency-domain approaches to the model reduction of delay system", *Annual Reviews in control*, vol.28, pp.65-73, 2004.

32. Rehm A. and Allgower F., "$H_\infty$ Control of Descriptor Systems with Norm-bounded Uncertainties in the System Matrices", *Proc. of the American Control Conf. Chicago, Illinois*, pp.3244-3248, Jun. 2000.

33. Schmitendorf W. E. and Stalford H. L., "Improving Stability Margins via Dynamic-State Feedback for Systems with Constant Uncertainty", *IEEE Trans. Automat. Contr.*, vol. 42, pp.1161-1163, Aug. 1997.

34. Verriest E. I., Fan M. K.H. and Kullstam J., "Frequency Domain Robust Stability Criteria for Linear Delay Systems", *Proc. of the 32nd Conf. on decision and Control San Antonio, Texas*, pp. 3473-3478, Dec.1993.

35. Verschelde J. and Wang Y., "Computing Dynamic Output Feedback Laws", *IEEE Trans. Automat. Contr.*, vol. 49, pp.1393-1397, Aug. 2004.

36. Young P. M., Newlin M. P., Doyle J. C., "$\mu$ Analysis with Real Parametric Uncertainty", *Proc. of the 30th Conf. on Decision and Control Brighton, England*, pp.1251-1256, Dec. 1991.

37. Ye D. and Yang G. H., "Delay-Dependent Reliable $H_\infty$ Control for Linear Time-Varying Delay Systems via Adaptive Approach", *Proc. of the 2007 American Control Conf. Marriott Marquis Hotel at Times Square New York City, USA*, pp. 3991-3996, Jul.2007.

38. Yang B. and Lin W., "Robust Output Feedback Stabilization of Uncertain Nonlinear Systems with Uncontrollable and Unobservable Linearization", *IEEE Trans. Automat. Contr.*, vol. 50, pp.619-630, May. 2005.

39. Zhang C. and Fu M., "A Revisit to the Gain and Phase Margins of Linear Quadratic Regulators", *IEEE Trans. Automat. Contr.*, vol. 41, pp.1527-1530, Oct. 1996.

40. Zelentsovsky A. L., "Non-quadratic Lyapunov Function for Robust Stability Analysis of Linear Uncertain Systems", *IEEE Trans. Automat. Contr.*, vol. 39, pp.135-138, Jan. 1994.

41. Zevin A. A. and Pinsky M., "Delay-Independent Stability Conditions for Time-Varying Nonlinear Uncertain Systems", *IEEE Trans. Automat. Contr.*, vol. 51, pp.1482-1485, Sept. 2006.

# Appendix-A

# Frequency Sweeping Test (FST)

## A.1 Introduction

Frequency sweeping test is very efficient computational tool for computing the tolerable delay margin of a given system.

Consider a system

$$\dot{x}(t) = Ax(t) + A_d x(t-\tau), \tau \geq 0$$

The characteristic equation of the above system may be written as:

$$a(s, e^{-\tau s}) = \det(sI - A - A_d e^{-\tau s})$$

The necessary and sufficient condition for stability of the above system is given in the following theorem that is the foundation of the frequency sweeping test.

**Theorem [12]:** Suppose that the system is stable at $\tau = 0$. Let $rank(A_d) = q$.

Furthermore, define

$$\overline{\tau}_i := \begin{cases} \min\limits_{1\leq k\leq n} \dfrac{\theta_k^i}{\omega_k^i} & \text{if } \lambda_i(j\omega_k^i I - A, A_d) = e^{-j\theta_k^i} \text{ for some } \omega_k^i \in (0,\infty), \theta_k^i \in [0, 2\pi] \\ \infty & \text{if } \underline{\rho}(j\omega I - A, A_d) > 1, \forall \omega \in (0,\infty) \end{cases}$$

Then $\overline{\tau} := \min\limits_{1\leq i\leq q} \overline{\tau}_i$

That is, the system is stable for all $\tau \in [0, \bar{\tau})$ , but becomes unstable at $\tau = \bar{\tau}$ .

The proof of the above theorem may be found in [12].

### A.2 Frequency Sweeping Test [12]

If the system matrices $A$ and $A_d$ are known, then one may compute the tolerable delay margin by using the following algorithm.

**Algorithmic A.1:**

Step 1: Calculate the maximum of the real parts of all the eigenvalues of $A + A_d$ . If it is less than zero then proceed.

Step 2: Obtain the rank of $A_d$ matrix. Let, $\text{rank}(A_d) = q$ . Then, there will be $q$ number of crossover points from closed left half plane to right half plane through the imaginary axis over frequency sweep and maximum tolerable $\bar{\tau} = \min_{1 \le i \le q} \bar{\tau}_i$ where $\bar{\tau}$ is the optimal value of $\tau$ , i.e., the system is stable for all $\tau \in [0, \bar{\tau})$, but unstable at $\tau = \bar{\tau}$ .

Step 3: Set a frequency range and fix the step size for frequency variation.

Step 4: For different frequencies ( $w_k^i$ ), calculate the absolute values of all the eigenvalues of the matrix pencil $(jw_k^i I - A, A_d)$ .

Step 5: From the above calculation, obtain the frequency at which the absolute value corresponding to each eigenvalue variation reaches one. If not then go to Step 3.

Step 6: Calculate the corresponding value of eigenvalue at that frequency.

Step 7: Get the angle corresponding to that eigenvalue in complex plane.

Step 8: Compute the value of delay ( $\tau$ ), where $\tau$ =angle/frequency.

# Appendix-B

# Using BIFTOOL

## B.1 Introduction

It is a tool for numerical bifurcation analysis of steady state solutions and periodic solutions of differential equation with constant delays. Basically, it is a collection of MATLAB routines for numerical bifurcation analysis of systems of delay differential equations and may be used to analyze the stability of a time-delay system by determining the rightmost or stability determining roots of the characteristics equation.

## B.2 Delay Differential Equations with Constant Delays

Consider the system of delay differential equations (DDEs) with constant delays

$$\frac{dx(t)}{dt} = f(x(t), x(t-\tau_1), \ldots\ldots\ldots x(t-\tau_m), \eta) \qquad \text{(B.1)}$$

where $x(t) \in R^n$, $f : R^{n(m+1)} \times R^p \to R^n$ is a nonlinear smooth function depending on a number of parameters $\eta \in R^p$ and delays $\tau_i > 0$, $i = 1, 2, \ldots\ldots m$

Maximal delay $\tau = \max_{i=1,2\ldots m} \tau_i$

The linearization of equation (B.1) around a solution $x^*(t)$ is the variational equation given by

$$\frac{dy(t)}{dt} = A_0(t)y(t) + \sum_{i=1}^{m} A_i(t)y(t-\tau_i) \qquad \text{(B.2)}$$

where using $f = f(x^0, x^1, \ldots\ldots\ldots x^m, \eta)$

$$A_i(t) = \left.\frac{\partial f}{\partial x^i}\right|_{(x^*(t), x^*(t-\tau_1)........x^*(t-\tau_m), \eta)} \quad , i = 1, 2, ........m \qquad \text{(B.3)}$$

If $x^*(t)$ corresponds to a steady state solution $x^*(t) \equiv x^* \in R^n$, with $f(x^*, x^* .......x^*, \eta) = 0$ then the matrices $A_i(t) = A_i$ and the corresponding variational equation (B.2) leads to a characteristics equation,

$$\Delta(\lambda) = \lambda I - A_0 - \sum_{i=1}^{m} A_i \exp(-\lambda \tau_i)$$

And the characteristics equation

$$\det(\Delta(\lambda)) = 0 \qquad \text{(B.4)}$$

Equation (B.4) has an infinite number of roots, which determines the stability of the steady state solution $x^*$ .The steady state solution is asymptotically stable, provided all roots of the characteristic equation (B.4) have negative real part; it is unstable if there exists a root with positive real part. It is known that the number of roots in any right half plane $R(\lambda) > \gamma, \gamma \in R$ is finite; hence the stability is always determined by a finite number of roots.

### B.3 How to use BIFFTOOL:

For stability analysis, computation of rightmost poles is very much required. To locate these poles or stability determining poles, we may define our system with feedback delay by replacing some routines by our defined routines in this tool like sys_rhs.m, sys_tau.m and sys_deri.m. Someone may define another default routine df_deriv.m in place of sys_deri.m because of less typing error and less complexity.

#### B.3.1 Defining sys_rhs.m :

The right hand side of the system is defined in sys_rhs.m. It has two arguments, $xx \in R^{n\times(m+1)}$ which contains the preset state variable(s) and the delayed state variable,

$xx = [x(t), x(t-\tau_1), ..........x(t-\tau_m)]$ .And $par \in R^{1\times p}$ which contains the parameter $par = \eta$ **.**The delays $\tau_i$ , i =1,2,3….m. are considered to be part of the parameter because the stability of the system and position of poles depend on the value s of the delays so the delay can occur both as a physical parameter as in **x** from these inputs the right hand side function ( $f$ ) is evaluated at time( $t$ ) and the representation of parameter has a specific order.

### B.3.2 Defining sys_tau.m:

Here, a function is required which returns the position of the delays in parameter list. The order in this list corresponds to the order in which they appears in $xx$ as passed to the function sys_rhs and syy_deri.

### B.3.3 Defining sys_deri.m:

If $f$ is a right hand side function, in this routine several derivatives of the function ( $f$ ) is evaluated. The input variable of the function $(f)$ are state variables $(xx)$, parameters $(par)$, no. of state variables $(nx)$, no. of parameter $(np)$ and $v$. Here $v \in C^{n\times 1}$ or empty. The $J$ is a matrix of partial derivatives of $f$ which depends on the type of derivative requested via $nx$ and $np$ multiplied with $v$ (when non empty),

$J$ is informally defined as follows: Initialize $J$ with $f$ .if $nx$ is non empty take the derivative of $J$ w.r.t. each of its elements. Each element is a number between 0 and $m$ based on $f \equiv f(x^0, x^1, .............x^m, \eta)$ .E.g., if $nx$ has only one element take the derivative w.r.t. $x^{nx(1)}$ .If it has two elements take of the result, the derivative w.r.t. $x^{nx(2)}$ and so on. Similarly, if $np$ is nonempty take of the resulting $J$ , the derivative w.r.t. $\eta_{np}(i)$ where $i$ ranges over all the element of $np$ ,$1 \le i \le p$ .finally, if $v$ is not an empty vector multiply the result with $v$ .The latter is used to prevent $J$ from being a tensor if two derivative w.r.t. state variables are taken (when $nx$ contains two elements).Not all possible combinations of these derivatives should be provided. In the current version, $nx$ has at most two elements and $np$ at most one. The element of $J$ are given by

$$J_{i,j} = \left[ \frac{\partial}{\partial x^{nx(2)}} A_{nx(1)} v \right]_{i,j} = \frac{\partial}{\partial x_j^{nx(2)}} \left( \sum_{k=1}^{n} \frac{\partial f_i}{\partial x_k^{nx(1)}} v_k \right)$$

| Length( $nx$ ) | Length( $np$ ) | $v$ | $J$ |
|---|---|---|---|
| 1 | 0 | Empty | $\frac{\partial f}{\partial x^{nx(1)}} = A_{nx(1)} \in R^{n \times n}$ |
| 0 | 1 | Empty | $\frac{\partial f}{\partial \eta_{np(1)}} \in R^{n \times 1}$ |
| 1 | 1 | Empty | $\frac{\partial^2 f}{\partial x^{nx(1)} \partial \eta_{np(1)}} \in R^{n \times n}$ |
| 2 | 0 | $\in C^{n \times 1}$ | $\frac{\partial (A_{nx(1)} v)}{\partial x^{nx(2)}} \in C^{n \times n}$ |

# Appendix-C

# MATLAB Programs

BIFTOOL has been used for computing the rightmost poles or stability determining poles of a closed loop system with feedback delay. In Appendix-B, we have mentioned the procedure to use the above tool. The programs written for our system are listed below.

## C.1 The case of Equal Delays ($\tau_a = \tau_s = \tau_c$)

### C.1.1 Define sys_tau.m

```
%dx(t)/dt=[a11 a12;a21 a22]*x(t)+[ad11 ad12;ad21 ad22]*x(t-tau)
%+[0 0;k1 k2]*x(t-tau);
%dx(t)/dt=[a11 a12;a21 a22]*[x1(t);x2(t)+[ad11 ad12;ad21
%ad22]*[x1(t-tau);x2(t-tau)]+[0 0;k1 k2]*[x1(t-tau);x2(t-tau)];
%dx1(t)/dt=a11x1(t)+a12x2(t)+ad11x1(t-tau)+ad12x2(t-tau)
%dx2(t)/dt=a21x1(t)+a22x2(t)+ad21x1(t-tau)+ad22x2(t-tau)+k1x1(t-tau)
%+k2x2(t-tau)
function tau=sys_tau()
%A(1,1),A(1,2),BK(1,1),BK(1,2),A(2,1),A(2,2),BK(2,1),BK(2,2),tau;
tau=[9];
return;
```

### C.1.2 Define sys_rhs.m

```
%dx(t)/dt=[a11 a12;a21 a22]*x(t)+[ad11 ad12;ad21 ad22]*x(t-tau)
%+[0 0;k1 k2]*x(t-tau);
%dx(t)/dt=[a11 a12;a21 a22]*[x1(t);x2(t)+[ad11 ad12;ad21
%ad22]*[x1(t-tau);x2(t-tau)]+[0 0;k1 k2]*[x1(t-tau);x2(t-tau)];
%dx1(t)/dt=a11x1(t)+a12x2(t)+ad11x1(t-tau)+ad12x2(t-tau)
%dx2(t)/dt=a21x1(t)+a22x2(t)+ad21x1(t-tau)+ad22x2(t-tau)+k1x1(t-tau)
%+k2x2(t-tau)
function f=sys_rhs(xx,par)
%A(1,1),A(1,2),BK(1,1),BK(1,2),A(2,1),A(2,2),BK(2,1),BK(2,2),tau;
f(1,1)=par(1)*xx(1,1)+par(2)*xx(2,1)+par(3)*xx(1,2)+par(4)*xx(2,2);
f(2,1)=par(5)*xx(1,1)+par(6)*xx(2,1)+par(7)*xx(1,2)+par(8)*xx(2,2);
return;
```

### C.1.3 Define sys_deri.m

```
%dx(t)/dt=[a11 a12;a21 a22]*x(t)+[ad11 ad12;ad21 ad22]*x(t-tau)
%+[0 0;k1 k2]*x(t-tau);
%dx(t)/dt=[a11 a12;a21 a22]*[x1(t);x2(t)+[ad11 ad12;ad21
%ad22]*[x1(t-tau);x2(t-tau)]+[0 0;k1 k2]*[x1(t-tau);x2(t-tau)];
%dx1(t)/dt=a11x1(t)+a12x2(t)+ad11x1(t-tau)+ad12x2(t-tau)
%dx2(t)/dt=a21x1(t)+a22x2(t)+ad21x1(t-tau)+ad22x2(t-tau)+k1x1(t-tau)
%+k2x2(t-tau)
function J=sys_deri(xx,par,nx,np,v)
%a11,a12,ad11,ad12,a21,a22,ad21,ad22,k1,k2,tau;
% par(1)=a11   x1(t)=xx(1,1)
% par(2)=a12   x2(t)=xx(2,1)
% par(3)=ad11  x1(t-tau)=xx(1,2)
% par(4)=ad12  x2(t-tau)=xx(2,2)
% par(5)=a21
% par(6)=a22
% par(7)=ad21
% par(8)=ad22
% par(9)=k1
% par(10)=k2
% par(11)=tau
J=[];
if length(nx)==1&length(np)==0&isempty(v)
    %first order derivative wrt state variable
    if nx==0 %derivative wrt x(t)
        J(1,1)=par(1);
        J(1,2)=par(2);
        J(2,1)=par(5);
        J(2,2)=par(6);
    elseif nx==1 %derivative wrt x(t-tau)
        J(1,1)=par(3);
        J(1,2)=par(4);
        J(2,1)=par(7)+par(9);
        J(2,2)=par(8)+par(10);
    end;
elseif length(nx)==1&length(np)==1&isempty(v)
     %mixed states and parameter derivatives
     if nx==0 %derivative wrt x(t)
         if np==1 %derivative wrt a11
             J(1,1)=1;
             J(1,2)=0;
             J(2,1)=0;
             J(2,2)=0;
         elseif np==2 %derivative wrt a12
             J(1,1)=0;
             J(1,2)=1;
             J(2,1)=0;
             J(2,2)=0;
         elseif np==3 %derivative wrt ad11
             J(1,1)=0;
             J(1,2)=0;
             J(2,1)=0;
             J(2,2)=0;
         elseif np==4 %derivative wrt ad12
```

```
            J(1,1)=0;
            J(1,2)=0;
            J(2,1)=0;
            J(2,2)=0;
        elseif np==5 %derivative wrt a21
            J(1,1)=0;
            J(1,2)=0;
            J(2,1)=1;
            J(2,2)=0;
        elseif np==6 %derivative wrt a22
            J(1,1)=0;
            J(1,2)=0;
            J(2,1)=0;
            J(2,2)=1;
        elseif np==7 %derivative wrt ad21
            J(1,1)=0;
            J(1,2)=0;
            J(2,1)=0;
            J(2,2)=0;
        elseif np==8 %derivative wrt ad22
            J(1,1)=0;
            J(1,2)=0;
            J(2,1)=0;
            J(2,2)=0;
        elseif np==9 %derivative wrt k1
            J(1,1)=0;
            J(1,2)=0;
            J(2,1)=0;
            J(2,2)=0;
        elseif np==10 %derivative wrt k2
            J(1,1)=0;
            J(1,2)=0;
            J(2,1)=0;
            J(2,2)=0;
        elseif np==11 %derivative wrt tau
            J(1,1)=0;
            J(1,2)=0;
            J(2,1)=0;
            J(2,2)=0;
        end;
    elseif nx==1 %derivative wrt x(t-tau)
        if np==1 %derivative wrt a11
            J(1,1)=0;
            J(1,2)=0;
            J(2,1)=0;
            J(2,2)=0;
        elseif np==2 %derivative wrt a12
            J(1,1)=0;
            J(1,2)=0;
            J(2,1)=0;
            J(2,2)=0;
        elseif np==3 %derivative wrt ad11
            J(1,1)=1;
            J(1,2)=0;
            J(2,1)=0;
```

```
        J(2,2)=0;
    elseif np==4 %derivative wrt ad12
        J(1,1)=0;
        J(1,2)=1;
        J(2,1)=0;
        J(2,2)=0;
    elseif np==5 %derivative wrt a21
        J(1,1)=0;
        J(1,2)=0;
        J(2,1)=0;
        J(2,2)=0;
    elseif np==6 %derivative wrt a22
        J(1,1)=0;
        J(1,2)=0;
        J(2,1)=0;
        J(2,2)=0;
    elseif np==7 %derivative wrt ad21
        J(1,1)=0;
        J(1,2)=0;
        J(2,1)=1;
        J(2,2)=0;
    elseif np==8 %derivative wrt ad22
        J(1,1)=0;
        J(1,2)=0;
        J(2,1)=0;
        J(2,2)=1;
    elseif np==9 %derivative wrt k1
        J(1,1)=0;
        J(1,2)=0;
        J(2,1)=1;
        J(2,2)=0;
    elseif np==10 %derivative wrt k2
        J(1,1)=0;
        J(1,2)=0;
        J(2,1)=0;
        J(2,2)=1;
    elseif np==11 %derivative wrt tau
        J(1,1)=0;
        J(1,2)=0;
        J(2,1)=0;
        J(2,2)=0;
    end;
end;
elseif length(nx)==0 &length(np)==1&isempty(v)
    if np==1 %derivative wrt a11
        J(1,1)=xx(1,1);
        J(2,1)=0;
    elseif np==2 %derivative wrt a12
        J(1,1)=xx(2,1);
        J(2,1)=0;

    elseif np==3 %derivative wrt ad11
        J(1,1)=xx(1,2);


        J(2,1)=0;
```

```
        elseif np==4 %derivative wrt ad12
            J(1,1)=xx(2,2);

            J(2,1)=0;

        elseif np==5 %derivative wrt a21
            J(1,1)=0;

            J(2,1)=xx(1,1);

        elseif np==6 %derivative wrt a22
            J(1,1)=0;

            J(2,1)=xx(2,1);

        elseif np==7 %derivative wrt ad21
            J(1,1)=0;

            J(2,1)=xx(1,2);

        elseif np==8 %derivative wrt ad22
            J(1,1)=0;

            J(2,1)=xx(2,2);

        elseif np==9 %derivative wrt k1
            J(1,1)=0;

            J(2,1)=xx(1,2);

        elseif np==10 %derivative wrt k2
            J(1,1)=0;

            J(2,1)=xx(2,2);

        elseif np==11 %derivative wrt tau
            J(1,1)=0;

            J(2,1)=0;
        end;
elseif length(nx)==2&length(np)==0&isempty(v)
    %second order derivatives wrt state variable
    J=0;
end;
if isempty(v)
    err=[nx np size(v)];
    error('SYS_DERI: requested derivative could not be computed!');
end;

return;
```

Note that, the above routine sys_deri.m may be optionally replaced by sys_deriv.m which may be used as a general routine for any cases.

### C.1.4 Define sys_deriv.m

```
function J=sys_deri(xx,par,nx,np,v)

% function J=sys_deri(xx,par,nx,np,v)
% INPUT:
%   xx state variable and delayed state variables columnwise
%   par list of parameter values
%   nx empty or list of requested state-derivatives (numbers of delay or
zero)
%   np empty or list of requested parameter-derivatives
%   v matrix to multiply result with
% OUTPUT:
%   J result of derivatives on righthandside multiplied with v
% COMMENT:
%   the numerical derivatives are evaluated using forward differences

% (c) DDE-BIFTOOL v. 1.00, 11/03/2000

% first order derivative discretisation parameters:

abs_eps_x1=1e-6;
abs_eps_x2=1e-6;
abs_eps_p1=1e-6;
abs_eps_p2=1e-6;
rel_eps_x1=1e-6;
rel_eps_x2=1e-6;
rel_eps_p1=1e-6;
rel_eps_p2=1e-6;

n=size(xx,1);

J=[];

% first order derivatives of the state:
if length(nx)==1 & length(np)==0 & isempty(v),
  f=sys_rhs(xx,par);
  for j=1:n
    xx_eps=xx;
    eps=abs_eps_x1+rel_eps_x1*abs(xx(j,nx+1));
    xx_eps(j,nx+1)=xx(j,nx+1)+eps;
    J(:,j)=(sys_rhs(xx_eps,par)-f)/eps;
  end;
% first order parameter derivatives:
elseif length(nx)==0 & length(np)==1 & isempty(v),
  f=sys_rhs(xx,par);
  par_eps=par;
  eps=abs_eps_p1+rel_eps_p1*abs(par(np));
```

```
  par_eps(np)=par(np)+eps;
  J=(sys_rhs(xx,par_eps)-f)/eps;
% second order state derivatives:
elseif length(nx)==2 & length(np)==0 & ~isempty(v),
  for j=1:n
    J(:,j)=sys_deri(xx,par,nx(1),[],[])*v;
    xx_eps=xx;
    eps=abs_eps_x2+rel_eps_x2*abs(xx(j,nx(2)+1));
    xx_eps(j,nx(2)+1)=xx_eps(j,nx(2)+1)+eps;
    J(:,j)=(sys_deri(xx_eps,par,nx(1),[],[])*v-J(:,j))/eps;
  end;
% mixed state parameter derivatives:
elseif length(nx)==1 & length(np)==1 & isempty(v),
  J=sys_deri(xx,par,nx(1),[],[]);
  par_eps=par;
  eps=abs_eps_p2+rel_eps_p2*abs(par(np));
  par_eps(np)=par(np)+eps;
  J=(sys_deri(xx,par_eps,nx(1),[],[])-J)/eps;
end;

if isempty(J)
  [nx np size(v)]
  error('SYS_DERI: requested derivative does not exist!');
end;

return;
```

### C.1.5 The Continuous Pole Placement Algorithm

#### C.1.5.1 The Control Gain Updating Program

```
how_many_eig=2;
K=[-1,-1];
[n,m]=size(K);
right_eig=1;iter=0;n_v_old=zeros(2,n*m);
while right_eig>=0
    iter=iter+1;
    for i2=1:1:how_many_eig
        which_eig=i2;
        for i1=1:1:n*m
            e=zeros(n*m,1);e(i1)=1;

[S(i2,i1),right_eigs(:,iter),how_many_eig_new,n_v_new]=sensitivity_control(K
,e,which_eig,how_many_eig,iter,n_v_old(:,i1));
            n_v_old(:,i1)=n_v_new;
        end
    end
if right_eig>1e-3
    Lamda_d=right_eig*[-0.1;-0.1];
else
    Lamda_d=1e-2*[-0.1;-0.1];
end
    del_K=(pinv(S)*Lamda_d)';
```

```
    K=K+real(del_K);
    right_eig=max(right_eigs(:,iter));
    right_eig
    how_many_eig=how_many_eig_new;
end
figure(2)
hold on
plot(1:iter,right_eigs(3,:),'r');
plot(1:iter,right_eigs(2,:),'g');
plot(1:iter,right_eigs(1,:),'k');
grid;
```

### C.1.5.2 Computing Sensitivity Function Program:

```
function[R,right_eigs,no_right_eig,n_v_new]=sensitivity_control(K,e,which_ei
g,no_right_eig,iter,n_v_old)

%dx(t)/dt=[a11 a12;a21 a22]*x(t)+[ad11 ad12;ad21 ad22]*x(t-tau)+[0 0;k1
k2]*x(t-tau);
%A=[a11 a12;a21 a22];AD=[ad11 ad12;ad21 ad22];K=[0 0;k1 k2];
 A=[2,0;0,0];B=[0;1];
% A=[2,0;0,0];B=[0;1];
BK=B*K;BK(1,2)=1;tau=0.3566;

stst.kind='stst';
stst.parameter=[A(1,1),A(1,2),BK(1,1),BK(1,2),A(2,1),A(2,2),BK(2,1),BK(2,2),
tau];
%stst.parameter=[A(1,1),A(1,2),AD(1,1),AD(1,2),A(2,1),A(2,2),AD(2,1),AD(2,2)
,K(2,1),K(2,2),tau];
stst.x=[0;0];
method=df_mthod('stst');
method.stability.minimal_real_part=-20;
method.stability.max_number_of_eigenvalues=6;
method.stability.max_newton_iterations=20;
%[stst,success]=p_correc(stst,[],[],method.point)

stst.stability=p_stabil(stst,method.stability);
% figure(1);clf;
% p_splot(stst);
% stst.stability.l1
if which_eig>1
    if imag(stst.stability.l1(1))~=0
        which_eig=which_eig+1;
    end
end

Abar=A+BK*exp(-(stst.stability.l1(which_eig))*tau);
[v,d]=eig(Abar);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
eigs=diag(d);
[Y,which_eig_vec]=min(abs(eigs-stst.stability.l1(which_eig)));
n_v_new(1)=v(2,which_eig_vec); n_v_new(2)=-v(1,which_eig_vec);
```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```
% n_v_new(1)=v(2,1); n_v_new(2)=-v(1,1);
if iter==1
    d_n_v=[1,1];
else
    d_n_v=(n_v_new-n_v_old');
end
if d_n_v==[0,0]
    d_n_v=[1,1];
end
P=[stst.stability.l1(which_eig)*eye(2)-A-BK*exp(-
stst.stability.l1(which_eig)*tau),(eye(2)+tau*exp(-
stst.stability.l1(which_eig)*tau)*BK)*v(:,which_eig_vec);d_n_v 0];
Q=[B*v(:,which_eig_vec)'*e*exp(-stst.stability.l1(which_eig)*tau);0];
        R1=inv(P)*Q;
        R=R1(3);
right_eigs=real(stst.stability.l1);
```

## C.2 The Case when $\tau_a \neq \tau_s$, Considering $\tau_a = \tau_1$ and $\tau_s = \tau_c = \tau_2$

### C.2.1 Define sys_tou.m

```
% dx(t)/dt=A*x(t)+B*xc(t-tau1);
% dxc(t)/dt=Ac*xc(t-tau2)+Bc*x(t-tau2);
% A=par(1),B=par(2),Ac=par(3),Bc=par(4),tau1=par(5),tau2=par(6);
% x(t)=xx(1,1),xc(t-tau1)=xx(2,2),x(t-tau2)=xx(1,3),xc(t-tau2)=xx(2,3);
function tau=sys_tau()
%A,B,Ac,,Bc,tau1,tau2;
tau=[5,6];
return;
```

### C.2.2 Define sys_rhs.m

```
% dx(t)/dt=A*x(t)+B*xc(t-tau1);
% dxc(t)/dt=Ac*xc(t-tau2)+Bc*x(t-tau2);
% A=par(1),B=par(2),Ac=par(3),Bc=par(4),tau1=par(5),tau2=par(6);
% x(t)=xx(1,1),xc(t-tau1)=xx(2,2),x(t-tau2)=xx(1,3),xc(t-tau2)=xx(2,3);
function f=sys_rhs(xx,par)
%A,B,Ac,,Bc,tau1,tau2;
f(1,1)=par(1)*xx(1,1)+par(2)*xx(2,2);
f(2,1)=par(3)*xx(1,3)+par(4)*xx(2,3);
return;
```

### C.2.3 The Continuous Pole Placement Algorithm

#### C.2.3.1 The Control Gain Updating Program:

```
clc;
clear all;
close(figure(1))
```

```
how_many_eig=2;
K=[-1,-1];
[n,m]=size(K);
right_eig=1;iter=0;n_v_old=zeros(2,n*m);
while right_eig>=0
    iter=iter+1;
    for i2=1:1:how_many_eig
        which_eig=i2;
        for i1=1:1:n*m
            e=zeros(n*m,1);e(i1)=1;

[S(i2,i1),right_eigs(:,iter),how_many_eig_new,n_v_new]=sensitivity_control2(
K,e,which_eig,how_many_eig,iter,n_v_old(:,i1));
            n_v_old(:,i1)=n_v_new;
        end
    end
if right_eig>1e-3
    Lamda_d=right_eig*[-0.1;-0.1];
else
    Lamda_d=1e-2*[-0.1;-0.1];
end
    del_K=(pinv(S)*Lamda_d)';
    K=K+real(del_K);
    right_eig=max(right_eigs(:,iter));
    right_eig
    how_many_eig=how_many_eig_new;
end
figure(1)
hold on
plot(1:iter,right_eigs(3,:),'r');
plot(1:iter,right_eigs(2,:),'g');
plot(1:iter,right_eigs(1,:),'k');
grid;
```

### C.2.3.2 Computing Sensitivity Function Program:

```
function[R,right_eigs,no_right_eig,n_v_new]=sensitivity_control(K,e,which_ei
g,no_right_eig,iter,n_v_old)

%dx(t)/dt=[a11 a12;a21 a22]*x(t)+[ad11 ad12;ad21 ad22]*x(t-tau)+[0 0;k1
k2]*x(t-tau);
%A=[a11 a12;a21 a22];AD=[ad11 ad12;ad21 ad22];K=[0 0;k1 k2];
A=[2,0;0,0];B=[0;1];
BK=B*K;Ad=[0 1;0 0];tau1=0.348;tau2=0.724;

stst.kind='stst';
stst.parameter=[A(1,1),Ad(1,2),BK(2,1),BK(2,2),tau1,tau2];
%
stst.parameter=[A(1,1),A(1,2),AD(1,1),AD(1,2),A(2,1),A(2,2),AD(2,1),AD(2,2),
K(2,1),K(2,2),tau];
stst.x=[0;0];
```

```
method=df_mthod('stst');
method.stability.minimal_real_part=-20;
method.stability.max_number_of_eigenvalues=6;
method.stability.max_newton_iterations=20;
%[stst,success]=p_correc(stst,[],[],method.point)

stst.stability=p_stabil(stst,method.stability);
%  figure(1);clf;
%  p_splot(stst);
% stst.stability.l1
if which_eig>1
    if imag(stst.stability.l1(1))~=0
        which_eig=which_eig+1;
    end
end
Abar=A+Ad*exp(-(stst.stability.l1(which_eig))*tau1)+BK*exp(-
(stst.stability.l1(which_eig))*tau2);
[v,d]=eig(Abar);
%n_v_new(1)=v(1,1); n_v_new(2)=-v(2,1);
eigs=diag(d);
[Y,which_eig_vec]=min(abs(eigs-stst.stability.l1(which_eig)));
n_v_new(1)=v(2,which_eig_vec); n_v_new(2)=-v(1,which_eig_vec);
if iter==1
    d_n_v=[1,1];
else
    d_n_v=(n_v_new-n_v_old');
end
if d_n_v==[0,0]
    d_n_v=[1,1];
end
P=[stst.stability.l1(which_eig)*eye(2)-A-Ad*exp(-
stst.stability.l1(which_eig)*tau1)-BK*exp(-
stst.stability.l1(which_eig)*tau2),(eye(2)+tau1*Ad*exp(-
stst.stability.l1(which_eig)*tau1)+tau2*exp(-
stst.stability.l1(which_eig)*tau2)*BK)*v(:,which_eig_vec);d_n_v 0];
Q=[B*v(:,which_eig_vec)'*e*exp(-stst.stability.l1(which_eig)*tau2);0];
        R1=inv(P)*Q;
        R=R1(3);
right_eigs=real(stst.stability.l1);
```

## C.3 The Case when $\tau_a \neq \tau_s \neq \tau_c$

### C.3.1 Define sys_tou.m

```
% dx(t)/dt=A*x(t)+B*xc(t-tau_a);
% dxc(t)/dt=Ac*xc(t-tau_c)+Bc*x(t-tau_s);
%
A=par(1),B=par(2),Ac=par(3),Bc=par(4),tau_a=par(5),tau_s=par(6),tau_c=par(7)
;
%x(t)=xx(1,1),xc(t-tau_a)=xx(2,2),x(t-tau_s)=xx(1,3),xc(t-tau_c)=xx(2,4);
 function tau=sys_tau()
%A,B,Ac,,Bc,tau_a,tau_s,tau_c;
```

```
 tau=[5,6,7];
 return;
```

### C.3.2 Define sys_rhs.m

```
%  dx(t)/dt=A*x(t)+B*xc(t-tau_a);
%  dxc(t)/dt=Ac*xc(t-tau_c)+Bc*x(t-tau_s);
%
A=par(1),B=par(2),Ac=par(3),Bc=par(4),tau_a=par(5),tau_s=par(6),tau_c=par(7)
;
% x(t)=xx(1,1),xc(t-tau_a)=xx(2,2),x(t-tau_s)=xx(1,3),xc(t-tau_c)=xx(2,4);
 function f=sys_rhs(xx,par)
% A,B,Ac,,Bc,tau_a,tau_s,tau_c;
 f(1,1)=par(1)*xx(1,1)+par(2)*xx(2,2);
 f(2,1)=par(3)*xx(2,4)+par(4)*xx(1,3);
 return;
```

### C.3.3 The Continuous Pole Placement Algorithm

#### C.3.3.1 The Control Gain Updating Program:

```
clc;
clear all;
%close(figure(1))

how_many_eig=2;
K=[-1,-1];
[n,m]=size(K);
right_eig=1;iter=0;n_v_old=zeros(2,n*m);
while right_eig>=0
    iter=iter+1;
    for i2=1:1:how_many_eig
        which_eig=i2;
        for i1=1:1:n*m
            e=zeros(n*m,1);e(i1)=1;

[S(i2,i1),right_eigs(:,iter),how_many_eig_new,n_v_new]=sensitivity_control1(
K,e,which_eig,how_many_eig,iter,n_v_old(:,i1),i1);
            n_v_old(:,i1)=n_v_new;
        end
    end
if right_eig>1e-3
    Lamda_d=right_eig*[-0.1;-0.1];
else
    Lamda_d=1e-2*[-0.1;-0.1];
end
    del_K=(pinv(S)*Lamda_d)';
    K=K+real(del_K);
    right_eig=max(right_eigs(:,iter));
    right_eig
    how_many_eig=how_many_eig_new;
end
```

```
figure(2)
hold on
plot(1:iter,right_eigs(3,:),'r');
plot(1:iter,right_eigs(2,:),'g');
plot(1:iter,right_eigs(1,:),'k');
grid;
```

### C.3.3.2 Computing Sensitivity Function Program:

```
function[S,right_eigs,no_right_eig,n_v_new]=sensitivity_control1(K,e,which_e
ig,no_right_eig,iter,n_v_old,which_k)

%[dx(t)/dt dxc(t)/dt]=[A 0;0 0][x(t) xc(t)]+[0 B;0 0][x(t-tau_a)
%xc(t-tau_a)]+[0 0;0 Ac][x(t-tau_c) xc(t-tau_c)]+[0 0;Bc 0][x(t-tau_s)
%xc(t-tau_s)];
A_bar=[2,0;0,0];B_bar=[0 1;0 0];Ac_bar=[0 0 ;0 K(1)];Bc_bar=[0 0;K(2) 0];
tau_a=0.3;tau_s=0.2565;tau_c=0.34;
%tau=0.35;
stst.kind='stst';
stst.parameter=[A_bar(1,1),B_bar(1,2),Ac_bar(2,2),Bc_bar(2,1),tau_a,tau_s,ta
u_c];
stst.x=[0;0];
method=df_mthod('stst');
method.stability.minimal_real_part=-20;
method.stability.max_number_of_eigenvalues=4;
method.stability.max_newton_iterations=5;
%[stst,success]=p_correc(stst,[],[],method.point)
stst.stability=p_stabil(stst,method.stability);
% figure(1);clf;
% p_splot(stst);
% stst.stability.l1
Abar=A_bar+B_bar*exp(-(stst.stability.l1(which_eig))*tau_a)+Ac_bar*exp(-
(stst.stability.l1(which_eig))*tau_c)+Bc_bar*exp(-
(stst.stability.l1(which_eig))*tau_s);
[v,d]=eig(Abar);
% n_v_new(1)=v(1,how_many_eig); n_v_new(2)=-v(2,how_many_eig);
eigs=diag(d);
[Y,which_eig_vec]=min(abs(eigs-stst.stability.l1(which_eig)));
n_v_new(1)=v(2,which_eig_vec); n_v_new(2)=-v(1,which_eig_vec);
if iter==1
    d_n_v=[1,1];
else
    d_n_v=(n_v_new-n_v_old');
end
if d_n_v==[0,0]
    d_n_v=[1,1];
end
P=[stst.stability.l1(which_eig)*eye(2)-A_bar-B_bar*exp(-
stst.stability.l1(which_eig)*tau_a)-Ac_bar*exp(-
stst.stability.l1(which_eig)*tau_c)-Bc_bar*exp(-
stst.stability.l1(which_eig)*tau_s),(eye(2)+tau_a*exp(-
stst.stability.l1(which_eig)*tau_a)*B_bar+tau_c*exp(-
stst.stability.l1(which_eig)*tau_c)*Ac_bar+tau_s*exp(-
stst.stability.l1(which_eig)*tau_s)*Bc_bar)*v(:,which_eig_vec);d_n_v 0];
```

```
if which_k==1
    Q=zeros(3,1);Q(2,1)=v(2,which_eig_vec);
else
    Q=zeros(3,1);Q(2,1)=v(1,which_eig_vec);
end
        S1=inv(P)*Q;
         S=S1(3);
right_eigs=real(stst.stability.l1);
```

## Publications From This Thesis

1. D. K. Das and S. Ghosh, "Tolerable Input-Delay Margin Improvement by using Feedback Controller with an Artificial Delay", *Proceedings of the International Conference on CAMIST*, NIT, Rourkela, pp.547-552, Jan.2010.
2. ---, "Tolerable feedback delay margin improvement using dynamic state feedback controller with an artificial delay", To be communicated in Journal.