To the Graduate Council:

I am submitting herewith a dissertation written by Daniel Rangel entitled "OPTIMIZATION APPROACH TO CONTROLLING CONSTRAINED NONLINEAR SYSTEMS." I have examined the final copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Chemical Engineering.

T. W. Wang, Major Professor

We have read this dissertation
and recommend its acceptance:

Accepted for the Council:

Associate Vice Chancellor
and Dean of The Graduate School

OPTIMIZATION APPROACH TO CONTROLLING CONSTRAINED NONLINEAR SYSTEMS

A Dissertation

Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Daniel Rangel

May 1992

Copyright © Daniel Rangel1992
All rights reserved

To my parents

ACKNOWLEDGEMENTS

I would like to express my deepest appreciation to my advisor, Dr. Tse-Wei Wang. Without her technical guidance, supervision and encouragement this research would not have been possible. I am also greatly indebted to Professor John D. Birdwell for his invaluable contributions and suggestions in this work.

I also wish to express my appreciation to Professor Charles Moore for his insight and instruction. He has a unique talent to illustrate the industrial application of process control systems. I want to thank Dr. Duane Bruns for his suggestions and advice.

Special thanks to my friends John Webb, Pramit Sarma, and Roger Horn for their invaluable help, friendship, and support.

Finally I would like to thank my parents. Their unfailing support, love, care and understanding have sustained me during the course of my doctoral work.

ABSTRACT

The focus of this dissertation is the development of an efficient controller for nonlinear systems that handle constraints explicitly without requiring excessive computation time. The algorithm developed uses the input-output linearization technique (Isidori (1989) [35]) to extend the application of the Dynamic Matrix Control (DMC) algorithm (Cutler (1983) [16]) to nonlinear systems. Transformation methods based on differential geometry have been the only techniques that successfully extend some of the linear control theory to nonlinear systems. There has been no attempt to use the differential geometry techniques to extend the application of the DMC to nonlinear systems with constraints, mainly because of the difficulty in handling the constraints after the transformation. The method presented in this work uses the input-output linearization technique to transform the original nonlinear process model into a pseudo-linear one. The obtained model is valid for the entire region of operation. Unconstrained DMC is then applied to the linear system. Constraint violations are then handled by a pointwise optimization method that computes a modified input such that, at each sampling time, this modified input would satisfy the system constraints and at the same time would be "closest" to the optimum input as dictated by DMC. In this work, we will refer to this controller as the Exact-Linearized-Constraint Controller (ELCC).

The feasibility and effectiveness of the ELCC is demonstrated through computer simulation of three processes: a Continuous Stirred Tank Bioreactor (CSTBR), an exothermic Continuous Stirred Tank Reactor (CSTR) and a catalytic reactor. In the first two processes, it is desirable to operate as close to the optimal point (a singular point) as possible, at the same time maintaining closed-loop stability in the face of disturbances and

model uncertainties. Traditional linear control strategies cannot simultaneously satisfy these demands.

The combination of exact linearization with linear DMC presented in this work allows the selection of an arbitrary set point, regardless of its feasibility. When the stipulated set point becomes unachievable due to model parameter variations and disturbances, the ELCC will still operate the system in a stable manner, and at the same time drive the system to a steady-state level very close to the new maximum achievable point.

TABLE OF CONTENTS

CHAPTER		PAGE
1. INTRODUCTION		**1**
1.1	Statement of the Problem	3
1.2	Methodology	3
1.3	Interpretation	6
2. BACKGROUND		**8**
2.1	Transformation Methods	9
2.1.1	Lie Derivative	10
2.1.2	Relative Degree	10
2.1.3	Minimum Phase System	12
2.1.4	Limitations of Transformation Systems	14
2.2	Optimization Methods	14
2.2.1	Unconstrained Dynamic Matrix Control	17
2.2.2	DMC and QDMC Extensions	21
2.2.3	Nonlinear Optimization Methods	23
2.2.4	General Nonlinear Approach	24
2.3	Summary	29
3. CONTROL ALGORITHM		**31**
3.1	Input-Output Linearization	32
3.2	Prediction	36

3.3 Projection 38
3.4 Controller Tuning 42
3.5 Global Optimization 43
3.5.1 Conjugate Gradient Method 44
3.5.2 Global Optimization Method 45
3.5.3 Open Loop Optimization 46
3.5.4 Initial Guess 47

4. APPLICATIONS 48

4.1 Continuous Stirred Tank Bioreactor 49
4.1.1 Controller Design 53
4.1.2 Simulation Results 54
4.1.3 Linear Controller for a CSTBR 67
4.1.4 Nonlinear Controller with Global Optimization for a CSTBR: a Reference Case 74
4.1.5 Exothermic Reactor 81
4.1.6 Controller Design for a CSTR 88
4.1.7 Simulation Results for the CSTR 89
4.1.8 Linear QDMC Controller for a CSTR 99
4.1.9 Nonlinear Controller with Global Optimization for a CSTR: A Reference Case 103
4.2 Catalytic CSTR 109
4.2.1 Controller Design for a Catalytic Reactor 114
4.2.2 Simulation Results for the Catalytic CSTR 115

4.2.3 Nonlinear Controller with Global Optimization for a Catalytic Reactor: The Base Case 120

5. DISCUSSION OF RESULTS 126

5.1 Stability of a System that Uses the ELCC algorithm 127

5.1.1 Role of the Tunable parameter 'ϵ' 127

5.1.2 Final Steady-State Value for CSTBR and CSTR 131

5.1.3 Effect of Noise in the System 133

5.2 Comparison of the ELCC with a Nonlinear Controller that Uses a Global Optimization Technique 135

5.3 ELCC With Proportional Integral Controller 136

6. CONCLUSIONS AND RECOMMENDATIONS 144

6.1 Summary of Results 144

6.2 Appropriateness of the ELCC Controller 147

6.2.1 Up Front Work 147

6.2.2 Work at Each Sampling Time 147

BIBLIOGRAPHY 148

APPENDICES 155

A. Simulation Programs for the CSTBR 156

A.1 ELCC 156

A.2 Nonlinear Controller with Global Linearization 162

B. Simulation Programs for the CSTR 170

B.1 ELCC 170

B.2 Nonlinear Controller with Global Linearization for the CSTR 176

C. Simulation Programs for the Catalytic Reactor 186

C.1 ELCC 186

C.2 Nonlinear Controller with Global Linearization 193

D. Simulation Results 202

D.1 CSTBR 202

D.1.1 ELCC: Nominal Performance 202

D.1.2 ELCC: Parameter Variations 206

D.1.3 Global Optimization: Nominal Performance 212

D.1.4 Global Optimization: Parameter Variations 216

D.2 CSTR 218

D.2.1 ELCC: Nominal Performance 218

D.2.2 ELCC: Parameter Variations 222

D.3 Catalytic CSTR 226

D.3.1 ELCC: Nominal Performance 226

D.3.2 ELCC: Parameter Variations 228

VITA **230**

LIST OF TABLES

TABLE		PAGE
4.1	Control Parameters	54
4.2	Nominal Set Point	68
4.3	Parameters of the exothermic reactor	85
4.4	CSTR Control Parameters	89
4.5	QDMC paramters	99
4.6	Parameters of the catalytic reactor	113
4.7	Control Parameters for the Catalytic Reactor	116
5.1	Computational Work of the ELCC algorithm	136
5.2	Computational Work for the Global Optimization	137
5.3	Control Settings	137

LIST OF FIGURES

FIGURE		PAGE
1.1	Closed-loop control system for the pointwise optimization method	5
2.1	An exothermic CSTR	26
2.2	Catalytic CSTR	29
3.1	Flowchart of the controller design algorithm	33
4.1	Bioreactor	50
4.2	Steady state diagram of a bioreactor	55
4.3	Set point change in the total rate of growth by 10%. System trajectory	56
4.4	Set point change in the total rate of growth by 10%. Output versus time	57
4.5	Set point change in the total rate of growth by 10%. Input versus time	57
4.6	Set point change in the total rate of growth by 10% with initial dilution rate higher than D_{opt}. System trajectory	58
4.7	Set point change in the total rate of growth by 10% with initial dilution rate higher than D_{opt}. Output versus time	58
4.8	Set point change in the total rate of growth by 10% with initial dilution rate higher than D_{opt}. Input versus time	59
4.9	Set point change in the total rate of growth by 15%. System trajectory	60
4.10	Steady state diagram of a bioreactor. Effect of y_s. Solid line corresponds to nominal value, dash-line an increase of 10%, and dot-line a decrease of 10%.	61

4.11 Steady state diagram of a bioreactor. Effect of μ_{max}. Solid line corresponds to nominal value, dash-line an increase of 10%, and dot-line for decrease of 10%. 61
4.12 Change in y_s by -10%. Unachievable set point. System trajectory 63
4.13 Change in y_s by -10%. Unachievable set point. Output versus time 63
4.14 Change in y_s by -10%. Unachievable set point. Input versus time 64
4.15 Change in μ_max by -10%. Unachievable set point. System trajectory . . . 64
4.16 Change in μ_{max} by -10%. Unachievable set point. Output versus time . . 65
4.17 Change in μ_{max} by -10%. Unachievable set point. Input versus time . . . 65
4.18 Change in μ_max by +10%. Achievable set point. System trajectory . . . 66
4.19 Change in μ_{max} by +10%. Achievable set point. Output versus time . . . 66
4.20 Change in μ_{max} by +10%. Achievable set point. Input versus time 67
4.21 Set point change in the total rate of growth by 5%. System trajectory . . 69
4.22 Set point change in the total rate of growth by 5%. Output versus time . 69
4.23 Set point change in the total rate of growth by 5%. Input versus time . . 70
4.24 Set point change in the total rate of growth by 15%. System trajectory . 70
4.25 Set point change in the total rate of growth by 15%. Output versus time . 71
4.26 Set point change in the total rate of growth by 15%. Input versus time . . 71
4.27 Change in μ_{max} by -10%. Unachievable set point. System trajectory . . . 73
4.28 Change in μ_{max} by -10%. Unachievable set point. Output versus time . . 73
4.29 Change in μ_{max} by -10%. Unachievable set point. Input versus time . . . 74
4.30 Set point change in the total rate of growth by 15%. System trajectory . 76
4.31 Set point change in the total rate of growth by 15%. Output versus time . 77
4.32 Set point change in the total rate of growth by 15%. Input versus time . . 77

4.33 Set point change in the total rate of growth by 15%. System trajectory . 78
4.34 Set point change in the total rate of growth by 15%. Output versus time . 78
4.35 Set point change in the total rate of growth by 15%. Input versus time . . 79
4.36 Change in y_s by -10%. Unachievable set point. System trajectory 80
4.37 Change in y_s by -10%. Unachievable set point. Output versus time 80
4.38 Change in y_s by -10%. Unachievable set point. Input versus time 81
4.39 Change in μ_{max} by -10%. Unachievable set point. System trajectory . . . 82
4.40 Change in μ_{max} by -10%. Unachievable set point. Output versus time . . 82
4.41 Change in μ_{max} by -10%. Unachievable set point. Input versus time . . . 83
4.42 Exothermic CSTR 83
4.43 Steady state diagram of an exothermic reactor 86
4.44 Set point change in the concentration of B by 20%. System trajectory . . 90
4.45 Set point change in the concentration of B by 20%. Output versus time . 91
4.46 Set point change in the concentration of B by 20%. Input versus time . . 91
4.47 Set point change in the concentration of B by 25%. System trajectory . . 92
4.48 Set point change in the concentration of B by 25%. Output versus time . 92
4.49 Set point change in the concentration of B by 25%. Input versus time . . 93
4.50 Steady state diagram of a CSTR. Effect of k_1. Dash line corresponds to nominal value, solid line to an increase of 20%, and dot line to a decrease of 20%. 94
4.51 Steady state diagram of a CSTR. Effect of k_{-1}. Dash line corresponds to nominal value, Solid line to an increase of 20%, and dot line to a decrease of 20%. 95
4.52 Change in k_{-1} by +20%. Unachievable set point. System trajectory . . . 95

4.53 Change in k_{-1} by +20%. Unachievable set point. Output versus time . . 96

4.54 Change in k_{-1} by +20%. Unachievable set point. Input versus time . . . 96

4.55 Change in k_{-1} by +50% and A_i by -20%. Unachievable set point. System trajectory 97

4.56 Change in k_{-1} by +50% and A_i by -20%. Unachievable set point. Output versus time 98

4.57 Change in k_{-1} by +50% and A_i by -20%. Unachievable set point. Input versus time 98

4.58 Set point change in the concentration of B by 15%. System trajectory . . 100

4.59 Set point change in the concentration of B by 15%. Output versus time . 100

4.60 Set point change in the concentration of B by 15%. Input versus time . . 101

4.61 Change in k_{-1} by +20%. Unachievable set point. System trajectory . . . 102

4.62 Change in k_{-1} by +20%. Unachievable set point. Output versus time . . 102

4.63 Change in k_{-1} by +20%. Unachievable set point. Input versus time . . . 104

4.64 Set point change in the concentration of B by 20%. System trajectory . . 104

4.65 Set point change in the concentration of B by 20%. Output versus time . 105

4.66 Set point change in the concentration of B by 20%. Input versus time . . 105

4.67 Set point change in the concentration of B by 20%. System trajectory . . 106

4.68 Set point change in the concentration of B by 20%. Output versus time . 107

4.69 Set point change in the concentration of B by 20%. Input versus time . . 107

4.70 Change in k_{-1} by +20%. Unachievable set point. System trajectory . . . 108

4.71 Change in k_{-1} by +20%. Unachievable set point. Output versus time . . 108

4.72 Change in k_{-1} by +20%. Unachievable set point. Input versus time . . . 109

4.73 Change in k_{-1} by +50% and A_i by -20%. Unachievable set point. System trajectory 110

4.74 Change in k_{-1} by +50% and A_i by -20%. Unachievable set point. Output versus time 110

4.75 Change in k_{-1} by +50% and A_i by -20%. Unachievable set point. Input versus time 111

4.76 Catalytic CSTR 112

4.77 Marginally unstable set point. Output versus time 117

4.78 Marginally unstable set point. Input versus time 117

4.79 Marginally unstable set point. Output versus time 118

4.80 Marginally unstable set point. Input versus time 118

4.81 System response for a change in k_1 of +20%. Output versus time 119

4.82 System response for a change in k_1 of +20%. Input versus time 120

4.83 System response for a change in C_{B1} of -20%. Output versus time 121

4.84 System response for a change in C_{B1} of -20%. Input versus time 121

4.85 Marginally unstable set point. Output versus time 123

4.86 Marginally unstable set point. Input versus time 123

4.87 Marginally unstable set point. Output versus time 124

4.88 Marginally unstable set point. Input versus time 125

5.1 Change in μ_max by -10%. Unachievable set point. System trajectory for $\epsilon = 0.5$ 129

5.2 Change in μ_{max} by -10%. Qualitative representation of $L_g h(x)$, $u(t)$, and ϵ. 129

5.3 Change in μ_max by -10%. Unachievable set point. System trajectory for two different ϵ values. The dash line corresponds to $\epsilon = 1$. The dot line corresponds to $\epsilon = 4.0$ 130
5.4 System trajectory for different set points 132
5.5 System trajectory for two different ϵ values. The dash line corresponds to $\epsilon = 0.5$. The dot line corresponds to $\epsilon = 1.0$ 132
5.6 Change in y_s by -10% in the presence of random noise. Unachievable set point. System trajectory 133
5.7 Change in y_s by -10% in the presence of random noise. Unachievable set point. Output versus time 134
5.8 Change in y_s by -10% in the presence of random noise. Unachievable set point. Input versus time 134
5.9 Set point change in the total rate of growth by 14%. System trajectory . 138
5.10 Set point change in the total rate of growth by 14%. Output versus time . 140
5.11 Set point change in the total rate of growth by 14%. Input versus time . . 140
5.12 Change in y_s by -10%. Unachievable set point. System trajectory 141
5.13 Change in y_s by -10%. Unachievable set point. Output versus time 141
5.14 Change in y_s by -10%. Unachievable set point. Input versus time 142
5.15 Change in μ_max by -10%. Unachievable set point. System trajectory . . . 142
5.16 Change in μ_{max} by -10%. Unachievable set point. Output versus time . . 143
5.17 Change in μ_{max} by -10%. Unachievable set point. Input versus time . . . 143

D.1 Set point change in the total rate of growth by 50%. System trajectory . 202
D.2 Set point change in the total rate of growth by 50%. Output versus time . 203
D.3 Set point change in the total rate of growth by 50%. Input versus time . . 203

D.4 Set point change in the total rate of growth by 50% with initial dilution rate higher than D_{opt}. System trajectory. 204
D.5 Set point change in the total rate of growth by 50% with initial dilution rate higher than D_{opt}. Output versus time 205
D.6 Set point change in the total rate of growth by 50% with initial dilution rate higher than D_{opt}. Input versus time 205
D.7 Change in y_s by -12%. Unachievable set point. System trajectory 206
D.8 Change in y_s by -12%. Unachievable set point. Output versus time 207
D.9 Change in y_s by -12%. Unachievable set point. Input versus time 207
D.10 Change in y_s by +10%. Achievable set point. System trajectory 208
D.11 Change in y_s by +10%. Achievable set point. Output versus time 209
D.12 Change in y_s by +10%. Achievable set point. Input versus time 209
D.13 Change in μ_{max} by -12%. Unachievable set point. System trajectory . . . 210
D.14 Change in μ_{max} by -12%. Unachievable set point. Output versus time . . 211
D.15 Change in μ_{max} by -12%. Unachievable set point. Input versus time . . . 211
D.16 Set point change in the total rate of growth by 5%. System trajectory . . 212
D.17 Set point change in the total rate of growth by 5%. Output versus time . 213
D.18 Set point change in the total rate of growth by 5%. Input versus time . . 213
D.19 Set point change in the total rate of growth by 30%. System trajectory . 214
D.20 Set point change in the total rate of growth by 30%. Output versus time . 215
D.21 Set point change in the total rate of growth by 30%. Input versus time . . 215
D.22 Change in μ_{max} by -10%. Unachievable set point. System trajectory . . . 216
D.23 Change in μ_{max} by -10%. Unachievable set point. Output versus time . . 217
D.24 Change in μ_{max} by -10%. Unachievable set point. Input versus time . . . 217

D.25 Set point change in the concentration of B by 50%. System trajectory . . 218
D.26 Set point change in the concentration of B by 50%. Output versus time . 219
D.27 Set point change in the concentration of B by 50%. Input versus time . . 219
D.28 Set point change in the concentration of B by 10% with initial temperature higher than $T_{i,opt}$. System trajectory. 220
D.29 Set point change in the concentration of B by 10% with initial temperature higher than $T_{i,opt}$. Output versus time 221
D.30 Set point change in the concentration of B by 10% with initial temperature higher than $T_{i,opt}$. Input versus time 221
D.31 Change in k_1 by -15%. Unachievable set point. System trajectory 222
D.32 Change in k_1 by -15%. Unachievable set point. Output versus time 223
D.33 Change in k_1 by -15%. Unachievable set point. Input versus time 223
D.34 Change in k_1 by -20% and A_i by -18%. Unachievable set point. System trajectory 224
D.35 Change in k_1 by -20% and A_i by -18%. Unachievable set point. Output versus time 225
D.36 Change in k_1 by -20% and A_i by -18%. Unachievable set point. Input versus time 225
D.37 Marginally unstable set point. Output versus time 226
D.38 Marginally unstable set point. Input versus time 226
D.39 Marginally unstable set point. Output versus time 227
D.40 Marginally unstable set point. Input versus time 227
D.41 System response for a change in k_2 of -30%. Output versus time 228
D.42 System response for a change in k_2 of -30%. Input versus time 228

D.43 System response for a change in k_1 by -20% and C_{B1} by +10%. Output versus time 229

D.44 System response for a change in k_1 by -20% and C_{B1} by +10%. Input versus time 229

CHAPTER 1

INTRODUCTION

Chemical processes are inherently nonlinear. However, they have traditionally been operated using linear controllers because linear controllers are easier to design and their properties are better understood. Generally, these controllers are designed based on a locally linearized model of the process. The linearized model is usually obtained by selecting a steady-state set point of operation and applying a truncated Taylor's series expansion around that point. This approach yields satisfactory results only for continuous processes with relatively mild nonlinearities that are operated in the neighborhood of the linearization point. However, for processes that are seldomly operated at steady state (such as batch processes), or for highly nonlinear processes (such as polymerization processes), this local linearization approach often results in poor controller performance and system instability, when system trajectory has shifted or when the model has drifted.

During the last few years, two approaches have been explored to try to develop feedback controllers for highly nonlinear processes: (1) the geometric transformation approach and (2) the optimization approach. The transformation approach uses differential geometry to transform the original nonlinear process model into a pseudo-linear model (Byrnes and Isidori 1985 [8], Kravaris and Chung 1987 [42]). While the linearized model obtained from a truncated Taylor series expansion is valid only for a small neighborhood around the linearization point, the pseudo-linear model obtained from differential geometry is valid for the entire region of operation. The second approach uses nonlinear

optimization techniques to determine values of the system's manipulated variables that will optimize a given performance criterion (Brengel D. and W. Seider, 1989 [6], Sistu and Bequette 1989 [75], Zafiriou and Chiou [87]). In the optimization approach, the nonlinear dynamic model of the system is included as a constraint in the optimization.

The differential geometric-based transformational technique has the attractive point that it allows linear control algorithms to be extended to nonlinear systems. However, this technique is limited to minimum phase and deterministic processes and does not explicitly handle constraints placed on the system variables. On the other hand, controllers based on nonlinear optimization would handle such constraints explicitly, but they are usually computationally intensive and difficult to tune and analyze. These features render these nonlinear approaches unattractive for real-time implementation.

The serious practical drawbacks associated with the existing nonlinear control strategies indicate the need for a nonlinear control algorithm that would not only explicitly handle constraints, but also not require excessive computational time. The work presented in this dissertation is an attempt to generate such an algorithm.

The dissertation is divided into six chapters documenting this attempt. Chapter 1 contains the statement of the problem to be addressed, the methodology to be used in solving this problem, and the significance of the results. Chapter 2 presents a review of the literature that is the basis for the definition of this problem and the source of the methodology that is used in this project. Chapter 3 contains the detailed description of the control algorithm. Chapter 4 demonstrates the feasibility and effectiveness of the control algorithm through the computer simulation of three continuous nonlinear processes. Chapter 5 presents a discussion of the results obtained, and a comparison of performance with other types of controllers. Finally, Chapter 6 contains the conclusions

of this work and some recommendations.

1.1 Statement of the Problem

As indicated by the inability of existing nonlinear control strategies to control nonlinear processes efficiently, the problem can be stated as follows:

Design an efficient controller for nonlinear systems that will handle constraints explicitly without requiring excessive computational time.

1.2 Methodology

There exist many algorithms for the optimization of the performance of a system that is subject to equality and inequality constraints. If the system to be optimized is modeled by a set of nonlinear differential equations, the problem is one of variational calculus. The solution of the conditions necessary for optimality of this problem using traditional methods requires the iterative solution of a two-point boundary problem, making it impractical for most engineering applications due to computational and storage requirements.

Several algorithms applicable to linear systems have been used to solve this optimization problem, such as Dynamic Matrix Control (DMC) and Quadratic Dynamic Matrix Control (QDMC) (Garcia and Morshedi in 1986 [20]). These algorithms give good results for near-linear systems. However, these same optimization techniques have also been used to address difficult control situations in chemical processes. The majority of such difficult processes are inherently highly non-linear, making linear models poor descriptions.

Extended versions of these linear algorithms have been developed in an attempt to accommodate the nonlinear dynamics. These extensions use a repeated local linearization to approximate the nonlinearities, and then the entire DMC or QDMC algorithm is applied at each point of linearization. For very nonlinear systems, the number of such points can be very high, leading to very large computational costs (Brengel and Seider (1989) [6]). If instead, differential geometry techniques were used to transform the original nonlinear process model into a pseudo-linear one, the obtained model will then be valid for the entire region of operation. This method has not been more widely used, mainly because of the difficulty in handling the constraints after the transformation.

In this dissertation, the approach is to integrate the input-output geometric transformation technique with the model-predictive-control technique. The input-output linearization technique is used first to generate a pseudo-linear model of the system that is valid for the entire region of operation; and then the model-predictive-control technique is designed for the pseudo-linear system, without regard to constraint consideration. Then at each sampling time, the calculated control input is checked for constraint violation. If found in violation, a pointwise projection is used to project the calculated input into the space of allowable inputs. A block diagram of the proposed approach is shown in Fig 1.1. Since this approach involves a point projection step, the computational time would be minimal. For future reference, our nonlinear controller shall hence forth be referred to as the Exact-Linearized-Constraint Controller (ELCC).

In order to establish the relative advantages and disadvantages of ELCC with reference to a given system, results obtained by applying the above scheme will be compared to those obtained using a global optimization method, which renders the best possible performance if computational time was not a consideration. The global optimization

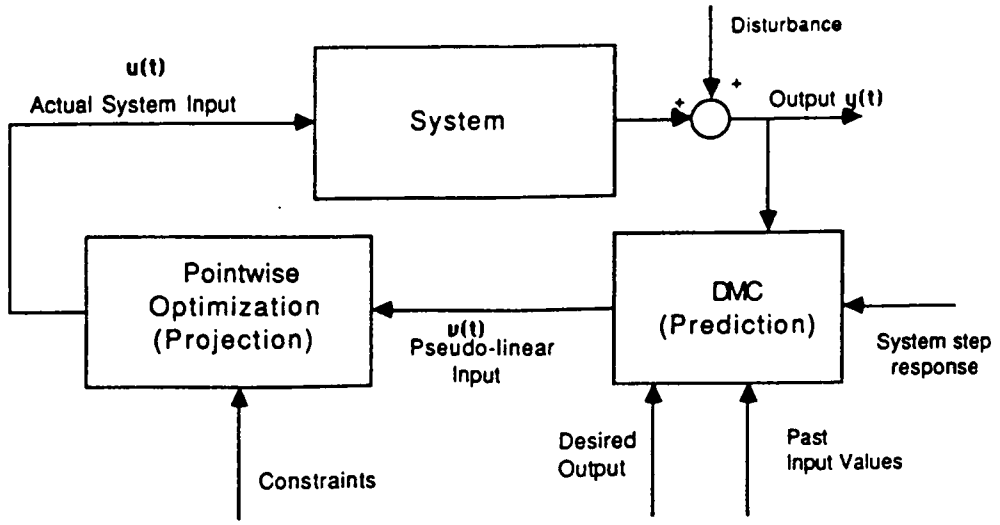

Figure 1.1: Closed-loop control system for the pointwise optimization method

will be performed using the conjugate gradient method in a closed-loop format in order to minimize the effect of modeling errors and disturbances.

The conjugate gradient method is one of the most widely used methods for minimizing a function. In this method, the optimization is performed by starting with an initial guess, and then by searching along the direction of the negative gradient of the function to be optimized, to reach the next guess. If the guessed value is relatively close to the optimal one, the optimization will converge after only a few iterations. The optimization is repeated at each sampling time, with the hope that the system conditions have not abruptly changed between successive sample intervals. Thus, information from the preceding optimization can be used for the estimation of the initial guess for the next optimization. This recycling of information from the system's past performances

is expected to reduce the number of iterations required to perform each successive optimization. If the system conditions change abruptly or when a new set point is specified, the DMC algorithm will be used to provide the new initial guess.

1.3 Interpretation

The performance of the ELCC is expected to surpass that of linear controllers in several respects:

- It will converge to the desired point of operation regardless of the initial conditions of the system.
- When the desired set point becomes unachievable, it will operate the system at its maximum possible optimal conditions without causing instability.

The ELCC is more attractive for real-time implementation than existing nonlinear controllers due to:

- Its ability to handle system constraints explicitly.
- It reduces computational requirements.

The new algorithm will use a model of the system to perform the optimization. Generally this model is not an exact representation of the system. Even when the model structure is relatively certain, the model parameters may not be known precisely; even if known precisely at time zero, they will likely vary during the course of operation. Therefore, a crucial test for the controller is its performance in the face of parameter variations. It is expected that the use of a pseudo-linear model valid for the entire region

of operation will allow the controller to maintain stability in the face of those parameter variations; whereas a locally linear controller is expected to fail, leading to instability under similar test conditions.

CHAPTER 2

BACKGROUND

Nonlinearity is a major obstacle encountered in process control. A traditional approach to this problem is to apply a linear control design strategy to a system linearized locally at a steady-state operating point by a Taylor series expansion around that point. The performance of these locally linearized systems is satisfactory (1) only in the neighborhood of the linearization point, or (2) if the process has only mild nonlinearities. However, in the presence of strong nonlinearities or when the process deviates significantly from the linearized operating point, the performance of the locally linearized system can be severely degraded. These manifestations have evoked the need for novel approaches to nonlinear control synthesis and analysis. Several new linearization methods that do not depend on the local point of operation have been developed during the last ten years. These methods allow the extension of linear control theory to nonlinear systems. They have been used in applications illustrated by Ashutosh *et al.* (1989) [3], Economou (1986) [17], and Peterson (1989) [60].

Most of the current nonlinear control strategies can be classified into two main categories: transformation methods and optimization methods. The transformation methods use differential geometry techniques to transform a system's nonlinear dynamics into pseudo-linear ones via state feedback, while trying to preserve the most important control features of the system (Kravaris (1987) [40], Isidori (1989) [35]). The optimization methods use nonlinear programming algorithms to calculate the values of the process

variables that best satisfy a given performance criterion (Ashutosh *et al.* (1989) [3], Sistu and Bequette (1989) [75]). Each approach has advantages and disadvantages that will be explored in this chapter.

Chapter 2 is divided into three sections. Section 2.1 presents an overview of the transformation methods and reviews some basic concepts of differential geometry. Section 2.2 reviews the optimization methods for both linear and nonlinear systems, including some of their limitations and advantages. Finally, Section 2.3 summarizes the contents of this chapter and points to the direction of the proposed research.

2.1 Transformation Methods

The transformation methods use differential geometry techniques to find a state feedback law such that the transformed system is linear. From a theoretical standpoint, there are two major types of transformations: global linearization and input-output linearization. Global linearization refers to making linear the dynamics between the transformed inputs and system states; input-output linearization refers to making linear only the transformed system input-output behavior. Though applications of global linearization have been reported by Hoo and Kantor (1985) [33] and others, Kravaris and Kantor (1990) [41] have pointed out two major limitations of this approach:

- Global linearization requires the solution to a system of partial differential equations, for which analytical solution rarely exists and the numerical solution is difficult to implement.

- Global linearization assumes the regulated output variables to be linear functions of the state variables of the globally transformed linear system.

These factors limit the application of the global linearization technique in the controller design in a general setting. In contrast, the input-output linearization method provides an algorithm for directly finding a nonlinear-state feedback law, such that the regulated outputs of the systems are linear functions of the transformed inputs. Thus a linear control algorithm can be directly applied to the input-output linearized system.

A brief review of basic differential geometric notations used in the transformation methods and of input-output linearization procedures is given below. The formulas and terminologies given in this section can be found in any standard textbook of differential geometry (e.g. Boothby, 1975 [7]). For further details regarding geometric linearization theory, one can refer to Isidori (1989) [35].

2.1.1 Lie Derivative

The Lie derivative of a scalar function $h(x) : \mathbf{R}^n \longrightarrow \mathbf{R}$ with respect to a vector field $f(x) : \mathbf{R}^n \longrightarrow \mathbf{R}^n$ is defined as:

$$L_f h = (\nabla h)^T f \tag{2.1}$$

where $(\nabla h)^T$ represents the transpose of the gradient of h with respect to x. Higher order Lie derivatives can be defined as follows:

$$L_f^k h = (\nabla (L_f^{k-1} h))^T f \tag{2.2}$$

2.1.2 Relative Degree

The relative degree represents the number of times the output of a system must be differentiated with respect to time in order to recover an explicit relationship between the system's output and input.

Consider a nonlinear system of the form

$$\begin{aligned} \dot{x} &= f(x) + g(x)u \\ y &= h(x) \end{aligned} \tag{2.3}$$

where x is an n-dimensional state vector, u is an m-dimensional input vector, and y is an m-dimensional output vector. It is assumed that the functions $f(x)$, $h(x)$ and $g(x)$ are smooth vector fields on R^n.

The system output can be differentiated with respect to time as:

$$\frac{dy}{dt} = \frac{\partial h}{\partial x}\frac{dx}{dt} \tag{2.4}$$

This equation can be rewritten as:

$$\dot{y} = \nabla h(x)^T f(x) + \nabla h(x)^T g(x)u \tag{2.5}$$

or

$$\dot{y} = L_f h(x) + L_g h(x)u \tag{2.6}$$

If $L_g h(x) \neq 0$, then the system has a relative degree of one because y has been differentiated only one time. If $L_g h(x) = 0$, then the output is differentiated again with respect to time. The process is repeated until the input u appears explicitly. Therefore, a system has relative degree r at x_i if there exists an open set $B(x_i) \in R^n$ such that for all $x \in B(x_i)$

$$\begin{aligned} L_g L_f^i h &= 0, \qquad \forall\, 0 \leq i < r-1 \\ L_g L_f^{r-1} h &\neq 0 \end{aligned}$$

For a nonlinear system with relative degree r, the rth-derivative of the output with

respect to time can be written as:

$$\frac{d^r y}{dt^r} = L_f^r h(x) + L_g L_f^{r-1} h(x) u \tag{2.7}$$

If the right side of Eq 2.7 is set equal to a new variable v, then the following linear system is obtained:

$$\frac{d^r y}{dt^r} \equiv v \tag{2.8}$$

Then the input transformation may be given by:

$$u(x, v) = (L_g L_f^{r-1} h(x))^{-1} (v - L_f^r h(x)) \tag{2.9}$$

Equation 2.8 gives a relationship between the new input v and the system output y, and allows selection of v that will produce a desired output. However, it does not guarantee that the internal dynamics of the system will remain bounded.

The internal behavior of a nonlinear system can be studied by studying the system's dynamics when the control input is such that the output y is made identically zero. These dynamics are called *zero dynamics*. The zero dynamics are used to study the internal stability of the system. The Input-Output linearization approach is feasible only when the zero dynamics of the system are stable.

2.1.3 Minimum Phase System

A nonlinear system is called minimum phase if its corresponding zero dynamics are stable; otherwise it is *non-minimum phase*.

Isidori (1989) [35] has shown that if the zero dynamics of a nonlinear system of relative degree one is asymptotically stable at x_i, then the system can be asymptotically stabilized locally at x_i by means of a feedback control:

$$u = -Ky + v \tag{2.10}$$

A similar proposition has been proven for nonlinear systems having a relative degree r (Isidori (1989) [35]). This implies that the states of a nonlinear system, with a feedback control in the form of Equation 2.10 and with a properly chosen k, will be guaranteed asymptotically stable even though they are unmeasured. This becomes a very important issue in the consideration of plant safety when only a limited number of output variables are to be controlled. The following section on the global linearization methodology shows that the form of the feedback u does follow the form as described in Equation 10.

Using the definitions stated in Sections 2.1.1 through 2.1.3, the condition for a nonlinear system to be *input-output* linearizable is presented as: (Slotine (1991) [76])

Proposition 1 *A nonlinear system is input-output linearizable if and only if it is minimum phase and its relative degree is* $r \leq n$.

Thus, if the linearizability condition is satisfied, the transformation

$$v = L_f^r h + L_g L_f^{r-1} h u \tag{2.11}$$

linearizes the input-output map to a reduced order system as

$$\dot{z} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} z + \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} v$$

$$y = [1\,0\,0 \ldots 0]\, z \;;$$

where z represents the states of the pseudo-linearize system; or in transfer function representation

$$\frac{y(s)}{v(s)} = \frac{1}{s^r}. \tag{2.12}$$

Thus, between the transformed input, $v(s)$, and the output, $y(s)$, the dynamic is linear. As a result, a linear controller can be design to control the transformed system.

2.1.4 Limitations of Transformation Systems

Some of the limitations of the transformation methods were noted by Sistu and Bequette (1989) [75]. The most important limitations of these methods are:

- They require that all state variables be measurable or estimatable.
- They do not explicitly handle constraints.
- They do not explicitly handle time-delays.
- They are only applicable to minimum phase processes.
- They do not allow the existence of model uncertainty, because exact cancellation of the model nonlinearity is achieved during the transformation procedures.

Though limited in application to minimum phase processes, the transformation methods have been the only technique that provides a theoretically rigorous approach to extend the applications of linear control algorithms to nonlinear systems. As a result, their performance is expected to be much superior for higly nonlinear systems.

2.2 Optimization Methods

A second general approach to nonlinear control design is global optimization. The objective of the optimization method is to select the set of values of the manipulated variables (system inputs), as function of time that would optimize the overall performance

of the system, as judged by a performance criterion. According to Athans and Falb (1966) [4], in order to accomplish this objective, the method requires the specification of the following elements:

- A model of the system.
- A set of desired outputs or target values.
- An objective function or performance index.
- A set of admissible input values.

The mathematical model is generally a set of differential equations, and is used to predict the effect of changes of the manipulated variables on the outputs of the system. The set of desired outputs is used to calculate the objective function. The most common objective function is the sum of the squares of the differences between the target values and expected system outputs over the prediction horizon. A more general cost function may be specified, which may include economic criteria or final condition relationships. Finally, optimization requires a set of admissible input values to avoid the selection of changes in the manipulated variables that are not physically achievable or that violate process specifications.

For a given cost function, the problem can be formulated as determining the admissible inputs that generate the desired outputs and that, in so doing, minimize (optimize) the chosen performance index. Several algorithms can be used to find the solution to this problem, as shown in Athans and Falb (1966) [4]. Due to the presence of disturbances, unmodeled dynamics, or parameter variations, the direct implementation of this solution may result in suboptimal or even undesirable performance. Therefore, the presence of

these conditions requires an iterative solution to the problem, such that a new set of inputs is calculated as the predicted output changes. Dynamic Matrix Control (DMC) (Garcia and Morshedi, 1986 [20]) is an example of an algorithm that uses an iterative solution of the optimization problem to minimize errors due to unmodeled dynamics or disturbances. DMC calculates a set of future changes in the manipulated variables, but only implements the first change. The output obtained after the first change in the input has been implemented is compared with the estimated output. Then the difference between the outputs is added to future estimations to account for the effect of unmodeled dynamics or disturbance, and the sequence is repeated again at the next sampling interval. This algorithm has been used in industrial applications with considerable success (Garcia and Morshedi, 1986 [20]).

The inherent linear model of DMC restricts its use in highly nonlinear processes. However, it possesses advantages that are highly desirable in any controller:

- It is not affected by the presence of time delays in the system.
- It does not require an explicit system model.
- It can be easily extended to systems having multiple inputs and multiple outputs.
- It can accommodate constraints.

Therefore, great efforts have been made to extend its applicability to nonlinear systems. Extended DMC (Arkun 1988) [60] and Industrial Model-Predictive Control (Garcia and Prett 1986 [21]) are examples of such attempts. These extensions have had limited success, as pointed out by Brengel and Seider (1989) [6]; therefore, a more general approach has been attempted. The general approach formulates the problem as selecting

the future values of the manipulated variables that optimize the performance of a system that is subject to nonlinear differential equation equality constraints. This problem is solved using nonlinear optimization routines iteratively. The applicability of this general approach has been limited by the large computational capability required.

In the following sections, a brief review is presented of the fundamental algorithms of several of the optimization methods mentioned. In Section 2.2.1, the DMC algorithm is presented with one of its variations, the Quadratic/Dynamic Matrix Control (QDMC). In Section 2.2.2 the extensions to nonlinear systems are presented, with each extension's advantages and limitations listed. Finally, in Section 2.2.4, the more general nonlinear approaches are reviewed.

2.2.1 Unconstrained Dynamic Matrix Control

Dynamic Matrix Control constitutes one of the first applications of large-scale dynamic optimization to chemical processes. The DMC algorithm was developed at Shell Oil in the late 1970s by the teams of Cutler and Ramaker and of Prett and Gillette [16]. It has been applied successfully to a variety of industrial processes (Garcia and Prett 1986 [21]). It uses the open-loop step response to estimate the system output, and attempts to minimize the sum of square deviations between the projected and desired outputs over a predictive horizon. DMC was the first control technique that could successfully handle process constraints. A brief review of the single-input, single-output (SISO) DMC algorithm is given here, from a more detailed description in Garcia (1986) [20]:

The output of a SISO system can be computed from the step response model coefficients, a_i, as follows:

$$y_k = y_o + \sum_{i=1}^{M} a_i m_{k-i} + d_k \tag{2.13}$$

where

- y_k is the value of the output y at sampling time k;
- a_i is the ith sample unit step response coefficient;
- m_{k-i} is the change in the input (or manipulated variable) at sampling time $k-i$;
- y_o is the value of y when $k = 0$;
- M is the number of time intervals required for the system to reach steady-state;
- d_k is included to take into account unmodeled effects and external disturbances on the measured output at the sampling time k.

If the output calculated by Equation 2.13 corresponds to a future output, the right side of the equation contains the effect of past and future moves of the manipulated variable. If we define Yb as the contribution to the output due to past inputs, the output for any future time $k+l$ can be written as:

$$y_{k+l} = \sum_{i=1}^{l} a_i m_{k+l-i} + Yb_{k+l} + d_{k+l} \tag{2.14}$$

where l is the number of future moves, and Yb is the contribution to the output due to the past $M-l$ moves, with respect to the kth sample interval of the manipulated variable, and is given by the equation:

$$Yb_{k+l} = y_o + \sum_{i=l+1}^{M} a_i m_{k+l-i} \tag{2.15}$$

The DMC algorithm uses equations 2.14 and 2.15 to express the future outputs as a function of the past and future changes in the manipulated variable. The projected

outputs for times $k+1$ up to $k+P$ can be written as:

$$\begin{bmatrix} y_{k+1} \\ \vdots \\ y_{k+P} \end{bmatrix} = \begin{bmatrix} Yb_{k+1} \\ \vdots \\ Yb_{k+P} \end{bmatrix} + Ax_k + \begin{bmatrix} d_{k+1} \\ \vdots \\ d_{k+P} \end{bmatrix}, \tag{2.16}$$

where A is given by

$$A = \begin{bmatrix} a_1 & 0 & \cdots & 0 \\ a_2 & a_1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ a_N & a_{N-1} & \cdots & a_1 \\ \vdots & \vdots & \vdots & \vdots \\ a_M & a_{M-1} & \cdots & a_{M-N+1} \\ \vdots & \vdots & \vdots & \vdots \\ a_M & a_M & \cdots & a_M \end{bmatrix},$$

where x is the vector of future input moves

$$x_k = \begin{bmatrix} m_k \\ \vdots \\ m_{k+N-1} \end{bmatrix}$$

and N represents the number of future input moves to be computed. Because the purpose of the algorithm is to calculate a set of future input moves that would produce a desired output trajectory, the projected outputs, by Equation 2.16, are replaced by the target value r.

The value of d_k is estimated as:

$$d_k = Yb_k - y_k \tag{2.17}$$

Equation 2.17 shows that future values of d_k cannot be calculated. Therefore, it is assumed that any future value of d_k is equal to the present d_k. Using this assumption, Equation 2.16 can be rewritten as:

$$\begin{bmatrix} r - Yb_{k+1} - d_k \\ \vdots \\ r - Yb_{k+P} - d_k \end{bmatrix} = ek + 1 = Ax_k \tag{2.18}$$

Then, the DMC problem is defined as finding the vector of N future input changes x_k that minimizes the magnitude of e_{k+1}, the length of the predicted future error vector. The least square solution to Equation 2.18 is given by

$$x_k = A^+ e_{k+1} \tag{2.19}$$

where A^+ denotes the pseudo-inverse of A. Equation 2.19 is used to calculate the next N future input moves at each sampling time, while only the first move is to be implemented at the next time interval. As a result, the algorithm allows for the correction of the effect of unmodeled dynamics or external disturbances, through the d_k term.

Though computationally more involved than standard linear time-invariant algorithms, DMC possesses distinct advantages that make it attractive for industrial applications. The ability to accommodate constraints is one of the most important features of the DMC algorithm. Arkun (1978) [1] and Prett and Gillette (1979) [63] demonstrated that in practice, the operating point of a plant that satisfies the overall economic goals of the process will usually lie at the intersection of constraints. This demonstration has motivated an effort to modify several existing control algorithms to improve their constraint handling capabilities. Quadratic/Dynamic Matrix Control (QDMC) is one such modification. QDMC was developed by Garcia and Morshedi in 1986 [20]. It is a

variation of the DMC algorithm that provides a rigorous handling of constraint violations by expressing the problem as a quadratic minimization, subject to linear inequality constraints. The problem is formulated as

$$\min_{x_k} J = \sum \frac{1}{2}(Ax_k - e_{k+1})^T(Ax_k - e_{k+1}) + \frac{1}{2}x_k^T x_k) \tag{2.20}$$

subject to

$$Cz_k \geq c_{k+1} \tag{2.21}$$

$$x_{min} \leq x_k \leq x_{max} \tag{2.22}$$

$$z_{min} \leq z_k \leq z_{max}, \tag{2.23}$$

where the matrix C contains dynamic information on the constraints; the vector c_{k+1} contains the projected deviations of the constrained variables from their limits; z is the vector of state variables; and the inequalities 2.22 and 2.23 contain the bounds on the state and manipulated variables. Equation 2.20 is solved using linear programming techniques.

2.2.2 DMC and QDMC Extensions

Even though DMC and QDMC optimization methods show certain superior characteristics among linear controllers, their inherent assumption of a linear model restricts their use in highly nonlinear systems. Therefore, extended versions of these methods have been developed in an attempt to accommodate nonlinear dynamics. Two of these methods are described below.

Industrial Model-Predictive Controller

Developed by Garcia and Prett in 1986 [21], the Industrial Model-Predictive Controller uses Taylor's series expansion to linearize the system locally; then, a DMC or QDMC algorithm is applied to the linearized process. As mentioned in Section 2, local linearization is valid only in the neighborhood of the linearization point. Therefore, to avoid degradation in the system performance, the system has to be linearized periodically.

The major advantage of the Industrial Model-Predictive Controller is its simplicity. If the system has only mild nonlinearities, the algorithm does not require large numbers of computations. However, if the system is highly nonlinear, the model of the process has to be linearized frequently. The algorithm does not state how often the linearization should be performed to avoid suboptimal performance. Another limitation of the Industrial Model-Predicted Controller is that it corrects for unmodeled dynamics and disturbances using the principle of superposition, which is valid only for linear systems. Brengel and Seider (1989) [6] pointed out that, in general, the Industrial Model-Predictive Controller produces suboptimal performance for highly nonlinear systems.

Extended DMC

Extended DMC was developed by Peterson, Hernandez, Arkun, and Schork in 1988 [60]. It uses the nonlinear model to calculate the system step response coefficients as follows: First, the output of the system is calculated for a given input by integrating the nonlinear differential equations that describe the system. Then, the output is calculated again for a different change in the input value. The step-response coefficients can be calculated by subtracting the two outputs at the corresponding time intervals and dividing the result by the magnitude of the change in the input. The DMC algorithm as described

in Section 2.2.1 is then applied. However, the original nonlinear dynamic equations are used to generate the disturbance vector for feedback.

The extended DMC algorithm has two major limitations:

- It does not take into consideration the presence of constraints.
- It uses an external nonlinear model that is different from the internal linear model of DMC. This difference violates the internal model control structure and results in steady state offsets.

2.2.3 Nonlinear Optimization Methods

Due to the difficulty encountered in the extension of DMC and QDMC to nonlinear systems, several iterative optimization methods based on nonlinear models have been developed. Two approaches have been used to develop the nonlinear optimization algorithms. The first approach modifies existing nonlinear feedback controllers to allow them to handle constraints explicitly. The second approach states the optimization problem in a general form subject to the nonlinear dynamic differential equation equality constraints, and tries to solve the problem using existing nonlinear techniques. An example of the first approach, the Constrained Pseudo-Newton Controller, is presented below. In section 2.2.4, the more general nonlinear approach is reviewed.

Constrained Pseudo-Newton Controller

The Constrained Pseudo-Newton Controller was developed by Biegler and Li in 1989 [46]. It is a modification of the Nonlinear Internal Model Controller (Nonlinear IMC) reported by Economou and Morari in 1986 [17]. The Nonlinear IMC uses Newton's method to

compute the value of the manipulated variable that, if held constant during the next sampling interval, will produce the desired output. Li and Biegler expressed the Newton-type control law as a quadratic programming problem, and solved it using successive quadratic programming (SQP). SQP approximates the cost function by a locally quadratic function and approximates the constraints by locally linear functions. By solving this approximate problem using quadratic programming, a search direction is obtained. The cost function is minimized in this search direction, and the process is repeated at the new location until the optimum is determined.

The Constrained Pseudo-Newton Controller can only predict the output of the system for the next sample interval, thus limiting its ability to foresee constraint violations. Therefore, the algorithm may experience difficulties in avoiding process constraints at a future time. A second undesirable consequence of the single prediction is that it may result in an overall suboptimal performance.

2.2.4 General Nonlinear Approach

A more general approach to the optimization problem has been suggested by Brengel and Seider (1989) [6], Renfro, Morshedi and Asbjornsen (1987) [67], and Sistu and Bequette (1990) [75]. In this approach, the system to be optimized is modeled by a set of nonlinear ordinary differential equations (ODEs) ($\dot{x} = f(x,u)$, $y = h(x)$). A mathematical formulation of this approach to the optimization problem is:

$$\min_{u_k,\cdots,u_{k+M}} \sum_{i=1}^{P} [(r - y_{k+i})^T (r - y_{k+i}) + (u_{k+i-1} - u_{k+i-2})^T (u_{k+i-1} - u_{k+i-2}) \quad (2.24)$$

subject to

$$\dot{x} = f(x,u) \quad (2.25)$$

$$z(x,u) = 0 \tag{2.26}$$

$$l(x,u) \leq 0, \tag{2.27}$$

where equations 2.26 and 2.27 represent the equality and inequality constraints on the process operation, Equations 2.25 represents the ODEs that describe the system, r represents the desired output vector, M represents the number of future input moves, and P represents the number of sample intervals for which the output of the system will be predicted.

There are three methods used to solve this general nonlinear optimization problem:

- Sequential Methods.
- Simultaneous Methods.
- Linear approximation Methods.

These methods are described in the sections below, along with some of their advantages and disadvantages.

Sequential Methods

The sequential methods (Jang 1987 [36]) use separate algorithms to solve the system of ODEs that describe the system, and to perform the optimization. The iterative solution of the ODEs is performed at an inner loop to provide the gradient information required to do the optimization. The optimization is performed at an outer loop.

Jang, Babu and Hiro [36] applied the sequential algorithm to the control of an exothermic reactor as shown in Fig 2.1. This reactor example is the same as that used by Economou and Morari (1986) [17].

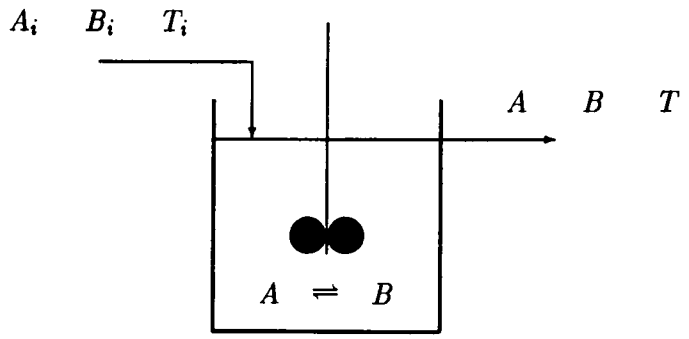

Figure 2.1: An exothermic CSTR

The process consists of an ideal Continuous Stirred Tank Reactor (CSTR), where a reversible chemical reaction

$$A \rightleftharpoons B$$

is carried out. The objective is to control the exit concentration of B by manipulating the inlet temperature T_i. Three different control strategies were implemented and compared. The first was a conventional PID controller; the second was a self-tuning adaptive controller; and the third was the sequential nonlinear optimization controller. The optimization approach proved to be superior to the other two strategies because it was the only strategy able to maintain stability when the desired set point was unachievable. The sequential nonlinear optimization technique uses a better description of the system than the traditional linear controllers. This improved description allows the controller to predict when the set point is unachievable. Then, if the set point is unachievable, the controller maintains the system at the maximum achievable point. However, the

sequential methods are computationally intensive. They require iterative integration of the system nonlinear ODEs. Jones and Finch (1984) [37] found that these methods use about 85% of the computational time integrating the model equations. These computational requirements can make the sequential methods impractical and unattractive for use in real-time applications.

Simultaneous Methods

The simultaneous solution technique (Eaton and Rawlings 1990 [18]) calls for the use of orthogonal collocation methods on finite elements to transform the system nonlinear ODEs into algebraic equations. This method involves the replacement of the independent and dependent variables by a set of approximating functions. Among the wide variety of approximating functions that can be used to replace the dependent variables, orthogonal polynomials are the most commonly used. After the differential equations have been transformed, the problem is solved using sequential quadratic programming.

Eaton and Rawlings tested this algorithm by simulating the set point tracking in a CSTR in which the reaction taking place is

$$A + A \rightarrow B$$

The controller adjusts the feed flow rate in order to regulate the concentration of A in the reactor. Parameter variations were implemented to study the effect of plant/model mismatch. The controller based on simultaneous optimization was able to track the desired achievable concentration even for large parameter variations.

Though the simultaneous methods reduce computational time considerably, they are generally difficult to implement and analyze. The solution of the nonlinear algebraic optimization problem is nontrivial. Also, the effect of the convergence tolerance of the

transformed system on the overall optimization problem is poorly understood. Brengel and Seider (1988) [6] pointed out that the selection of the trial functions significantly impacts the performance of the algorithm, thus making the controller structure difficult to tune and analyze.

Linear Approximation Methods

Linear approximation was developed by Brengel and Seider (1988) [6]. It uses a Taylor series expansion to develop a recursive formula that locally linearizes the system differential equations. To avoid performance degradation, the recursive formula is applied at every sampling time. If the system is highly nonlinear, it may require linearization in between sampling times. After the equality constraint is linearized, the problem can be solved using nonlinear programming.

Brengel and Seider used the algorithm to control a two-input, two-output catalytic reaction that is carried out in a CSTR. The reaction examined was:

$$A + B \rightarrow P$$

The system uses two feed streams with different concentrations of B, as shown in Fig 2.2. Based on the linear approximation method, the controller should maintain the product concentration and the liquid level in the reactor at specified values by adjusting the flow rates of the incoming reactants. This is the same problem used by Li and Biegler (1988) [46] to test the Pseudo-Newton Controller. Brengel and Seider used the same sampling time as Li and Biegler to facilitate a direct comparison between the two methods. The results show that the linear approximation algorithm produces smaller overshoot and settling time than the Pseudo Newton Controller, thus reducing the computational time considerably. Nevertheless, the implementation of the linear approximation algo-

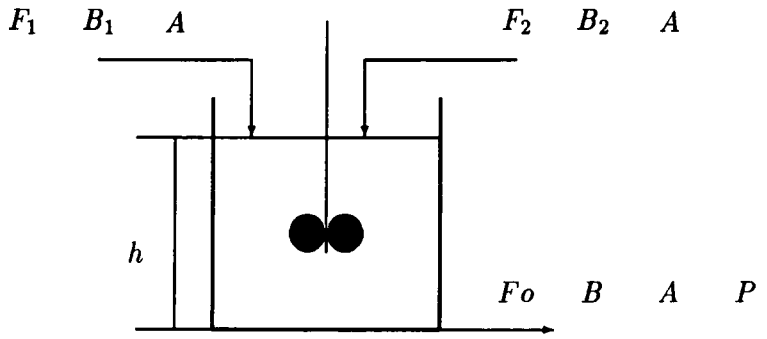

Figure 2.2: Catalytic CSTR

rithm in real time may require a large computer capability. According to Brengel and Seider [6], the computational requirements increase linearly with the number of linearizations performed.

2.3 Summary

The control of chemical processes may be stated naturally as a nonlinear optimization problem, subject to differential constraints. To minimize the effect of modeling errors and disturbances on the system performance, the solution of this problem should be carried out in a closed-loop form. The existing algorithms applicable to nonlinear systems either require large computer facilities and/or are difficult to tune, analyze and implement, thus making them unattractive for applications in real-time processes.

The transformation method based on differential geometry (Hoo and Kantor [33],

Kravaris and Kantor [41]) has been the only technique that successfully extend some of the linear control theory to nonlinear systems. There has been no attempt to use differential geometry techniques to extend the application of DMC and QDMC optimization methods to nonlinear systems, mainly because of the difficulty in handling the constraints after the transformation. A method that would incorporate the constraint handling capability of the DMC and QDMC algorithms, and is applicable to nonlinear systems without requiring excessive computational time, is highly desirable. The work described in the next section is an attempt to achieve this goal.

CHAPTER 3

CONTROL ALGORITHM

The development of the Exact-Linearized-Constraint Controller (ELCC) algorithm is divided into three parts: linearization, prediction, and projection. In the linearization part, the original nonlinear system model is transformed into a pseudo-linear one, using the differential geometry techniques introduced in Chapter 2. In the prediction part, an unconstrained linear model-predictive-control technique is applied to the linearized model in order to find the values of the pseudo-linear inputs that will be required to optimize an arbitrarily chosen performance criteria. Finally, in the projection part, these desirable predicted future input values are checked for constraint violation and if found in violation are projected onto the domain of the allowable inputs. The projection is computed at each sampling instant to minimize the distances between the actual inputs and the desirable predicted inputs, subject to the system's constraints.

The first two parts of the ELCC algorithm integrate the input-output linearization technique with the model-predictive-control technique. The input-output linearization technique is used to generate a linear model of the system that is valid for the entire region of operation, and the model-predictive-control technique is used to control the system based on the linearized model. However, if the system is subject to constraints, the input trajectory calculated using the model-predictive-technique may not be implementable. Therefore, a pointwise optimization procedure is used to project the desired input trajectory onto the space of allowable system inputs.

Three parts are presented individually in Sections 3.1, 3.2, and 3.3. A flowchart of the algorithm is shown in Fig 3.1. Section 3.4 presents recommended guidelines for tuning the controller.

In order to establish the relative advantages and disadvantages of the developed algorithm, the results obtained by applying the ELCC algorithm are to be compared to those obtained using a global optimization method, which serves as a reference for best possible performance if computational time was not a consideration. The global optimization algorithm will be described in Section 3.5.

3.1 Input-Output Linearization

Consider a minimum phase, nonlinear system of the form

$$\begin{aligned} \dot{x} &= f(x) + g(x)u \\ y &= h(x) \end{aligned} \tag{3.1}$$

where x is an n-dimensional state vector, u is an m-dimensional input vector, and y is an m-dimensional output vector. The system is input-output linearizable if and only if the relative degree of the system r is less than or equal to n [35]. As shown in Section 2.1.3 (Proposition 1), the transformation

$$v \equiv L_f^r h(x) + L_g L_f^{r-1} h(x) u \tag{3.2}$$

yields a linear input-output model of the form

$$\frac{d^r y}{dt^r} = v \tag{3.3}$$

where v is the pseudo-linear input.

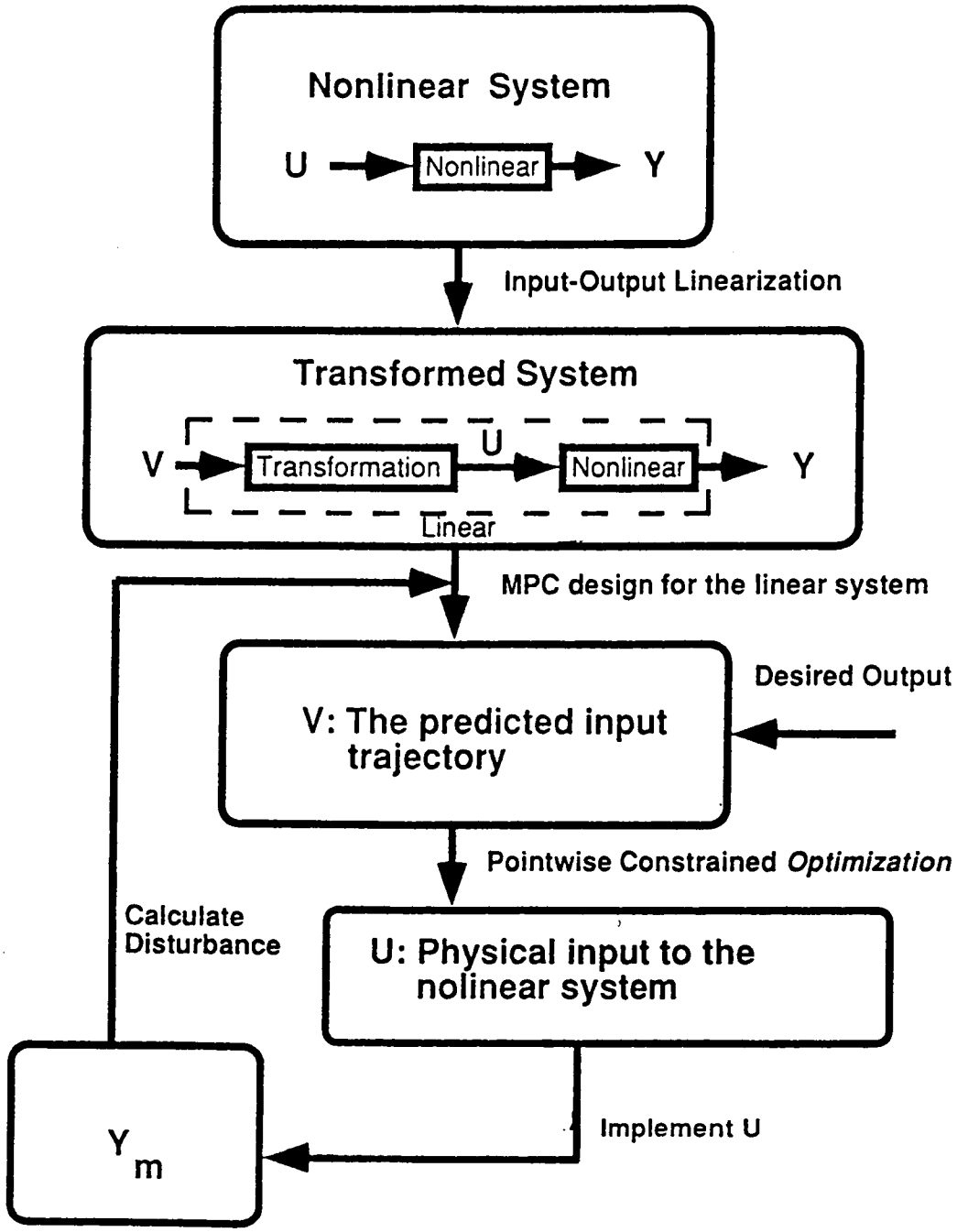

Figure 3.1: Flowchart of the controller design algorithm

From Equation 3.2, the actual input of the system can be calculated as

$$u(x,v) = \frac{(v - L_f^r h(x))}{L_g L_f^{r-1} h(x)} \tag{3.4}$$

From inspection, Equation 3.4 is valid only if $L_g L_f^{r-1} h(x) \neq 0$. Some nonlinear systems satisfy this condition over the whole operational manifold except at some finite points, the so-called "singular points". Maximum and minimum points in the steady-state system's input-output profile are examples of singular points. The existence of singular points in a given system renders the concept of "relative degree" impractical, and application of the feedback linearization technique infeasible at these points. Yet, very often, these points (especially the maximum points) are the very points one wishes to operate at. They usually correspond to the maximum yield or rate of the output parameter. In order to overcome the singularity problem, Lien and Wang [50] modified Equation 3.4 to incorporate a parameter 'ϵ':

$$u(x,v) = \frac{(v - L_f^r h(x))(L_g L_f^{r-1} h(x))}{\epsilon + (L_g L_f^{r-1} h(x))^2}, \tag{3.5}$$

where 'ϵ' is a tunable positive parameter, selected to have values that are in the neighborhood of the relative singular point. Under this modification, the output dynamics of the nonlinear system become

$$\frac{d^r y}{dt^r} = L_f^r h(x) + (v - L_f^r h(x)) \frac{(L_g L_f^{r-1} h(x))^2}{\epsilon + (L_g L_f^{r-1} h(x))^2} \tag{3.6}$$

There are two extreme approximations of Equation 3.5 according to the relative magnitudes of ϵ and $L_g L_f^{r-1} h(x)$:

- First, when the system is relatively far from the singular point: $| L_g L_f^{r-1} h(x) | \gg \epsilon$, the input becomes

$$u \simeq \frac{(v - L_f^r h(x))}{L_g L_f^{r-1} h(x)}. \tag{3.7}$$

and the linearized system approximates

$$\frac{d^r y}{dt^r} \simeq v \tag{3.8}$$

Equation 3.8 is the same as earlier without ϵ.

- At the other extreme, when the system is very close to the singular point: $| L_g L_f^{r-1} h(x) | \ll \epsilon$, u becomes

$$u \simeq 0 \ . \tag{3.9}$$

Hence, the output of the nonlinear system approximately evolves according to

$$\frac{d^r y}{dt^r} \simeq L_f^r h(x). \tag{3.10}$$

The dynamics described in Equation 3.10 is stable since the nonlinear system is of minimum phase.

The parameter 'ϵ' prevents a disproportional increase in the fraction for u, and at the same time serves as a tunable parameter to adjust the precision of the linearization process. It is desirable to choose the smallest possible value of 'ϵ' in order to increase the precision of the exact linearization. However, small values of 'ϵ' may introduce instability into the system, since they result in large changes in the input variable u. Therefore, 'ϵ' is selected as a tradeoff between stability and linearization precision.

Once the system is linearized, any type of linear controller can be used to find the values of the pseudo-linear input v that will yield a desirable output y. Then, the values of v are used to compute u according to Equation 3.5. However, if the linear model-predictive-control method is to be applied, a slightly different linear model from that of Equation 3.5 has to be employed in order to satisfy the associated system constraints, as described below.

3.2 Prediction

For the prediction part of the ELCC controller design, the linear Dynamic Matrix Control (DMC) algorithm [16] is used to compute the values of the pseudo linear input v (Equation 3.2), such that the difference between the desired and predicted system output is minimized over some future time horizon. As explained in Section 2.2.1, DMC uses the system's open-loop step response to estimate the predicted system output. Input movements are then computed that would minimize the sum of the square deviations between the predicted and the desired output.

One of the requirements for maintenance of stability of DMC is that the numerical representation of the process dynamics reaches a steady state at a finite future step, in response to a change in the manipulated variable (Cutler (1983) [16]). In order to satisfy this condition, Equation 3.3 has to be modified, because, as it stands, it is an integrator. If the system has a relative degree of one, one modification is to add the output of the system to the left side of 3.3 as in

$$\frac{dy}{dt} + y = v \tag{3.11}$$

Then v and u are related by

$$v = L_f h(x) + L_g h(x)u + h(x) \tag{3.12}$$

With the introduction of 'ϵ', the input of the system can be calculated as

$$u(x, v) = \frac{(v - L_f h(x) - h(x))(L_g h(x))}{\epsilon + (L_g h(x))^2}. \tag{3.13}$$

The system as modeled by Equation 3.11 is open loop stable and, for a step change in v, will reach steady state in finite steps. For a system with a relative degree higher

than one, similar modification can be made by adding the output and time derivatives of the output to the left side of Equation 3.3. The order of the highest derivative term added should be less than the relative degree of the system. The coefficients of the terms added to the left side of the equation should be chosen such that all the roots of the characteristic equation of the system lie in the open left half of the plane to ensure stability.

The system is modified to render it open-loop stable, and its step response is computed. Then, the predicted desired values of the pseudo-linear input are calculated according to

$$\Delta v(k) = A^{+} e(k+1) \tag{3.14}$$

where $e(k+1)$ is the predicted future error vector at the $(k+1)$th sampling point, and A^{+} is the pseudo-inverse of the system dynamic matrix A, as shown in Section 2.2.1 (Equation 2.16). In order to avoid large changes in the magnitude of the pseudo-linear input, the DMC equation (Garcia (1986) [20]) is modified by augmenting the A matrix by Λ as:

$$\Delta v(k) = \begin{bmatrix} A \\ \Lambda \end{bmatrix}^{+} \begin{bmatrix} e(k+1) \\ 0 \end{bmatrix}$$

where Λ is given by

$$\Lambda = \begin{bmatrix} \lambda & 0 & \cdots & 0 \\ 0 & \lambda & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \lambda \end{bmatrix}$$

where λ is the input suppression coefficient. The use of λ reduces undesirable oscillations in the output and limits the maximum amplitude of the input signal.

The computational requirements of the prediction part of the controller are small. After the system is linearized, the step response can be calculated off-line and the pseudo-inverse of the dynamic matrix A^+ can be hard-coded into the control algorithm. Therefore, the prediction part requires only the multiplication of the stored A^+ matrix with the on-line calculated error vector, making the algorithm computationally efficient.

3.3 Projection

The trajectory of v computed in the prediction part of the controller does not take into account the presence of system constraints. In the absence of constraints, the first of the predicted inputs is to be implemented at the next sampling point, and the process starts over again. However, the calculated predicted input would often violate the constraints imposed on the system, such as the existence of maximum flow rates, heat input or removal rates, and concentration levels, among others. Therefore, the trajectory predicted in the absence of constraints is projected onto the space of allowable system inputs, using a pointwise optimization technique (Spong, Thorp, and Kleinwaks 1986 [77]).

The value of the predicted v at a given sampling instant t is used to choose the value of the input u of the system. The system input is chosen so that, at each t, the distance between the projected v and the originally predicted v is minimum in the least squares sense. For a system of relative degree one, the value of $u(t)$ that meets the constraints will correspond to the solution of minimization of the following weighted quadratic norm:

$$\min_{u(t)}(L_f h(x) + L_g h(x)u(t) + h(x) - v(t))^T Q(L_f h(x) + L_g h(x)u(t) + h(x) - v(t)) \quad (3.15)$$

subject to

$$l_1(x,u) = 0 \tag{3.16}$$

$$l_2(x,u) \leq 0 \tag{3.17}$$

where $l_1(x,u)$ and $l_2(x,u)$ represent equality and inequality constraints on the process operation, Q is a positive definite weighting matrix, $v(t)$ denotes the original predicted unconstrained input value at t, and x denotes the state vector at time t, a known variable to be measured on-line at each sampling time.

If the system has constraints in the inputs of the form $u_{i,min} \leq u_i \leq u_{i,max}$, Equation 3.15 can be rewritten as

$$\min_{u(t)} (L_g h(x)u(t) + L_f h(x) + h(x) - v(t))^T Q (L_g h(x)u(t) + L_f h(x) + h(x) - v(t)) \tag{3.18}$$

subject to

$$Nu(t) \leq C \tag{3.19}$$

where C is a column vector given by

$$C = \begin{bmatrix} u_{1,max} \\ -u_{1,min} \\ u_{2,max} \\ -u_{2,min} \\ \vdots \\ u_{n,max} \\ -u_{n,min} \end{bmatrix},$$

N is a matrix given by

$$N = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ -1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ 0 & -1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 \\ 0 & \cdots & 0 & -1 \end{bmatrix}$$

and $v(t)$ is the ideal input as calculated from the prediction step without regard to constraints.

The minimization of 3.18 is equivalent to:

$$\min_{u(t)}(\frac{1}{2}u(t)^T L_g h(x)^T Q L_g h(x) u(t) - b(x,t)^T u(t)) \tag{3.20}$$

subject to 3.19

$$Nu(t) \leq C$$

where $b(x,t)$ is defined as

$$b(x,t) = L_g h(x)^T Q(v(t) - L_f h(x) - h(x)), \tag{3.21}$$

all known values at each t.

Assuming feasibility, the problem is equivalent to

$$\max_{\gamma \geq 0} \min_{u(t)} \{\frac{1}{2}u(t)^T L_g h(x)^T Q L_g h(x) u(t) - b(x,t)^T u(t) + \gamma^T (Nu(t) - C)\} \tag{3.22}$$

where γ is a costate vector. The minimization over $u(t)$ is unconstrained and is attained by

$$u(t)^* = (L_g h(x)^T Q L_g h(x))^{-1}(b(x,t) - N^T \gamma^*) \tag{3.23}$$

where γ^* denotes the optimal value of γ. Then, replacing the value of $u(t)^*$ in Equation 3.22 yields

$$\max_{\gamma \geq 0}(-\frac{1}{2}\gamma^T P(x,t)\gamma - \gamma^T d(x,t)) \tag{3.24}$$

where $P(x,t)$ is given by

$$P(x,t) = N^T(L_g h(x)^T Q L_g h(x))^{-1} N \tag{3.25}$$

and $d(x,t)$ is defined as

$$d(x,t) = C - N(L_g h(x)^T Q L_g h(x))^{-1} b(x,t) \tag{3.26}$$

The value of γ that maximizes Equation 3.24 is substituted into Equation 3.23 to arrive at the optimum system input $u(t)^*$. The maximization of Equation 3.24 can be accomplished by applying Hildreth's Quadratic Programming Procedure (Luenberger (1969) [54]) to

$$J = \min_{\gamma \geq 0}(\frac{1}{2}\gamma^T P(x,t)\gamma + \gamma^T d(x,t)) \tag{3.27}$$

The procedures consist of initially setting the vector $\gamma = 0$, then calculating the value of J and obtaining a new value of γ by using

$$\gamma_i^{k+1} = max(0, w_i^{k+1}). \tag{3.28}$$

where w_i^{k+1} is given by

$$w_i^{k+1} = -\frac{1}{P_{i,i}}(d_i + \sum_{j=1}^{i-1} P_{i,j}\gamma_j^{k+1} + \sum_{j=i+1}^{n} P_{i,j}\gamma_j^k) \tag{3.29}$$

where $P_{i,j}$ is the i,jth component of the matrix P, d_i is the ith component of the vector d, and γ_i^{k+1} is the ith component of the vector γ for the $(k+1)$th iteration. After a new value of γ is obtained, the value of the corresponding J is calculated. If the new value of

J is less than the previously calculated one, the procedure is repeated. If the new value of J is larger than the previously calculated one, the value of γ corresponding to the smaller J will be the optimal value γ^*. The value of γ^* is then used in Equation 3.23 to calculate the value of the optimum system input $u(t)^*$. The input $u(t)^*$ is then implemented in the system and a new output is generated. The process is repeated in the next step. This calculated optimum input $u(t)^*$ is the projection of the original predicted unconstrained input that satisfies the system constraints.

3.4 Controller Tuning

The tuning of the ELCC would require the setting of the DMC parameters and the selection of ϵ, as in Equation 3.13. The DMC parameters to be selected are:

- The number of time intervals necessary for the output to come to steady state, M.
- The number of future input moves to be computed, N.
- The number of future time intervals to be considered, P.
- The input suppression factor, λ.

According to Garcia and Morshedi (1986) [20], the selection of $P = M + N$ will yield a stable DMC controller. The use of λ suppresses the amplitude of the input moves. Larger λ improves the stability properties of the controller in the face of model inaccuracies (Garcia and Morari (1982) [23]). However, larger λ will yield more sluggish response and results in larger offset in the output. The size of 'ϵ' corresponds to the size of the neighborhood around the singular point, in which the linearization is only approximate. An increase in ϵ decreases the range of exact linearization, but would accommodate

larger model parameter variations at the expense of producing larger steady-state offset in the output.

The values of P, M, and N should be first selected, following the existing guide lines for tuning a DMC controller [23]. Then, the values of λ and ϵ are to be adjusted to eliminate chattering and undesired oscillations in both input and output responses.

The ELCC algorithm uses a linear DMC to predict the values of the pseudo-linear input v, based on the linearized system. However, any linear controller can be used in the prediction step. In the work described here, results using DMC are to be compared with results obtained using a simple Proportional Integral Controller. Details will be presented in Chapter 5.

3.5 Global Optimization

The ELCC algorithm described above yields a suboptimal solution in the system input u. The solution is suboptimal because a pointwise optimization approach is employed at every sampling point. The motivation for employing pointwise optimization is that a global optimization problem, complete with applicable system constraints, cannot be properly formulated after the feedback linearization step. A solution based on a global optimization method that uses the original nonlinear model is developed to serve as a performance reference against which the performance of the ELCC controller is to be evaluated. The global optimization algorithm uses the conjugate gradient method to calculate the optimum values of the input of the system that will optimize a given performance criteria. The optimization is performed in a closed-loop form to minimize the effect of modeling errors and disturbances. The conjugate gradient method is presented

in Section 3.5.1 and the global approach is described in sections 3.5.2, 3.5.3 and 3.5.4.

3.5.1 Conjugate Gradient Method

The conjugate gradient method used to find a minimizing argument x^* of a scalar cost function $J(x)$, $x \in R^n$ consists of the following steps (Hansdorff, 1976 [27]):

- Making an initial guess.
- Constructing a sequence $x_0, x_1, x_2, \ldots$ such that

$$J(x_{i+1}) < J(x_i) \tag{3.30}$$

The sequence is extended until no x_{i+1} can be found, such that Equation 3.30 is satisfied, or until the sequence x_i approaches a constraint. To choose the sequence so that it converges to x^*, the gradient of the function $J(x)$ is calculated. If $J(x)$ has continuous partial derivatives, the gradient can be calculated as:

$$\nabla J(x) = \frac{\partial J}{\partial x} \tag{3.31}$$

Then the sequence is found by selecting successive direction vectors as a linear combination of the previous direction vectors and the current negative gradient (Luenberger, 1984 [53]). The iterative algorithm will have the form:

$$x_{k+1} = x_k + \alpha_k p_k \tag{3.32}$$

where α_k is a nonnegative scalar minimizing $J(x_k - \alpha_k \nabla J(x)_k^T)$ and p_k is given by the equation:

$$p_k = -\nabla J(x_k) + \frac{< \nabla J(x_k), \nabla J(x_k) >}{< \nabla J(x_{k-1}), \nabla J(x_{k-1}) >} \tag{3.33}$$

In other words, from the point x_k, we search along the direction of p_k to a minimum point on this line; this minimum point is taken to be x_{k+1}.

3.5.2 Global Optimization Method

The problem to be solved by the controller may be stated as follows: consider a nonlinear process of the form

$$\begin{aligned} \dot{x} &= f(x,u) \\ y &= g(x,u) \end{aligned} \tag{3.34}$$

where x is an n-dimensional state vector u is an m-dimensional input vector, and y is an m-dimensional output vector. An open-loop optimal control problem may be defined as:

$$\min_{u(t)} J(t) = \int_{to}^{\infty} \alpha^t h(x,u)dt \tag{3.35}$$

subject to

$$\dot{x} = f(x,u) \tag{3.36}$$

$$l_1(x,u) = 0 \tag{3.37}$$

$$l_2(x,u) \leq 0 \tag{3.38}$$

where the α^t term models a decreasing interest in the cost function with time, $h(x,t)$ is a scalar performance index, and $l_1(x,u)$ and $l_2(x,u)$ represent equality and inequality constraints on the process operation.

A corresponding closed-loop control problem can be expressed as:

Find a $\gamma^* : x(t) \rightarrow u(t)$ which minimizes

$$\hat{J}(\gamma) = \int_{to}^{\infty} \alpha^t h(x,\gamma(x))dt \tag{3.39}$$

Such that a γ^* will be implemented in real-time to compute the optimal $u^*(t)$ from the current measurement of $x(t)$.

If the solution to the problem reaches a steady-state operating point at time τ, such that

$$0 = f(x(\tau), u(\tau)) \tag{3.40}$$

then the control problem will take the form

$$\min_{u(t)} J(t) = \int_{t_0}^{\tau} \alpha^t h(x,u)dt - \frac{\alpha^\tau h(x(\tau), u(\tau))}{ln(\alpha)} \tag{3.41}$$

subject to

$$\dot{x} = f(x,u) \tag{3.42}$$

$$0 = f(x(\tau), u(\tau)) \tag{3.43}$$

$$l_1(x,u) = 0 \tag{3.44}$$

$$l_2(x,u) \leq 0 \tag{3.45}$$

The solution to this problem will provide a near-optimal control vector $u(t)$ for $t_0 \leq t \leq \tau$. If the computation of the solution to the optimization problem requires an interval of time t_i ($t_0 \leq t_i \leq \tau$), then a practical application of this algorithm would be to apply, in real-time, only the values of u for $t_0 \leq t \leq t_i$ and to solve a new optimization for $t_i \leq t \leq \tau + t_i$.

3.5.3 Open Loop Optimization

The solution to the optimization problem will be obtained by applying the conjugate gradient method as explained in Section 3.5.1. The convergence of this method can be speeded up considerably if a good initial guess is selected.

3.5.4 Initial Guess

The initial guess is chosen based upon the solution to the previous optimization step as follows:

Consider the previous solution to be $(x(t), u(t))$ for $t_0 \leq t \leq \tau$, and assume that the vector $u(t)$ has been applied for $t_0 \leq t \leq t_i$ resulting in the state vector $x(t_i)$. Then the initial value for the next optimization is given by

$$u(t) = \begin{cases} u(t) \ for \ t_i \leq t \leq \tau \\ u(\tau) \ for \ \tau \leq t \leq \tau + t_i \end{cases}$$

If the measured values of the state vector are not substantially different from the values calculated by the optimization, then this initial value should be close to the optimal trajectory. If the conditions of the system have changed substantially, or if a new output trajectory is desired, then a new initial guess will need to be calculated using the DMC algorithm for the linearized system. This is followed by transforming the estimated pseudo-linear input into the system's physical input u, setting the values of the input that do not satisfy the system constraints equal to the limit values.

CHAPTER 4

APPLICATIONS

The Exact-Linearized-Constraint Controller (ELCC) algorithm was tested by computer simulation on three processes: a Continuous Stirred Tank Bioreactor (CSTBR), an exothermic Continuous Stirred Tank Reactor (CSTR) and a catalytic CSTR. The description of the bioreactor and the results of applying the nonlinear controller to this simulated process are presented in Section 4.1. The controller design for the CSTR and the simulation results for this system are presented in Section 4.2. Finally, the description of the catalytic reactor and its simulation results are presented in Section 4.3.

The first process used in the testing was a CSTBR, which has proven to be difficult to control due to the following factors: (Wang, 1987 [84])

- The lack of instrumentation suitable for on-line supervision of critical variables, such as biomass concentration.
- The time-varying and nonlinear nature of the system.
- The high degree of model uncertainty associated with a biological system.

These factors show the need for a type of nonlinear controller that is robust enough to accommodate an imprecisely known dynamic model that changes its nature during the course of operation.

A nonlinear optimal controller appears to be well-suited to control this type of process. It will operate the system at applicable optimal conditions whenever they are

reachable. If the desired optimal point cannot be reached, the controller should still maintain stable operation as much within the set of specified constraints as possible. If the optimization is carried out in a closed-loop form, the controller should be able to make adjustments to correct for disturbances and unmodeled dynamics.

The second process, an exothermic CSTR, was initially used by Economou and Morari to test the performance of their proposed nonlinear IMC controller [17]. Since then, it has been utilized by several other investigators to test different types of nonlinear controllers. The CSTR was used as a test case to compare the ELCC algorithm with other nonlinear controllers proposed in the literature.

The third process used to test the control algorithm was a catalytic reaction in a CSTR with multiple equilibrium points. This process has been studied by Li and Biegler (1988) [46], Brengel and Seider (1989) [6], and Gattu and Zafiriou (1991) [25] to test their proposed nonlinear controllers. This process has two inputs and two outputs and a marginally unstable set point; it was used here to show the performance of the ELCC with a multiple-input, multiple-output (MIMO) system. The third test illustrated the ELCC's capacity to stabilize a marginally unstable system and was used to compare our controller with other, previously reported nonlinear control strategies, reported previously.

4.1 Continuous Stirred Tank Bioreactor

This system is a continuous bioreactor in which living microorganisms (biomass) utilize externally fed material (substrate) to grow, reproduce, and excrete complex biochemicals. In the simpler case where no secondary product is formed, the products of the reaction are additional microbial mass and the end-products of aerobic respiration.

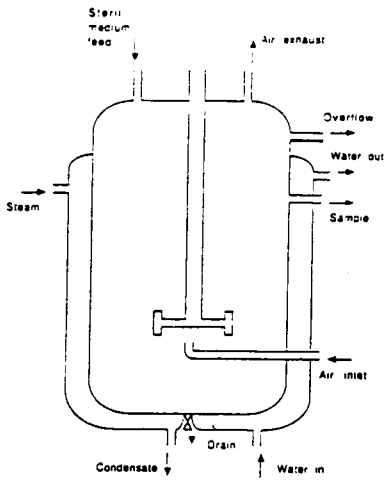

Figure 4.1: Bioreactor

The general equation for growth and end-product formation can be written as [82]:

$$A(C_aH_bO_c) + B(O_2) + D(NH_3) = M(C_dH_eO_fN_g) + P(CO_2) + Q(H_2O) \qquad (4.1)$$

where $C_aH_bO_c$ denotes the carbon energy source (substrate) and $C_dH_eO_fN_g$ is the biomass. The vessel in which this process is carried out is referred to as a bioreactor. A schematic representation of a bioreactor is shown in Fig 4.1. Strict first-level control of the operating conditions, such as pH, temperature and dissolved oxygen, must be maintained in order for the microorganisms to grow properly.

Performance Specifications

The main objective for this process is to control the overall productivity of the biomass in the bioreactor at a desired level. The control is subject to a time-varying growth model and uncertainty in the model parameters. The performance of the system was evaluated through simulations with set point changes and parameter variations.

Model of the Process

In this study, a SISO process with multiple states is considered. The model that describes the system can be obtained from the overall material balance on the biomass and the substrate concentrations as follows:

Biomass Balance

The balance of biomass around the bioreactor is given by

$$Biomass_{in} - Biomass_{out} + Biomass_{growth} - Biomass_{death} = Biomass_{accumulation} \quad (4.2)$$

Each term in 4.2 refers to a rate term. In this process, the feed is sterile. Therefore, the first term is 0. Because the growth rate is generally much greater than the death rate, the fourth term can be ignored. Then, the corresponding mathematical equation is

$$\dot{b} = \mu b - Db \quad (4.3)$$

where D is the dilution rate, b is the biomass concentration, and μ is the specific growth rate. ($\mu = \frac{db}{dt}\frac{1}{b}$, where $\frac{db}{dt}$ refers to growth only).

Substrate Balance

The balance of substrate around the bioreactor is given by

$$Substrate_{in} - Substrate_{Out} - Substrate_{Consumed} = Substrate_{accumulation} \quad (4.4)$$

The final equation takes the form:

$$\dot{s} = D(s_f - s) - \frac{\mu b}{y_s} \quad (4.5)$$

where s is the substrate concentration, y_s is the biomass yield with respect to substrate, and s_f is the substrate concentration in the feed.

In order to describe the growth dependence on the substrate, only empirical formulae have been considered. The most widely used is the Monod equation:

$$\mu = \frac{\mu_{max} s}{K_s + s} \tag{4.6}$$

where K_s is a constant in the Monod equation and μ_{max} is the maximum specific growth rate. Equation 4.6 was used in this study to model the growth kinetics μ.

Manipulated, Controlled and State Variables

In a bioreactor, the total rate of growth is equal to the biomass productivity at steady state R. R can be indirectly calculated on-line from measurements of the inlet and outlet air flow rates and concentrations of oxygen and carbon dioxide, as shown by San and Stephanopoulos (1984) [79]. In this work, R was used as measurable output variable. The manipulated variable, whose value can be adjusted freely, was the dilution rate D.

For a continuous bioreactor, the total rate of growth is defined as:

$$R = \mu b \tag{4.7}$$

The control problem in the predictive control phase of the controller design becomes

$$\min_{D_k,\cdots,D_{k+N}} \sum_{i=1}^{P} [(r - R_{k+i})^T (r - R_{k+i})] \tag{4.8}$$

subject to

$$\dot{b} = \mu(b) - D(b) \tag{4.9}$$

$$\dot{s} = D(s_f - s) - \frac{\mu b}{y_s} \tag{4.10}$$

$$\mu = \frac{\mu_{max} s}{K_s + s} \tag{4.11}$$

$$D_{min} \leq D(k) \leq D_{max} \tag{4.12}$$

where Equation 4.12 refers to the inequality constraints on the process input, r represents the desired total rate of growth, N represents the number of future input moves, and P represents the number of sample intervals for which the output of the system was predicted.

4.1.1 Controller Design

The first step in the design of the ELCC is the linearization of the system. Following that outlined in Section 2.1.2, the output of the system must be differentiated with respect to time in order to generate an explicit relationship between the system's output and input. The first derivative of R with respect to time is

$$\frac{dR}{dt} = \mu(\mu b - Db) + \frac{bK_s\mu_{max}}{(K_s + s)^2}(D(s_f - s) - \frac{\mu b}{y_s}). \tag{4.13}$$

Since $\frac{dR}{dt}$ depends on D explicitly, the relative order of the system is one, and the linearized system can be written as

$$\frac{dR}{dt} + R = v. \tag{4.14}$$

The state feedback is given by

$$D = \frac{(v - L_f h(x) - h(x))L_g h(x)}{\epsilon + (L_g h(x))^2} \tag{4.15}$$

where $L_f h(x)$, $L_g h(x)$ and $h(x)$ are

$$L_f h(x) = \mu^2 b - \frac{\mu b^2 K_s \mu_{max}}{(K_s + s)^2 y_s} \tag{4.16}$$

$$L_g h(x) = \frac{b(s_f - s)K_s \mu_{max}}{(K_s + s)^2} - b\mu \tag{4.17}$$

$$h(x) = \mu b \tag{4.18}$$

Table 4.1: Control Parameters

Parameter	Value
ϵ	0.5
λ	0.8
P	100
M	70
N	30
D_{max}	1.2 hr^{-1}
D_{min}	0.1 hr^{-1}

The Dynamic Matrix Control algorithm is applied to the system, represented by Equation 4.14. Using the procedure described in Section 3.3, the value of v is used to calculate D. The values of the control parameters are shown in Table 4.1.

4.1.2 Simulation Results

Two tests of the ELCC were performed. In the first test, it was assumed that the model of the system is known exactly. Under this assumption, the nominal performance of the controller was evaluated. However, some of the parameters of the system may not be known exactly or can be subject to variations, due to changes in the conditions or changes in the metabolic state of the system. Therefore in the second test, the performance of

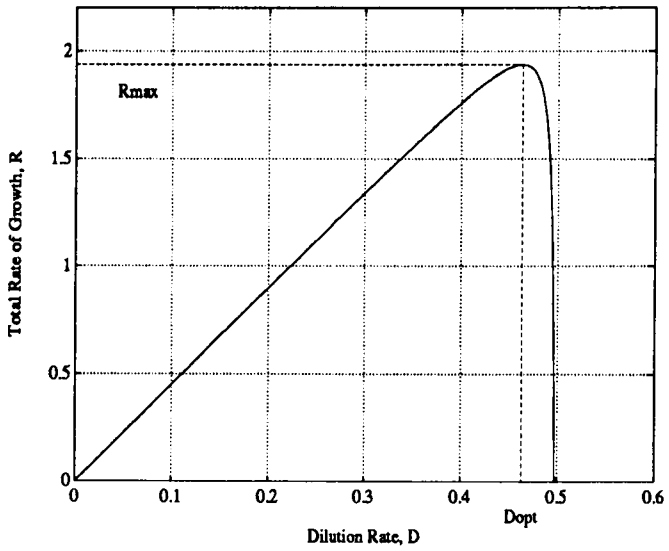

Figure 4.2: Steady state diagram of a bioreactor

the controller is evaluated in the presence of parameter variations. A sampling time of 0.05 hours was used in all the simulations.

Nominal Performance

The steady-state plot of the total rate of growth as a function of the dilution rate has a well-defined maximum, as shown in Fig 4.2. In this figure R_{max} corresponds to the steady-state maximum rate of growth, and D_{opt} corresponds to the dilution rate that yields R_{max}. It is desirable to work as close to R_{max} as possible while maintaining stability. Traditionally, an output lower than R_{max} is selected as the set point to ensure that its value is achievable, even when the parameters of the system change. In general, this selection will yield a suboptimal performance, but it will give some margin of stability. In the first test, the initial steady-state dilution rate was chosen to have a value lower

than D_{opt}. A step change of +10% on the output was then introduced. Figures 4.3, 4.4 and 4.5 show that the system attained the desired value in less than two hours.

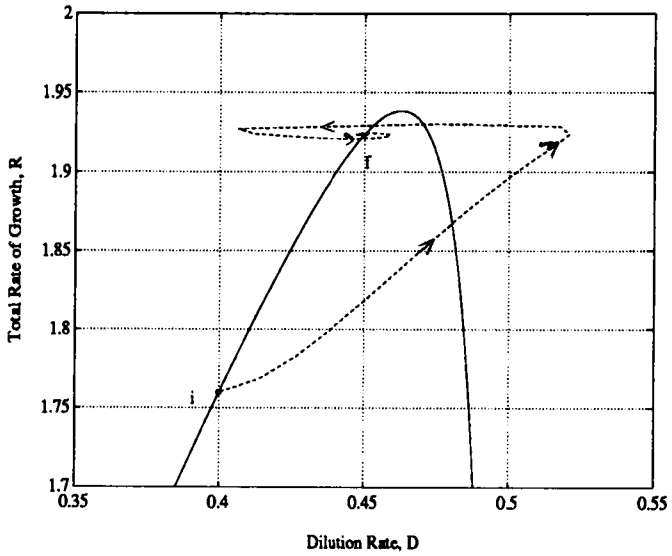

Figure 4.3: Set point change in the total rate of growth by 10%. System trajectory

One of the limitations of a linear controller for this system is that it can only work on one side or the other of the steady state maximum because of the change in the signs of the associated gains. The side in which the control maintains stability depends on the point of local linearization. If the conditions of the system are such that the initial steady state of the system corresponds to the side opposite to the side of linearization, the system will eventually become unstable, as will be shown in Section 4.1.3. The ELCC overcame this limitation, since the linearization used in the algorithm is valid for the entire region of operation. An initial value of the dilution rate larger than D_{opt} was selected to demonstrate the ELCC's ability to work on either side of the maximum point. Figures 4.6, 4.7 and 4.8 show that the ELCC was able to maintain stability and

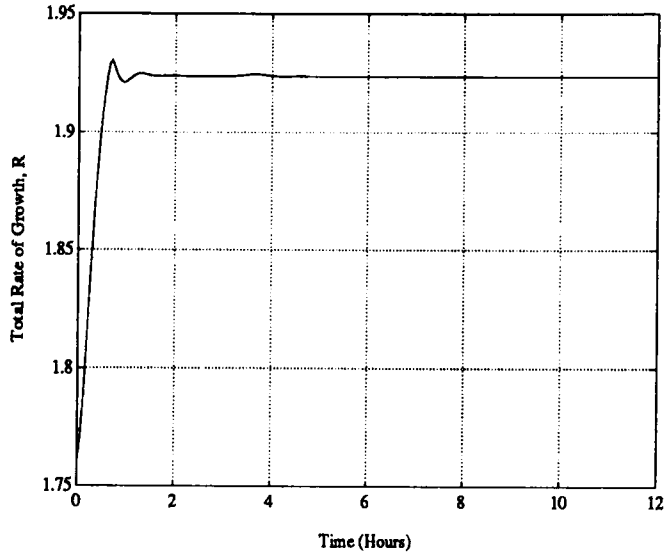

Figure 4.4: Set point change in the total rate of growth by 10%. Output versus time

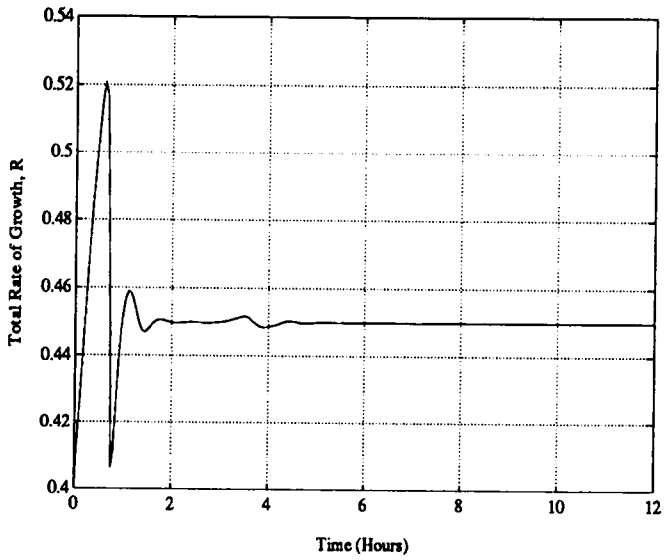

Figure 4.5: Set point change in the total rate of growth by 10%. Input versus time

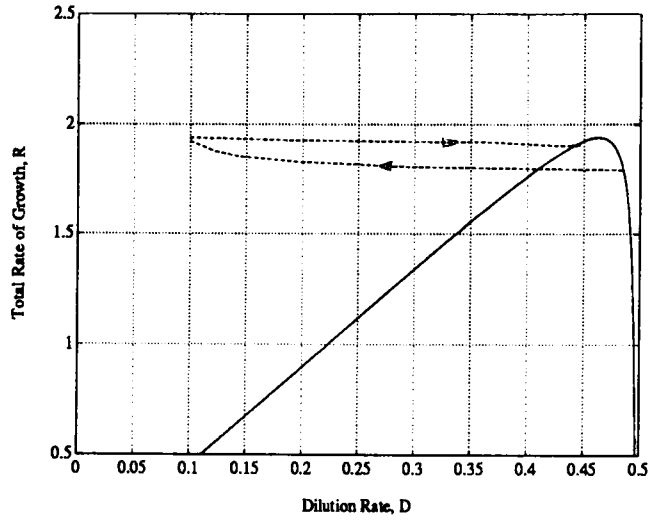

Figure 4.6: Set point change in the total rate of growth by 10% with initial dilution rate higher than D_{opt}. System trajectory

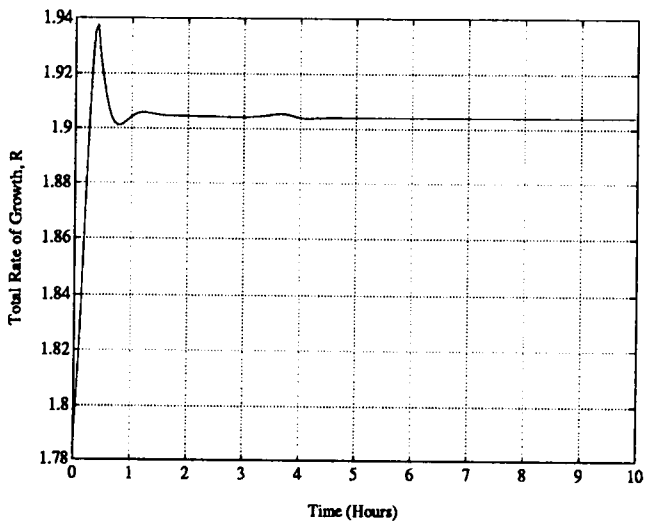

Figure 4.7: Set point change in the total rate of growth by 10% with initial dilution rate higher than D_{opt}. Output versus time

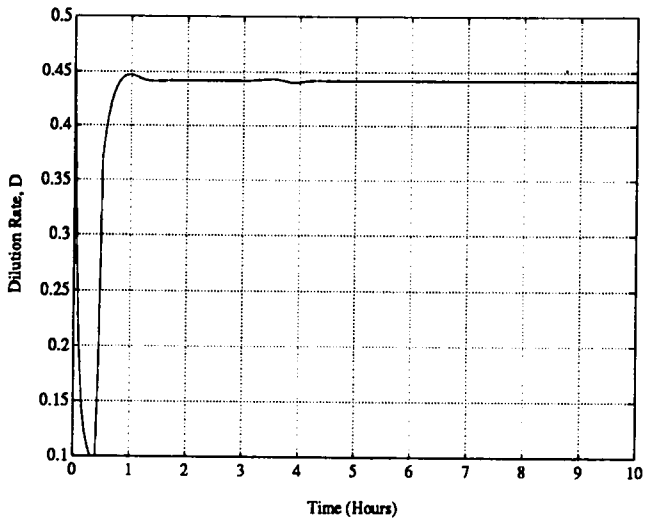

Figure 4.8: Set point change in the total rate of growth by 10% with initial dilution rate higher than D_{opt}. Input versus time

that the desired output value was attained in less than two hours, with the final dilution rate settling on the left side of D_{opt}. Section 5.1 will elaborate on which side of D_{opt} the system settles.

Additional simulations were carried out with different initial values of the dilution rate and set points. These simulations are shown in Appendix D. These simulations show that the performance of the ELCC does not change for different initial conditions. In all simulations, the system achieved the desired set point in less than 3 hours. The simulations that started with a dilution rate lower than D_{opt} finished in the left side of the steady state curve. The simulations that started with a dilution rate higher than D_{opt}, and that began relatively far from the maximum point, ended in the right side of the steady state curve (see Figure 4.9). The simulations that started with a dilution rate higher than D_{opt}, but that began relatively close to the maximum point, ended in

the left side of the steady state curve (see Figure 4.6). Details of this occurrence will be elaborated in Section 5.1.

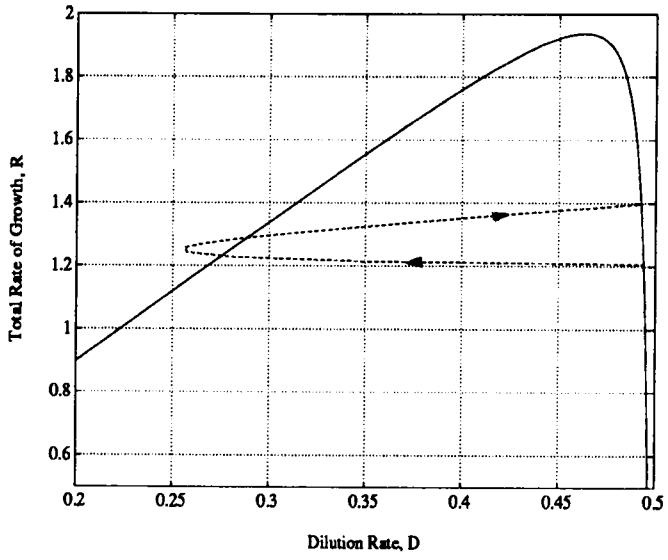

Figure 4.9: Set point change in the total rate of growth by 15%. System trajectory

Accommodation of Model Uncertainty

One of the major difficulties in the control of a bioreactor is the high degree of model uncertainty due to the lack of understanding of microbial pathways and cellular control mechanisms (Wang et al [82]). Model uncertainty may also result when some of the system parameters are not known exactly, and vary due to changes in the conditions in the bioreactor.

The effect of a change in the biomass yield y_s and the effect of a change in the maximum specific growth rate μ_{max} on the steady-state curve are shown in Figures 4.10 and 4.11, respectively. Changes in those parameters modify the value of R_{max}. Therefore,

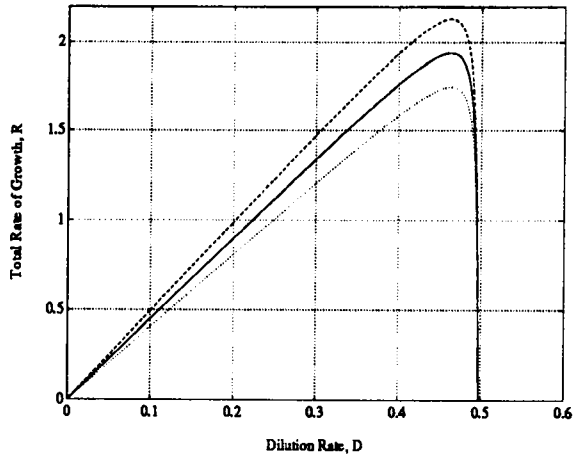

Figure 4.10: Steady state diagram of a bioreactor. Effect of y_s. Solid line corresponds to nominal value, dash-line an increase of 10%, and dot-line a decrease of 10%.

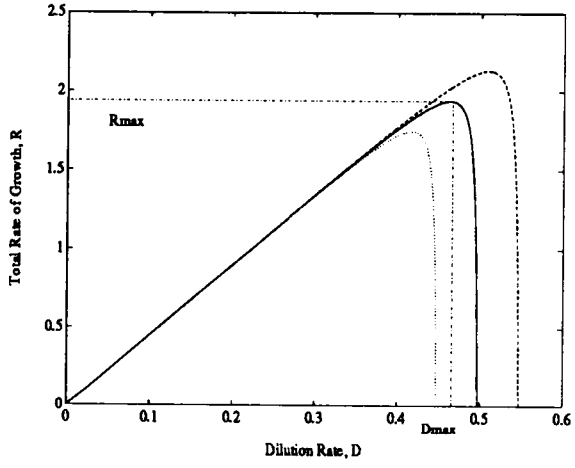

Figure 4.11: Steady state diagram of a bioreactor. Effect of μ_{max}. Solid line corresponds to nominal value, dash-line an increase of 10%, and dot-line for decrease of 10%.

if the desired output is close to the maximum total rate of growth, it may become unachievable. The traditional solution to the unachievability problem is the selection of a suboptimal set point. However, the ELCC algorithm presented in this work was able to maintain stability even when the set point became unachievable, thus allowing the selection of set points higher than any actually achievable by the system. Note that changes in y_s result in changes of R_{max} at constant D_{opt}; whereas, changes in μ_{max} result in changes R_{max} at different D_{opt}. Therefore, the nominal position of the R_{max} cannot be used to predict what would be the maximum condition in the bioreactor.

In order to demonstrate the ability of the controller to maintain stability in the presence of parameter variations that result in the set point being unachievable, two changes were made in the system. First, the value of y_s was decreased by 10%, with a set point remaining at 1.87. Figures 4.12, 4.13 and 4.14 show that, under these altered conditions the ELCC maintained stability and set the system at a point close to the new maximum possible achievable set point.

A second parameter variation was implemented by introducing a change in μ_{max} with the set point remaining unchanged. A change in μ_{max} not only changed the value of R_{max}, it also changed the dilution rate at which R_{max} is achieved (see Fig 4.11). The results for a decrease in the value of μ_{max} by 10% are shown in Figures 4.15, 4.16 and 4.17. These figures show that the ELCC operated the system close to the new maximum possible achievable point without causing system instability. Note that the lower constraint of D became active during the first half hour of operation.

When the parameter variation was such that the set point is still achievable, the controller set the system in the desired output, as shown in Figures 4.18, 4.19 and 4.20. Additional parameter variations were implemented, starting with different dilution rates.

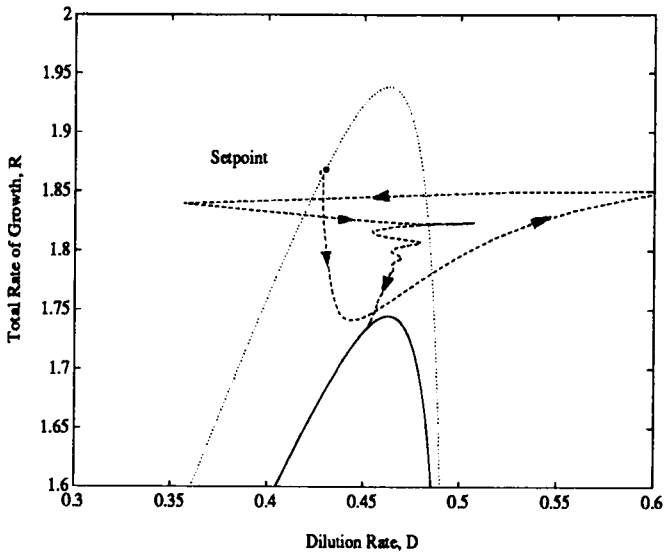

Figure 4.12: Change in y_s by -10%. Unachievable set point. System trajectory

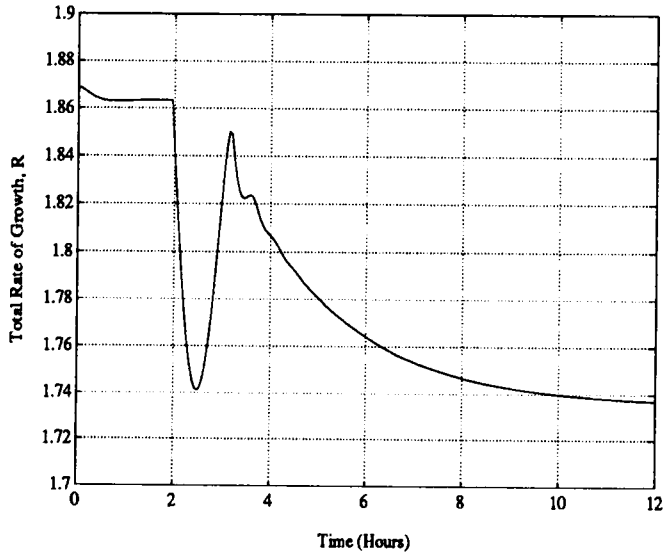

Figure 4.13: Change in y_s by -10%. Unachievable set point. Output versus time

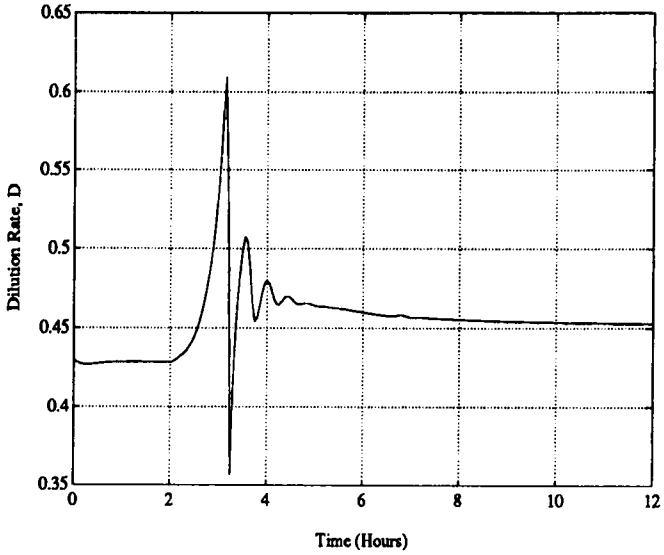

Figure 4.14: Change in y_s by -10%. Unachievable set point. Input versus time

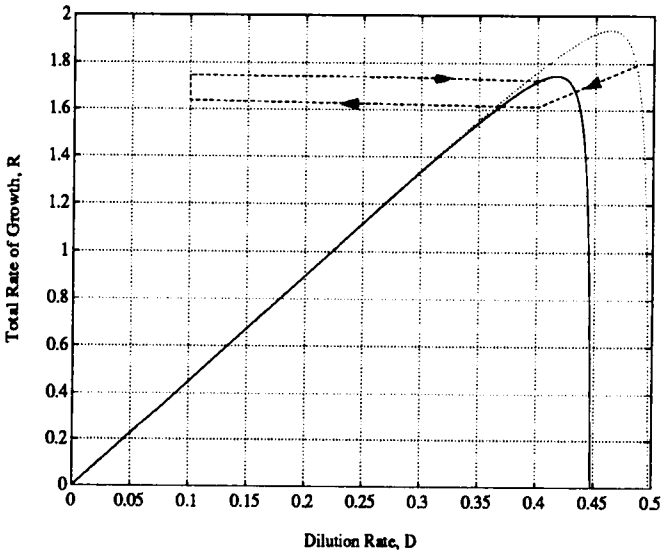

Figure 4.15: Change in $\mu_m ax$ by -10%. Unachievable set point. System trajectory

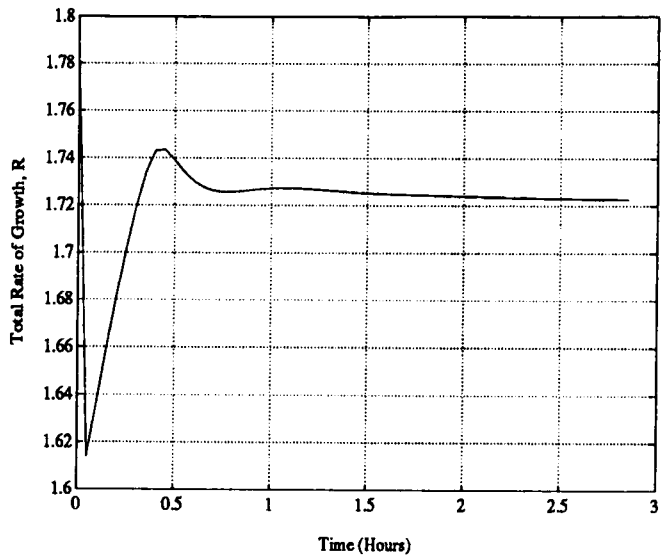

Figure 4.16: Change in μ_{max} by -10%. Unachievable set point. Output versus time

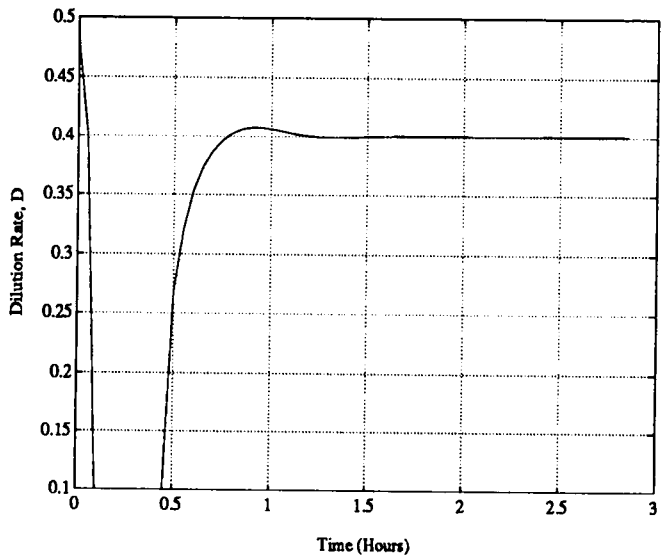

Figure 4.17: Change in μ_{max} by -10%. Unachievable set point. Input versus time

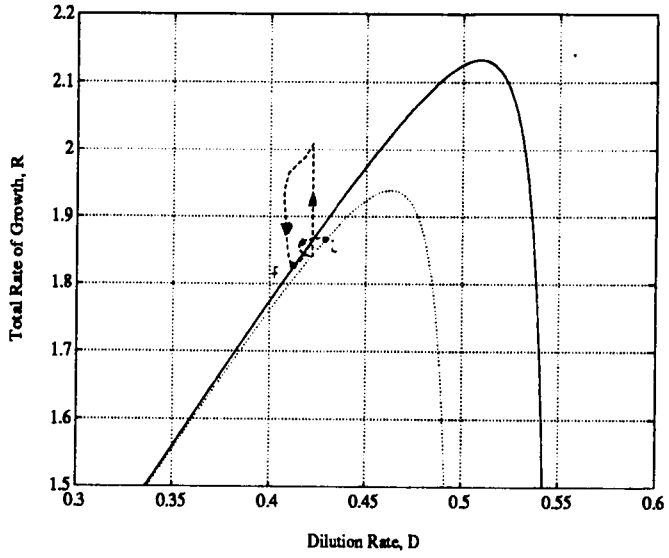

Figure 4.18: Change in μ_max by +10%. Achievable set point. System trajectory

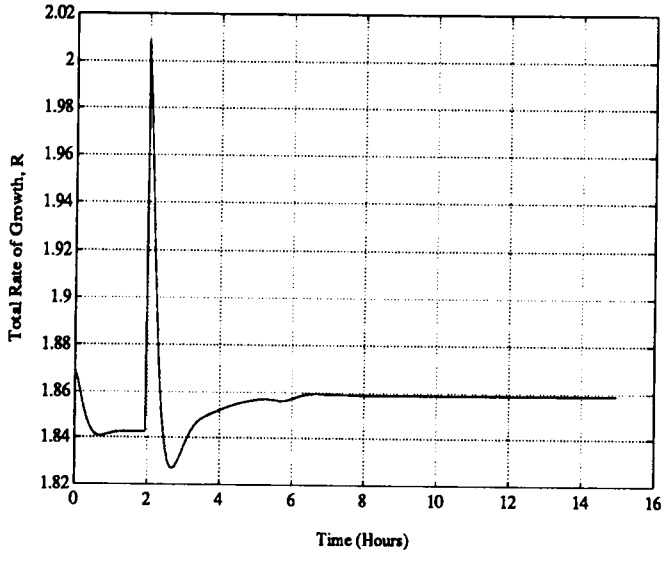

Figure 4.19: Change in μ_{max} by +10%. Achievable set point. Output versus time

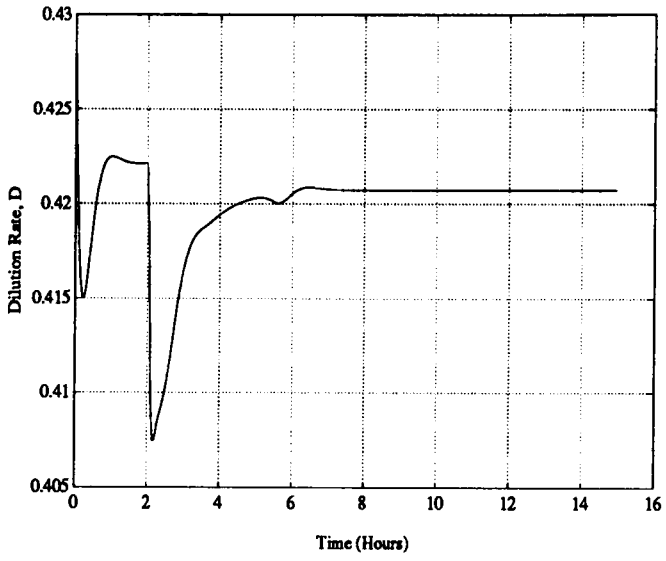

Figure 4.20: Change in μ_{max} by +10%. Achievable set point. Input versus time

They are shown in Appendix D. The controller was able to maintain stability in all the simulations. When the set point was achievable, the controller operated the system very close to the maximum achievable total rate of growth.

4.1.3 Linear Controller for a CSTBR

The results obtained with the ELCC were constrasted with those obtained with a classical (linear) Proportional Integral Derivative (PID) controller. This controller was used to demonstrate the inability of a linear controller of its kind to maintain stability in the presence of parameter variations that render the desired set point unachievable. Traditionally, a PID is tuned to control a system that is linearized locally at a steady state operating point by a truncated Taylor series expansion around that point. In these simulations, the values shown on Table 4.2 were used as a nominal set point, as suggested

by Wang (1986) [84].

Table 4.2: Nominal Set Point

Parameter	Value
D	0.42 hr^{-1}
s_f	9.0 $\frac{g}{l}$
μ_{max}	0.5 hr^{-1}
k_s	0.05 $\frac{g}{l}$
y_s	0.5
b	4.36875
s	0.2625 $\frac{g}{l}$
μ	0.42 hr^{-1}

After the nominal set point was selected, the system was linearized, and then the PID controller was tuned using the Cohen-Coon method [72]. The nominal performance was first tested with the conditions near the neighborhood of the nominal set point. Then, a step change of 5% in the desired output was introduced. The results of this simulation are shown on Figures 4.21, 4.22 and 4.23. These figures show that the system performed satisfactorily and achieved the desired set point in less than 4 hours. For this mild perturbation the PID was able to maintain stability, and its performance was close to that of the globally linearized system.

A second simulation was performed, starting with a dilution rate higher than D_{opt} (the opposite side of the bioreactor steady-state curve). The results are presented in Figures 4.24, 4.25 and 4.26. These figures show clearly that the PID controller was not

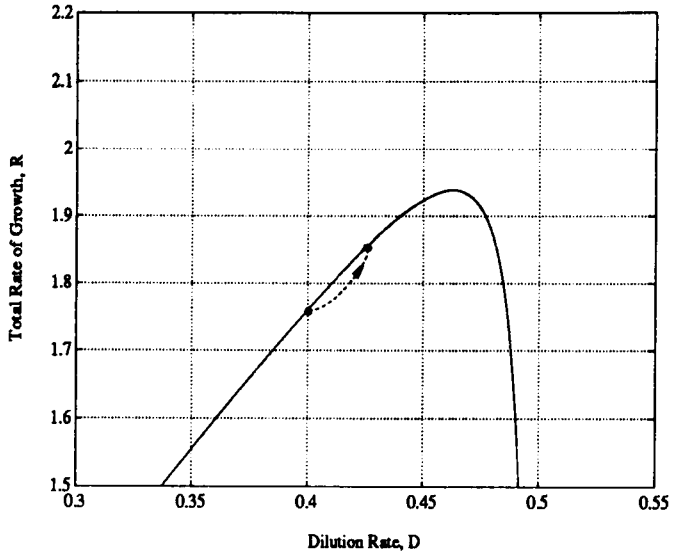

Figure 4.21: Set point change in the total rate of growth by 5%. System trajectory

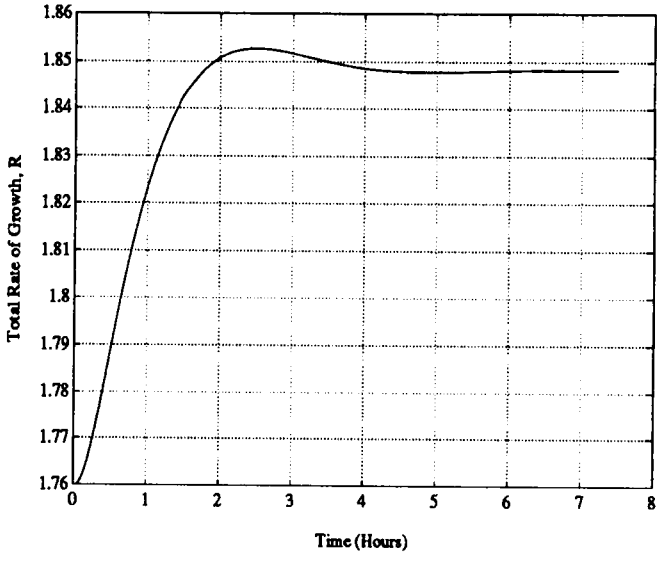

Figure 4.22: Set point change in the total rate of growth by 5%. Output versus time

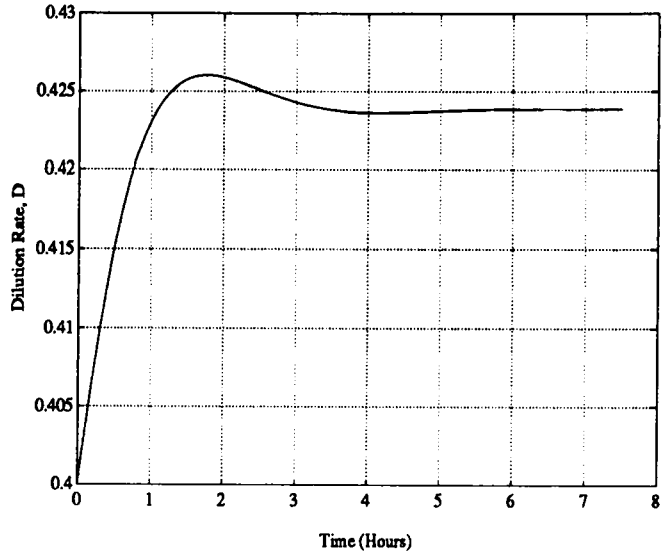

Figure 4.23: Set point change in the total rate of growth by 5%. Input versus time

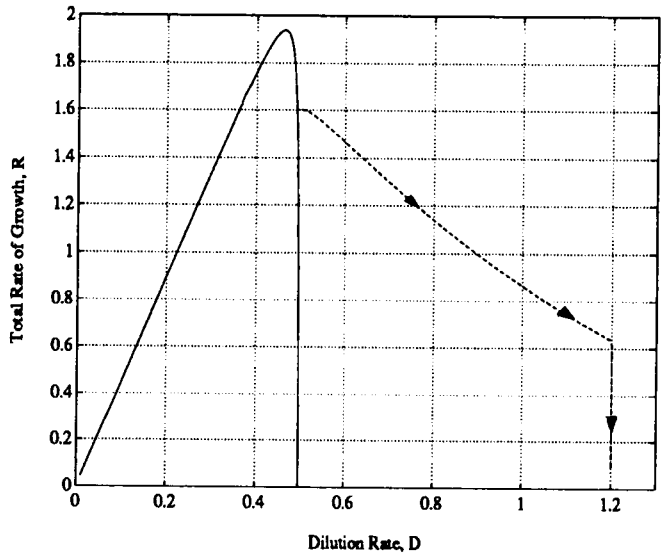

Figure 4.24: Set point change in the total rate of growth by 15%. System trajectory

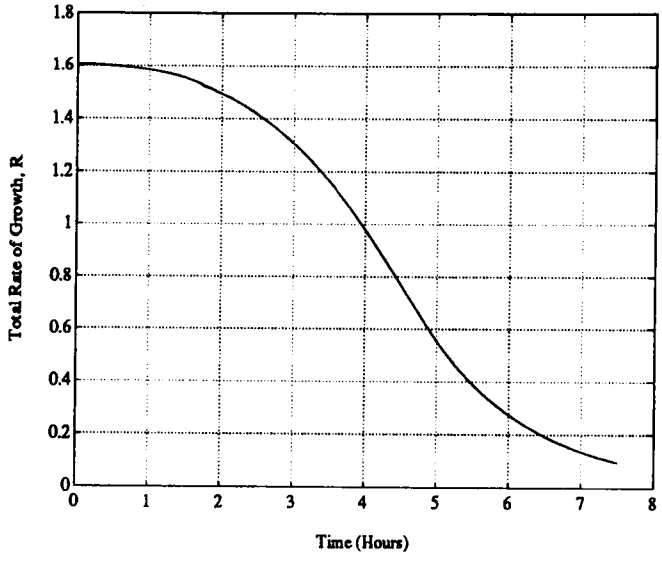

Figure 4.25: Set point change in the total rate of growth by 15%. Output versus time

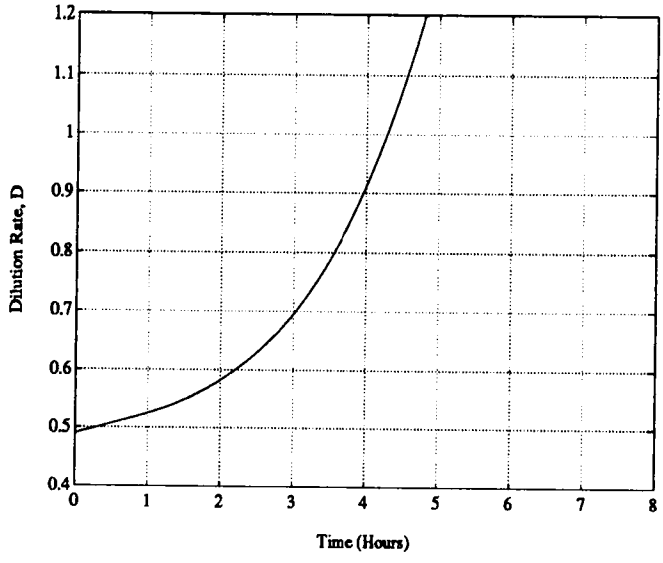

Figure 4.26: Set point change in the total rate of growth by 15%. Input versus time

able to drive the system toward the desired set point. In order to achieve the desired set point, the controller steadily increased the dilution rate. Since the relationship between the input and output of the system was now opposite in the sign of the steady state gain, the result was a drop in the total rate of growth. This result was expected, since the linearization is only valid close to the nominal set point and to the left of D_{opt}. In order to maintain stability with a linear controller, large deviations from the nominal set point normally are avoided. For this type of system, a set point well below and to the left of R_{max} must be selected to render a safe margin of stability. This selection results in poor performance of the system.

Although the set point may be selected close to the linearization point, a change in parameters can make the set point unachievable. The system response for an unachievable set point is shown in Figures 4.27, 4.28 and 4.29. In the simulations presented in these figures, the value of μ_{max} was decreased by 10%, making the desired output unachievable. The response of the system was similar to that obtained when the starting point was in the right side of R_{max}. The PID controller could not maintain the desired set point and tried to achieve it by increasing the dilution rate, until finally the system washed out.

These results are independent of the type of linear controller used, as long as they are based on a locally linearized model. Therefore, the set point should be chosen such that it will be achievable for all possible values of parameter variations. This selection will only maintain stability at the expense of suboptimal performance.

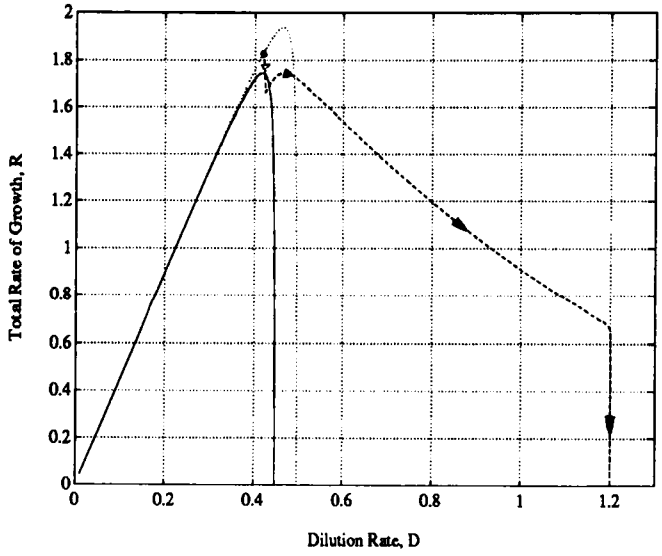

Figure 4.27: Change in μ_{max} by -10%. Unachievable set point. System trajectory

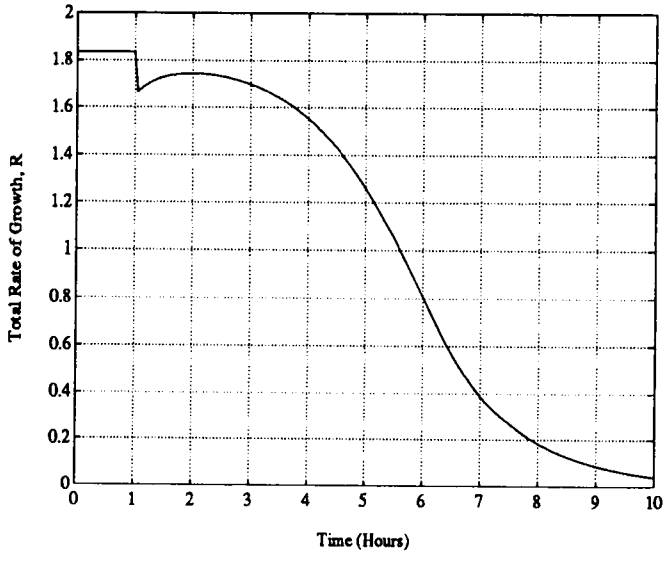

Figure 4.28: Change in μ_{max} by -10%. Unachievable set point. Output versus time

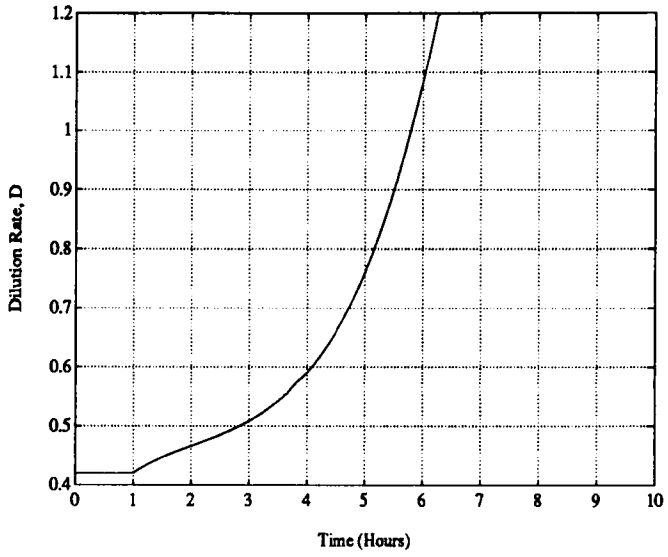

Figure 4.29: Change in μ_{max} by -10%. Unachievable set point. Input versus time

4.1.4 Nonlinear Controller with Global Optimization for a CSTBR: a Reference Case

Using the original nonlinear model, the same simulations described in Section 4.1.3, were performed with a controller that uses a global optimization algorithm. The function to be minimized is given by

$$\min_{D(t)} J(t) = \int_{to}^{\tau} \alpha^t [(r - R(t))^T (r - R(t))] dt - \frac{\alpha^\tau [(r - R(\tau))^T (r - R(\tau))]}{ln(\alpha)} \tag{4.19}$$

subject to

$$\dot{x} = f(x, D) \tag{4.20}$$

$$0 = f(x(\tau), D(\tau)) \tag{4.21}$$

$$D_{min} \leq D(t) \leq D_{max} \tag{4.22}$$

where x is the state vector

$$x = \begin{bmatrix} b \\ s \end{bmatrix},$$

and $f(x, D)$ is given by

$$f(x, D) = \begin{bmatrix} \mu x(1) - Dx(1) \\ D(s_f - x(2)) - \frac{\mu x(1)}{y_s} \end{bmatrix},$$

and μ is calculated using Equation 4.11 with a value of $\alpha = 0.99$.

The Hamiltonian function for the system is given by

$$H(x, \zeta, D, t) = \zeta^T f(x, D) + \alpha^t[(r - R(t))^T(r - R(t))]. \tag{4.23}$$

where ζ is a costate vector.

The conjugate gradient method is applied, as explained in Section 3.5.1. Given an initial estimate of the dilution rate D^i, the optimal dilution rate is found by calculating the gradient of H with respect to D. The gradient is used to determine a new value of D that, when clipped off at the upper and lower bounds, gives D^{i+1}. The iterative algorithm has the following form:

$$D(t)^{i+1} = \begin{cases} D(t)^i + \beta^i p(t)^i & \text{if } D_{min} \leq D(t) \leq D_{max} \\ D_{max} & \text{if } D(t) > D_{max} \\ D_{min} & \text{if } D(t) < D_{min} \end{cases}$$

where β^i is a nonnegative scalar minimizing $H(D^i - \beta^i \nabla H(D)^{iT})$ and p^i is given by the equation:

$$p^i = -\nabla H(D)^i + \frac{\int_{to}^{\tau} < \nabla H(D)^i, \nabla H(D)^i > dt}{\int_{to}^{\tau} < \nabla H(D)^{i-1}, \nabla H(D)^{i-1} > dt} \tag{4.24}$$

The computer program that implements this optimization procedure is shown in Appendix A. A sampling time of 0.05 hours was used in all the simulations.

Results of the first test of the global-optimum controller, in which the initial dilution rate has a value lower than D_{opt}, is shown in Figures 4.30, 4.31 and 4.32. These figures show that the system attained the desired value in about the same time as that of ELCC; but in this case, there was no overshoot.

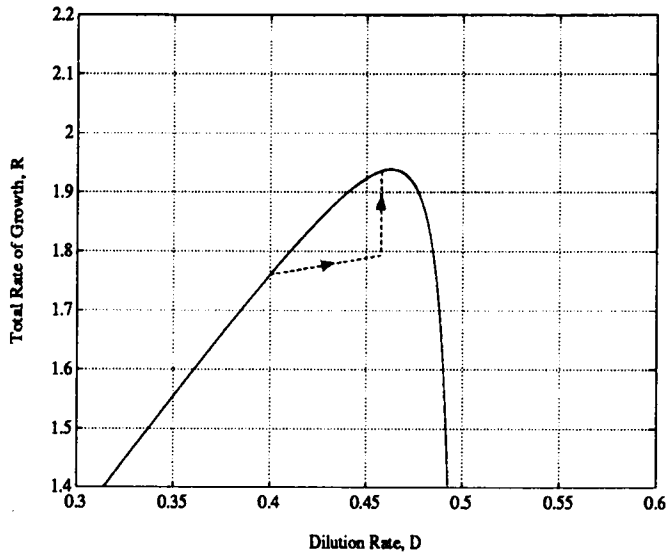

Figure 4.30: Set point change in the total rate of growth by 15%. System trajectory

In the second simulation, an initial dilution rate larger than D_{opt} was selected to demonstrate the global-optimum controller's ability to work on both sides of the maximum point. Figures 4.33, 4.34 and 4.35 show that the controller was able to maintain stability. However, the system used more time than the pointwise optimization to reach the desired output value.

Additional simulations were performed using the nominal system (see Appendix D). In all the simulations, stability was maintained. However, the simulations that started with a dilution rate larger than D_{opt} reached the desired set point in longer time than

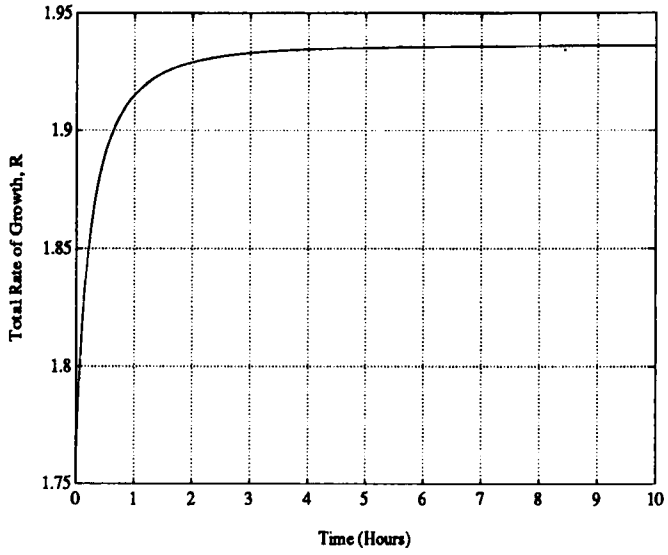

Figure 4.31: Set point change in the total rate of growth by 15%. Output versus time

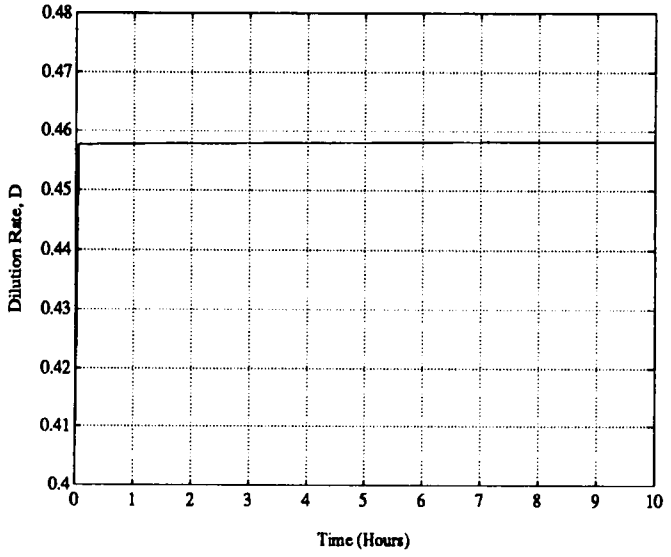

Figure 4.32: Set point change in the total rate of growth by 15%. Input versus time

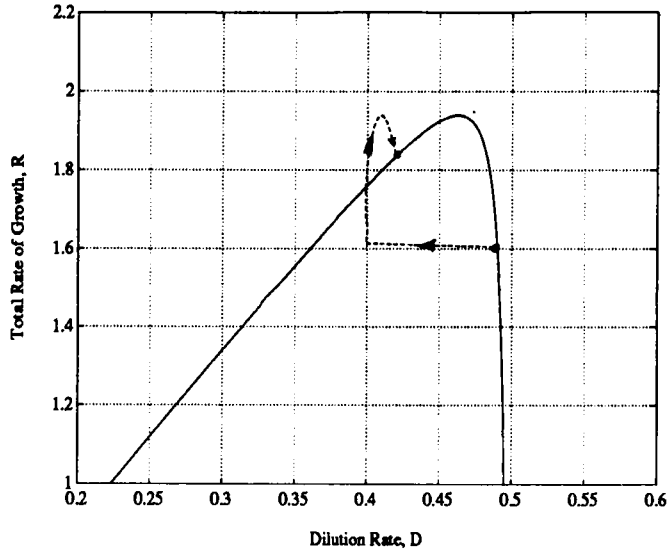

Figure 4.33: Set point change in the total rate of growth by 15%. System trajectory

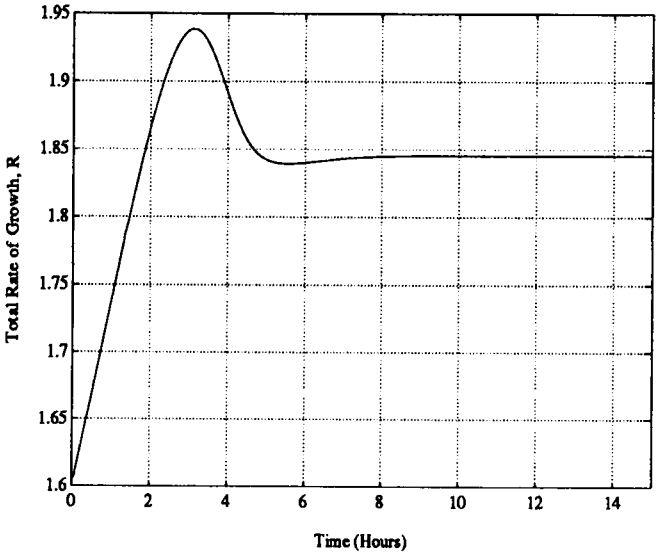

Figure 4.34: Set point change in the total rate of growth by 15%. Output versus time

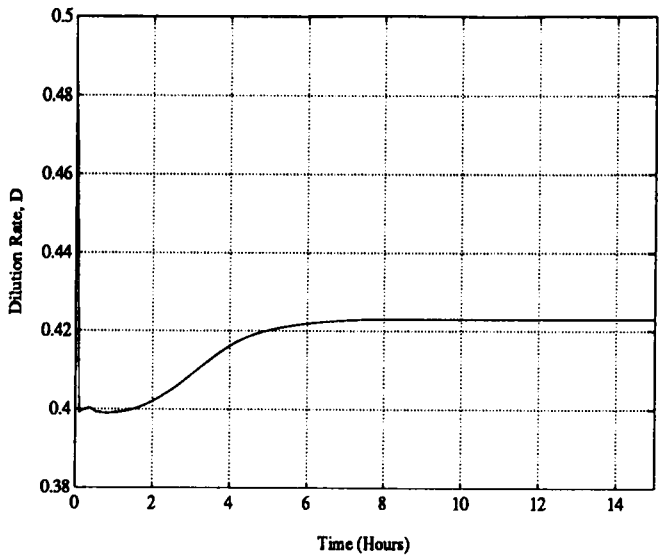

Figure 4.35: Set point change in the total rate of growth by 15%. Input versus time

the corresponding simulations with the pointwise optimization algorithm.

In order to the demonstrate global-optimum controller's ability to maintain stability in the presence of parameter variations, the value of y_s was decreased by 10%. Figures 4.36, 4.37 and 4.38 show that the controller did maintain stability. However, the final steady state point reached was lower than that obtained using ELCC (see Figures 4.12, 4.13 and 4.14). For these changes, the final total rate of growth with the global optimal controller was 1.64, whereas the final value obtained with ELCC was 1.73. The final value of the dilution rate obtained with the optimal controller was very close to the original value. The controller did not try to look for the maximum point.

A second parameter variation was implemented. The value of μ_{max} was decreased by 10%. The initial dilution rate was selected to the right side of nominal R_{max}. If the controller maintained the same dilution rate it would result in a system wash out.

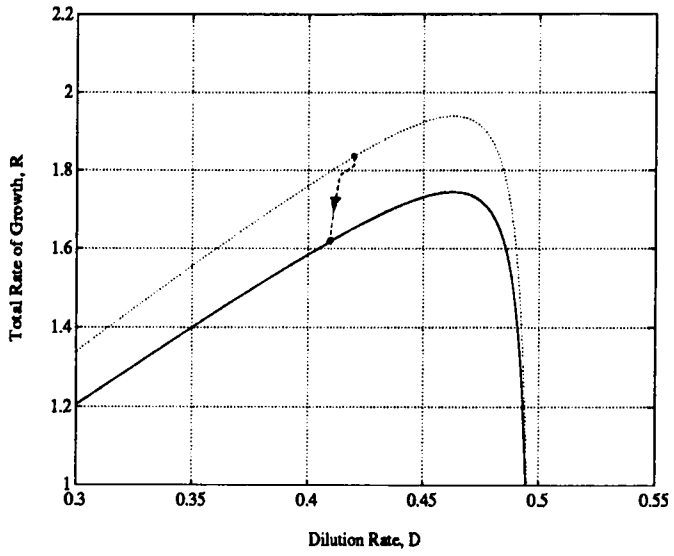

Figure 4.36: Change in y_s by -10%. Unachievable set point. System trajectory

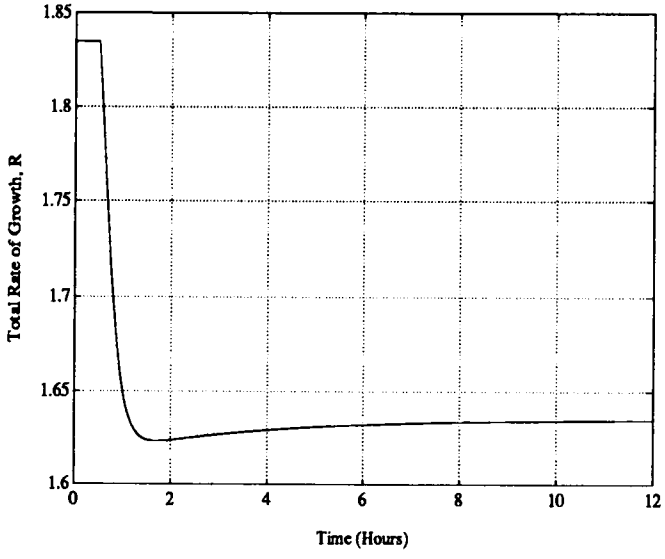

Figure 4.37: Change in y_s by -10%. Unachievable set point. Output versus time

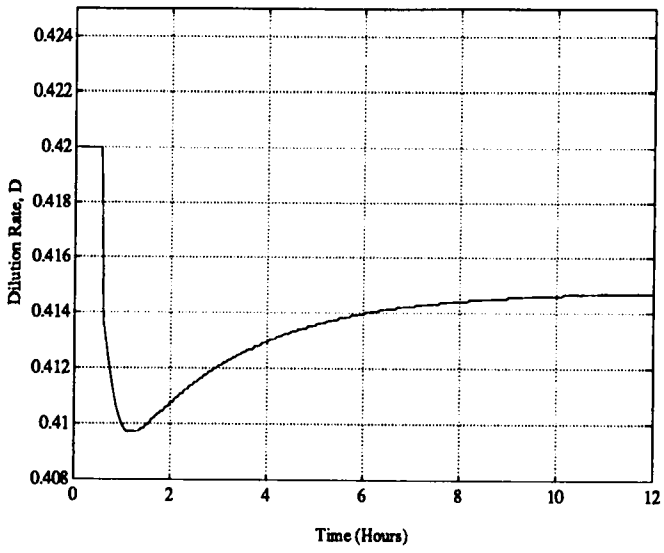

Figure 4.38: Change in y_s by -10%. Unachievable set point. Input versus time

The results of this simulation are shown in Figures 4.39, 4.40 and 4.41. The controller was able to maintain stability; but again, the position of the maximum point was lower than the value obtained using a pointwise optimization algorithm (see Figures 4.15, 4.16 and 4.17).

4.1.5 Exothermic Reactor

The ELCC scheme was also applied to the control of an exothermic reactor, as shown in Fig 4.42. This reactor example was the same as that used by Economou and Morari [17]. The process consisted of an ideal CSTR where a reversible chemical reaction was carried out

$$A \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} B$$

From component balance equations, one can obtain the process model as:

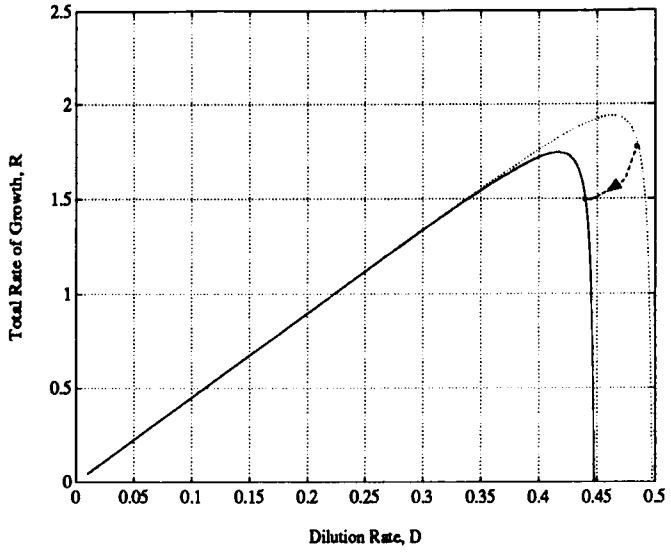

Figure 4.39: Change in μ_{max} by -10%. Unachievable set point. System trajectory

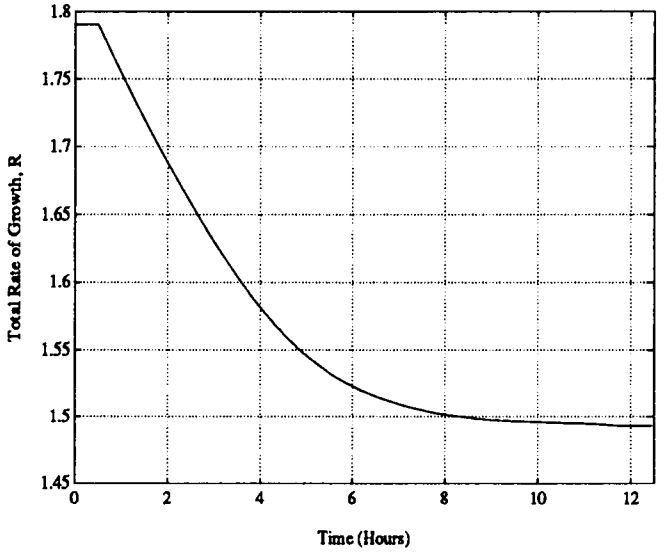

Figure 4.40: Change in μ_{max} by -10%. Unachievable set point. Output versus time

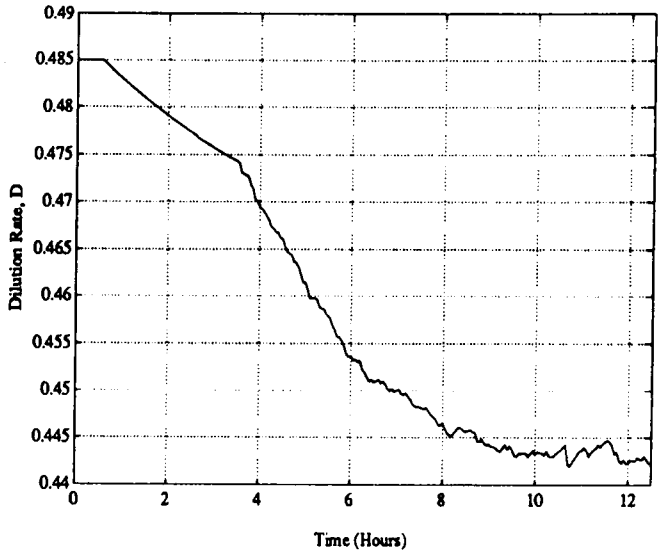

Figure 4.41: Change in μ_{max} by -10%. Unachievable set point. Input versus time

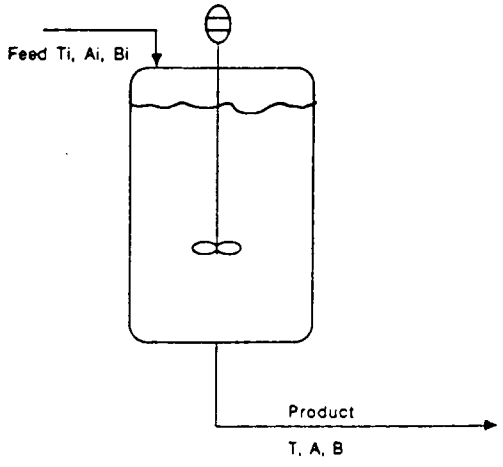

Figure 4.42: Exothermic CSTR

- *State Equations:*

$$\frac{dA}{dt} = \frac{A_i - A}{\tau} - k_1 A + k_{-1} B \tag{4.25}$$

$$\frac{dB}{dt} = \frac{B_i - B}{\tau} + k_1 A - k_{-1} B \tag{4.26}$$

$$\frac{dT}{dt} = \frac{-\Delta H}{\rho C_p}(k_1 A - k_{-1} B) + \frac{T_i - T}{\tau} \tag{4.27}$$

- *Output Equation:*

$$y = B \tag{4.28}$$

where:

- τ is the residence time;
- k_1 and k_{-1} are the rate constants of the forward and reverse reaction respectively, and are defined by the equations $k_1 = C_1 e^{-Q_1/RT}$ and $k_{-1} = C_{-1} e^{-Q_{-1}/RT}$;
- ΔH is the heat of reaction;
- C_p is the specific heat of the solution;
- T is the temperature of the reactor; and
- ρ is the density of the solution.

The parameters used in the example and simulation study are shown in Table 4.3.

The steady state of the exothermic reactor can be obtained from the solution to the following coupled algebraic equations:

$$\frac{A_i - A}{\tau} - k_1 A + k_{-1} B = 0 \tag{4.29}$$

Table 4.3: Parameters of the exothermic reactor

τ	$60\ s^{-1}$	A_i	$1.0\ mol/l$
C_1	$5 \times 10^3\ s^{-1}$	B_i	$0.0\ mol/l$
C_{-1}	$1 \times 10^6\ s^{-1}$	T_i	$427\ K$
Q_1	$10000\ cal/mol$	A	$0.492\ mol/l$
Q_{-1}	$15000\ cal/l$	B	$0.508\ mol/l$
R	$1.987\ cal/mol/K$	T	$430\ K$
ΔH	$-5000\ cal/mol$	T_{opt}	$434\ K$
ρ	$1\ kg/l$		
C_p	$1000\ cal/kg/K$		

$$\frac{B_i - B}{\tau} + k_1 A - k_{-1} B = 0 \tag{4.30}$$

$$\frac{-\Delta H}{\rho C_p}(k_1 A - k_{-1} B) + \frac{T_i - T}{\tau} = 0 \tag{4.31}$$

The steady state can be easily solved as functions of the steady-state reactor temperature T as:

$$A = A_i \frac{\frac{1}{\tau} + C_{-1} e^{-Q_{-1}/RT}}{\frac{1}{\tau} + C_1 e^{-Q_1/RT} + C_{-1} e^{-Q_{-1}/RT}} \tag{4.32}$$

$$B = A_i - A \tag{4.33}$$

$$T_i = T + \frac{\Delta H\, \tau\, (C_1 e^{-Q_1/RT} A - C_{-1} e^{-Q_{-1}/RT} B)}{\rho\, C_p} \tag{4.34}$$

As a function of the feed temperature, the reactor equilibrium concentration (conversion)

at a fixed A_i has a well-defined maximum, shown as the solid line in Figure 4.43, where $T_{i,op}$ is the inlet temperature corresponding to the optimal B.

The control objective is to operate the reactor very near to this optimal point, at the same time maintaining the stability of the closed-loop system in the presence of disturbances and/or model uncertainties. The inlet stream feed temperature T_i and the desired product output concentration B are selected as the manipulated (u) and controlled (y) variables, respectively.

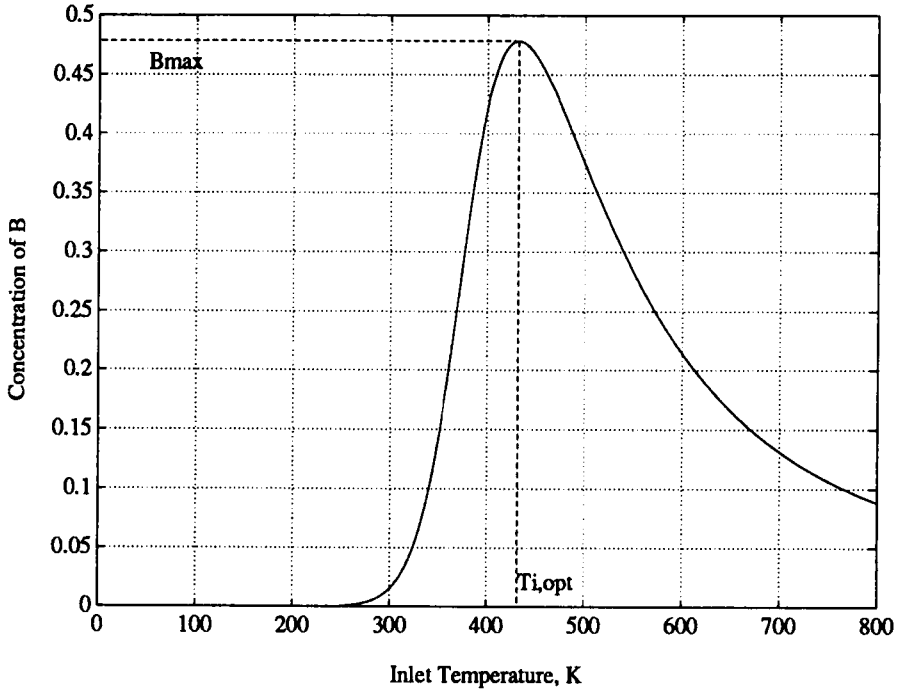

Figure 4.43: Steady state diagram of an exothermic reactor

Rearrange equations 4.25, 4.26, 4.27, and 4.28 according to

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

$$y = h(x)$$

yielding

$$x = \begin{bmatrix} A \\ B \\ T \end{bmatrix}$$

$$u = T_i$$

$$f(x) = \begin{bmatrix} \frac{A_i - A}{\tau} - k_1 A + k_{-1} B \\ \frac{B_i - B}{\tau} + k_1 A - k_{-1} B \\ \frac{-\Delta H}{\rho C_p}(k_1 A - k_{-1} B) - \frac{T}{\tau} \end{bmatrix}$$

$$g(x) = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{\tau} \end{bmatrix}$$

$$h(x) = B$$

The control problem in the predictive phase of the control design becomes:

$$\min_{u_k, \cdots, u_{k+M}} \sum_{i=1}^{P} [(r - y_{k+i})^T (r - y_{k+i})] \tag{4.35}$$

subject to

$$\dot{x} = f(x, u) \tag{4.36}$$

$$u_{min} \leq u_k \leq u_{max} \tag{4.37}$$

where 4.37 represents the inequality constraints on the process input, r represents the desired concentration of B, N represents the number of future input moves, and P represents the number of sample intervals for which the output of the system will be predicted.

4.1.6 Controller Design for a CSTR

In order to linearize the system, the relative degree is determined by calculating the first derivative of the output with respect to time as follows

$$\frac{dy}{dt} = \frac{B_i - x_2}{\tau} + k_1 x_1 - k_{-1} x_2. \tag{4.38}$$

Since the first derivative does not depend on the input explicitly, the second derivative of the output with respect to time is calculated as

$$\begin{aligned}\frac{d^2y}{dt^2} &= k_1\left(\frac{A_i - A}{\tau} - k_1 A + k_{-1} B\right) - \left(\frac{1}{\tau} + k_1\right)\left(\frac{B_i - B}{\tau} + k_1 A - k_{-1} B\right) + \\ &\frac{k_1 Q_1 A - k_{-1} Q_{-1} B}{RT^2}\left(\frac{\Delta H}{\rho C_p}(k_1 A - k_{-1} B) - \frac{T}{\tau}\right) + \frac{k_1 Q_1 A - k_{-1} Q_{-1} B}{\tau R T^2} T_i\end{aligned}$$

Since the second derivative of the output depends on T_i explicitly, the system has a relative degree of two, and the linearizing feed back can be written as

$$u = \frac{(v - L_f^2 h(x) - 3L_f h(x) - h(x)) L_g L_f h(x)}{\epsilon + (L_g L_f h(x))^2}, \tag{4.39}$$

where $L_f h(x)$ is

$$L_f h = \frac{B_i - B}{\tau} + k_1 A - k_{-1} B,$$

$L_f^2 h(x)$ is

$$\begin{aligned}L_f^2 h &= k_1\left(\frac{A_i - A}{\tau} - k_1 A + k_{-1} B\right) \\ &\quad - \left(\frac{1}{\tau} + k_1\right)\left(\frac{B_i - B}{\tau} + k_1 A - k_{-1} B\right) \\ &\quad + \frac{k_1 Q_1 A - k_{-1} Q_{-1} B}{RT^2}\left(\frac{\Delta H}{\rho C_p}(k_1 A - k_{-1} B) - \frac{T}{\tau}\right),\end{aligned}$$

and $L_g L_f h(x)$ is

$$L_g L_f h = \frac{k_1 Q_1 A - k_{-1} Q_{-1} B}{\tau R T^2}.$$

The linearized system was chosen to have the form

$$\frac{d^2y}{dt^2} + 3\frac{dy}{dt} + y = v. \tag{4.40}$$

Note that the system as described by Equation 4.40 is open-loop stable, as required by

Table 4.4: CSTR Control Parameters

Parameter	Value
ϵ	4×10^{-12}
λ	10
P	30
M	20
N	10

the DMC algorithm. The DMC algorithm was applied to the linearized system using the procedures described in Section 3.3. The calculated values of v were used to calculate T_i. The values of the control parameters are shown in Table 4.4.

4.1.7 Simulation Results for the CSTR

Because of the presence of a maximum in the steady state input-output profile of the CSTR system, and since the operating point of interest is often at or near this maximum point, traditional linear controller is not applicable. These requirements are similar to those of the CSTBR introduced in Section 4.1. The performance of the ELCC was tested in two steps. In the first step, the nominal performance of the system was tested by introducing both step changes in the desired set point and disturbances. In the second step, the performance of the ELCC in the presence of system parameter variations that

result in an unachievable set points was tested by introducing changes in C_1 and C_2. A sampling time of 1 second, the same sampling time used by Economou and Morari [17] was used in all the simulations.

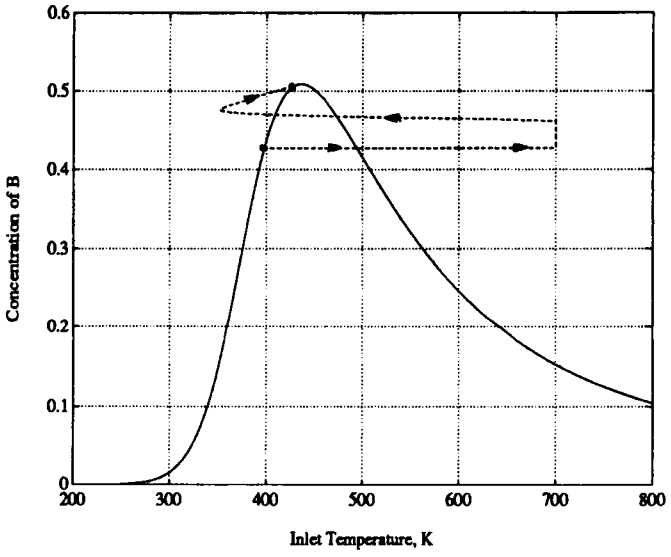

Figure 4.44: Set point change in the concentration of B by 20%. System trajectory

Nominal Performance of a CSTR

In the first test of the controller, the initial inlet temperature was chosen to have a value lower than $T_{i,op}$. Then a step change of +20% in the output was introduced. Figures 4.44, 4.45 and 4.46 show that the system attained the desired value in less than 3 minutes. The input dynamics show that the upper constraint became active during the first minute of operation. The controller was able to maintain the system within the specified input constraints and was able to drive the system to the desired set point.

In the second test, the initial T_i was chosen to be larger than $T_{i,op}$. Figures 4.47, 4.48

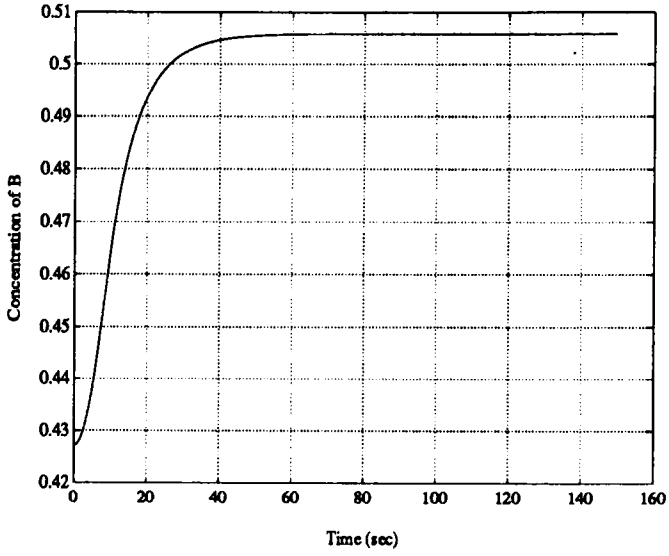

Figure 4.45: Set point change in the concentration of B by 20%. Output versus time

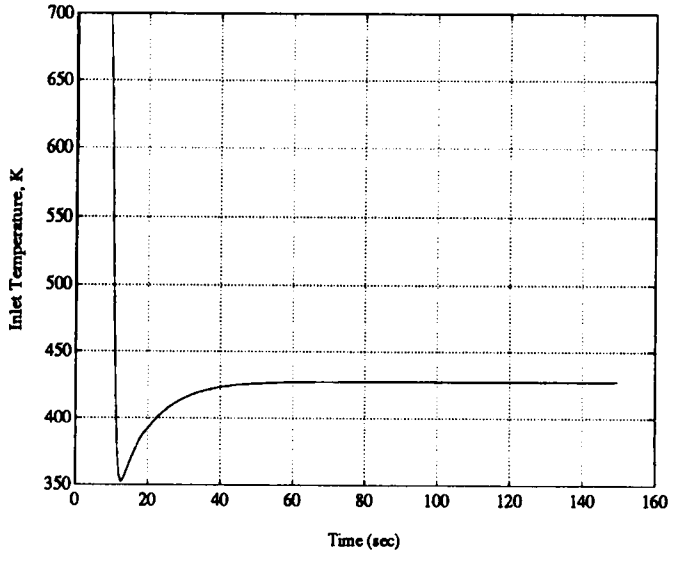

Figure 4.46: Set point change in the concentration of B by 20%. Input versus time

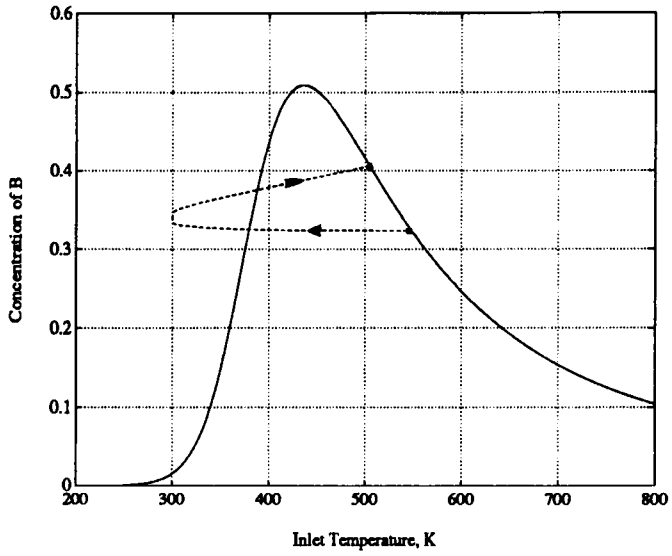

Figure 4.47: Set point change in the concentration of B by 25%. System trajectory

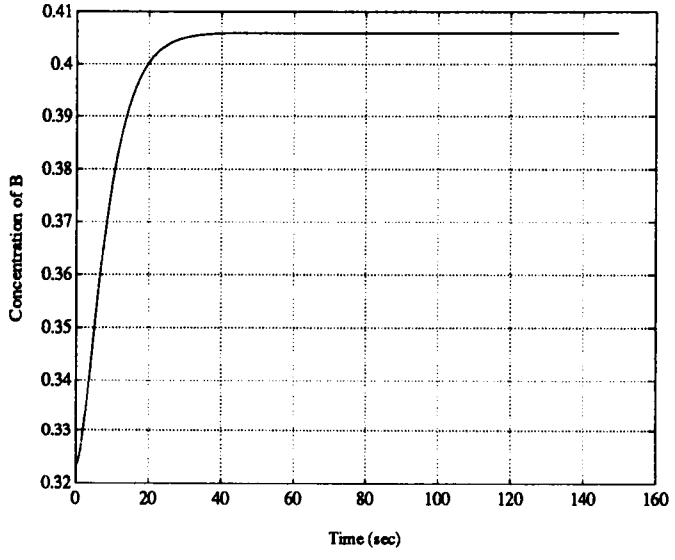

Figure 4.48: Set point change in the concentration of B by 25%. Output versus time

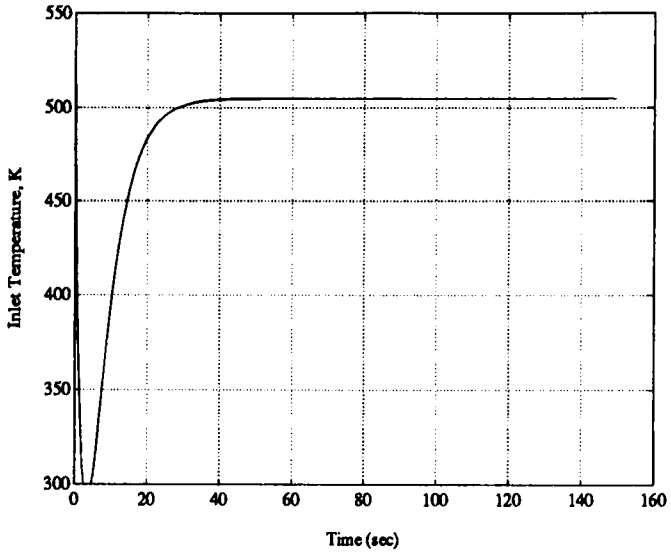

Figure 4.49: Set point change in the concentration of B by 25%. Input versus time

and 4.49 show that the controller was able to drive the system to the new set point, although the relationship between the input and the output of the system changed. This result agrees with those obtained with the bioreactor. Since the linearization is valid for the entire region of operation, the performance of the controller is independent of the initial point of operation. Other simulations using the nominal plant are shown in Appendix D. These results show that the performance of the controller was not affected by the selection of the initial conditions in the reactor. In all the simulations, the system reached the desired set point in less than 40 seconds.

Parameter Variations for a CSTR

Figures 4.50 and 4.51 show the effect of changes in k_1 or k_{-1} in the steady state profile for the CSTR. A change in the value of these parameters changes the value of the steady-

state maximum concentration of B. If the change was such that the maximum possible value was smaller than the desired set point, the set point became unachievable. It was shown with the bioreactor that a linear controller could not maintain stability under these conditions. It will be shown in the following simulations that under the same conditions the ELCC was able to maintain stability and to drive the system close to the maximally achievable concentration, thus reducing the offset from the specified set point.

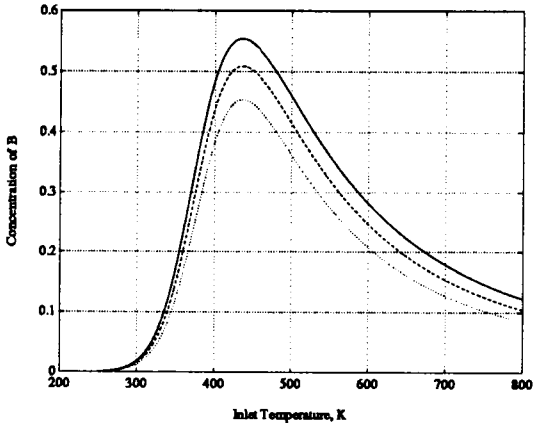

Figure 4.50: Steady state diagram of a CSTR. Effect of k_1. Dash line corresponds to nominal value, solid line to an increase of 20%, and dot line to a decrease of 20%.

In order to demonstrate the ability of the ELCC to maintain stability in the presence of parameter variations, two disturbances were introduced in the system. First, the value of k_{-1} was increased by 20%. Figures 4.52, 4.53 and 4.54 show that the controller maintained stability and drove the system close to the maximum achievable point under these conditions. In general, the value of ϵ of Equation 4.39 determines how close to the maximum point the final value of the output would be.

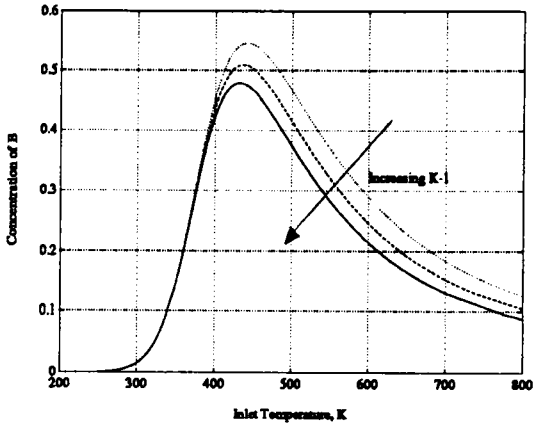

Figure 4.51: Steady state diagram of a CSTR. Effect of k_{-1}. Dash line corresponds to nominal value, Solid line to an increase of 20%, and dot line to a decrease of 20%.

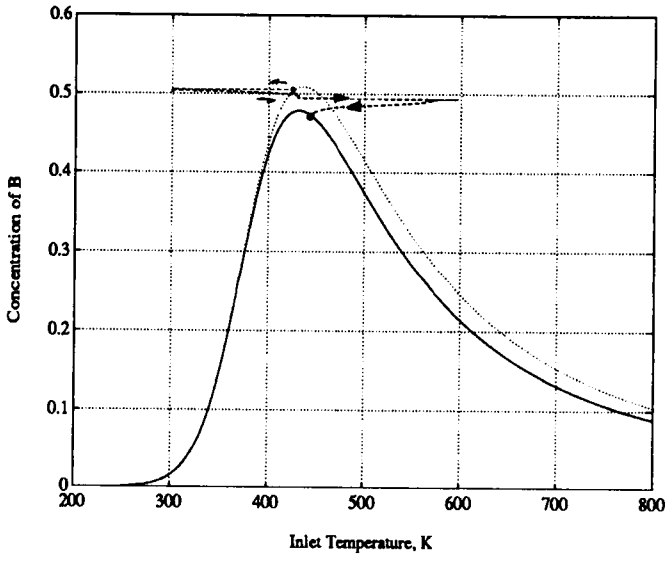

Figure 4.52: Change in k_{-1} by +20%. Unachievable set point. System trajectory

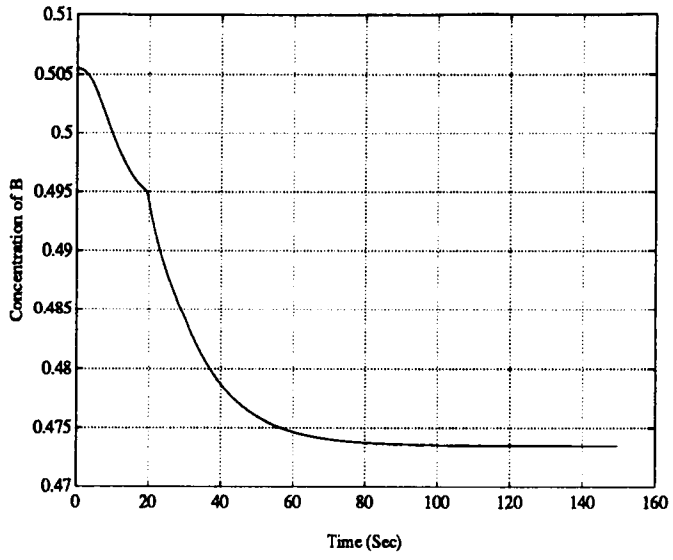

Figure 4.53: Change in k_{-1} by +20%. Unachievable set point. Output versus time

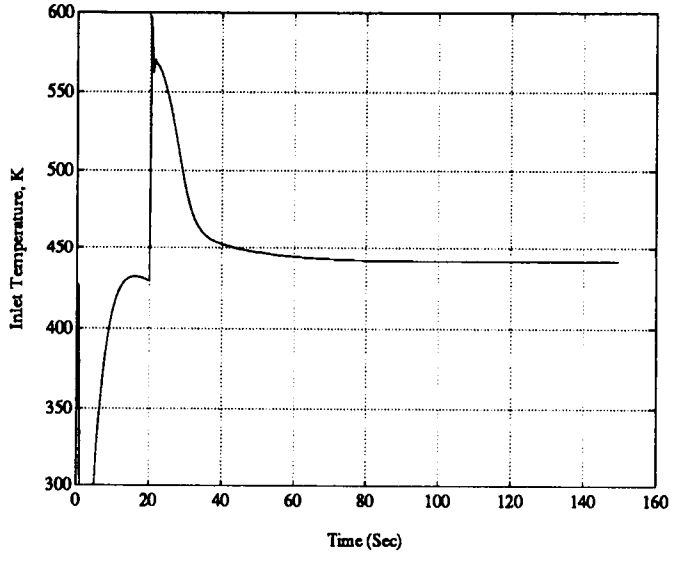

Figure 4.54: Change in k_{-1} by +20%. Unachievable set point. Input versus time

A second parameter variation was implemented by introducing simultaneously an increase in k_{-1} and a change in the inlet concentration of A. An increase in k_{-1} decreased the value of the maximum achievable concentration of B, and a variation in the inlet concentration A_i changed the the inlet temperature for which the maximum was achieved. The results for an increase in the value of k_{-1} by 50% and a decrease in A_i by 20% are shown in Figures 4.55, 4.56 and 4.57. These figures show that, although the maximum point was shifted to the left and to a lower position, the controller still was able to maintain stability and to operate the system close to the maximum possible point. These two simulation scenarios were not study by other researchers.

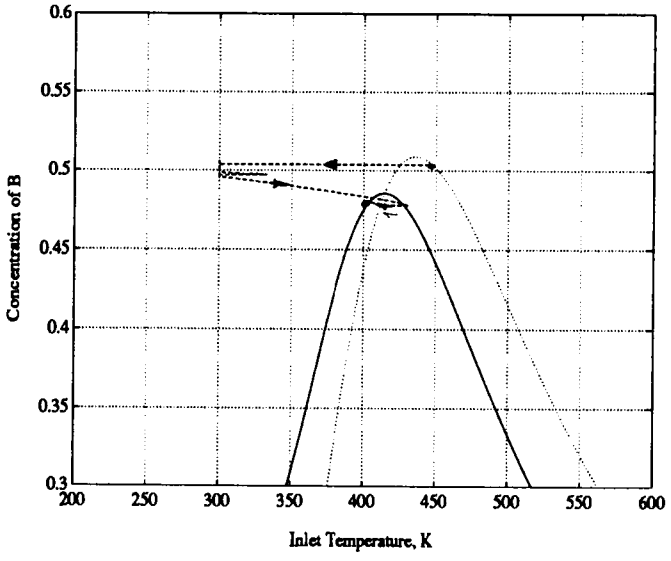

Figure 4.55: Change in k_{-1} by +50% and A_i by -20%. Unachievable set point. System trajectory

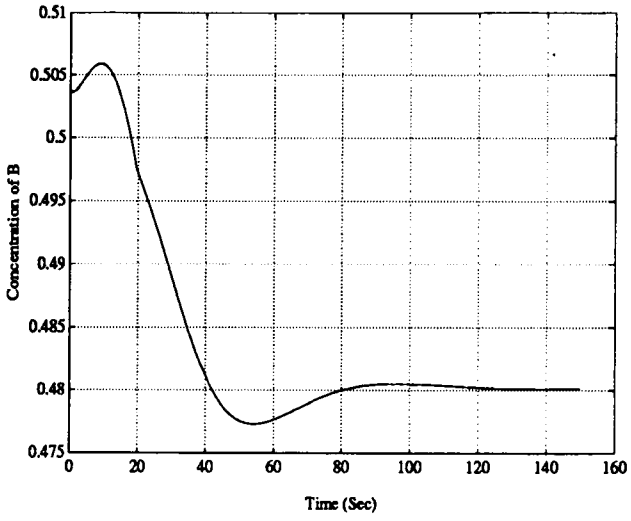

Figure 4.56: Change in k_{-1} by +50% and A_i by -20%. Unachievable set point. Output versus time

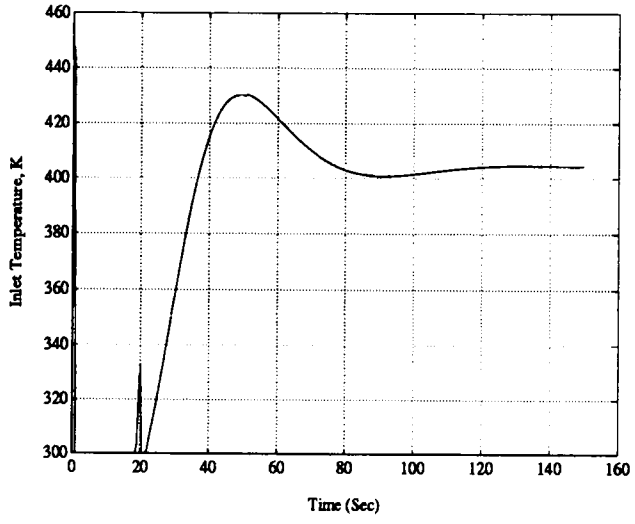

Figure 4.57: Change in k_{-1} by +50% and A_i by -20%. Unachievable set point. Input versus time

4.1.8 Linear QDMC Controller for a CSTR

The results obtained with the ELCC were compared with those obtained with a linear Quadratic Programming Solution of the Dynamic Matrix Control controller (QDMC). This controller was used to demonstrate the inability of a linear controller of this kind to maintain stability in the presence of parameter variations that render the desired set point unachievable. As expalined in Section 4.1.3, a linear controller is tuned to control a system that is linearized locally at a steady state operating point by a truncated Taylor series expansion around that point. The control parameters used in this simulation are shown on Table 4.5

Table 4.5: QDMC paramters

Parameter	Value
M	80
N	20
P	100
Sampling Time	2.0 (sec)

The nominal performance was first tested with the conditions near the neighborhood of the nominal set point. Then, a step change of 15% in the desired output was introduced. The results of this simulation are shown on Figures 4.58, 4.59 and 4.60. These figures show that the system performed satisfactorily and achieved the desired set point in 800 seconds. For this perturbation the linear QDMC was able to maintain stability, and its performance was close to that of the globally linearized system.

Although the set point may be selected close to the linearization point, a change

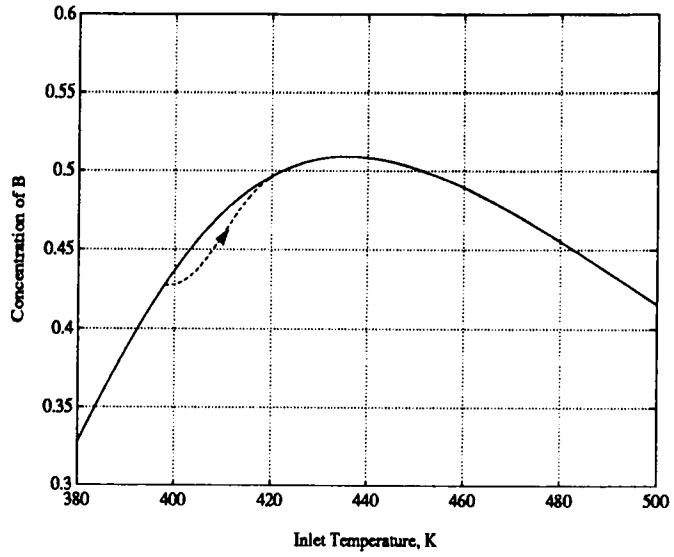

Figure 4.58: Set point change in the concentration of B by 15%. System trajectory

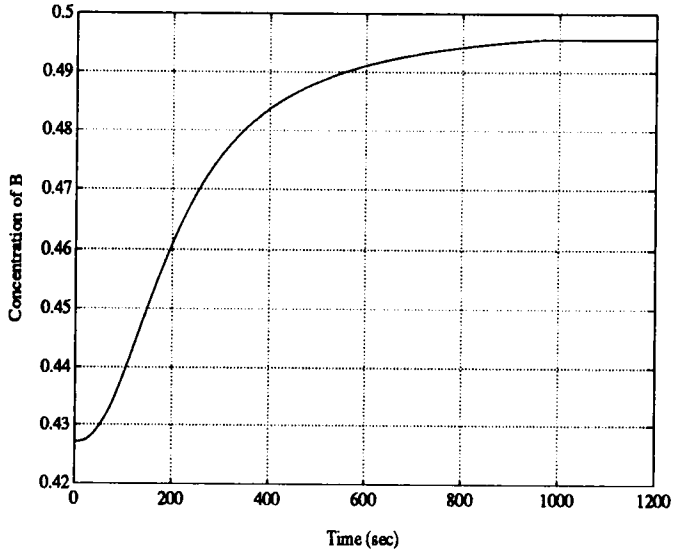

Figure 4.59: Set point change in the concentration of B by 15%. Output versus time

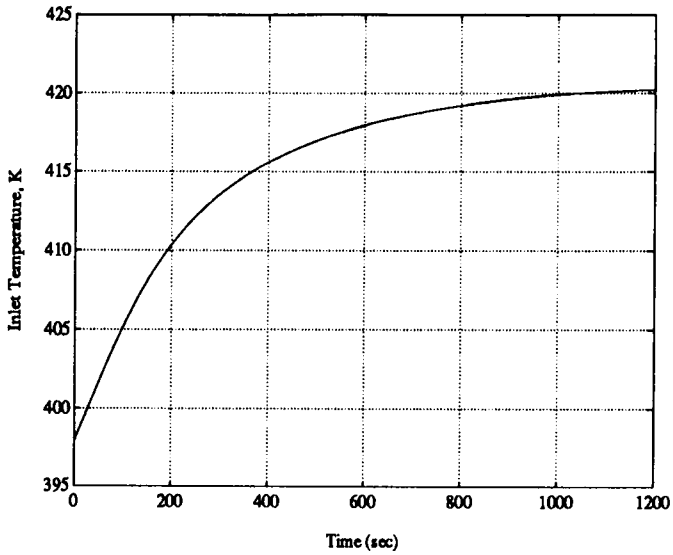

Figure 4.60: Set point change in the concentration of B by 15%. Input versus time

in a parameter can make the set point value unachievable. The system response for an unachievable set point is shown in Figures 4.61, 4.62 and 4.63. In the simulations presented in these figures, the value of k_{-1} was decreased by 20%, making the desired output unachievable. The linear QDMC controller could not maintain the desired set point and tried to achieve it by increasing the inlet temperature, producing a drop in concentration.

These results are independent of the type of linear controller used, as long as they are based on a locally linearized model. Therefore, the set point should be chosen such that it will be achievable for all possible values of parameter variations. This selection will maintain stability only at the expense of suboptimal performance.

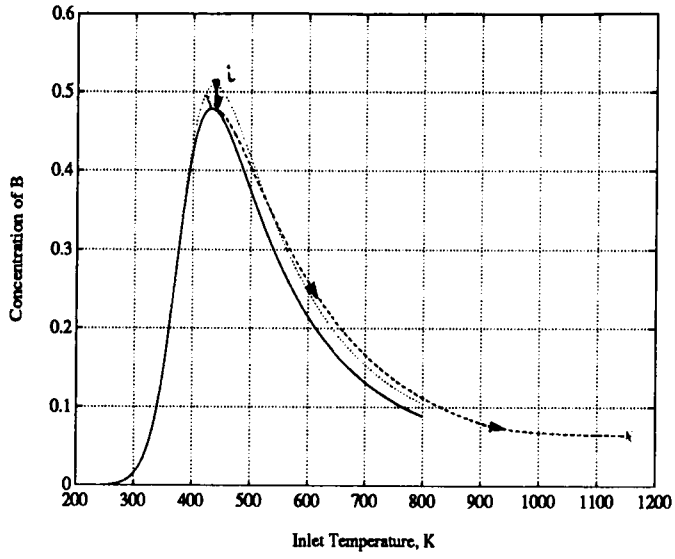

Figure 4.61: Change in k_{-1} by +20%. Unachievable set point. System trajectory

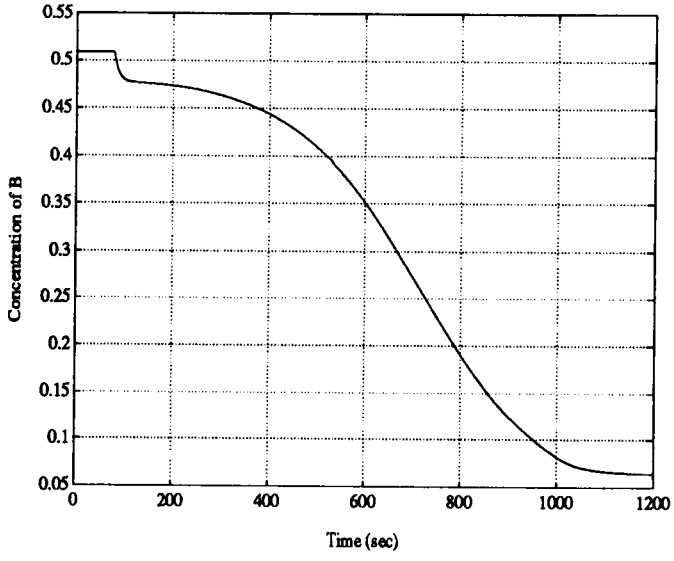

Figure 4.62: Change in k_{-1} by +20%. Unachievable set point. Output versus time

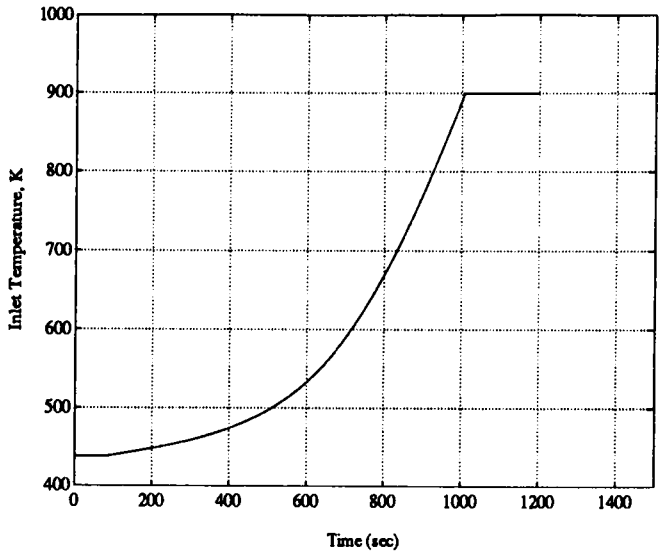

Figure 4.63: Change in k_{-1} by +20%. Unachievable set point. Input versus time

4.1.9 Nonlinear Controller with Global Optimization for a CSTR: A Reference Case

The same simulations described in Section 4.1.7 were performed with a controller that uses the global optimization algorithm described in Section 3.5. The function to be minimized is given by

$$\min_{u(t)} J(t) = \int_{to}^{\tau} \alpha^t[(r - y(t))^T(r - y(t))]dt - \frac{\alpha^\tau[(r - y(\tau))^T(r - y(\tau))]}{ln(\alpha)} \tag{4.41}$$

subject to

$$\dot{x} = f(x, u) \tag{4.42}$$

$$0 = f(x(\tau), u(\tau)) \tag{4.43}$$

$$u_{min} \leq u(t) \leq u_{max}. \tag{4.44}$$

The Hamiltonian function for the system is given by

$$H(x,\zeta,u,t) = \zeta^T f(x,u) + \alpha^t[(r - y(t))^T(r - y(t))], \tag{4.45}$$

where ζ is a costate vector. The same optimization algorithm used with the bioreactor was used to estimate the inlet reactor temperature u that minimizes 4.41 (see Appendix B). A sampling time of 5 seconds was used in all the simulations and a value of $\alpha = 0.9$.

In the first simulation, a change in the desired set point of +20% was implemented. Results of this simulation are shown in Figures 4.64, 4.65 and 4.66. In this simulation the initial, inlet temperature was chosen to have a value smaller than $T_{i,op}$. The new desired set point was achieved in less than 200 seconds. This result is close to the one obtained using the ELCC, but without overshoot.

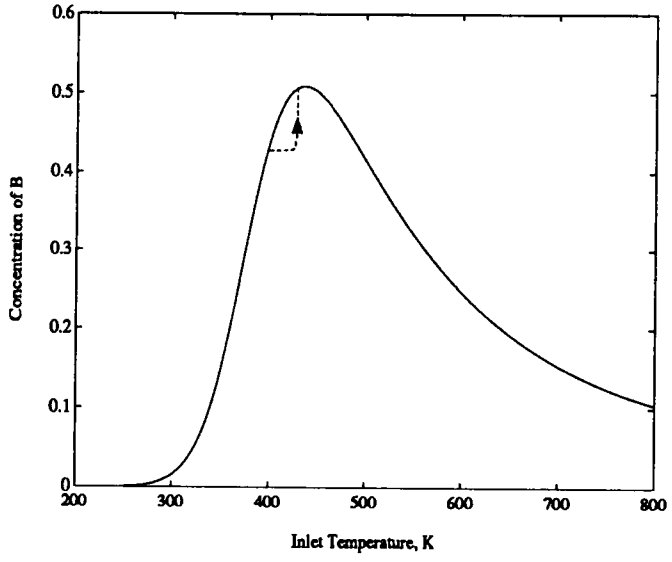

Figure 4.64: Set point change in the concentration of B by 20%. System trajectory

In the second simulation, an initial value of T_i larger than $T_{i,op}$ was selected to

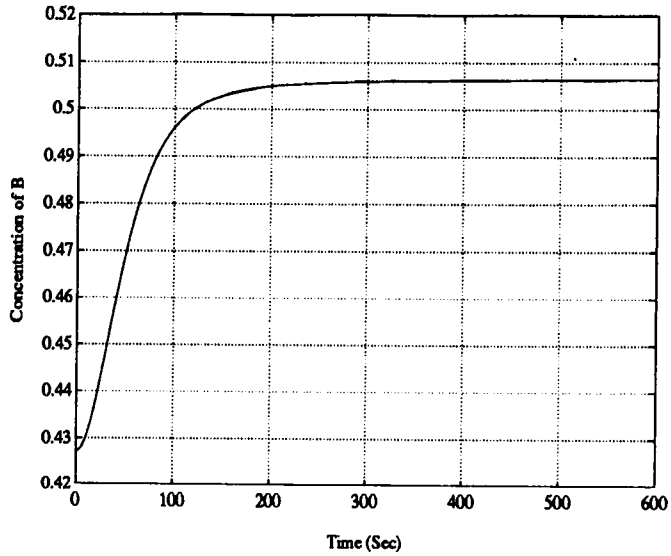

Figure 4.65: Set point change in the concentration of B by 20%. Output versus time

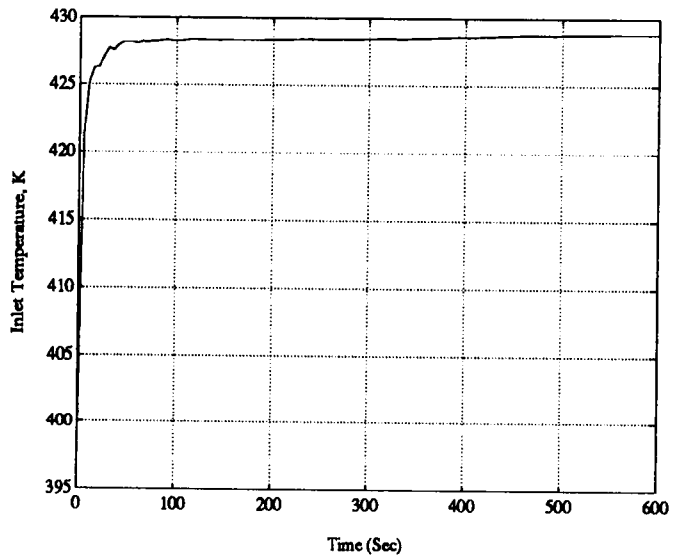

Figure 4.66: Set point change in the concentration of B by 20%. Input versus time

demonstrate the global-optimum controller's ability to work in both sides of the maximum point. Figures 4.67, 4.68 and 4.69 show that the global-optimum controller was able to maintain stability. However, the system used more time than the pointwise optimization to reach the desired output value (see Figures 4.44, 4.45 and 4.46).

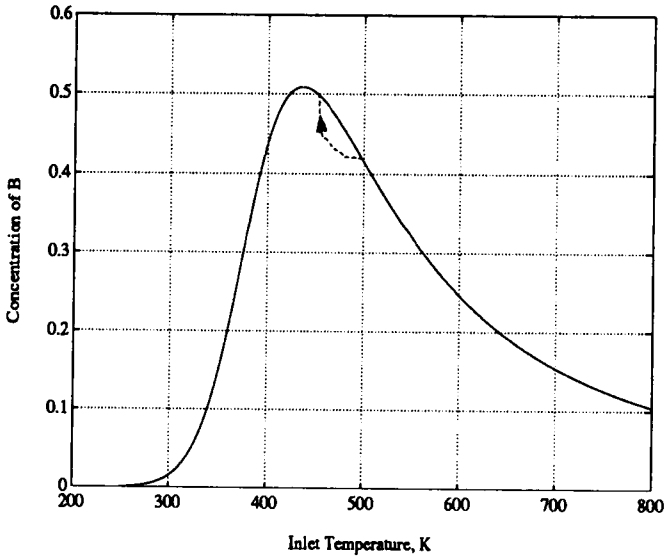

Figure 4.67: Set point change in the concentration of B by 20%. System trajectory

In order to the demonstrate the global-optimum controller's ability to maintain stability in the presence of parameter variations that lead to set point unachievability, and to compare the results with the ELCC, the value of k_{-1} was increased by +20%. Figures 4.70, 4.71 and 4.72 show that the controller was again able to maintain stability. However, a significant difference is that the controller set the output to a value lower than that obtained using the ELCC (Figure 4.52).

A second parameter variation was implemented by introducing an increase of 50% in k_{-1} and a decrease of 20% in A_i. Results of this simulation are shown in Figures 4.73, 4.74

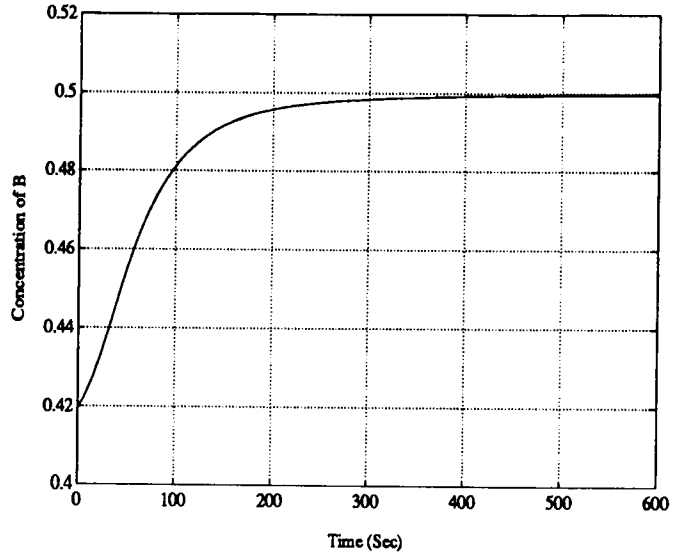

Figure 4.68: Set point change in the concentration of B by 20%. Output versus time

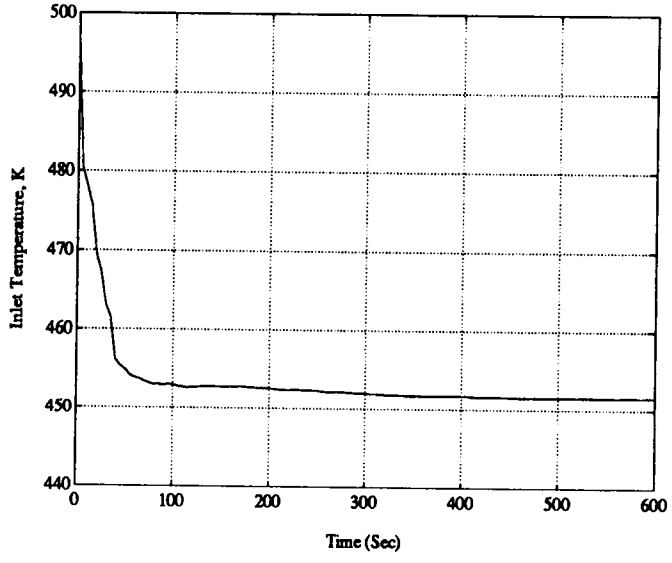

Figure 4.69: Set point change in the concentration of B by 20%. Input versus time

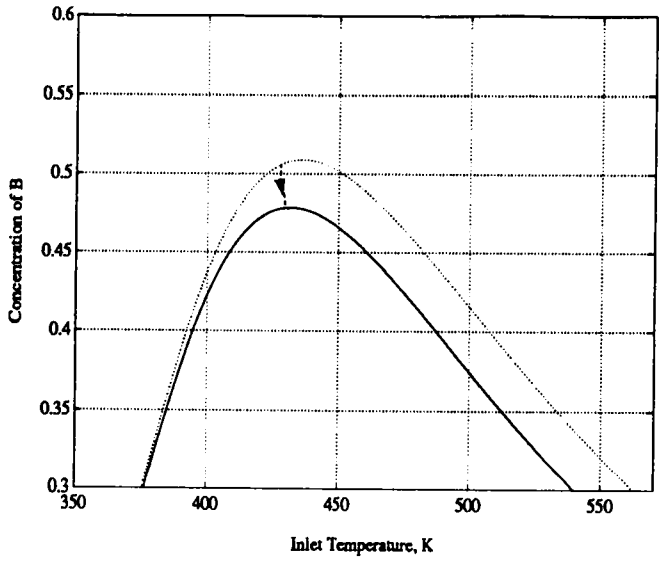

Figure 4.70: Change in k_{-1} by +20%. Unachievable set point. System trajectory

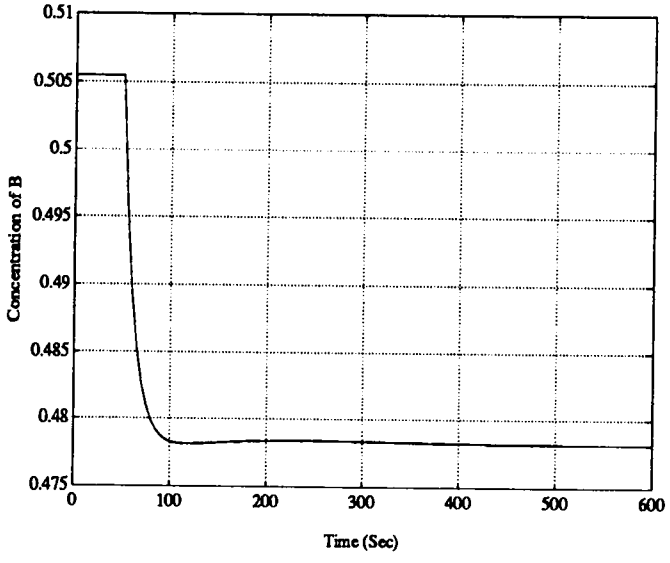

Figure 4.71: Change in k_{-1} by +20%. Unachievable set point. Output versus time

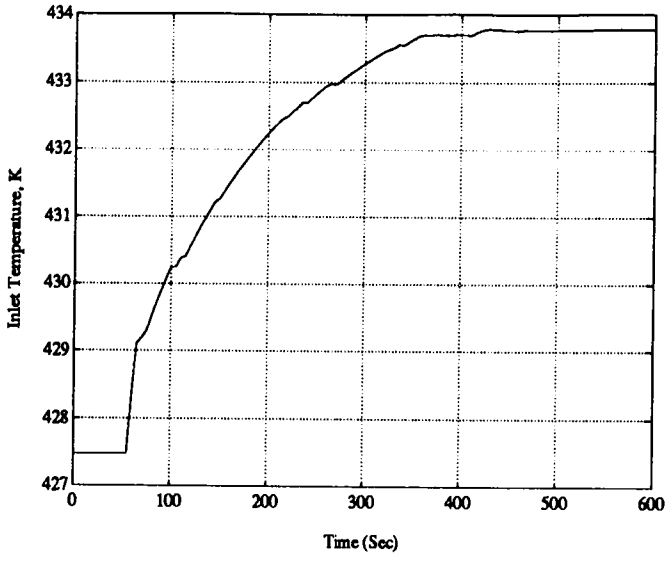

Figure 4.72: Change in k_{-1} by +20%. Unachievable set point. Input versus time

and 4.75. In this simulation, the maximum point was shifted to a left lower value. The controller still maintained stability, but again the final set point was lower than the value achieved with the ELCC (Figure 4.55).

4.2 Catalytic CSTR

The last process used to test the performance of the ELCC was a catalytic reactor, shown in Fig 4.76. This problem was used by Li and Biegler (1988) [46] to test their proposed Pseudo-Newton Controller. It was also used by Brengel and Seider (1989) [6] to test their proposed Multistep Nonlinear Predictive Controller, and was also used by Gattu and Zafiriou (1991) [25] to test their Nonlinear Quadratic Dynamic Matrix Control

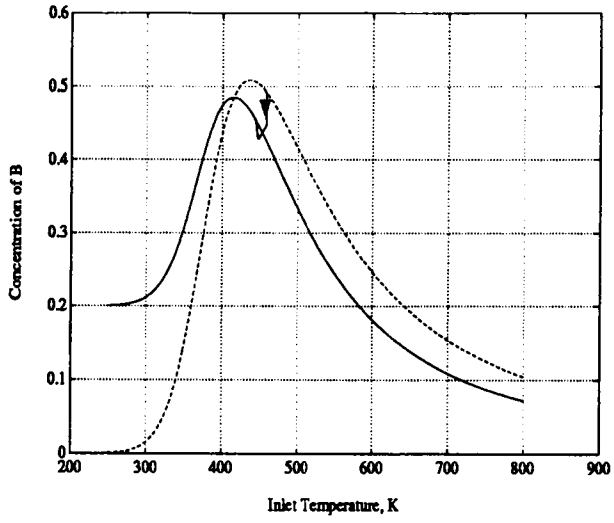

Figure 4.73: Change in k_{-1} by +50% and A_i by -20%. Unachievable set point. System trajectory

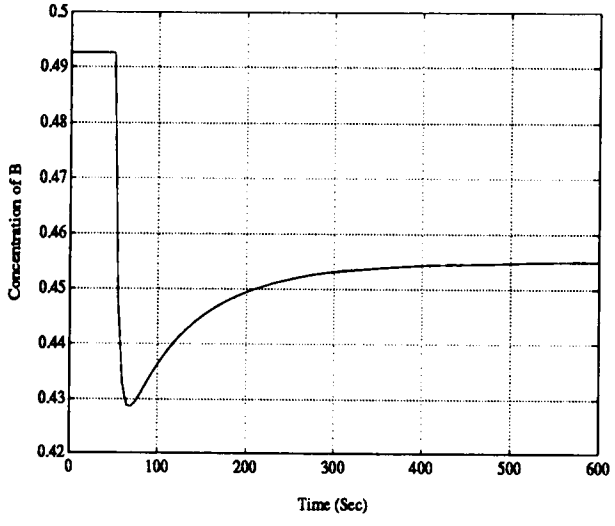

Figure 4.74: Change in k_{-1} by +50% and A_i by -20%. Unachievable set point. Output versus time

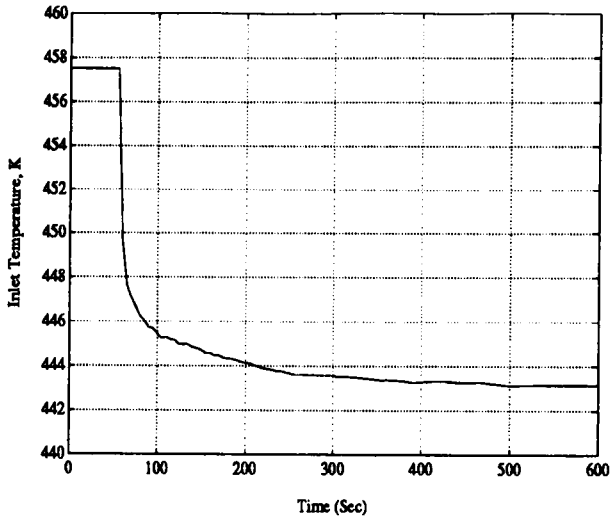

Figure 4.75: Change in k_{-1} by +50% and A_i by -20%. Unachievable set point. Input versus time

algorithm. The process consists of a CSTR, where a catalytic reaction

$$A + B \underset{k_2}{\overset{k_1}{\rightleftharpoons}} P$$

is carried out, where A is in excess.

The system uses two feed streams with different concentrations of B. It is assumed that the reactor operates isothermally. The dynamic model has the form:

- *State Equations:*

$$\frac{dx_1}{dt} = u_1 + u_2 - 0.2x_1^{0.5} \tag{4.46}$$

$$\frac{dx_2}{dt} = (C_{B1} - x_2)\frac{u_1}{x_1} + (C_{B2} - x_2)\frac{u_2}{x_1} - \frac{k_1 x_2}{(1 + k_2 x_2)^2} \tag{4.47}$$

- *Output Equations:*

$$y_1 = x_1 \tag{4.48}$$

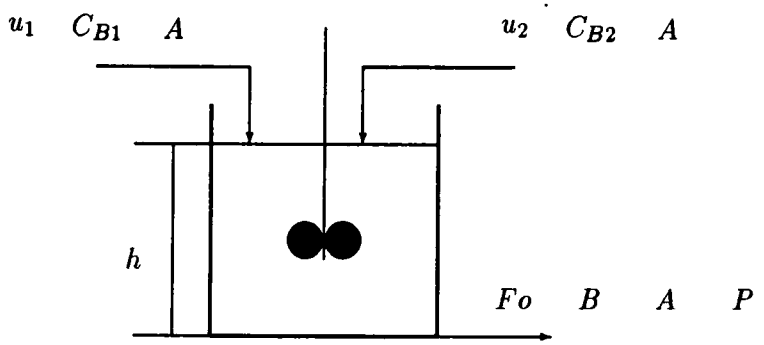

Figure 4.76: Catalytic CSTR

$$y_2 = x_2 \tag{4.49}$$

where:

- x_1 is the liquid height in the tank;
- x_2 is the concentration of B in the reactor;
- u_1 is the inlet flow rate of the concentrated B stream;
- u_2 is the inlet flow rate of the dilute B stream;
- C_{B1} is the concentration of B in the concentrate feed;
- C_{B2} is the concentration of B in the dilute feed; and
- k_1 and k_2 are the rate constants of the forward and reverse reaction respectively.

This is a 2-input-2-output control problem. The parameters used in the simulations are shown in Table 4.6. These parameter values are the same as those used by Li and Biegler [46]. Li and Biegler showed that for some values of u_1 and u_2, there exists

Table 4.6: Parameters of the catalytic reactor

Parameter	Value
C_{B1}	24.1
C_{B2}	0.01
k_1	1.0
k_2	1.0
x_1	2.787
x_2	100

more than one steady state value of y_2. Previous tests of different nonlinear control algorithms [46], [6], and [25] have concentrated in demonstrating the ability of those algorithms to control the system at a liquid level y_1 of 100 units and at steady-state values of u_1 and u_2 of 1 unit. At these conditions, there are three possible steady-state values of y_2; two of them are stable and one is marginally unstable [6]. The performance of the algorithms was tested by demonstrating the controllability of the reactor in reaching the unstable steady state.

The ELCC was tested using the same set point used by other researchers in order to compare our results with theirs. Additional tests were performed to demonstrate the ability of the ELCC to maintain stability in the presence of parameter variations, which are not being demonstrated by other researchers.

4.2.1 Controller Design for a Catalytic Reactor

The control problem can be stated as

$$\min_{u_k,\cdots,u_{k+N}} \sum_{i=1}^{P}[(r-y_{k+i})^T(r-y_{k+i})+(u_{k+i-1}-u_{k+i-2})^T(u_{k+i-1}-u_{k+i-2}) \quad (4.50)$$

subject to

$$\dot{x} = f(x,u) \quad (4.51)$$

$$u_{min} \leq u_k \leq u_{max}, \quad (4.52)$$

where Equation 4.52 represents the inequality constraints on the process input, r represents the desired value of y, N represents the number of future input moves, and P represents the number of sample intervals for which the output of the system will be predicted.

In order to pseudo-linearize the system, the first derivative of the output with respect to time is calculated as

$$\frac{dy_1}{dt} = u_1 + u_2 - 0.2x_1^{0.5} \quad (4.53)$$

$$\frac{dy_2}{dt} = (C_{B1}-x_2)\frac{u_1}{x_1} + (C_{B2}-x_2)\frac{u_2}{x_1} - \frac{k_1 x_2}{(1+k_2x_2)^2}. \quad (4.54)$$

Since the first derivative of the output does depend explicitly on the input u, the system has a relative degree of one. The linearizing feedback can be written as

$$\mathbf{u} = (\mathbf{L_g h(x)})^{-1}(v - \mathbf{L_f h(x)} - \mathbf{h(x)}) \quad (4.55)$$

where $\mathbf{L_f h(x)}$ is

$$\mathbf{L_f h(x)} = \begin{bmatrix} -0.2x_1^{0.5} \\ -\frac{k_1 x_2}{(1+k_2x_2)^2} \end{bmatrix}$$

$\mathbf{L_g h(x)}$ is written as

$$\mathbf{L_g h(x)} = \begin{bmatrix} 1 & 1 \\ \frac{C_{B1}-x_2}{x_1} & \frac{C_{B2}-x_2}{x_1} \end{bmatrix}$$

and $\mathbf{h(x)}$ ir given by

$$\mathbf{h(x)} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}.$$

The inverse of the matrix $\mathbf{L_g h(x)}$ is given by

$$(\mathbf{L_g h(x)})^{-1} = \frac{\begin{bmatrix} C_{B2}-x_2 & -x_1 \\ C_{B1}-x_2 & x_1 \end{bmatrix}}{C_{B2}-C_{B1}} \tag{4.56}$$

Equation 4.56 shows that the only singular point in the system is at $C_{B2} = C_{B1}$. Since the two inlet flows always have different concentrations, this condition is not part of the operation, and the parameter 'ϵ' is not needed in Equation 4.55.

The pseudo-linearized system has the form

$$\frac{dy_1}{dt} + y_1 = v_1 \tag{4.57}$$

$$\frac{dy_2}{dt} + y_2 = v_2. \tag{4.58}$$

This pseudo-linearized system was decoupled, simplifying the prediction part of the controller, a commoun feature of using the exact linearization approach. The DMC algorithm was applied to the linearized system, and using procedures described in Section 3.3, the values of v were used to calculate u. The values of the model parameters are shown in Table 4.7. The sampling time used in the simulations was 1 minute.

4.2.2 Simulation Results for the Catalytic CSTR

The performance of the ELCC was initially tested using the marginally unstable set point of $y_1 = 100$ and $y_2 = 2.7927$, and the initial conditions as specified by Li and Biegler

Table 4.7: Control Parameters for the Catalytic Reactor

Parameter	Value
λ	0.7
P	4
M	4
N	2
u_{max1}	100
u_{min1}	0.05
u_{max2}	40
u_{min}	0.05

(1988) [46] ($y_1 = 40$ and $y_2 = 0.1$). Results of this simulation are shown in Figures 4.77 and 4.78. These figures show that the system achieved the desired set point after 5 minutes. These results are comparable to those presented by Li and Biegler (1988) [46], Brengel and Seider (1989) [6], and Gattu and Zafiriou (1991) [25]. The ELCC maintained the input within the specified limits.

A second test was performed by reducing the maximum value of u_1 and u_2 to 20 and implementing the same change in the desired output. Figures 4.79 and 4.80 show that the ELCC was able to drive the system to the desired set point, at the same time keeping the inputs under desired constraints.

Additional simulations involving changes in the desired set point were performed, using different values of P and M. The results are shown in Appendix D. Again, these results are comparable to the ones obtained in previous studies, in that the ELCC took

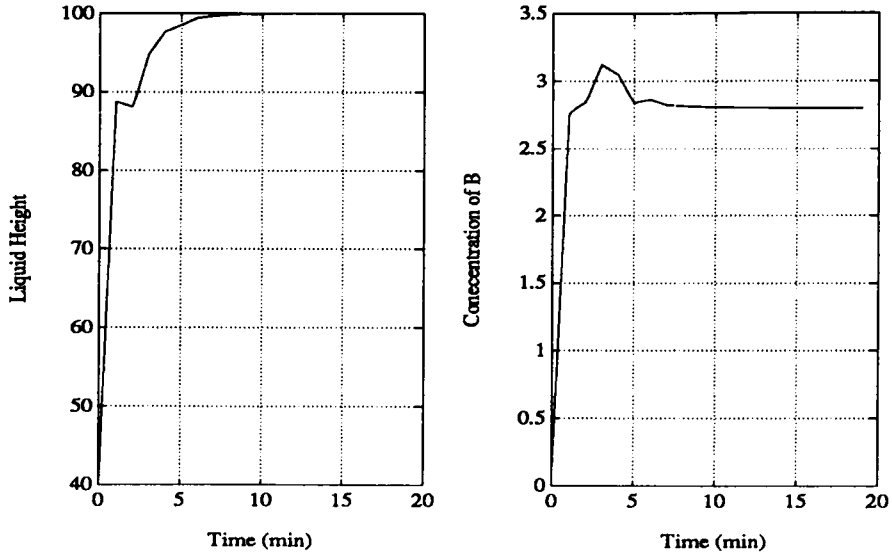

Figure 4.77: Marginally unstable set point. Output versus time

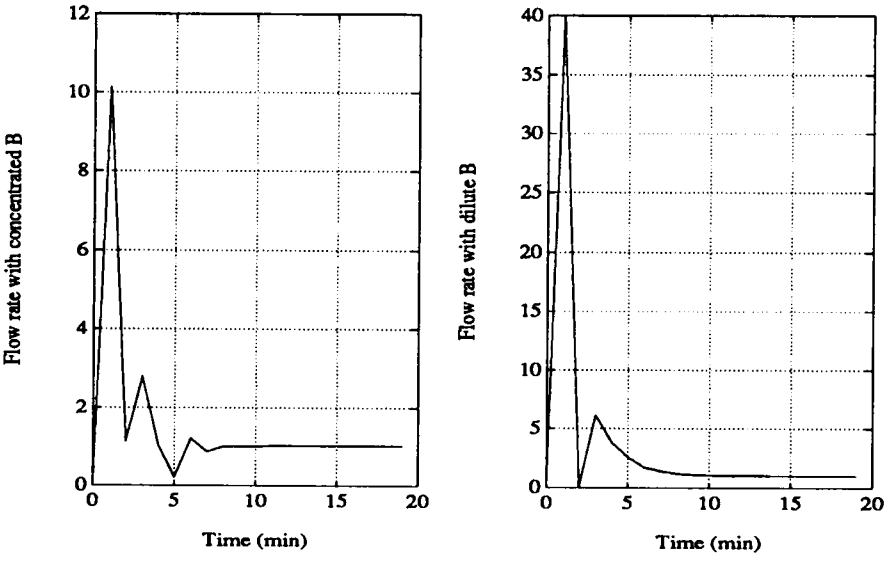

Figure 4.78: Marginally unstable set point. Input versus time

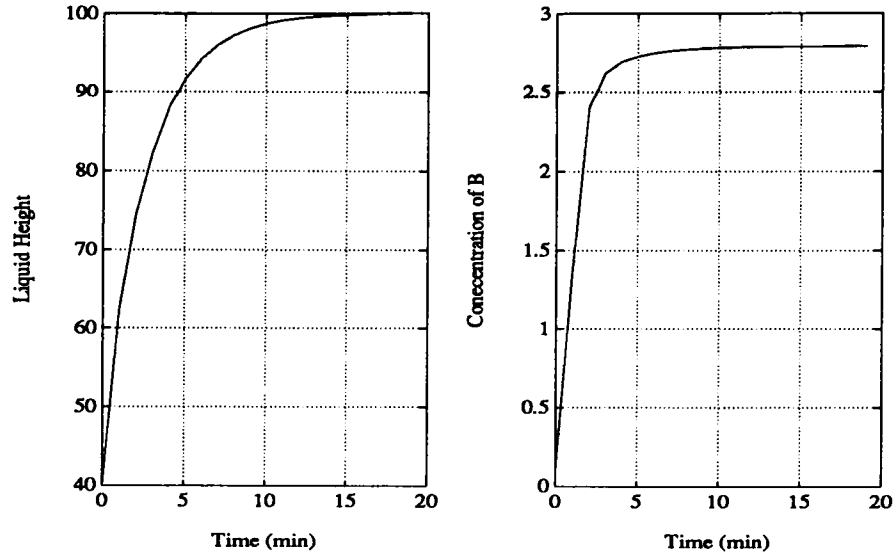

Figure 4.79: Marginally unstable set point. Output versus time

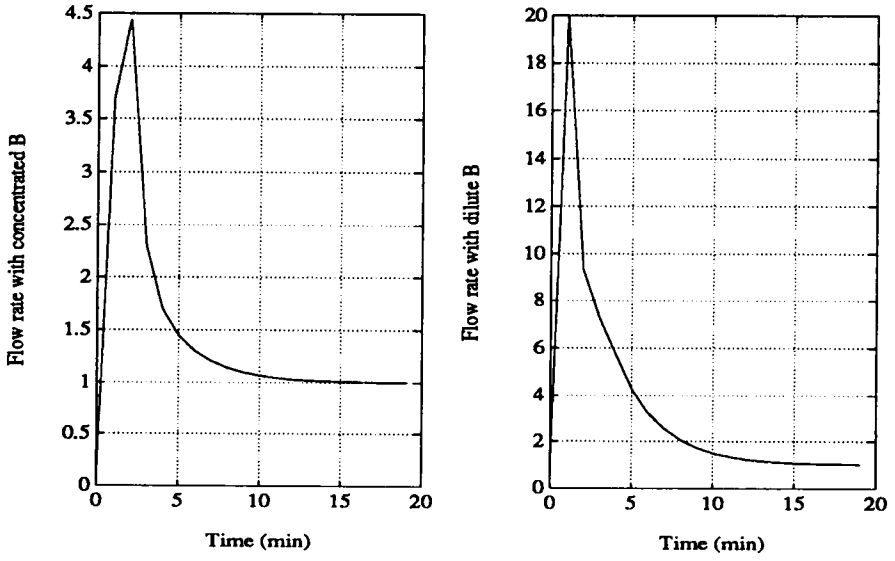

Figure 4.80: Marginally unstable set point. Input versus time

less than 10 minutes to drive the system to the desired set point and maintained the input u within the desired constraints.

In order to the demonstrate the ELCC's ability to maintain stability in the presence of parameter variations, a change in k_1 of +20% was implemented. Figures 4.81 and 4.82 show that the controller was able to maintain stability, and adjusted the value of the inputs to maintain the desired set point. A change in the value of k_1 only affected the concentration of B in the reactor. Since the linearized model was decoupled, the controller was able to correct for this problem without affecting the liquid height.

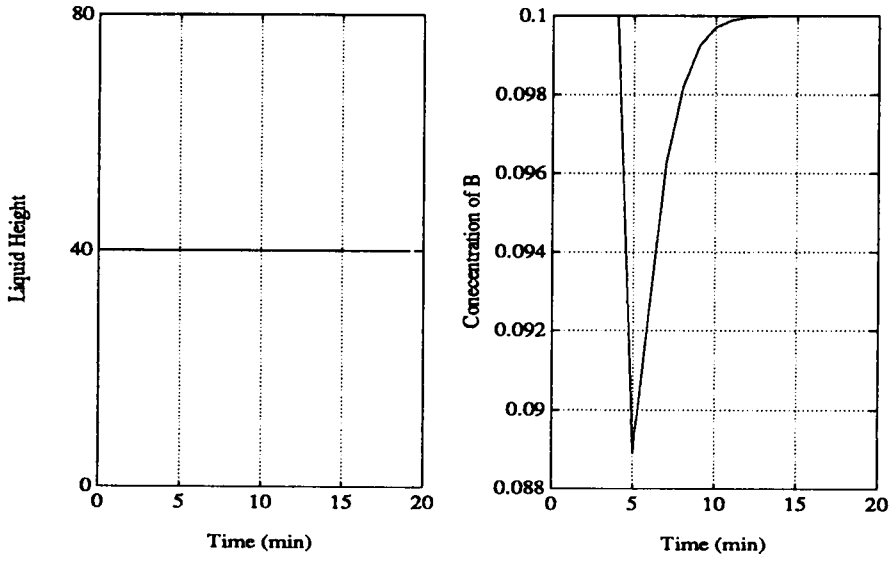

Figure 4.81: System response for a change in k_1 of +20%. Output versus time

Finally, a disturbance in the inlet concentration of B of the concentrated feed was implemented to test the disturbance rejection capability of the ELCC. Figures 4.83 and 4.84 show the system response for a change in C_{B1} of −20%. The controller again was able to maintain the desired set point. Additional simulations are shown in Appendix D.

In general, They show that the performance of the ELCC is not affected by the presence of parameter variations. These simulations were not done in similar studies by previous researchers.

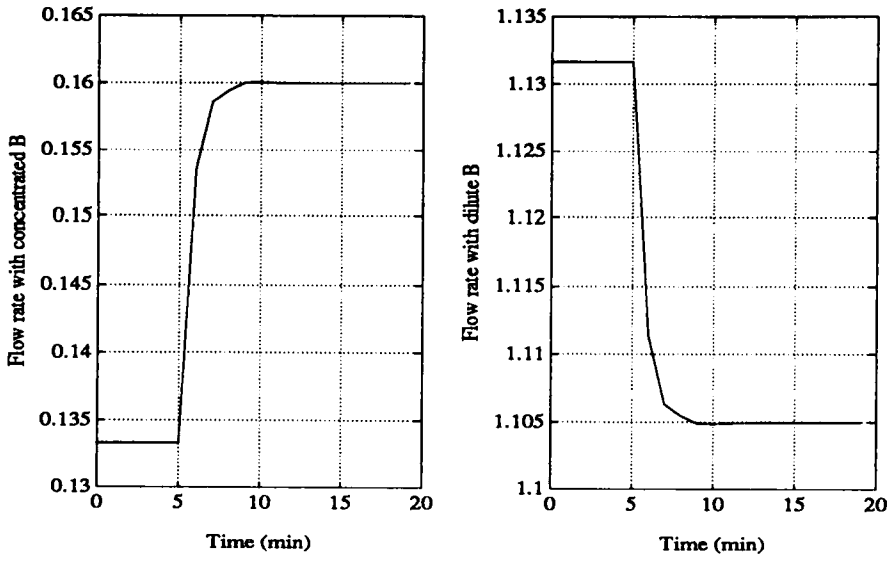

Figure 4.82: System response for a change in k_1 of +20%. Input versus time

4.2.3 Nonlinear Controller with Global Optimization for a Catalytic Reactor: The Base Case

The same simulations as described in Section 4.2.2 were performed with a controller that uses a global optimization algorithm in order to assess the suboptimality of the ELCC. The function to be minimized is given by

$$\min_{u(t)} J(t) = \int_{to}^{\tau} \alpha^t [(r - y(t))^T (r - y(t))] dt - \frac{\alpha^\tau [(r - y(\tau))^T (r - y(\tau))]}{ln(\alpha)} \qquad (4.59)$$

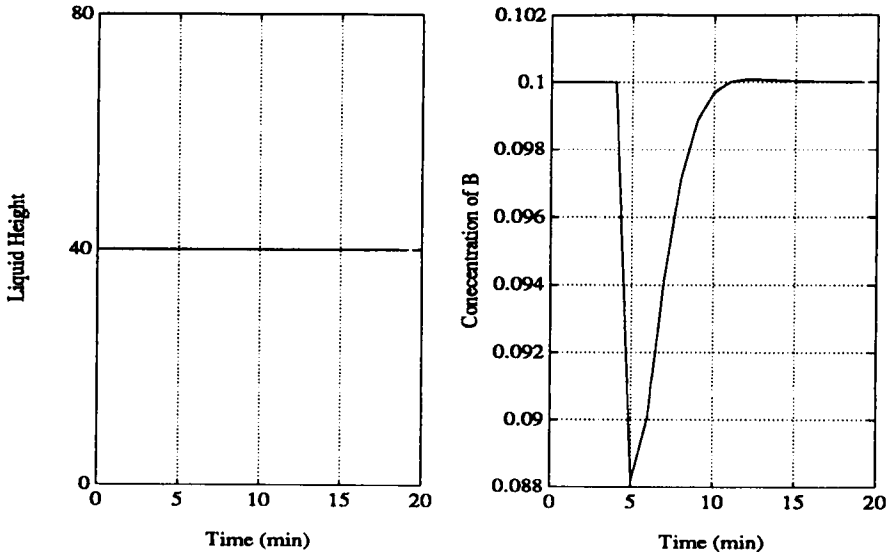

Figure 4.83: System response for a change in C_{B1} of -20%. Output versus time

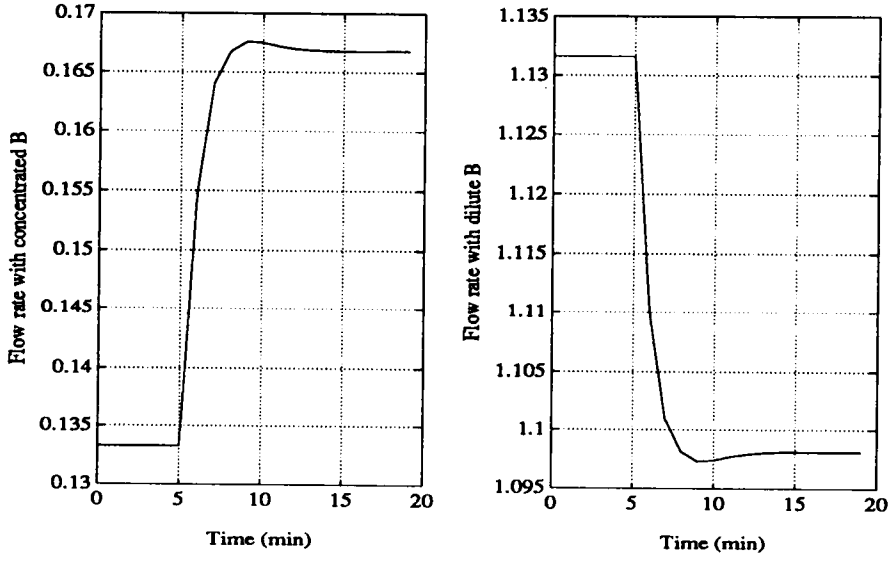

Figure 4.84: System response for a change in C_{B1} of -20%. Input versus time

subject to

$$\dot{x} = f(x, u) \tag{4.60}$$

$$0 = f(x(\tau), u(\tau)) \tag{4.61}$$

$$u_{min} \leq u(t) \leq u_{max}, \tag{4.62}$$

where τ denotes the time required to reach steady state. A value of $\alpha = 0.85$ was used. The Hamiltonian function for the system is given by

$$H(x, \zeta, u, t) = \zeta^T f(x, u) + \alpha^t[(r - y(t))^T (r - y(t))]. \tag{4.63}$$

where ζ is a costate vector. The optimization algorithm as used with the CSTBR was used with the catalytic reactor to estimate the inlet flow rates of concentrated B, u_1 and of diluted B, u_2 that minimize 4.59 (see Appendix C).

The performance of the global-optimum controller was tested initially using the marginally unstable set points $y_1 = 100$ and $y_2 = 2.7927$, and the initial conditions as specified by Li and Biegler (1988) [46] ($y_1 = 40$ and $y_2 = 0.1$). The results of this simulation are shown in Figures 4.85 and 4.86. These figures show that the system achieved the desired set point after 10 minutes. These results are comparable to the results obtained using the pointwise optimization. The value of α changed the speed of convergence to the desired conditions. A value of 0.99 was used in the simulations. For a value of 0.9, the system took more than 15 minutes to achieve the final condition.

A second test was performed by reducing the maximum value of u_1 and u_2 to 20, followed by the same change in the desired output. Figures 4.87 and 4.88 show that the controller was able to drive the system to the desired set point, at the same time maintaining the input under desired constraints. The response was not affected by this change in limits. The global-optimum controller did not call for changes in u_2 larger than

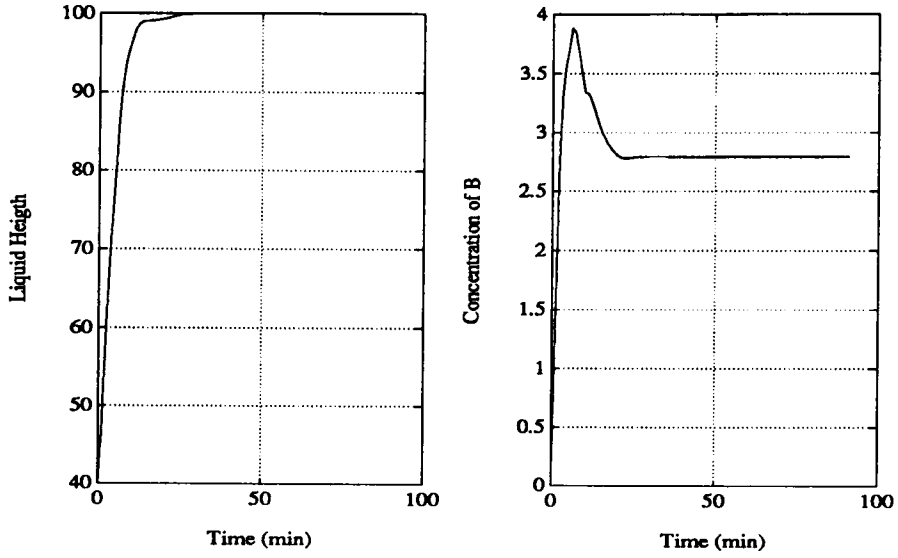

Figure 4.85: Marginally unstable set point. Output versus time

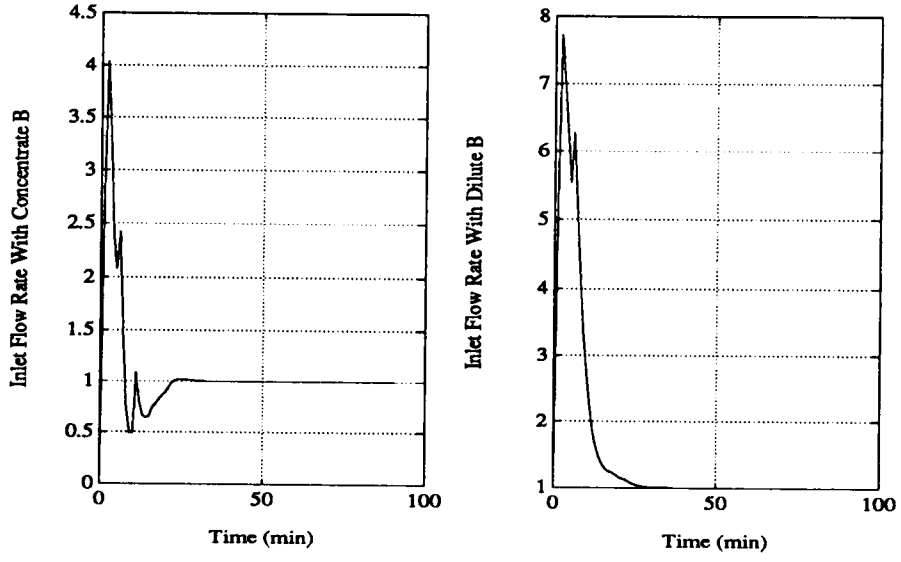

Figure 4.86: Marginally unstable set point. Input versus time

8, whereas the ELCC called for changes in u_2 up to 20 (see Figure 4.86 and Figure 4.78). Since the global-optimum controller used smaller changes in the input, it took more time to reach the desired condition. This suggests that the ELCC did not perform that suboptimally when compared to the global-optimum controller; moreover, the ELCC's computer time requirements were significantly less. This point will be discussed in more detail in Chapter 5.

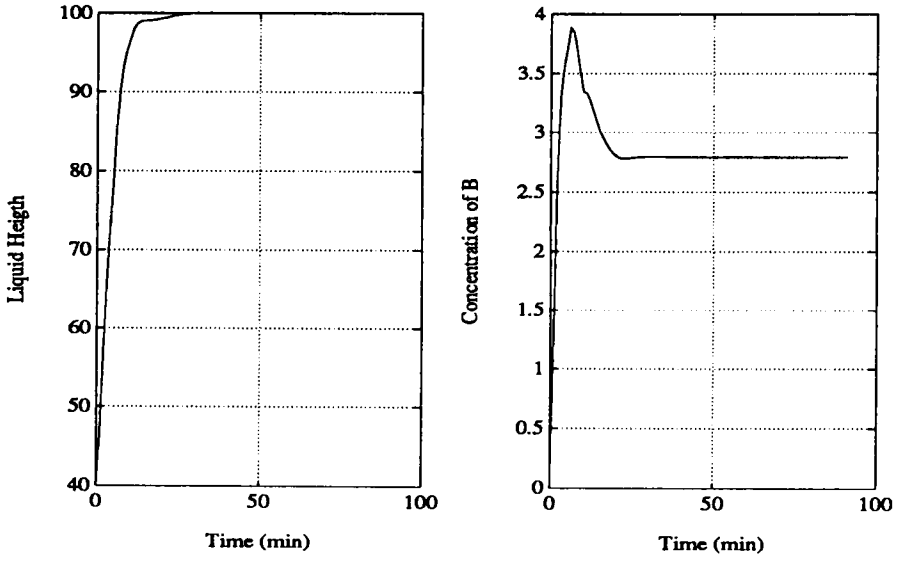

Figure 4.87: Marginally unstable set point. Output versus time

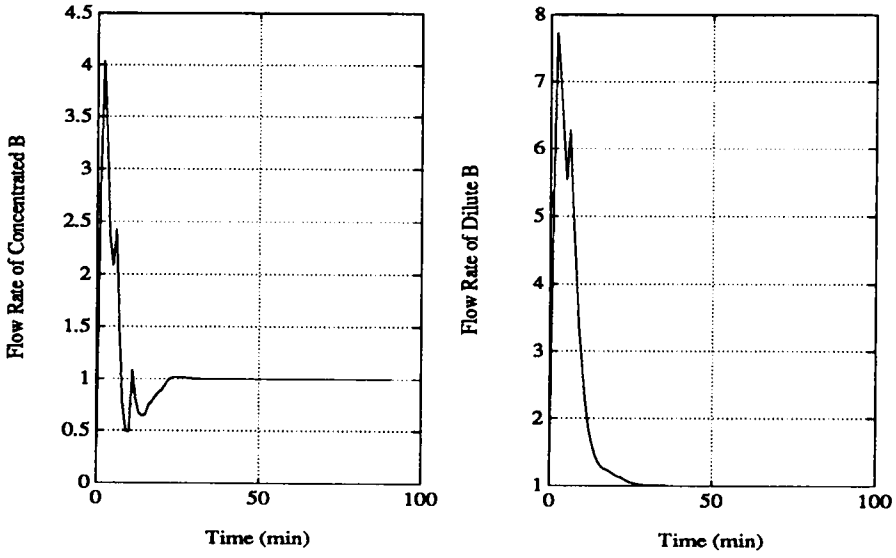

Figure 4.88: Marginally unstable set point. Input versus time

CHAPTER 5

DISCUSSION OF RESULTS

The computer simulation results of the CSTBR, exothermic CSTR and catalytic reactor presented in Chapter 4 show that the ELCC has the following practical advantages:

- The ELCC is able to maintain stability in the presence of parameter variations that render the desired set point unachievable.

- When the desired conditions are achievable, the ELCC is able to drive the system to the desired point of operation regardless of the initial conditions of the system.

- For the simulations presented in this work, the performance of the ELCC is comparable or superior to that of a nonlinear controller that uses a global optimization algorithm, while requiring at least one order of magnitude less computational time than the global optimal controller.

- The ELCC allows the use of different types of linear controllers in its prediction step, does not require an estimator to counteract the effect of parameter variations, and explicitly handles constraints.

Each of the above advantages is discussed in this chapter. Section 5.1 presents a discussion of the first two advantages. Section 5.2 follows with a comparison of the performance of the nonlinear global-optimum controller and the ELCC. Finally, Section 5.3 presents the results obtained when a Proportional Integral linear controller is used in the prediction step of the ELCC algorithm.

5.1 Stability of a System that Uses the ELCC algorithm

The ELCC algorithm uses the input-output linearization technique to generate a pseudo-linear model of the system that is independent of the operating point. The input-output linearization technique uses the Lie derivatives of the output to develop an explicit expression relating the output of the system to the input. The Lie derivatives required in the linearizing feedback use the current state values in the relationship between the input and output of the system, making the linearization valid for the entire region of operation. Therefore, the controller can operate the system starting at either of the two sides of the input-output steady-state curve, when it exhibits a maximum.

5.1.1 Role of the Tunable parameter 'ϵ'

Consider a nonlinear system of relative degree of one of the form

$$\begin{aligned} \dot{x} &= f(x) + g(x)u \\ y &= h(x) \end{aligned} \tag{5.1}$$

where x is an n-dimensional state vector, u is an m-dimensional input vector, and y is an m-dimensional output vector. The linearized model has the general form

$$\frac{dy}{dt} + y = L_f h(x) + L_g h(x)u + h(x) = v \tag{5.2}$$

and the linearizing feedback is given by

$$u(x, v) = \frac{(v - L_f h(x) - h(x))(L_g h(x))}{\epsilon + (L_g h(x))^2}. \tag{5.3}$$

When the desired condition is not achievable, the ELCC controller will call for larger values of v to try to achieve the set point. As the value of v increases, the value of y

increases linearly (Equation 5.2). When the value of y approaches the maximum value, its rate of change with respect to u decreases and is exactly zero at the maximum point.

In the input-output linearization technique, $L_g h(x)$ is defined to bear a simple linear relation with the rate of change of the output y with respect to the input u. Therefore, the magnitude of $L_g h(x)$ decreases as the system approaches its optimal value. If 'ϵ' is suppressed in Equation 5.3, the value of $L_g h(x)$ will become vanishingly small as the system nears the maximum point. If the value of $L_g h(x)$ is very small, the controller will call for very large values of u, producing system instability, or "chattering" (see Equation 5.3). However, if the value of 'ϵ' in Equation 5.3 is nonzero, v will increase until the magnitude of $L_g h(x)^2$ is comparable to that of 'ϵ'. At this point, due to the presence of 'ϵ', the value of u decreases regardless of the value of v. Once the input value has been reduced, the controller will start increasing the value of u again until it reaches steady state. Figures 5.1 and 5.2 show the system response for a value of ϵ equal to 0.5. These figures show that the controller increases the value of u until the system reaches a region where the magnitude of $L_g h(x)$ becomes comparable to the magnitude of 'ϵ' (point a in the figures). Then, the value of u decreases sharply to a value lower than that corresponding to the maximum point. The input is then increased until the system reaches steady state.

The value of 'ϵ' determines how much the magnitude of u is increased before its value finally reaches a steady-state point. Therefore, 'ϵ' ultimately controls how close the final output value is to the maximum point. Figure 5.3 shows the system response for the values of 'ϵ' at 1 and 4. It can be seen in these figures that as the value of ϵ increases, the maximum value of u decreases (0.598 vs 0.498 respectively), thereby placing the final steady state farther from the optimal point (1.734 vs 1.71 respectively). Thus the value

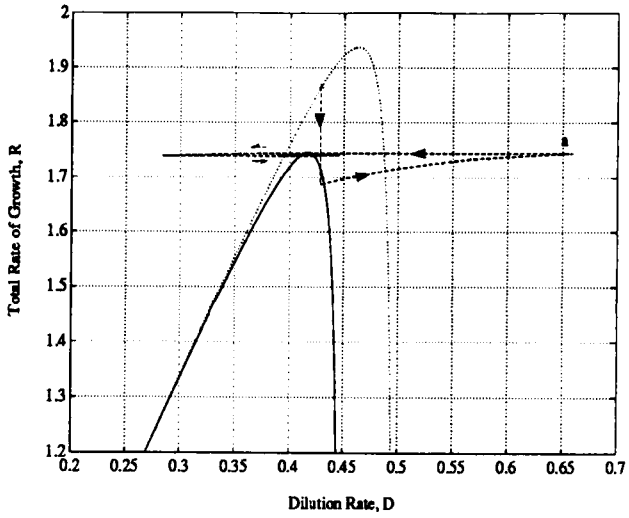

Figure 5.1: Change in μ_max by -10%. Unachievable set point. System trajectory for $\epsilon = 0.5$

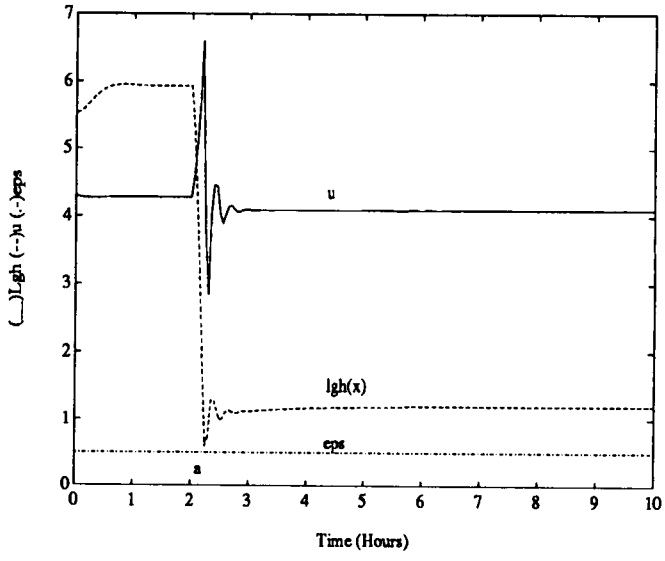

Figure 5.2: Change in μ_{max} by -10%. Qualitative representation of $L_g h(x)$, $u(t)$, and ϵ.

of 'ϵ' should be chosen to be as small as possible, taking into account the knowledge of the parameters of the system. A small value of 'ϵ' will yield a more precise linearization and lead to higher final output steady-state levels; but it will not accommodate large model uncertainties, as they will lead to excessively large values of v, whose effect will not be compensated by small values of 'ϵ'.

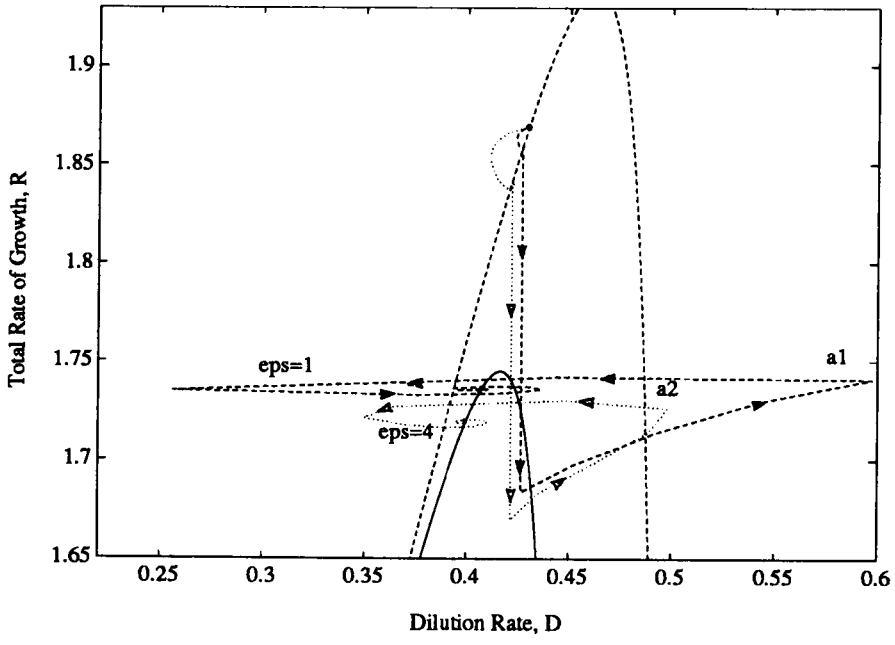

Figure 5.3: Change in $\mu_m ax$ by -10%. Unachievable set point. System trajectory for two different ϵ values. The dash line corresponds to $\epsilon = 1$. The dot line corresponds to $\epsilon = 4.0$

The ELCC described in Chapter 3 is able to accommodate operations on both sides of the maximum point, where the steady state system gains are opposite in sign. Most importantly, it is able to maintain stability in the presence of parameter variations that render the set point unachievable. These qualities of the ELCC allow the selection of

higher set points, thereby increasing system productivity. Even small changes in system productivity may be translated into large increases in profits, due to the high prices of some products manufactured in a typical commercial bioreactor, such as human growth hormone ($1700000/kg), renin ($100000/kg), insulin ($10000/kg), etc. (Sikdar, Bier and Todd (1989) [74]).

5.1.2 Final Steady-State Value for CSTBR and CSTR

Figures 4.2 and 4.43 show that there are two possible steady-state input values for each output value. During the simulations of the CSTBR and CSTR presented in Chapter 4, it was observed that the final steady-state value was generally to the left of the maximum point. The systems tend to settle at smaller input values due to the presence of 'ϵ' in the linearizing feedback. When a step change in the desired output was introduced, the controller changed the input to try to reach the desired condition. The system starts with $L_g h(x) \gg \epsilon$. Then, as the input changes, the value of $L_g h(x)$ alters (Equation 5.2). With the stipulation of a higher set point, the value of $L_g h(x)$ decreases as the system approaches the optimal point (as explained in Section 5.1.1). If the system enters the region where the magnitude of $L_g h(x)$ is comparable to the magnitude of 'ϵ', the value of u in Equation 5.3 decreases, shifting the system to the left side of the optimal point. Therefore, the system can be forced to operate with input values bigger than D_{opt} by either limiting the magnitude of the changes in the set point or by reducing the value of ϵ, as shown in Figures 5.4 and 5.5, respectively.

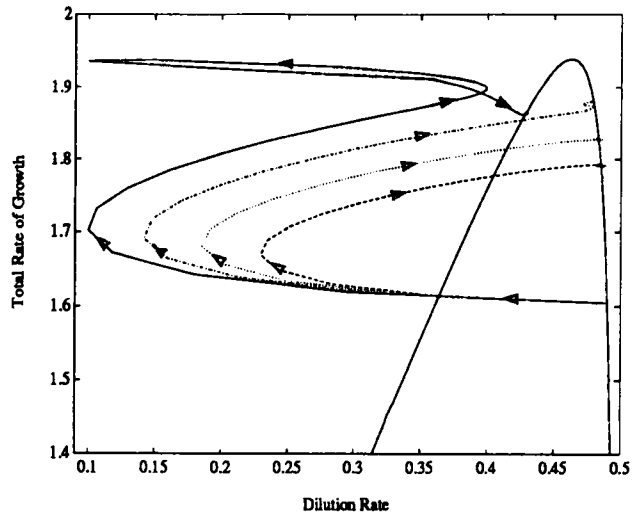

Figure 5.4: System trajectory for different set points

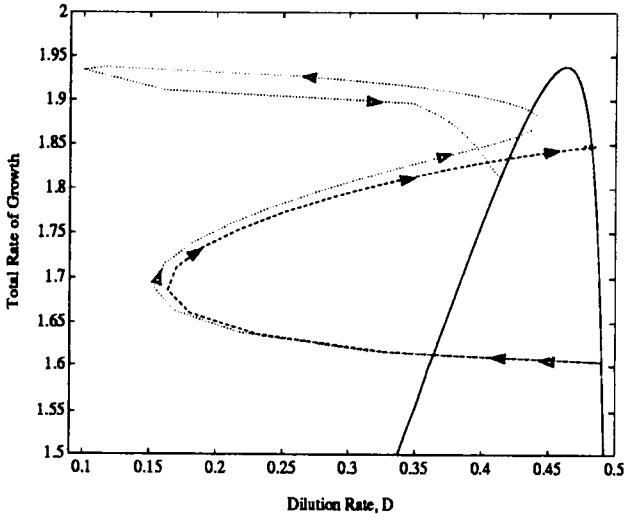

Figure 5.5: System trajectory for two different ϵ values. The dash line corresponds to $\epsilon = 0.5$. The dot line corresponds to $\epsilon = 1.0$

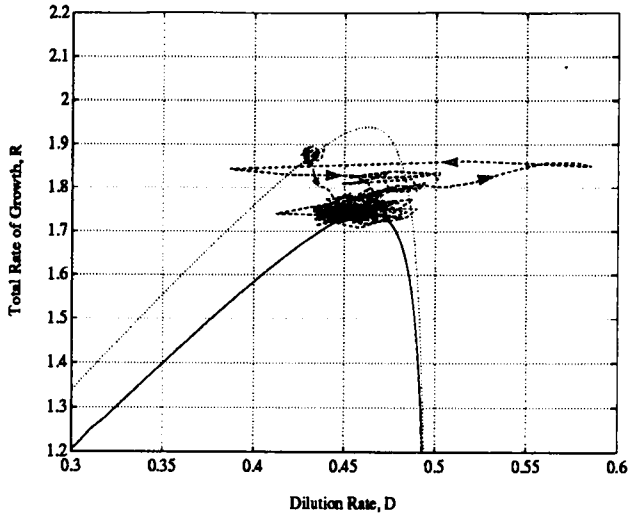

Figure 5.6: Change in y_s by -10% in the presence of random noise. Unachievable set point. System trajectory

5.1.3 Effect of Noise in the System

Several simulations that include gaussian noise in the output measurements were carried out to show the ability of the ELCC controller to maintain stability under these conditions. Figures 5.6, 5.7, and 5.8 show the system response for unachievable set point. As shown, the controller is able to maintain stability and to drive the system close to the maximum point. The presence of white noise in the output of the system does not affect the controller ability of maintaining stability when the set point is unachievable.

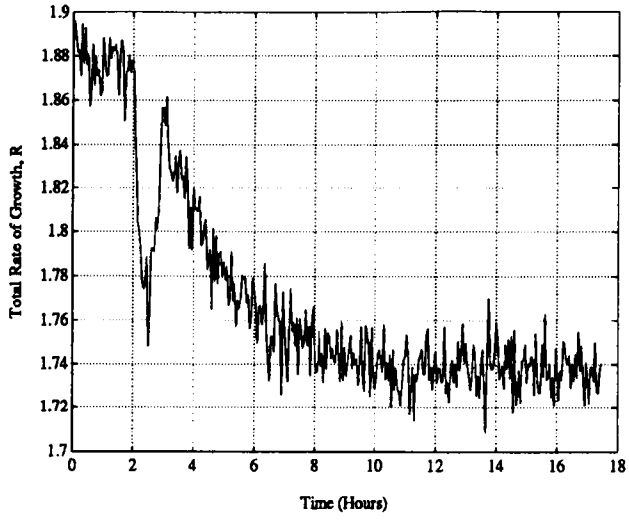

Figure 5.7: Change in y_s by -10% in the presence of random noise. Unachievable set point. Output versus time

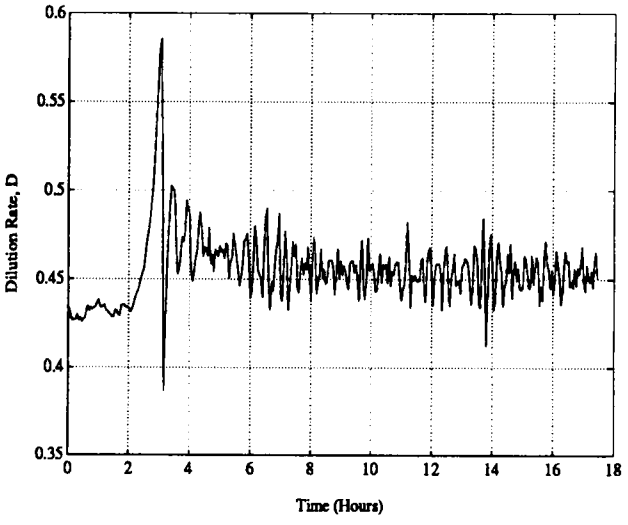

Figure 5.8: Change in y_s by -10% in the presence of random noise. Unachievable set point. Input versus time

5.2 Comparison of the ELCC with a Nonlinear Controller that Uses a Global Optimization Technique

The simulation results presented in Chapter 4 (sections 4.1, 4.2 and 4.3) show that the nominal performance of the ELCC algorithm and the nominal performance of the nonlinear optimal controller were very similar. The response of the system controlled by the ELCC generally had a larger overshoot than that of the global linearization, but its settling time was generally smaller. When the desired conditions were unachievable, both controllers were able to maintain stability. However, the ELCC drove the system close to the maximum achievable point, whereas the nonlinear optimal controller tended to settle at a level close to the initial value. This result makes the ELCC a better choice for higher productivity, in the presense of model uncertainty.

The most practical advantage of the ELCC over a nonlinear optimal controller is its low computational requirements. In order to compute a system trajectory, the ELCC requires only the multiplication of the pseudo-inverse of the dynamic matrix A, with the predicted error (see Chapter 2). This trajectory is then projected onto the space of allowable inputs using a pointwise optimization. On the other hand, the global optimization technique requires the iterative integration of the state and costate equations of the system until the cost function is minimized. On a Sun Workstation, a comparison of the work required for each algorithm is presented in Table 5.1 and Table 5.2. A typical computer time requirement for implementing the optimal nonlinear controller is about 5 hours, while the ELCC only requires 3 minutes.

The performance results shown in Chapter 4 for the MIMO system for the ELCC and for the nonlinear optimal controller are comparable. Both controllers yield small

Table 5.1: Computational Work of the ELCC algorithm

ELCC
Up front work Required (One time)
• Linearization
• Calculation of A^+
Work at each sampling time
• Measure output, y
• Calculate Error from A, set point, and y
• $v = A^+ Error$, $v \rightarrow u$
• Check for constraint violations
• Projection of desired trajectory to the space of allowable inputs

overshoot and comparable settling time. However, the computational requirements for the ELCC are much smaller. After the input-output linearization step in the ELCC algorithm, the system becomes decoupled, simplifying the prediction step of the algorithm even more.

5.3 ELCC With Proportional Integral Controller

In this dissertation, the DMC algorithm was used as the linear controller for the pseudo-linear system. It was used to predict the values of the pseudo-linear input v that minimizes a given performance criteria. However, any linear controller could have been used in this step of the ELCC algorithm. A Proportional Integral controller was also

Table 5.2: Computational Work for the Global Optimization

GLOBAL OPTIMIZATION
Up front work Required (One time)
• Calculation of the gradient of the cost function
• Derivation of the costate equations
Work at each sampling time
• Measure output, y
• Evaluate the cost function
• Integrate forward the state equations
• Integrate backwards the costate equations
• Evaluate the cost function
• Iterate until convergence

Table 5.3: Control Settings

Proportional Gain	10
Integral Time	1
Sampling Time	0.05

implemented in the prediction step, and was tested through simulation of the Bioreactor described in Section 4.1. The other steps of the control algorithm were implemented as explained in Chapter 3. The control settings for this system are shown in Table 5.3. The results for a step change in the desired total growth rate are shown in Figures 5.9, 5.10, and 5.11. These results are comparable to those obtained when DMC is used in the prediction step (Section 4.1, Figures 4.3, 4.4, and 4.5). The settling time for both controllers is about 1 hour, and both produce a small overshoot.

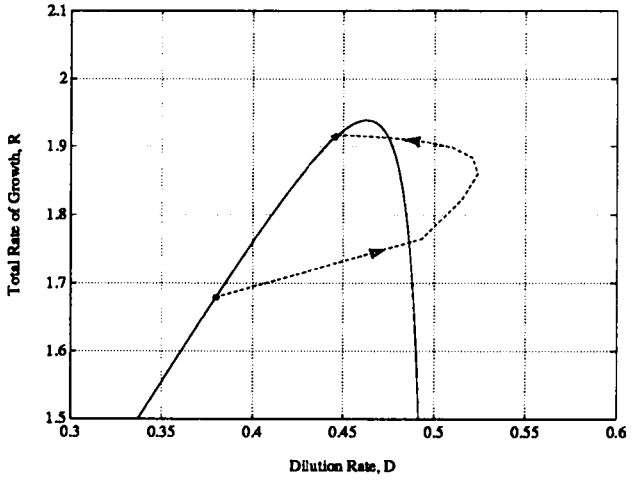

Figure 5.9: Set point change in the total rate of growth by 14%. System trajectory

In order to demonstrate the ability of the controller to maintain stability in the presence of parameter variations, two changes were made in the system. First, the value of y_s was decreased by 10%. Figures 5.12, 5.13 and 5.14 show that the ELCC-PI controller does maintain stability and drives the system to a point near the maximum achievable set point under these conditions. This result agrees with those obtained when ELCC-DMC is used as the linear controller (see Figures 4.12, 4.13 and 4.14)

A second parameter variation was implemented by introducing a change in μ_{max}. The results for a decrease in the value of μ_{max} by 10% are shown in Figures 5.15, 5.16 and 5.17. These figures show that the controller operates the system close to the maximum possible point without causing system instability, as expected. Therefore, in general, any well-tuned linear controller can be used as the linear controller for the pseudo-linear system.

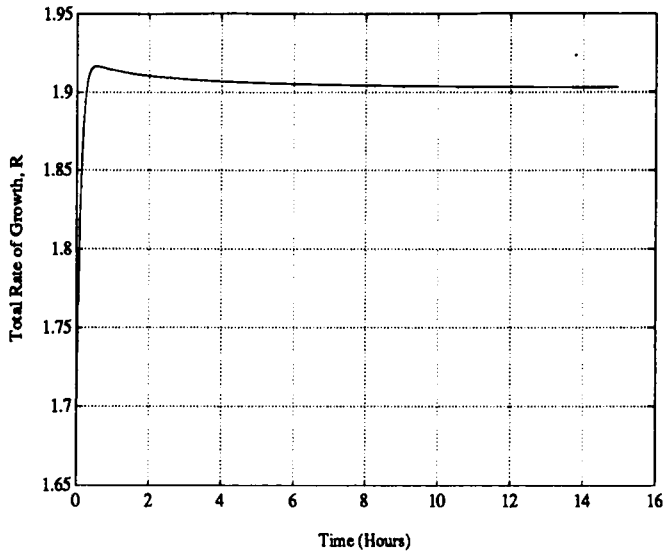

Figure 5.10: Set point change in the total rate of growth by 14%. Output versus time

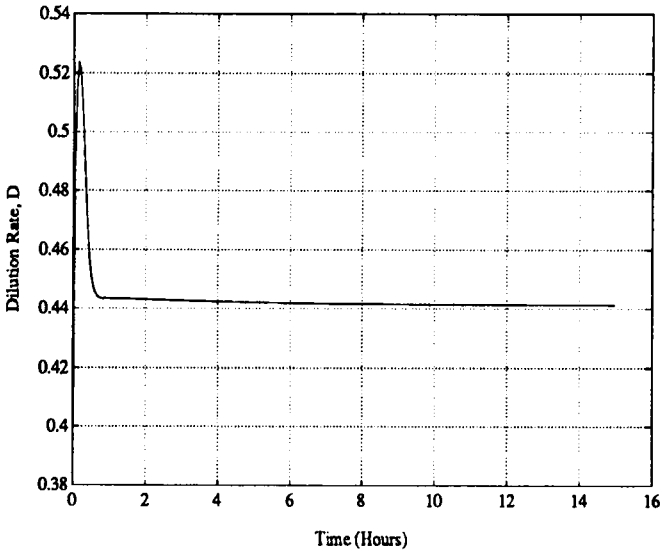

Figure 5.11: Set point change in the total rate of growth by 14%. Input versus time

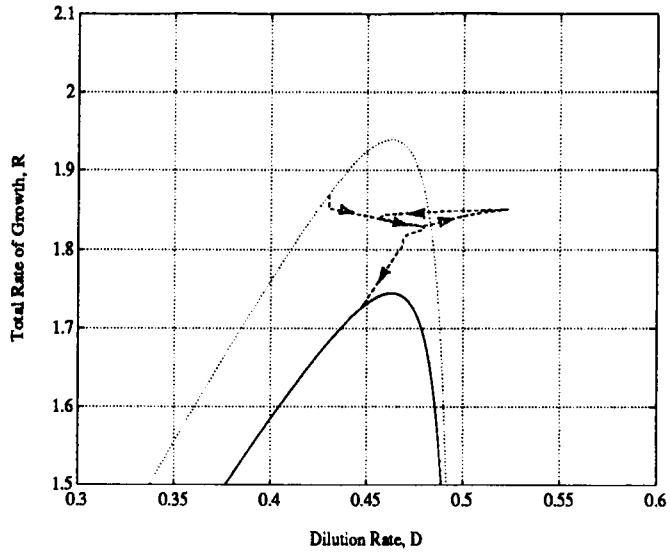

Figure 5.12: Change in y_s by -10%. Unachievable set point. System trajectory

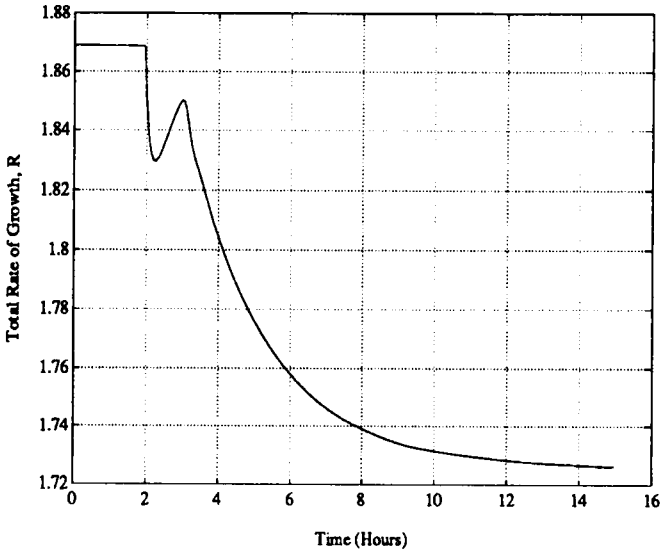

Figure 5.13: Change in y_s by -10%. Unachievable set point. Output versus time

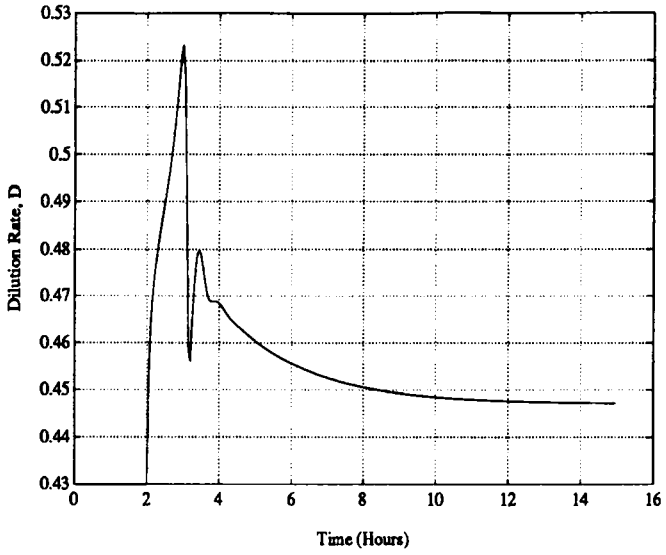

Figure 5.14: Change in y_s by -10%. Unachievable set point. Input versus time

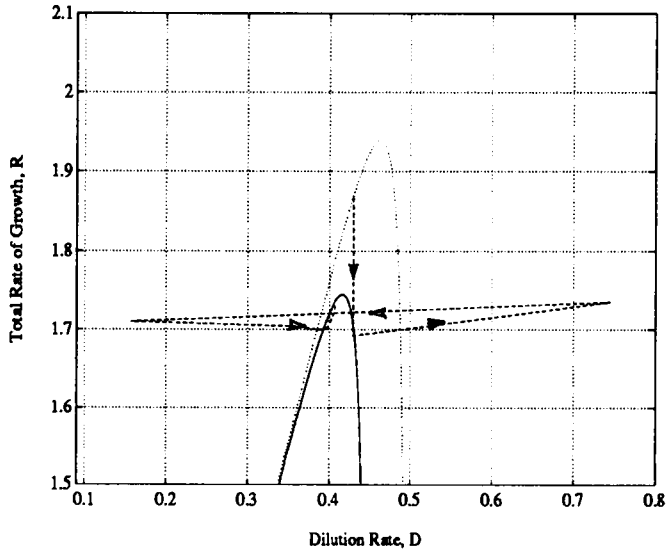

Figure 5.15: Change in $\mu_m ax$ by -10%. Unachievable set point. System trajectory

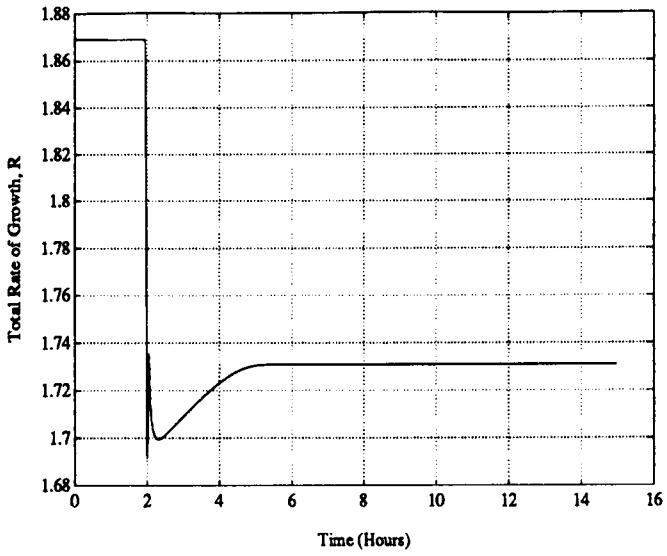

Figure 5.16: Change in μ_{max} by -10%. Unachievable set point. Output versus time

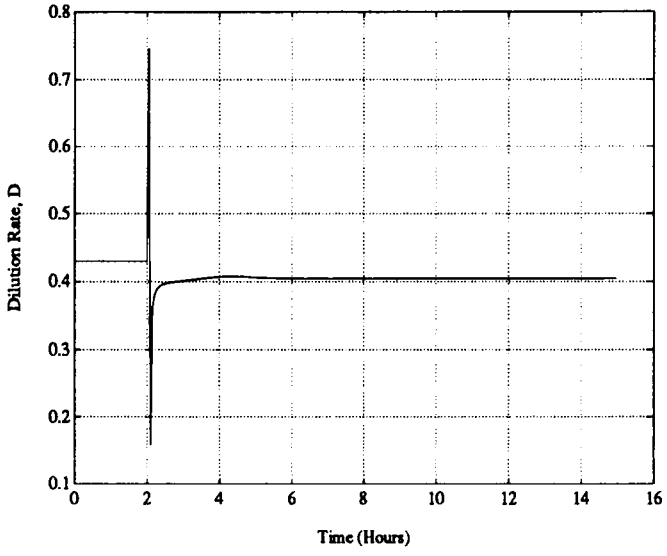

Figure 5.17: Change in μ_{max} by -10%. Unachievable set point. Input versus time

CHAPTER 6

CONCLUSIONS AND RECOMMENDATIONS

This dissertation presented the results of applying the exact input-output linearization technique to nonlinear systems to extend the application of DMC to nonlinear systems with constraints. This technique not only permits the controller to explicitly handle constraints, but also does not require excessive computational time, thus allowing real-time implementation. The on-line computations are greatly reduced, since the system only has to be pseudo-linearized once, initially. The pseudo-inverse of the dynamic matrix of the system can be hard-coded into the algorithm, and the product of its value and the predicted error will yield the desired trajectory at every sampling time.

6.1 Summary of Results

Based on the simulation studies, the Exact-Linearized-Constraint Controller (ELCC) technique has the following features:

1. The performance of ELCC is independent of the point of operation and can be applied to systems that have singular points in their operational manifolds.

2. When the desired conditions are unachievable, the ELCC is able to maintain stability and to drive the system close to the maximum achievable condition. The selection of the tunable parameter 'ϵ' determines how close to the maximum point the system can be driven and how much parameter uncertainty can be accom-

modated. This provides an insightful way to detune the controller in the case of unexpected, large perturbations. For systems that exhibit a maximum point in their steady-state relationship, the value of 'ϵ' also determines at which side of the maximum the final state settles when a change in set point is made.

3. When the trajectory predicted by the ELCC does not violate any system constraints, the performance of the system is optimal. When the constraints do become active, the controller implements the trajectory closest to the desired one in the least squares sense. Though simulation studies will never be exhaustive for the cases presented in this work, the performance of the ELCC, using the projection method, is comparable to the performance of systems that use the global optimization methods. The ELCC, however is much less computationally intensive, and has the added feature that steady-state offset is less than that given by the global optimization method. Global optimization minimizes the total cost J only with respect to a given model as described by Equation 3.42, whereas the cost computed from ELCC may actually be smaller than J.

4. The tuning of the ELCC control system should be carried out in two steps. In the first step, the DMC parameters are selected following the existing guide lines for tuning a DMC controller [23]. Then, through simulations the values of λ and ϵ are adjusted to eliminate chattering and undesired oscillations in both input and output responses.

5. In comparing the results with other nonlinear approaches [17], [46], [6], [25], the ELCC algorithm presented in this paper is more attractive from both the design and implementation aspects. It is computationally efficient, the selection of the

DMC tuning parameter can be made based on the established criteria for linear systems, and it does not need *a priori* knowledge of the system response. The methods described in [17] and [46] require the implementation of an inversion, point by point, of the model of the system, and require knowledge of the output domain of the system. The methods described in [6] and [25] require repeated linearization of the system at intermittent time points. According to Brengel and Seider [6], the computational requirements increase linearly with the number of linearizations performed, making the methods impractical for real time application.

In conclusion, the ELCC approach does not require excessive computation time, handles constraints explicitly, and performs excellently. With respect to performance, ELCC has demonstrated its ability to mantain stability even in the presence of parameter variations and disturbances that render the set point unachievable. Other nonlinear approaches have not demonstrated this particular capability. The ELCC is especially efficient in the control of nonlinear systems that exhibit a maximum or minimum point in the steady-state input-output curve, allowing the system to operate very close to those points.

Though the global optimization technique can be used with more general models, it requires at least an order of magnitude more computational effort than the ELCC method, and its performance degrades in the presence of parameter variations and model uncertainties. Additional research work would be required to develop adaptive control strategies that counteract these negative effects.

The simulation results obtained in this work are very promising and should be tested experimentally. There are still a number of important limitations associated with the input-output linearization that are topics of current research. Wright and Kravaris

(1992) [85] presented a nonminimum-phase compensator for nonlinear processes. Additional research effort is required to extend the application of the geometric linearization techniques to discrete time systems.

6.2 Appropriateness of the ELCC Controller

The ELCC controller is especially applicable to nonlinear systems that exhibit a maximum or minimum point in the steady-state input-output curve. The controller is able to maintain stability in the presence of parameter variations and disturbances. For MIMO nonlinear systems the linearized model used in the ELCC becomes decoupled, thus greatly simplifying the subsequent controller design.

6.2.1 Up Front Work

The up front work required to implement the ELCC is

- Linearization of the model of the system.
- Calculation of the step response coefficients of the linearized model to be used in the DMC unit of the ELCC.

6.2.2 Work at Each Sampling Time

The work required to be done at each sampling time includes

- Measure the output of the system.
- Calculation of the error.
- Prediction of the new input trajectory.

- Checking for constraint violations.

- Projection of the desired trajectory to the space of allowable inputs.

A typical computer time requirement for implementing the ELCC controller over a time horizon of 6 hours is about 3 minutes on a Sun/4 Workstation.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Arkun Y., "Design of Steady-State Optimization Control Structures for Chemical Process," Ph. D. Thesis, University of Minnesota. 1978.

[2] Asbjornsen O. "Stablilty and Robustness of the DMC Algorithm as Compared With the Traditional ARMA Model," *ACC Conference,* Atlanta, 1988.

[3] Ashutosh, P., J. Rawlings and E. Thomas, "Nonlinear Model Predictive Control," *Chem. Eng. Comm.* 87, 123-141, 1990.

[4] Athans, M. and P. Falb, *Optimal Control*, McGraw-Hill Book Company, New York, 1966.

[5] Bhat, J., M. Chidambara, and K. Madhavan, "Nonlinear Feedback Control of a CSTR," *Chem. Eng. Comm.* 101, 131-145, 1991.

[6] Brengel D. and W. Seider, "Multistep Nonlinear Predictive Controller" *Ind. Eng. Chem. Res.*, 28, 1812-1822, 1989.

[7] Boothby, W. M., *An Introduction to Differentiable Manifolds and Riemannian Geometry*, Academic Press, New York, 1975.

[8] Byrnes, C. I. and A. Isidori, "Global Feedback Stabilization of Nonlinear Minimum Phase Systems," *Proc. 24th IEEE Conf. Decision Contr.,* 1031, 1985.

[9] Calvet, J. and Y. Arkun, "Feedforward and Feedback Linearization of Nonlinear Systems and Its Implementation Using Internal Model Control (IMC)," *Ind. Eng. Chem. Res.*, 28, 1812-1822, 1989.

[10] Casti, J. L., *Nonlinear System Theory*, Academic Press, Inc., 1985.

[11] Campo, P. and M. Morari, "Model Predictive Optimal Averaging Level Control," *AIChE*, 35, 579-591, 1989.

[12] Campo, P. and M. Morari, "Robust Control of Process Subject to Saturation Nonlinearities," *Computers Chem. Eng.*, 14, 343-358, 1990.

[13] Clarke, D., C. Mohtadi and P. Tuffs, "Generalized Predictive Control Par I. The Basic Algorithm," *Automatica*, 23, 137-148, 1987.

[14] Clarke, D., C. Mohtadi and P. Tuffs, "Generalized Predictive Control Par II. Extensions and Interpretations," *Automatica*, 23, 149-160, 1987.

[15] Cutler, C., "Experience with the DMC Inverse for Identification," *Fourth International Conference on Chemical Process Control*, South Padre Island, 1991.

[16] Cutler, C., "Dyanmic Matrix Control. An Optimal Multivariable Control Algorithm with Constraints," *Ph.D Thesis,* University of Houston, 1983.

[17] Economou, C. G. and M. Morari, "Internal Model Control. 5. Extension to Nonlinear Systems," *Ind. Eng. Chem. Process Des. Dev.*, 25, 403, 1986.

[18] Eton J. W. and J. B. Rawlings, " Feedback Control of Chemical Process Using On-Line Optimization Techniques," *Computers Chem. Eng.* 14 469-479, 1990.

[19] Finlayason, B. *Nonlinear Analysis in Chemical Engineering,* McGraw-Hill Inc. New York, 1980.

[20] Garcia, C. and A. Morshedi, " Quadratic Programming Solution of Dynamic Matrix Control (QDMC)," *Chem. Eng. Commun.*, 46, 73-87, 1986.

[21] Garcia, C. and D. Prett, "Advances in Industrial Model-Predictive Control," *Third International Conference on Chemical Process Control,* New York, 1986.

[22] Garcia, C., D. Prett and M. Morari, "Model Predictive Control: Theory and Practice a Survey" *Automatica*, 25, 3, 335-348, 1989.

[23] Garcia, C. and M. Morari, "Internal Model Control. 1. A Unifying Review and Some New Results" *Ind. Eng. Process Des. Dev*, 21, 308-323, 1982.

[24] Garcia, C., B. Ramaker, and J. Pollard, "Total Process Control: Beyond the Design of Model Predictive Controllers," *Fourth International Conference on Chemical Process Control,* South Padre Island, 1991.

[25] Gattu, G. and E. Zafiriou, "Nonlinear Quadratic Dynamic Matrix Control With State Estimation," Submitted to Ind. and Eng. Chem. Res. 1991.

[26] Georgiou, A., C. Georagakis, and W. Luyben, "Nonlinear Dynamic Matrix Control for High-Purity Distillation Columns," *AIChE Journal,* 34, 1287-1298, 1988.

[27] Hasdorff, L, *Gradient Optimization and Nonlinear Control,* John Wiley and Sons, New York, 1976.

[28] Henson, A. and D. Seborg, "A Unified Differential Geometric Approach to Nonlinear Process Control," *AIChE Annual Meeting,* San Francisco, 1989.

[29] Henson, A. and D. Seborg, "A Critique of Differential Geometric Control Strategies for Process Control," *IFAC World Congress,* Tallinn, 1990.

[30] Hicks, J., A. Mohan, and W. Ray, "The Optimal Control of Polymerization Reactors," *Can. J, Chem. Eng.*, 47, 590-596, 1969.

[31] Hirschorn, R. and J. Davis, "Output Tracking for Nonlinear Systems With Singular Points," *SIAM J. Control and Optimization*, 25, 547-557, 1987.

[32] Hirschorn, R. and J. Davis, "Global Output Tracking for Nonlinear Systems," *SIAM J. Control and Optimization*, 26, 1321-1330, 1987.

[33] Hoo, K. and J. Kantor, "An Exothermic Continuous Stirred Tank Reactor is Feedback Equivalent to a Linear System," *Chem. Eng. Commun.* 37, 1-10, 1985.

[34] Hsiao, J. and L. Bezanson, "Offset Elimination Properties of Adaptive/Predictive Controllers," *Computers Chem. Eng.* 13, 1097-1104,1989.

[35] Isidori, A., *Nonlinear Control System,* Springer-Verlag, Berlin, 1989.

[36] Jang, S., J. Babu, and H. Mukai, "Control of Constrained Multivariable Nonlinear Process Using a Two-Phase Approach," *Ind. Eng. Chem. Res.*, 26, 2106-2114, 1987.

[37] Jones, D. and J. Finch, "Comparison of Optimal Algorithms," *Int. J. Control*, 40, 747, 1984.

[38] Kappos, E., "A Geometric Linearization Theory," *Proc. 28th IEEE Conf. Decision Contr.,* 77, 1989.

[39] Khorasani, K., "Robust Stabilization of Nonlinear Systems With Unmodelled Dynamics," *Int. J. Control*, 50, 827-844, 1989.

[40] Kravaris, C. and C. Chung, "Nonlinear State Feedback Synthesis by Global Input/Output Linearization,"*AIChE J.*, 33, 592-603, 1987.

[41] Kravaris, C. and J. Kantor, "Geometric Methods for Nonlinear Process Control. 1. Background," *Ind. Eng. Chem. Res.* 29, 2295-2310, 1989.

[42] Kravaris, C. and J. Kantor, "Geometric Methods for Nonlinear Process Control. 2. Controller Synthesis," *Ind. Eng. Chem. Res.* 29, 2310-2323, 1989.

[43] Kravaris, C. and M. Soroush, " Synthesis of Multivariable Nonlinear Controllers by Input/Output Linearization," *AIChE Journal*, 36, 249-264, 1990.

[44] Kravaris, C., R. Wright and J. Carrier, "Nonlinear Controllers for Trajectory Tracking in Batch Processes," *Computers Chem. Eng.*, 13, 73-82, 1989.

[45] Lennartson, B. E., "Combining Infrequent and Indirect Measurements by Estimation and Control," *Ind. Eng. Chem. Res.*, 28, 1653-1658, 1989.

[46] Li W. and L. Biegler, "Process Control Strategies For Constrained Nonlinear Systems," *Ind. Eng. Chem. Res.* 27, 1421-1433, 1988.

[47] Li W. and L. Biegler, "Newton-Type Controllers for Constrained Nonlinear Process with Uncertainty," *Ind. Eng. Chem. Res.* 29, 1647-1657, 1990.

[48] Li W., L. Biegler, C. Economou and M. Morari, "A Constrained Pseudo-Newton Control Strategy for Nonlinear Systems," *Computers Chem. Eng.*, 14, 451-468, 1990.

[49] Limqueco, L. C. and J. C. Kantor, "Nonlinear Output Feedback Control of an Exothermic Reactor," *Computers Chem. Eng.* 14, 427-437, 1990.

[50] Lien, C. Y. and T. W. Wang, *"Application of Feedback Linearization to Bioreactor Control," ACC Conf.*, San Diego, 1990.

[51] Lin H., J. Lewis, and R. Luecke. "Simulation of QDMC on a Fed-Batch Bioreactor," *ACC Conference,* Atlanta, 1988.

[52] Longwell, E., "Chemical Process and Nonlinear Control Technology," *Fourth International Conference on Chemical Process Control*, South Padre Island, 1991.

[53] Luenberger, D., *Linear and Nonlinear Programming*, Addison-Wesley Publishing Company, Reading, 1984.

[54] Luenberger, D., *Optimization by Vector Space Methods*, John Wiley and Sons, Inc. New York, 1969.

[55] McLellan, P., T. Harris, and D. Bacon, " Error Trajectory Descriptions of Nonlinear Controller Designs," *Chemical Engineering Science*, 45, 3017-3034, 1990.

[56] Moholer, R. *Nonlinear Systems*, Prentice Hall, Englewood Cliffs, 1991

[57] Morari, M. and E. Zafiriou, *Robust Process Control*, Prentice Hall, Englewood Cliffs, New Jersey, 1989.

[58] Morari, M. and J. Lee, " Model Predictive Control: The Good, the Bad, and the Ugly" *Fourth International Conference on Chemical Process Control*, South Padre Island, 1991.

[59] Nikolaou, M. and V. Manousiothakis, "A Hybrid Approach to Nonlinear System Stability and Performance," *AIChE Journal*, 35, 559-572, 1989.

[60] Peterson, T., E. Hernandez, Y. Arkun and F. J. Schork, "A Nonlinear DMC Algorithm and Its Applications to Semi-Batch Polymerization Reactor," *ACC Conf.*, 1989.

[61] Pike, R., *Optimization for Engineering Systems*, Van Nostrand Reihnold Company, New York, 1986.

[62] Prett, D. M. and C. Garcia, *Fundamental Process Control*, Butterworths Boston, 1988.

[63] Prett, D. and R. Gillette, "Optimization and Constrained Multivariable Control of a Catalytic Cracking Unit," *AIChE Annual Metting,* Houston, 19779.

[64] Prett, D. and M. Morari, *The Shell Process Control Workshop*, Butterworths, Boston 1987.

[65] Quintana, V. and E. Davison, "Clipping-off Gradient Algorithms to Compute Optimal Controls With Constrained Magnitude" Int. J. Control, 20, 243-255, 1974.

[66] Ramarajan, S. "Stability Criterion for Nonlinear Systems: Integral Restrictions," *Int. J. Control*, 48, 1433-1437, 1989.

[67] Renfro, J., A. Morshedi and O. Asbjornsen, " Simultaneous Optimization and Solution of Systems described by Differential/Algebraic Equations" *Comput. Chem. Eng.*, 11, 503-517, 1987.

[68] Ricker, L., "Model Predictive Control With State Estimation," *Ind. Eng. Chem. Res.* 29, 374-382, 1990.

[69] Ricker, L., "Model Predictive Control: State of the Art," *Fourth International Conference on Chemical Process Control*, South Padre Island, 1991.

[70] Riggs, J, and R. Rhinehart, "Comparison Between Process Model Based Controllers," *ACC Conference*, Atlanta, 1988.

[71] Seider, W., D. Brengel, A. Provost and S. Widago, "Nonlinear Analysis in Process Design. Why Overdesign To Avoid Complex Nonlinearities?" *Ind. Eng. Chem. Res.*, 29, 805-818, 1990.

[72] Seborg, D., T. Edgar, and D. Mellichamp, *Process Dynamics and Control*, John Wiley & Sons Inc., New York, 1989.

[73] Sifu, L., K. Lim, and D. G. Fisher, " A State Space Formulation for Model Predictive Control," *AIChE Journal*, 35, 241-249, 1989.

[74] Sikdar, S., M. Bier, and P. Todd, *Frontiers in Bioprocessing*, CRC Press, Inc. Boca Raton, 1989.

[75] Sistu, P. and W. Bequette, "Nonlinear Predictive Control of Uncertain Chemical Process," *AIChE Annual Meeting*, Chicago, 1990.

[76] Slotine J. and W. Li, *Applied Nonlinear Control*, Prentice Hall, Englewood Cliffs, 1991.

[77] Spong, M., J. Thorp, and J. Kleinwaks, "The Control of Robot Manipulators With Bounded Input," IEEE Transactions on Automatic Control, AC-31, 483-490, 1986.

[78] Srinivas, P., K. Kravaris, and Y. Wang, "On-Line Yield Optimization for Batch Processes," *AIChE Annual Meeting*, San Francisco, 1989.

[79] Stephanopoulos, G., and K. L. San, "Studies on On-line Bioreactor Identification I," *Biotechnol. Bioeng.*, 26, 1176, 1984.

[80] Swiniarski, R, "State Estimation and Optimal Control of Nonlinear Continuous Fermentation Processes" *ACC Conference,* Atlanta, 1988.

[81] Villadsen, J. and M. Michelsen, *Solution of Differential Equations Models by Polynomial Approximation*, Prentice-Hall Inc. Englewood Cliffs, 1978.

[82] Wang, D., C. Cooney, A. Demain, P. Dunnill, A. Humphrey, and M. Lilly, *Fermentation And Enzyme Technology*, John Wiley & Sons, New York, 1979.

[83] Zafiriou, E., "Robust Model Predictive Control of Processes With Hard Constraints" *Computers Chem. Eng.* 14, 359-371, 1990.

[84] Wang, T., "Feasibility of Applying a Robust Multivariable Controller to a Nonlinear Bacterial Growth System," *M.S. Thesis,* University of Tennesee, 1986.

[85] Wright, R. and C. Kravaris, "Nonminimum-Phase Compensation for Nonlinear Processes" *AIChe Journal* 38, 26-40, 1992.

[86] Yocum, F and Zimmerman J. "Real Time Control of Inverse Response Using Dynamic Matrix Control," *ACC Conference,* Atlanta, 1988.

[87] Zafiriou, E. and H. Chiou, "An Operator Control Theory Approach to the Design and Tuning of Constrained Model Predictive Controllers" *AIChE Annual Meeting,* San Francisco, 1989.

[88] Zames, G. and B. A. Francis, "Feedback, Minimax Sensitivity, and Optimal Robustness," *IEEE Transactions on Automatic Control*, AC-28, 5, 585-600, 1983.

[89] Zhou, J. and E. Zafiriou, "Iterative Batch-to-Batch Input Profile Determination for Semi-Batch Processes," *AIChE Annual Meeting,* San Francisco, 1989.

APPENDICES

APPENDIX A

Matlab Simulation Programs for the CSTBR

A.1 ELCC

This program simulates the response of the bioreactor introduced in Section 4.1 when the ELCC is used as a controller

```
%=================================================================
%
%                  DMC + Linearization + Minimization
%
%=================================================================
%
%  This program solves the Dynamic Matrix Control problem then it
%  uses input-output linearization to transform the calculated input
%  to the original input of the nonlinear system then minimizes
%  the distance between the linearized input and the real input.
%
%=================================================================
%
%             Solution to the Dynamic Matrix Control
%
%                     v(k) = pinv(A)*e(k+1)
%
%=================================================================
%
%  clear variables
    clear;
%
   global us umaxm ksm ysm sfm lfh h lgh umaxim umin vgl G;
%
%=================================================================
%
%  Nonlinear Plant
%
%     x(1) = (mu-u)*x(1)
%
%     x(2) = u*(sf-x(2))-mu*x(1)/ys
%
%     mu = umax*x(2)/(ks+x(2))
%
%
%     y=x(1)*x(3)
%
%
      umaxim=1.2   % Maximum input
%
%
      umin=0.1;       % Minimum input
%
```

```
%     Where:
%
      umax=0.5;     %  Maximum specific growth rate
      umaxm=umax;
      ks=0.05;      %  A constant in the Monod equation
      ksm=ks;
      yss=0.5;       %  Biomass yield with respect to substrate
      ysm=yss;
      sf=9.0;       %  Substrate concentration in the feed
      sfm=sf;
%
%
%
%=================================================================
%
%  Nominal Steady State Plant:
    d=0.001;
    i=1;
    while d<umax-.0002
          mu=d;
          s=mu*ks/(umax-mu);
          b=d*(sf-s)*yss/mu;
          uis(i)=d;
          ros(i)=b*mu;
          if d<=0.45
             d=d+0.001;
          end,
          if d>0.45
             d=d+0.0005;
          end,
          i=i+1;
    end,
%
%=================================================================
%
%  Steady State Plant:
    d=0.001;
    i=1;
    while d<umaxm-.0002
          mu=d;
          s=mu*ksm/(umaxm-mu);
          b=d*(sfm-s)*ysm/mu;
          ui(i)=d;
          ro(i)=b*mu;
          if d<=0.43
             d=d+0.001;
          end,
          if d>0.43
             d=d+0.0005;
          end,
          i=i+1;
    end,
%
%=================================================================
%  Linearized  plant:
%
%              .
%              x = H x + B u
%
%              y = C x
%
```

```
      H=[-1];
      B=[1];
      C=[1];
      ni=1;           % Number of inputs
      no=1;           % Number of outputs
%=================================================================
%
%  Control Parameters
%
     dt=0.05;     % Sampling time
     m=70;        % Number of time intervals used to describe the system
     n=30;        % Number of future input moves to be computed
     np=100;      % Predicted horizon (generally n+m)
     hor=250;
%
%=================================================================
%
%  Initial Conditions for the linearized model
%
      x=[0];
      ud=1.0;
      t(1)=0.0;
      y(1)=0;
%
%=================================================================
%
%  Discretization
%
%    Discrete system:
%
%                    x((k+1)dt)=Ad x((k)dt) + Bd u((k)dt)
%                    y((k)dt)= C x((k)dt)
%
      em=expm([dt*H dt*B; zeros(no,ni+no)]);
      Ad=em(1:ni,1:ni);
      Bd=em(1:no,(ni+1):(ni+no));
%
%=================================================================
%
%  System Step response
%
 for i=2:m+1
 t(i)=dt*(i-1);
         x=Ad*x+Bd*ud;
 y(i)=C*x;
 end,
%
% The values of y from m to np are taken as y(m)
%
 if np>m
tm=m+1;
for i=tm:np
y(i)=y(m);
        end,
 end,
 y(np+1)=y(np);
%
%=================================================================
```

```
%
%  Construction of the dynamics matrix A
%
    A=zeros(np,n);
    WU=0.8*eye(np,n);
 for i=1:n
 l=1;
 for j=i:np
                  l=l+1;
 A(j,i)=y(l);
        end,
 end,
A=[A;WU];
%
%================================================================
%
%  Calculation of the inverse of the dynamic matrix
%  and save only the first column
 dv=A(1:np,1);
 p=zeros(np,1);
 wr=inv(A'*A)*A';
 aiv=wr(1,:);
%
%================================================================
%  Initial Conditions for the simulation
%
eps=0.5;
niu=5;
% select initial d
         dss=0.49;
         miu=dss;
         s=miu*ks/(umax-miu);
         b=dss*(sf-s)*yss/miu;
 u=0;
 yr = dv(1)*u;
 tl(1) = 0.0;
 xx=0;
 x= [b;s];
 bp(1)=b;
 sp(1)=s;
 yy(1)=x(1)*miu;
 xs=yy(1);
 ys(1)=yy(1);
 h=yy(1);
 lgh=-miu*x(1)+x(1)*(sf-x(2))*ks*umax/((ks+x(2))^2);
 lghp(1)=lgh;
 lfh=-x(1)*(umax*ks/((ks+x(2))^2))*miu*x(1)/yss+miu*miu*x(1);
 vv(1)=lgh*dss+lfh+h;
 vf=dss*lgh+lfh+h;
 dd=vv(1)-vf;
 yl(1) = 0;
 v(1)=0;
 us=dss;
 uu(1)=us;
 ubs=us;
%
%================================================================
```

```
%
% Desired output or desired trajectory
%
   for i=1: hor+hor
            r(i)=yy(1)*1.15-yy(1);
   end,
%
%================================================================
%
%    Parameters for the optimization
%
     N=[1;-1];
     C=[umaxim;-umin];
     Q=1;
% simulation for the interval np
%
  for mk = 2:hor
          qq=0;
          po = p(1)+dv(1)*u;
          d = yr - po;
          for i=1:np-1
                p(i) = p(i+1) + dv(i+1)*u + d;
          end,
          p(np) = p(np-1);
          ll=mk-1;
          for i =1:np
              e(i) = r(ll+i) - p(i);
          end
          u=0;
          for i=1:m
                u=u+aiv(i)*e(i);
          end,
          v(mk)=v(mk-1)+u;
          vv(mk)=vv(mk-1)+u;
          lgh=-miu*x(1)+x(1)*(sf-x(2))*ks*umax/((ks+x(2))^2);
          lghp(mk)=lgh;
          lfh=-x(1)*(umax*ks/((ks+x(2))^2))*miu*x(1)/yss+miu*miu*x(1);
          h=x(1)*miu;
      us=(vv(mk)-lfh-h)*sign(lgh)*abs(lgh^niu)/(abs(lgh^(niu+1))+eps);
      if us<umin,
            qq=1;
      end,
      if us>umaxim,
             qq=1;
      end,
      lggh=(lgh^2+eps)/lgh;
        if qq==1
            P=N*inv(lggh'*Q*lggh)*N';
            bv=lggh'*Q*(vv(mk)-lfh-h);
            D=C-N*inv(lggh'*Q*lggh)*bv;
            gamma=[0;0];
            J=gamma'*P*gamma/2+gamma'*D;
            wv(1)=-(D(1) +P(2,2)*gamma(2))/P(1,1),
            gamman(1,1)=max(0,wv(1));
            wv(2)=-(D(2) +P(2,1)*gamman(1))/P(2,2),
            gamman(2,1)=max(0,wv(2));
            Jn=gamman'*P*gamman/2+gamman'*D;
```

```
            err=J-Jn,
            while err>0
                  gamma=gamman;
                  J=Jn;
                  wv(1,1)=-(D(1) +P(2,2)*gamma(2))/P(1,1),
                  gamman(1,1)=max(0,wv(1));
                  wv(2)=-(D(2) +P(2,1)*gamman(1))/P(2,2),
                  gamman(2,1)=max(0,wv(2));
                  Jn=gamman'*P*gamman/2+gamman'*D;
                  err=J-Jn,
            end,
            us=inv(lggh'*Q*lggh)*(bv-N'*gamma),
         end,
           vv(mk)=us*lgh+lfh+h;
           u=vv(mk)-vv(mk-1);
           uu(mk)=us;
           tl(mk)=dt*(mk-1);
           ti=[tl(mk-1) tl(mk)];
           xi=x;
           if mk<=40
                [tr,xr]=lsodar('fcstba',ti,xi);
                x=[xr(2,1);xr(2,2)];
                bp(mk)=x(1);
                sp(mk)=x(2);
                miu=umax*x(2)/(ks+x(2));
           end,
           if mk>40
                [tr,xr]=lsodar('fmcstba',ti,xi);
                x=[xr(2,1);xr(2,2)];
                bp(mk)=x(1);
                sp(mk)=x(2);
                miu=umaxm*x(2)/(ksm+x(2));
           end,
           yy(mk)=x(1)*miu;
         yr=yy(mk)-yy(1);
  end,
%
%=================================================================
%
% Plot results
%
%=================================================================
    axis([0.3 0.6 1.2 2.2])
   plot(ui,ro,uu,yy,uis,ros)
   ylabel('Total Rate of Growth, R')
   xlabel('Dilution Rate, D');
   axis;
%=================================================================
%
%                    THE END
%
%=================================================================
      function FX=fcstba(t,X)
      umax=0.5;
      ks=0.05;
      ys=0.5;
      sf=9.0;
      U=us;
```

```
      miu=umax*X(2)/(ks+X(2));
      FX(1) = (miu-U)*X(1);
      FX(2) = U*(sf-X(2))-miu*X(1)/ys;
%================================================================
      function FX=fmcstba(t,X)
      umax=umaxm;
      ks=ksm;
      ys=ysm;
      sf=sfm;
      U=us;
      miu=umax*X(2)/(ks+X(2));
      FX(1) = (miu-U)*X(1);
      FX(2) = U*(sf-X(2))-miu*X(1)/ys;
%================================================================
```

A.2 Nonlinear Controller with Global Linearization

This program simulates the response of the bioreactor introduced in Section 5.1 when a nonlinear controller that uses a global linearization technique is used as a controller

```
%
%============================================================
%
%    This program uses the conjugate gradient to solve
%    the  following problem:
%
%                           tf
%    J(u)=M[x(tf),tf]+  S   l(x,u)dt
%                           to
%    subject to:
%
%              .
%              x=f(x,u); x(to)=xo
%
%              umin<=uj<=umaxim
%
%============================================================
%
% The following functions are specified:
%
%            nlcost (t,x)    : Calculates the index of performance
%
%            fcstbcc(t,x)    : System equations
%
%                                                  .   -dH
%            flambda(t,ld)   : Costate equation  ld=----
%                                                    dx
%
%
%                              dH
%            fg(ld,x,u)      : ----
%                              du
%
%            funalf(af)      : Use to find the alfa that minimizes
%                              J(u+alf*p)
%
%            fbeta(gn,gold)  : Determine beta
```

```
%
%=======================================================
%
   clear,
%  The following variables are assumed to be global
%
   global Q R xi tv to tter dt xr us uu vn al;
   global rs umaxim umin pp jk fcv umaxm ksm ysm sfm;
 %
 %=======================================================
 %
 % Process parameters
 %
 %
      umax=0.5;     %  Maximum specific growth rate
      umaxm= umax*.9;
      ks=0.05;      %  A constant in the Monod equation
      ksm=ks;
      ys=0.5;       %  Biomass yield with respect to substrate
      ysm=ys;
      sf=9.0;       %  Substrate concentration in the feed
      sfm=sf;
      umaxim=1.2;   %  Maximum dilution rate
      umin=0;       %  Minimum dilution rate
 %
 %=======================================================
 %
 % Definition of Q,R,C,al
 %
     Q=1;
     R=0;
     al=0.999;
 %
 %=======================================================
 %
 % Definition of the time interval for optimization
 %
     to=0;
     dt=0.05;
     trr=round(8/dt);
     tter=trr*dt;
     tgg=to:trr;
     tv=tgg*dt;
     tgg=trr:-1:to;
     tb=tgg*dt;
     [nd vn]=size(tv);
 %
 %=======================================================
 %
 % Number of loops that the optimization will be performed
 %
    nl=vn;
    nloops=nl+150;
 %
 %=======================================================
 %
 %  Steady State Nominal Plant:
 %
    d=0.01;
```

```
    i=1;
    wl=1;
    while d<umax-.001
          mu=d;
          s=mu*ks/(umax-mu);
          b=d*(sf-s)*ys/mu;
          uis(i)=d;
          ros(i)=b*mu;
          if d<=0.45
             d=d+0.01;
          end,
          if d>0.45
             d=d+0.001;
          end,
          i=i+1;
    end,
%
 %=============================================================
 %
 %  Steady State Plant:
 %
    d=0.01;
    i=1;
    wl=1;
    while d<umaxm-.001
          mu=d;
          s=mu*ksm/(umaxm-mu);
          b=d*(sfm-s)*ysm/mu;
          ui(i)=d;
          ro(i)=b*mu;
          if d<=0.8*umaxm
             d=d+0.01;
          end,
          if d>0.8*umaxm
             d=d+0.001;
          end,
          i=i+1;
    end,
%
%=====================================================
%  Initial Conditions for the simulation
%
% select initial d
          dss=0.485;
          miu=dss;
          sss=miu*ks/(umax-miu);
          bss=dss*(sf-sss)*ys/miu;
          xi=[bss;sss];
 %
 %=====================================================
 %
 % Initial values of the vector used to plot results
 %
    yp(1)=xi(1)*umax*xi(2)/(ks+xi(2)),
 %=====================================================
 %
 % Desired output
 %
    rorig=yp(1)*1,
 %
```

```
%=========================================================
%
   dc=0;
   bp(1)=xi(1);
   sp(1)=xi(2);
   tp(1)=to;
   up(1)=dss;
   options(2)=1e-3;
   options(3)=1e-3;
   options(14)=30.0;
%
%=========================================================
%
% Initial values of the input vector
%
   u(1)=dss;
   for i=2:vn
       u(i)=dss;
   end,
   fcv=1e-5;
%
%
%=========================================================
%
% Optimization loop
%
 for i=1:nloops
    %  rs=rorig+dc;
      rs=rorig;
      uu=u;
      [tr,xr]=lsodar('fcstbcc',tv,xi);
      xx=xr;
      jn(i)=nlcost(xx);
      miu=umax*xr(vn,2)/(ks+xr(vn,2));
      lfinal=[al^(tv(vn))*Q*(2*xr(vn,1)*miu^2-2*rs*miu)/log(al);0];
      [tf,ldf]=lsodar('flambda',tb,lfinal);
      for kk=1:vn
         ld(kk,:)=ldf(vn+1-kk,:);
      end,
      g=fg(ld,u);
      gn=g;
      pp=-g;
      afp=fmin('funalfo',-10,10,options);
      for kk=1:vn
            unew(kk)=u(kk)+afp*fcv*pp(kk);
            if unew(kk)>umaxim
                  unew(kk)=umaxim;
            end,
            if unew(kk)<umin
                  unew(kk)=umin;
            end,
      end,
      uu=unew;
      [tr,xx]=lsodar('fcstbcc',tv,xi);
      jt=nlcost(xx),
      error=jn(i)-jt;
      kt=1;
      mult=1;
      while error>0.0001,
```

```
        g=gn;
        jn(i)=jt;
        u=uu;
        xr=xx;
        miu=umax*xr(vn,2)/(ks+xr(vn,2));
lfinal=[al^(tv(vn))*Q*(2*xr(vn,1)*miu^2-2*rs*miu)/log(al);0];
        [tf,ldf]=lsodar('flambda',tb,lfinal);
        for ik=1:vn
            ld(ik,:)=ldf(vn+1-ik,:);
        end,
        gn=fg(ld,u);
        beta=fbeta(gn,g);
        pp=-gn+beta*pp;
        af=afp;
        afp=fmin('funalfo',-10,10,options);
        for kk=1:vn
            unew(kk)=u(kk)+afp*fcv*pp(kk);
            if unew(kk)>umaxim
                    unew(kk)=umaxim;
            end,
            if unew(kk)<umin
                  unew(kk)=umin;
            end,
        end,
        uu=unew;
        [tr,xx]=lsodar('fcstbcc',tv,xi);
        jt=nlcost(xx),
        error=jn(i)-jt;
        kt=kt+1;
        if kt>25
            error=-1000;
        end,
end,
us=u(1);
tc=[to to+dt];
[tyy,xyy]=lsodar('fcstba',tc,xi);
ycc=xyy(2,1)*umax*xyy(2,2)/(ks+xyy(2,2));
if i<=10
    [ty,xy]=lsodar('fcstba',tc,xi);
end,
if i>10
    [ty,xy]=lsodar('fmcstba',tc,xi);
end,
xi=[xy(2,1);xy(2,2)];
to=to+dt;
tter=tter+dt;
tv=tv+ones(tv)*dt;
tb=tb+ones(tb)*dt;
tp(i+1)=to;
yp(i+1)=xi(1)*umax*xi(2)/(ks+xi(2));
dc=yp(i+1)-ycc;
eval=rs-yp(i+1);
bp(i+1)=xi(1);
sp(i+1)=xi(2);
tp(i+1)=to;
up(i+1)=us;
for il=1:vn-1
    u(il)=u(il+1);
```

```
      end,
   end,
%
%=============================================================
%
% Plot results
%
% ========================
%
%    Output
%
   fn=max([umax umaxm]);
   vv=[0 fn 0 2.5];
   axis(vv);
   plot(ui,ro,up,yp,uis,ros);
   ylabel('Total Rate of Growth, R');
   xlabel('Dilution Rate, D');
% ========================
%
%    Save Results
%
   save test1
%
% ============================================================
%
%     THE END
%
% ============================================================
     function FX=fcstbcc(t,X)
%
%    System equations
%
     umax=0.5;     %  Maximum specific growth rate
     ks=0.05;      %  A constant in the Monod equation
     ys=0.5;       %  Biomass yield with respect to substrate
     sf=9.0;       %  Substrate concentration in the feed
     U=spline(tv,uu,t);
     miu=umax*X(2)/(ks+X(2));
     FX(1) = (miu-U)*X(1);
     FX(2) = U*(sf-X(2))-miu*X(1)/ys;
%  ============================================================
%
  function hv=funalfo(afp)
%
%    The following is assumed to be global:
%
%          Q,R,xi,tv,xr,us,uu,n,al,r,umax,ks,umaxim,umin,p
%
%    Calculate the new input
%
%       check if u+alfa is in the feasible set otherwise set
%       alfa to zero
%
    for kk=1:vn
          unew(kk)=uu(kk)+afp*fcv*pp(kk);
        if unew(kk)>umaxim
                unew(kk)=umaxim;
```

```
            end,
            if unew(kk)<umin
                  unew(kk)=umin;
            end,
      end,
      uold=uu;
      uu=unew;
      [tr,xx]=lsodar('fcstbcc',tv,xi);
      hv=nlcost(xx);
      uu=uold;
%  ============================================================
%
      function Fl=flambda(tx,ld)
%
%     Costate equations
%
      umax=0.5;     % Maximum specific growth rate
      ks=0.05;      % A constant in the Monod equation
      ys=0.5;       % Biomass yield with respect to substrate
      sf=9.0;       % Substrate concentration in the feed
      U=spline(tv,uu,tx);
      x(1)=spline(tv,xr(:,1),tx);
      x(2)=spline(tv,xr(:,2),tx);
      x(3)=spline(tv,xr(:,3),tx);
      miu=umax*x(2)/(ks+x(2));
      df=[miu-U -miu/ys;0 -U];
      dl=[al^(tx)*Q*(2*x(1)*miu^2-2*rs*miu);0];
      Fl=-df*ld-dl;
%  ============================================================
%
% This subroutine calculates the cost J
%
%   J=INT(al^(t)((Cx(t)-r(t))'Q(Cx(t)-r(t))+u(t)'Ru(t))))
%     - al^(tf)((Cx(tf)-r(tf))'Q(Cx(tf)-r(tf))+u(tf)'Ru(tf)))/ln(al)
%
% global Q,R,xi,tv,us,uu,n,al,rs,umax,ks,umaxim,umin
%
   function J=nlcost(x)
      umax=0.5;     % Maximum specific growth rate
      ks=0.05;      % A constant in the Monod equation
      ys=0.5;       % Biomass yield with respect to substrate
      sf=9.0;       % Substrate concentration in the feed
   J=0;
   dt=tv(2)-tv(1);
   nf=vn-1;
   for i=2:nf,
       mu=umax*x(i,2)/(ks+x(i,2));
 J=J+2*al^(tv(i))*((mu*x(i,1)-rs)'*Q*(mu*x(i,1)-rs)+uu(i)'*R*uu(i));
   end,
   mu=umax*x(1,2)/(ks+x(1,2));
   J=J+al^(tv(1))*((mu*x(1,1)-rs)'*Q*(mu*x(1,1)-rs)+uu(1)'*R*uu(1));
   mu=umax*x(vn,2)/(ks+x(vn,2));
tmp=al^(tv(vn))*((mu*x(vn,1)-rs)'*Q*(mu*x(vn,1)-rs)+uu(vn)'*R*uu(vn));
   J=J+tmp;
   J=J*dt/2-tmp/log(al);
```

```
% ============================================================
   function hv=fbeta(gn,g)
%
%    The following is assumed to be global:
%
%          Q,R,xi,tv,xr,us,uu,n,al,r,umax,ks,umaxim,umin,p
%
%    Calculate the new input
%
%       check if u+alfa is in the feasible set otherwise set
%       alfa to zero
%
   dt=tv(2)-tv(1);
   nf=vn-1;
   I1=0;
   I2=0;
   I3=0;
   for i=2:nf,
       I1=I1+2*gn(i)*gn(i);
       I2=I2+2*g(i)*g(i);
       I3=I3+2*gn(i)*pp(i);
   end,
   I1=I1+gn(1)*gn(1)+gn(vn)*gn(vn);
   I1=dt*I1/2;
   I2=I2+g(1)*g(1)+g(vn)*g(vn);
   I2=dt*I2/2;
   I3=I3+gn(1)*pp(1)+pp(vn)*gn(vn);
   I3=dt*I3/2;
   if I2 ~= 0
        Bo=I1/I2;
   end,
   if I3 ~= 0
        Bm=I1/I3;
   if I2 == 0
        Bo=0;
        hv=0;
   end,
   if I3 == 0
        Bm=0;
        hv=0;
   end,
   if Bo<Bm
      if I2~=0
        hv=Bo;
      end,
   end,
   if Bo>=Bm
      if I3 ~=0
        hv=Bm/2;
      end,
   end,
% ============================================================
%
function g=fg(ld,u)
sf=9.0;
for kk=1:vn
g(kk)=-ld(kk,1)*xr(kk,1)+ld(kk,2)*(sf-xr(kk,2))+2*al^(tv(kk))*R*u(kk);
end,
```

APPENDIX B

Simulation Programs for the CSTR

B.1 ELCC

This program simulates the response of the CSTR introduced in Section 4.2 when the ELCC is used as a controller

```
%================================================================
%
%                  DMC + Linearization + Minimization
%
%================================================================
%
%  This program solves the Dynamic Matrix Control problem then it
%  uses input-output linearization to transform the calculated input
%  to the original input of the nonlinear system then minimizes
%  the distance between the linearized input and the real input.
%
%================================================================
%
%             Solution to the Dynamic Matrix Control
%
%                      v(k)=inv(A'A)*A'e(k+1)
%
%================================================================
%
%  clear variables
    clear;
%
   global us Aim C1m;
   global C2m Q1m Q2m lflfh lfh h lglfh umaxim umin vgl G;
%
%================================================================
%
%  Nonlinear Plant
%
%     x(1) = (Ai-x(1))/tau - k1 x(1) + k2 x(2)
%
%     x(2) = (Bi-x(2))/tau + k1 x(1) - k2 x(2)
%
%     x(3) = -dH(k1 x(1) - k2 x(2))/(rho Cp) + (u-x(3))/tau
%
%
%     y=x(2)
%
%
      umaxim=700;   % Maximum input
%
%
      umin=300;      % Minimum input
```

```
%
%     Where:
%
      C1=5000;     %  Frequency factor
      C1m=C1;
      C2=1e6;      %  Frequency factor
      C2m=C2*1.2;
      Q1=10000;        %  Activation energy
      Q1m=Q1;
      Q2=15000;        %  Activation energy
      Q2m=Q2;
      rho=1;
      Ai=1.0;
      Aim=Ai;
      Bi=1-Ai;
      Bim=1-Aim;
      tau=60;
      Cp=1000;
      R=1.987;
      dH=-5000;
%
%=================================================================
%
%  Steady State Plant:
i = 1;
T = 250.0;
        lv=1;
while T <= 800.0
k1 = C1m*exp(-Q1m/R/T);
k2 = C2m*exp(-Q2m/R/T);
A = (Aim*(1./tau + k2)+Bim*k2)/(1./tau + k1 + k2);
B = Aim - A + Bim;
ter = dH*tau*(k1*A - k2*B)/rho/Cp;
Ti = T + ter;
Bp(i) = B;
Tip(i) = Ti;
i = i + 1;
T = T + 1.0;
end
%
%=================================================================
%
%  Steady State Nominal Plant:
i = 1;
T = 250.0;
        lv=1;
while T <= 800.0
k1 = C1*exp(-Q1/R/T);
k2 = C2*exp(-Q2/R/T);
A = (Ai*(1./tau + k2)+Bi*k2)/(1./tau + k1 + k2);
B = Ai - A + Bi;
ter = dH*tau*(k1*A - k2*B)/rho/Cp;
Ti = T + ter;
Bpm(i) = B;
Tipm(i) = Ti;
i = i + 1;
T = T + 1.0;
end
%
```

```
%=================================================================
%  Linearized  plant:
%
%              .
%              x = H x + B u
%
%              y = C x
%
     H=[0 1;-1 -3];
     B=[0;1];
     C=[1 0];
     ni=2;           % Number of states
     no=1;           % Number of outputs
%=================================================================
%
%  Control Parameters
%
    dt=0.5;     % Sampling time sec
    m=20;       % Number of time intervals used to describe the system
    n=10;       % Number of future input moves to be computed
    np=30;      % Predicted horizon (generally n+m)
    hor=300;
%
%=================================================================
%
%  Initial Conditions for the linearized model
%
     x=[0;0];
     ud=1.0;
     t(1)=0.0;
     y(1)=0;
%
%=================================================================
%
%  Discretization
%
%    Discrete system:
%
%                    x((k+1)dt)=Ad x((k)dt) + Bd u((k)dt)
%                    y((k)dt)= C x((k)dt)
%
     em=expm([dt*H dt*B; zeros(no,ni+no)]);
     Ad=em(1:ni,1:ni);
     Bd=em(1:ni,(ni+1):(ni+no));
%
%=================================================================
%
%  System Step response
%
 for i=2:m+1
 t(i)=dt*(i-1);
        x=Ad*x+Bd*ud;
 y(i)=C*x;
 end,
%
% The values of y from m to np are taken as y(m)
%
 if np>m
tm=m+1;
```

```
for i=tm:np
y(i)=y(m);
        end,
 end,
 y(np+1)=y(np);
%
%=================================================================
%
%  Construction of the dynamics matrix A
%
    A=zeros(np,n);
    WU=10*eye(np,n);
 for i=1:n
 l=1;
 for j=i:np
                  l=l+1;
 A(j,i)=y(l);
         end,
 end,
 A=[A;WU];
%
%=================================================================
%
%  Calculation of the inverse of the dynamic matrix
%  and save only the first column
 dv=A(1:np,1);
 p=zeros(np,1);
 wr=inv(A'*A)*A';
 aiv=wr(1,:);
%
%=================================================================
%  Initial Conditions for the simulation
%
  eps=4e-12;
% select initial Ti
         T=450;
         %T=430;
         k1 = C1*exp(-Q1/R/T);
         k2 = C2*exp(-Q2/R/T);
 A = (Ai*(1./tau + k2)+Bi*k2)/(1./tau + k1 + k2);
         B = Ai - A + Bi;
         ter = dH*tau*(k1*A - k2*B)/rho/Cp;
         Ti = T + ter;
 u=0;
 yr = dv(1)*u;
 tl(1) = 0.0;
 xx=[0;0];
 x= [A;B;T];
 bp(1)=B;
 ap(1)=A;
 tp(1)=T;
 yy(1)=x(2);
 xs=yy(1);
 ys(1)=yy(1);
 h=yy(1);
 k1s=C1*exp(-Q1/(R*T));
```

```
k2s=C2*exp(-Q2/(R*T));
f1=(Ai-A)/tau-k1s*A+k2s*B;
f2=(Bi-B)/tau+k1s*A-k2s*B;
f3=-dH*(k1s*A-k2s*B)/(rho*Cp)-T/tau;
h=B;
lfh=f2;
lfh1=k1s;
lfh2=-1/tau-k2s;
lfh3=(A*k1s*Q1-B*k2s*Q2)/(R*T*T);
lglfh=lfh3/tau;
lflfh=lfh1*f1+lfh2*f2+lfh3*f3;
vv(1)=(lglfh^2+eps)*Ti/lglfh+lflfh+h+3*lfh;
yl(1) = 0;
v(1)=0;
us=Ti;
uu(1)=us;
%
%================================================================
%
% Desired outpu or desired trajectory
%
   for i=1: hor+hor
  r(i)=yy(1)*1.0-yy(1);
   end,
%
%================================================================
%
% simulation for the interval np
%
  for mk = 2:hor
          mk,
          po = p(1)+dv(1)*u;
          d = yr - po;
          for i=1:np-1
                p(i) = p(i+1) + dv(i+1)*u + d;
          end,
          p(np) = p(np-1);
          ll=mk-1;
          for i =1:np
              e(i) = r(ll+i) - p(i);
          end
          u=0;
          for i=1:m
                u=u+aiv(i)*e(i);
          end,
          v(mk)=v(mk-1)+u;
          vv(mk)=vv(mk-1)+u;
          k1=C1*exp(-Q1/(R*x(3)));
          k2=C2*exp(-Q2/(R*x(3)));
          f1=(Ai-x(1))/tau-k1*x(1)+k2*x(2);
          f2=(Bi-x(2))/tau+k1*x(1)-k2*x(2);
          f3=-dH*(k1*x(1)-k2*x(2))/(rho*Cp)-x(3)/tau;
          h=x(2);
          lfh=f2;
          lfh1=k1;
          lfh2=-1/tau-k2;
          lfh3=(x(1)*k1*Q1-x(2)*k2*Q2)/(R*x(3)*x(3));
          lglfh=lfh3/tau;
```

```
        lflfh=lfh1*f1+lfh2*f2+lfh3*f3;
        us=(vv(mk)-lflfh-h-3*lfh)*lglfh/(lglfh^2+eps);
        qq=0;
      if us<umin,
          qq=1;
      end,
      if us>umaxim,
           qq=1;
      end,
      lggh=(lglfh^2+eps)/lglfh;;
        if qq==1
          P=N*inv(lggh'*Q*lggh)*N';
          bv=lggh'*Q*(vv(mk)-3*lfh-h-lflfh);
          D=C-N*inv(lggh'*Q*lggh)*bv;
          gamma=[0;0];
          J=gamma'*P*gamma/2+gamma'*D;
          wv(1)=-(D(1) +P(2,2)*gamma(2))/P(1,1),
          gamman(1,1)=max(0,wv(1));
          wv(2)=-(D(2) +P(2,1)*gamman(1))/P(2,2),
          gamman(2,1)=max(0,wv(2));
          Jn=gamman'*P*gamman/2+gamman'*D;
          err=J-Jn,
          while err>0
                gamma=gamman;
                J=Jn;
                wv(1,1)=-(D(1) +P(2,2)*gamma(2))/P(1,1),
                gamman(1,1)=max(0,wv(1));
                wv(2)=-(D(2) +P(2,1)*gamman(1))/P(2,2),
                gamman(2,1)=max(0,wv(2));
                Jn=gamman'*P*gamman/2+gamman'*D;
                err=J-Jn,
          end,
          us=inv(lggh'*Q*lggh)*(bv-N'*gamma),
      end,
       vv(mk)=lglfh*us+lflfh+h+3*lfh;
        u=vv(mk)-vv(mk-1);
        uu(mk)=us;
        tl(mk)=dt*(mk-1);
        ti=[tl(mk-1) tl(mk)];
        xi=x;
        if mk<=40
            [tr,xr]=lsodar('fcstra',ti,xi);
        end,
        if mk>40
            [tr,xr]=lsodar('fmcstra',ti,xi);
        end,
        x=[xr(2,1);xr(2,2);xr(2,3)];
        bp(mk)=x(2);
        ap(mk)=x(1);
        tp(mk)=x(3);
        yy(mk)=x(2);
       yr=yy(mk)-yy(1);
  end,
%
plot(Tip,Bp,uu,yy,Tipm,Bpm);
xlabel('Inlet Temperature, K');
```

```
ylabel('Concentration of B');
%=============================================================
%
%                    THE END
%
%=============================================================
%
      function FX=fcstra(t,X)
      TAU=60.0;
      AI=1.0;
      CK=5000;
      CT=1e6;
      Q1=10000;
      Q2=15000;
      RHO=1;
      DH=-5000;
      RR=1.987;
      CP=1000;
      U(1)=us;
FX(1) = (AI-X(1))/TAU+X(2)*CT*exp(-Q2/(X(3)*RR));
FX(1) = FX(1)-X(1)*CK*exp(-Q1/(X(3)*RR));
FX(2) = -X(2)/TAU-X(2)*CT*exp(-Q2/(X(3)*RR))+X(1)*CK*..
exp(-Q1/(X(3)*RR));
FX(3) = (U(1)-X(3))/TAU;
tw1=-DH*(X(1)*CK*exp(-Q1/(X(3)*RR))-X(2)*CT*..
exp(-Q2/(X(3)*RR)))/(CP*RHO);
FX(3) = FX(3)+tw1;
%
%=============================================================
      function FX=fmcstra(t,X)
      TAU=60.0;
      AI=Aim;
      CK=C1m;
      CT=C2m;
      BI=1-AI;
      Q1=Q1m;
      Q2=Q2m;
      RHO=1;
      DH=-5000;
      RR=1.987;
      CP=1000;
      U(1)=us;
FX(1) = (AI-X(1))/TAU+X(2)*CT*exp(-Q2/(X(3)*RR));
FX(1) = FX(1)-X(1)*CK*exp(-Q1/(X(3)*RR));
FX(2) = (BI-X(2))/TAU-X(2)*CT*exp(-Q2/(X(3)*RR));
FX(2) = FX(2)+X(1)*CK*exp(-Q1/(X(3)*RR));
FX(3) = (U(1)-X(3))/TAU;
tw1=-DH*(X(1)*CK*exp(-Q1/(X(3)*RR))-X(2)*CT*..
exp(-Q2/(X(3)*RR)))/(CP*RHO);
FX(3) = FX(3)+tw1;
```

B.2 Nonlinear Controller with Global Linearization for the CSTR

This program simulates the response of the CSTR introduced in Section 5.1 when a nonlinear controller that uses a global linearization technique is used as a controller

```
%=================================================
```

```
%
% This program uses the conjugate gradient to solve
% the  following problem:
%
%                          tf
%    J(u)=M[x(tf),tf]+  S   l(x,u)dt
%                          to
%    subject to:
%                .
%               x=f(x,u); x(to)=xo
%
%               umin<=uj<=umaxim
%
%=================================================
%
% The following functions are specified:
%
%            costcstr (t,x) : Calculates the index of performance
%
%            fcstrcc(t,x)    : System equations
%                                                    .   -dH
%            flamcstr(t,ld)  : Costate equation ld= ----
%                                                     dx
%
%
%                                  dH
%            fgcstr(ld,x,u)     : ----
%                                  du
%
%            falfcstr(af)     : Use to find the alfa that
%                               minimizes J(u+alf*p)
%
%            fbeta(gn,gold) : Determine beta
%
%=================================================
%
   clear,
%  The following variables are assumed to be global
%
   global Q R xi tv to;
   global tter dt xr us uu vn al rs umaxim;
   global umin pp jk fcv C1m C2m Aim Q1m Q2m;
 %
%=================================================
%  Nonlinear Plant
%     .
%     x(1) = (Ai-x(1))/tau - k1 x(1) + k2 x(2)
%     .
%     x(2) = (Bi-x(2))/tau + k1 x(1) - k2 x(2)
%     .
%     x(3) = -dH(k1 x(1) - k2 x(2))/(rho Cp) + (u-x(3))/tau
%
%     y=x(2)
%
%
      umaxim=700;   % Maximum input
%
      umin=280;      % Minimum input
```

```
%
% Process parameters
%
%
%
      C1=5000;     %  Frequency factor
      C1m=C1;
      C2=1e6;      %  Frequency factor
      C2m=C2*1.5;
      Q1=10000;        %  Activation energy
      Q1m=Q1;
      Q2=15000;        %  Activation energy
      Q2m=Q2;
      rho=1;
      Ai=1.0;
      Aim=Ai*0.8;
      Bi=1-Ai;
      Bim=1-Aim;
      tau=60;
      Cp=1000;
      RR=1.987;
      dH=-5000;
 %
 %=================================================
 %
 % Definition of Q,R,C,al
     Q=1;
     R=0;
     al=0.999;
 %
 %=================================================
 %
 % Definition of the time interval for optimization
 %
     to=0;
     dt=5;
     trr=round(150/dt);
     tter=trr*dt;
     tgg=to:trr;
     tv=tgg*dt;
     tgg=trr:-1:to;
     tb=tgg*dt;
     [nd vn]=size(tv);
 %
 %=================================================
 %
 % Number of loops that the optimization will be performed
 %
    nl=vn;
    nloops=nl+100;
 %
 %=================================================
%
%  Steady State Plant for new Parameter:
i = 1;
T = 250.0;
while T <= 800.0
k1 = C1m*exp(-Q1m/RR/T);
         k2 = C2m*exp(-Q2m/RR/T);
```

```
A = (Aim*(1./tau + k2)+Bim*k2)/(1./tau + k1 + k2);
B = Aim - A + Bim;
ter = dH*tau*(k1*A - k2*B)/rho/Cp;
Ti = T + ter;
Bp(i) = B;
Tip(i) = Ti;
i = i + 1;
T = T + 1.0;
end
%
%=================================================
%
%  Steady State Plant for new Parameter:
i = 1;
T = 250.0;
while T <= 800.0
k1 = C1*exp(-Q1/RR/T);
k2 = C2*exp(-Q2/RR/T);
A = (Ai*(1./tau + k2)+Bi*k2)/(1./tau + k1 + k2);
B = Ai - A + Bi;
ter = dH*tau*(k1*A - k2*B)/rho/Cp;
Ti = T + ter;
Bps(i) = B;
Tips(i) = Ti;
i = i + 1;
T = T + 1.0;
end
%
%=================================================
%  Initial Conditions for the simulation
%
% select initial Ti
        T=460;
        k1 = C1*exp(-Q1/RR/T);
        k2 = C2*exp(-Q2/RR/T);
 A = (Ai*(1./tau + k2)+Bi*k2)/(1./tau + k1 + k2);
        B = Ai - A + Bi;
        ter = dH*tau*(k1*A - k2*B)/rho/Cp;
        Ti = T + ter;
        xi=[A;B;T];
 %
 %=================================================
 %
 % Initial values of the vector used to plot results
 %
    yp(1)=xi(2),
    dc(1)=0;
    ap(1)=xi(1);
    Tp(1)=xi(3);
    tp(1)=to;
    up(1)=Ti;
    options(2)=1e-3;
    options(3)=1e-3;
    options(14)=40.0;
 %
 %=================================================
 %
 % Desired output
```

```
%
      rorig=xi(2)*1.1,
%
%================================================
%
% Initial values of the input vector
%
   for i=1:vn
       u(i)=Ti;
   end,
   fcv=1;
%
%================================================
%
% Optimization loop
%
 for i=1:nloops
       rs=rorig-dc(i);
       uu=u;
       [tr,xr]=lsodar('fcstrcc',tv,xi);
       xx=xr;
       jn(i)=costcstr(xx),
       lfinal=[0;-al^(tv(vn))*Q*(2*xr(vn,2)-2*rs)/log(al);0];
       [tf,ldf]=lsodar('flamcstr',tb,lfinal);
       for kk=1:vn
          ld(kk,:)=ldf(vn+1-kk,:);
       end,
       g=fgcstr(ld,u);
       I1=0;
       gn=g;
       pp=-g;
       nf=vn-1;
       for io=2:nf,
           I1=I1+2*g(io)*g(io);
       end,
       I1=I1+g(1)*g(1)+g(vn)*g(vn);
       I1=dt*I1/2;
       jk=jn(i);
       af=100;
       afp=fmin('falfcstr',-10000,10000,options),
       for kk=1:vn
             unew(kk)=u(kk)+afp*pp(kk);
             if unew(kk)>umaxim
                    unew(kk)=umaxim;
             end,
             if unew(kk)<umin
                    unew(kk)=umin;
             end,
       end,
       uu=unew;
       [tr,xx]=lsodar('fcstrcc',tv,xi);
       jt=costcstr(xx),
       error=jn(i)-jt;
       kt=1;
       mult=1;
       if error>0
          u=uu;
       end,
       while error>0.001,
             g=gn;
```

```
        jn(i)=jt;
        u=uu;
        xr=xx;
        if i==1
            hold;
        end,
        lfinal=[0;-al^(tv(vn))*Q*(2*xr(vn,2)-2*rs)/log(al);0];
        [tf,ldf]=lsodar('flamcstr',tb,lfinal);
        for ik=1:vn
            ld(ik,:)=ldf(vn+1-ik,:);
        end,
        gn=fgcstr(ld,u);
        beta=fbeta(gn,g);
        pp=-gn+beta*pp;
        af=afp;
        afp=fmin('falfcstr',-10000,10000,options);
        for kk=1:vn
            unew(kk)=u(kk)+afp*fcv*pp(kk);
            if unew(kk)>umaxim
                    unew(kk)=umaxim;
            end,
            if unew(kk)<umin
                   unew(kk)=umin;
            end,
        end,
        uu=unew;
        [tr,xx]=lsodar('fcstrcc',tv,xi);
        jt=costcstr(xx),
        error=jn(i)-jt,
        kt=kt+1;
        if kt>10
            error=-1000;
        end,
end,
us=u(2);
tc=[to to+dt];
[tyy,xyy]=lsodar('fcstra',tc,xi);
ycc=xyy(2,2),
if i<=10
    [ty,xy]=lsodar('fcstra',tc,xi);
end,
if i>10
    [ty,xy]=lsodar('fmcstra',tc,xi);
end,
xi=[xy(2,1);xy(2,2);xy(2,3)];
to=to+dt;
tter=tter+dt;
tv=tv+ones(tv)*dt;
tb=tb+ones(tb)*dt;
tp(i+1)=to;
yp(i+1)=xi(2);
dc(i+1)=yp(i+1)-ycc;
yxr=xr(:,2);
ap(i+1)=xi(1);
Tp(i+1)=xi(3);
tp(i+1)=to;
up(i+1)=us;
for il=1:vn-1
    u(il)=u(il+1);
```

```
      end,
   end,
%
%==========================================
%
% Plot results
%
% ==========================================
%
%    Output
%
   plot(Tip,Bp,up,yp,Tips,Bps);
   ylabel('Concentration of B');
   xlabel('Inlet Temperature, K');
%
%
%  ======================
%
%    Save Results
%
   save test2
%
%  ==========================================
%
%     THE END
%
%  ==========================================
%
% This subroutine calculates the cost J
%
%   J=INT(al^(t)((Cx(t)-r(t))'Q(Cx(t)-r(t))+u(t)'Ru(t))))
%   -al^(tf)((Cx(tf)-r(tf))'Q(Cx(tf)-r(tf))+u(tf)'Ru(tf)))/ln(al)
%
% global Q,R,xi,tv,us,uu,n,al,r,umax,ks,umaxim,umin
%
   function J=costcstr(x)
   J=0;
   dt=tv(2)-tv(1);
   nf=vn-1;
   for i=2:nf,
     J=J+2*al^(tv(i))*((x(i,2)-rs)'*Q*(x(i,2)-rs)+uu(i)'*R*uu(i));
   end,
   J=J+al^(tv(1))*((x(1,2)-rs)'*Q*(x(1,2)-rs)+uu(1)'*R*uu(1));
   tmp=al^(tv(vn))*((x(vn,2)-rs)'*Q*(x(vn,2)-rs)+uu(vn)'*R*uu(vn));
   J=J+tmp;
   J=J*dt/2-tmp/log(al);
%  ==========================================
      function FX=fcstrcc(t,X)
%
%     System equations
%
      TAU=60.0;
      AI=1.0;
      CK=5000;
      CT=1e6;
      Q1=10000;
      Q2=15000;
      RHO=1;
      DH=-5000;
```

```
      RR=1.987;
      CP=1000;
   U=spline(tr,uu,t);
   FX(1) = (AI-X(1))/TAU+X(2)*CT*exp(-Q2/(X(3)*RR));
   FX(1) = FX(1)-X(1)*CK*exp(-Q1/(X(3)*RR));
   FX(2) = -X(2)/TAU-X(2)*CT*exp(-Q2/(X(3)*RR))+X(1)*..
   CK*exp(-Q1/(X(3)*RR));
   FX(3) = (U-X(3))/TAU;
 tw1   =-DH*(X(1)*CK*exp(-Q1/(X(3)*RR))-X(2)*CT..
   *exp(-Q2/(X(3)*RR)))/(CP*RHO);
    FX(3) = FX(3)+tw1;
 %  ==============================================
      function Fl=flamcstr(tx,ld)
%
%     Costate equations
%
      TAU=60.0;
      AI=1.0;
      BI=1-AI;
      C1=5000;
      C2=1e6;
      Q1=10000;
      Q2=15000;
      RHO=1;
      DH=-5000;
      RR=1.987;
      CP=1000;
   U=spline(tv,uu,t);
   x(1)=spline(tv,xr(:,1),tx);
   x(2)=spline(tv,xr(:,2),tx);
   x(3)=spline(tv,xr(:,3),tx);
r1=-1/TAU-C1*exp(-Q1/(RR*x(3)));
r2=C2*exp(-Q2/(RR*x(3)));
r3=-x(1)*C1*Q1*exp(-Q1/(RR*x(3)))/(RR*x(3)^2)+x(2)*C2..
*Q2*exp(-Q2/(RR*x(3)))/(RR*x(3)^2);
r4=C1*exp(-Q1/(RR*x(3)));
r5=-1/TAU-C2*exp(-Q2/(RR*x(3)));
r6=x(1)*C1*Q1*exp(-Q1/(RR*x(3)))/(RR*x(3)^2)-x(2)*C2..
*Q2*exp(-Q2/(RR*x(3)))/(RR*x(3)^2);
r7=-DH*C1*exp(-Q1/(RR*x(3)))/(RHO*CP);
r8=DH*C2*exp(-Q2/(RR*x(3)))/(RHO*CP);
r9=-DH*(x(1)*C1*Q1*exp(-Q1/(RR*x(3)))/(RR*x(3)^2)-x(2)*C2*Q2*..
exp(-Q2/(RR*x(3)))/(RR*x(3)^2))/(RHO*CP)-1/TAU;
df=[r1 r2 r3;r4 r5 r6;r7 r8 r9];
dl=[0;al^(tx)*Q*(2*x(2)-2*rs);0];
Fl=-df'*ld-dl;
%
%  ==============================================
%
function g=fgcstr(ld,u)
tau=60;
for kk=1:vn
    g(kk)=ld(kk,3)/tau+2*al^(tv(kk))*R*u(kk);
end,
%  ==============================================
 function hv=falfcstr(afg)
```

```
%
%    The following is assumed to be global:
%
%          Q,R,xi,tv,xr,us,uu,n,al,r,umaxim,umin,pp
%
%    Calculate the new input
%
%       check if u+alfa is in the feasible set otherwise set
%       alfa to zero
%
    for kk=1:vn
          unew(kk)=uu(kk)+afg*pp(kk);
         if unew(kk)>umaxim
                unew(kk)=umaxim;
          end,
          if unew(kk)<umin
                unew(kk)=umin;
          end,
    end,
    uold=uu;
    uu=unew;
    [tr,xx]=lsodar('fcstrcc',tv,xi);
    hv=costcstr(xx);
    uu=uold;
% ================================================
   function hv=fbeta(gn,g)
%
%    The following is assumed to be global:
%
%          Q,R,xi,tv,xr,us,uu,n,al,r,umax,ks,umaxim,umin,p
%
%    Calculate the new input
%
%       check if u+alfa is in the feasible set otherwise set
%       alfa to zero
%
%
   dt=tv(2)-tv(1);
   nf=vn-1;
   I1=0;
   I2=0;
   I3=0;
   for i=2:nf,
       I1=I1+2*gn(i)*gn(i);
       I2=I2+2*g(i)*g(i);
       I3=I3+2*gn(i)*pp(i);
   end,
   I1=I1+gn(1)*gn(1)+gn(vn)*gn(vn);
   I1=dt*I1/2;
   I2=I2+g(1)*g(1)+g(vn)*g(vn);
   I2=dt*I2/2;
   I3=I3+gn(1)*pp(1)+pp(vn)*gn(vn);
   I3=dt*I3/2;
   if I2 ~= 0
        Bo=I1/I2;
   end,
   if I3 ~= 0
        Bm=I1/I3;
   end,
```

```
if I2 == 0
      Bo=0;
      hv=0;
end,
if I3 == 0
      Bm=0;
      hv=0;
end,
if Bo<Bm
   if I2~=0
      hv=Bo;
   end,
end,
if Bo>=Bm
   if I3 ~=0
      hv=Bm/2;
   end,
end,
```

APPENDIX C

Simulation Programs for the Catalytic Reactor

C.1 ELCC

This program simulates the response of the catalytic reactor introduced in Section 4.3 when the ELCC is used as a controller

```
%=========================================
%
%      DMC + Linearization + Minimization
%
%=========================================
%
%  This program solves the Dynamic Matrix Control
%  problem then it uses input-output linearization
%  to transform the calculated inputs to the original
%  inputs of the nonlinear system then minimizes the
%  distance between the linearized inputs and the real
%  inputs.
%
%=========================================
%
%             Solution to the Dynamic Matrix Control
%
%                     v(k)=inv(A'A)*A'e(k+1)
%
%=========================================
%
%  clear variables
    clear;
%
   global us k1m k2m cb1m cb2m;
%
%=========================================
%
%  Nonlinear Plant
%     .                          0.5
%     x(1) = u(1)+u(2)-0.2 x(1)
%
%     .
%     x(2) = (CB1-x(2))u(1)   (CB2-x(2))u(2)     k1x(2)
%            ------------- + ------------- - ---------- 2
%                  x(1)           x(1)        (1+k2x(2))
%
%     y(1)=x(1)
%
%     y(2)=x(2)
%
      umax1=100;  % Maximum input1
%
```

```
      umin1=0.05;     % Minimum input1
%
      umax2=100;  % Maximum input1
%
      umin2=0.05;     % Minimum input1
%
%     Where:
%
      k1=1.0;
      k1m=k1*1.2;
%
      k2=1.0;
      k2m=k2;
%
      CB1=24.9;
      cb1m=CB1;
%
      CB2=0.1;
      cb2m=CB2;
%
%=========================================
%
%  Relative Degree of the system 1
%
%=========================================
%
%  Linearized  plant 1 = Linearized plant 2:
%
%             .
%             x = H x + B u
%
%             y = C x
%
     H=[-1];
     B=[1];
     C=[1];
     ni=1;           % Number of inputs
     no=1;           % Number of outputs
%=========================================
%
%  Control Parameters
%
    dt=1;           % Sampling time in minutes
    m=4;            % Number of time intervals
                    % used to describe the system
    n=4;            % Number of future input moves to be computed
    np=4;           % Predicted horizon (generally n+m)
    hor=20;
%
%=========================================
%
%  Initial Conditions for the linearized model
%
     x=[0];
     ud=1.0;
     t(1)=0.0;
     y(1)=0;
%
%=========================================
```

```
%
%  Discretization
%
%    Discrete sytem:
%
%          x((k+1)dt)=Ad x((k)dt) + Bd u((k)dt)
%          y((k)dt)= C x((k)dt)
%
     em=expm([dt*H dt*B; zeros(no,ni+no)]);
     Ad=em(1:ni,1:ni);
     Bd=em(1:no,(ni+1):(ni+no));
%
%=============================================
%
%  System Step response
%
 for i=2:m+1
 t(i)=dt*(i-1);
        x=Ad*x+Bd*ud;
 y(i)=C*x;
 end,
%
% The values of y from m to np are taken as y(m)
%
 if np>m
tm=m+1;
for i=tm:np
y(i)=y(m);
        end,
 end,
 y(np+1)=y(np);
%
%================================================
%
%  Construction of the dynamics matrix A
%
    A=zeros(np,n);
    WU=0.7*eye(np,n);
 for i=1:n
 l=1;
 for j=i:np
                l=l+1;
 A(j,i)=y(l);
        end,
 end,
 A=[A;WU];
%
%================================================
%
%  Calculation of the inverse of the dynamic matrix
%  and save only the first column
 dv=A(1:np,1);
 p1=zeros(np,1);
 wr=inv(A'*A)*A';
 aiv=wr(1,:);
 p2=zeros(np,1);
%
%================================================
```

```
%  Initial Conditions for the simulation
%
% select initial h and cb
          hss=40;
          cbss=0.1;
          co1=0.2*hss^0.5;
          co2=(CB1-cbss)/hss;
          co3=(CB2-cbss)/hss;
          co4=k1*cbss/((1+k2*cbss)^2);
          u2ss=(co4-co2*co1)/(co3-co2);
          u1ss=co1-u2ss;
 u1=0;
 u2=0;
 U=[u1ss;u2ss];
 yr1 = dv(1)*u1;
 yr2 = dv(1)*u2;
 t1(1) = 0.0;
 xx1=0;
 xx2=0;
 x= [hss;cbss];
 xim=x;
 yy1(1)=x(1);
 yy2(1)=x(2);
 hh1=yy1(1);
 hh2=yy2(1);
 lgh=[1 1;(CB1-cbss)/hss (CB2-cbss)/hss];
 lfh=[-0.2*hss^0.5;-k1*cbss/((1+k2*cbss)^2)];
 hy=[yy1(1);yy2(1)];
 vt=lgh*U+lfh+hy;
 vv1(1)=vt(1);
 vv2(1)=vt(2);
 y1(1) = 0;
 v1(1)=0;
 v2(1)=0;
 uu1(1)=U(1);
 uu2(1)=U(2);
%
%=============================================
%
% Desired outpu or desired trajectory
%
   for i=1: hor+hor
         r1(i)=yy1(1)-yy1(1);
         r2(i)=yy2(1)-yy2(1);
         %r1(i)=100-yy1(1);
         %r2(i)=2.787-yy2(1);
   end,
%
%=========================================
%
%
%    Parameters for the optimization
%
     N=[1 0;-1 0;0 1;0 -1];
     C=[umax1;-umin1;umax2;-umin2];
     Q=1;
% simulation for the interval np
```

```
%
  rand('normal');
  for mk = 2:hor
          qq1=0;
          po1 = p1(1)+dv(1)*u1;
          po2 = p2(1)+dv(1)*u2;
          d1 = yr1 - po1;
          d2 = yr2 - po2;
          for i=1:np-1
                p1(i) = p1(i+1) + dv(i+1)*u1 + d1;
          end,
          p1(np) = p1(np-1);
          for i=1:np-1
                p2(i) = p2(i+1) + dv(i+1)*u2 + d2;
          end,
          p2(np) = p2(np-1);
          ll=mk-1;
          for i =1:np
              e1(i) = r1(ll+i) - p1(i);
          end
          u1=0;
          for i =1:np
              e2(i) = r2(ll+i) - p2(i);
          end
          u2=0;
          for i=1:m
                u1=u1+aiv(i)*e1(i);
          end,
          for i=1:m
                u2=u2+aiv(i)*e2(i);
          end,
          v1(mk)=v1(mk-1)+u1;
          vv1(mk)=vv1(mk-1)+u1;
          v2(mk)=v2(mk-1)+u2;
          vv2(mk)=vv2(mk-1)+u2;
          vt=[vv1(mk);vv2(mk)];
          lgh=[1 1;(CB1-x(2))/x(1) (CB2-x(2))/x(1)];
          lfh=[-0.2*x(1)^0.5;-k1*x(2)/((1+k2*x(2))^2)];
          hy=[x(1);x(2)];
          us=inv(lgh)*(vt-lfh-hy);%+rand(2,1);
          if us(1)<umin1,
            qq1=1;
          end,
          if us(1)>umax1,
             qq1=1;
          end,
          if us(2)<umin2,
            qq1=1;
          end,
          if us(2)>umax2,
             qq1=1;
          end,
        if qq1==1
            P=N*inv(lgh'*Q*lgh)*N';
            bv=lgh'*Q*([vv1(mk);vv2(mk)]-lfh-hy);
            D=C-N*inv(lgh'*Q*lgh)*bv;
            gamma=[0;0;0;0];
            gamman=[0;0;0;0];
```

```
    J=gamma'*P*gamma/2+gamma'*D;
    for ii=1:4
         sum=0;
         for jj=1:4
             if jj < ii
                 sum=sum+P(ii,jj)*gamman(jj);
             end,
             if jj > ii
                 sum=sum+P(ii+1,jj)*gamma(jj);
             end,
         end,
         wv(ii)=-(D(ii)+sum)/P(ii,ii);
         gamman(ii)=max(0,wv(ii));
    end,
    Jn=gamman'*P*gamman/2+gamman'*D;
    err=J-Jn;
    while err>0
          gamma=gamman;
          J=Jn;
          for ii=1:4
               sum=0;
               for jj=1:4
                   if jj < ii
                      sum=sum+P(ii,jj)*gamman(jj);
                   end,
                   if jj > ii
                      sum=sum+P(ii+1,jj)*gamma(jj);
                   end,
               end,
               wv(ii)=-(D(ii)+sum)/P(ii,ii);
               gamman(ii)=max(0,wv(ii));
          end,
          Jn=gamman'*P*gamman/2+gamman'*D;
          err=J-Jn;
    end,
    us=inv(lgh'*Q*lgh)*(bv-N'*gamma),
end,
  vt=lgh*us+lfh+hy;
  vv1(mk)=vt(1);
  vv2(mk)=vt(2);
  u1=vv1(mk)-vv1(mk-1);
  u2=vv2(mk)-vv2(mk-1);
  uu1(mk)=us(1);
  uu2(mk)=us(2);
  tl(mk)=dt*(mk-1);
  ti=[tl(mk-1) tl(mk)];
  xi=x;
  if mk<=5
       [tr,xr]=lsodar('mimoeq',ti,xi);
       x=[xr(2,1);xr(2,2)];
       yy1(mk)=x(1);
       yy2(mk)=x(2);
  end,
  if mk>5
       [tr,xr]=lsodar('mimoeqm',ti,xi);
       x=[xr(2,1);xr(2,2)];
       yy1(mk)=x(1);
       yy2(mk)=x(2);
```

```
          end,
          yr1=yy1(mk)-yy1(1);
          yr2=yy2(mk)-yy2(1);
  end,
%
%================================================
%
% Plot results
%
clg
subplot(121)
plot(tl,yy1)
grid
xlabel('Time (min)')
ylabel('Liquid Height')
plot(tl,yy2)
grid
xlabel('Time (min)')
ylabel('Conecentration of B')
clg
subplot(121)
plot(tl,uu1)
grid
xlabel('Time (min)')
ylabel('Flow rate with concentrated B')
plot(tl,uu2)
grid
xlabel('Time (min)')
ylabel('Flow rate with dilute B')
%
%================================================
%
%                    THE END
%
%================================================
%
      function FX=mimoeq(t,X)
      k1=1.0;
      k2=1.0;
      cb1=24.9;
      cb2=0.1;
      U(1)=us(1);
      U(2)=us(2);
      FX(1) = U(1) + U(2) - 0.2*X(1)^0.5;
      FX(2)=(cb1-X(2))*U(1)/X(1)+(cb2-X(2))*U(2)/X(1)..
            -k1*X(2)/((1+k2*X(2))^2);
%================================================
%
      function FX=mimoeqm(t,X)
      k1=k1m;
      k2=k2m;
      rand('normal')
      cb1=cb1m;
      cb2=cb2m;
      U(1)=us(1);
      U(2)=us(2);
      FX(1) = U(1) + U(2) - 0.2*X(1)^0.5;
```

```
      FX(2)=(cb1-X(2))*U(1)/X(1)+(cb2-X(2))*U(2)/X(1)..
            -k1*X(2)/((1+k2*X(2))^2);
%
%=========================================
```

C.2 Nonlinear Controller with Global Linearization

This program simulates the response of the catalytic reactor introduced in Section 4.3 when a nonlinear controller that uses a global linearization technique is used as a controller

```
%=======================================================
%
%    This program uses the conjugate gradient to solve
%    the  following problem:
%
%                          tf
%    J(u)=M[x(tf),tf]+  S  l(x,u)dt
%                          to
%    subject to:
%
%               .
%               x=f(x,u); x(to)=xo
%
%               umin<=uj<=umaxim
%
%=====================================================
%
% The following functions are specified:
%
%    cost (t,x)       : Calculates the index of performance
%
%    fmimo(t,x)       : System equations
%
%                                          .   -dH
%    flambda(t,ld)    : Costate equation ld= ----
%                                            dx
%
%
%                         dH
%    fg(ld,x,u)       :  ----
%                         du
%
%    falf(af)         : Use to find the alfa that minimizes
%                       J(u+alf*p)
%
%    fbeta(gn,gold)   : Determine beta
%
%=========================================================
%
   clear,
%  The following variables are assumed to be global
%
   global Q  R xi tv;
   global to tter dt xr us uu vn al rs umax1 umin1;
   global umax2 umin2 pp jk fcv k1m k2m cb1m cb2m;
 %
```

```
%=========================================================
%  Nonlinear Plant
%                                  0.5
%     .
%     x(1) = u(1)+u(2)-0.2 x(1)
%
%
%     .
%     x(2) = (CB1-x(2))u(1)  (CB2-x(2))u(2)     k1x(2)
%            ------------- + ------------- -  ---------- 2
%                  x(1)          x(1)          (1+k2x(2))
%
%     y(1)=x(1)
%     y(2)=x(2)
%
      umax1=20;  % Maximum input1
%
      umin1=0.05;    % Minimum input1
%
      umax2=20;  % Maximum input1
%
      umin2=0.05;    % Minimum input1
%
% Process parameters
%
%
%
      k1=1.0;
      k1m=k1;
%
      k2=1.0;
      k2m=k2;
%
      CB1=24.9;
      cb1m=CB1;
%
      CB2=0.1;
      cb2m=CB2;
%
%=====================================================
%
% Definition of Q,R,C,al
%
    Q=[0.1 0;0 2];
    R=0;
    al=0.99;
%
%=====================================================
%
% Definition of the time interval for optimization
%
    to=0;
    dt=1;
    trr=round(10/dt);
    tter=trr*dt;
    tgg=to:trr;
    tv=tgg*dt;
    tgg=trr:-1:to;
    tb=tgg*dt;
    [nd vn]=size(tv);
```

```
%
%=====================================================
%
% Number of loops that the optimization will be performed
%
   nl=vn;
   nloops=nl+80;
%
%=====================================================
%  Initial Conditions for the simulation
%
% select initial h and cb
        hss=40;
        cbss=0.1;
        co1=0.2*hss^0.5;
        co2=(CB1-cbss)/hss;
        co3=(CB2-cbss)/hss;
        co4=k1*cbss/((1+k2*cbss)^2);
        u2ss=(co4-co2*co1)/(co3-co2);
        u1ss=co1-u2ss;
        xi=[hss;cbss];
%
%=====================================================
%
% Initial values of the vector used to plot results
%
   yp1(1)=xi(1),
   yp2(1)=xi(2),
   dc1(1)=0;
   dc2(1)=0;
   tp(1)=to;
   up1(1)=u1ss;
   up2(1)=u2ss;
   options(2)=1e-4;
   options(3)=1e-4;
   options(14)=30.0;
%
%=====================================================
%
% Desired output
%
      rorig1=100.00;
      rorig2=2.7927;
%
%=====================================================
%
% Initial values of the input vector
%
   for i=1:vn
       u(i,1)=u1ss;
       u(i,2)=u2ss;
   end,
   fcv=1;
%
%=====================================================
%
% Optimization loop
%
```

```
for i=1:nloops
     rs=[rorig1;rorig2];
     uu=u;
     [tr,xr]=lsodar('fmimo',tv,xi);
     xx=xr;
     jn(i)=cost(xx),
     lfinal=[-al^(tv(vn))*Q(1,1)*(2*xr(vn,1)-2*rs(1))/log(al)..
      ;-al^(tv(vn))*Q(2,2)*(2*xr(vn,2)-2*rs(2))/log(al)],
     [tf,ldf]=lsodar('flambda',tb,lfinal);
     for kk=1:vn
        ld(kk,:)=ldf(vn+1-kk,:);
     end,
     g=fg(ld,xr,u);
     I1=0;
     gn=g;
     pp=-g;
     afp=fmin('falf',-100,100,options),
     for kk=1:vn
          ppd=[pp(kk,1);pp(kk,2)];
          ud=[uu(kk,1);uu(kk,2)];
          udnew=ud+afp*fcv*ppd;
          if udnew(1)>umax1
                  udnew(1)=umax1;
          end,
          if udnew(2)>umax2
                 udnew(2)=umax2;
          end,
          if udnew(1)<umin1
                   udnew(1)=umin1;
          end,
          if udnew(2)<umin2
                 udnew(2)=umin2;
          end,
          unew(kk,1)=udnew(1);
          unew(kk,2)=udnew(2);
     end,
     uu=unew;
     [tr,xx]=lsodar('fmimo',tv,xi);
     jt=cost(xx),
     error=jn(i)-jt;
     kt=1;
     mult=1;
     if error>0
        u=uu;
     end,
     while error>0.001,
       g=gn;
       jn(i)=jt;
       u=uu;
       xr=xx;
     lfinal=[-al^(tv(vn))*Q(1,1)*(2*xr(vn,1)-2*rs(1))/log(al)..
          ;-al^(tv(vn))*Q(2,2)*(2*xr(vn,2)-2*rs(2))/log(al)];
            [tf,ldf]=lsodar('flambda',tb,lfinal);
            for ik=1:vn
                ld(ik,:)=ldf(vn+1-ik,:);
            end,
            gn=fg(ld,xr,u);
            beta=fbeta(gn,g),
            pp=-gn+beta*pp;
            afo=afp;
```

```
      % afp=fmin('falf',-100,100,options);
        afp=fmins('falf',afo,options);
        for kk=1:vn
            ppd=[pp(kk,1);pp(kk,2)];
            ud=[uu(kk,1);uu(kk,2)];
            udnew=ud+afp*fcv*ppd;
            if udnew(1)>umax1
                  udnew(1)=umax1;
            end,
            if udnew(2)>umax2
                  udnew(2)=umax2;
            end,
            if udnew(1)<umin1
                  udnew(1)=umin1;
            end,
            if udnew(2)<umin2
                  udnew(2)=umin2;
            end,
            unew(kk,1)=udnew(1);
            unew(kk,2)=udnew(2);
        end,
        uu=unew;
        [tr,xx]=lsodar('fmimo',tv,xi);
        jt=cost(xx),
        error=jn(i)-jt,
        kt=kt+1;
        if kt>20
            error=-1000;
        end,
end,
us(1)=u(1,1);
us(2)=u(1,2);
tc=[to to+dt];
[tyy,xyy]=lsodar('mimoeq',tc,xi);
ycc1=xyy(2,1);
ycc2=xyy(2,2);
if i<=10
    [ty,xy]=lsodar('mimoeq',tc,xi);
end,
if i>10
    [ty,xy]=lsodar('mimoeqm',tc,xi);
end,
xi=[xy(2,1);xy(2,2)];
to=to+dt;
tter=tter+dt;
tv=tv+ones(tv)*dt;
tb=tb+ones(tb)*dt;
tp(i+1)=to;
yp1(i+1)=xi(1),
yp2(i+1)=xi(2),
dc1(i+1)=yp1(i+1)-ycc1;
dc2(i+1)=yp2(i+1)-ycc2;
tp(i+1)=to;
up1(i+1)=us(1);
up2(i+1)=us(2);
for il=1:vn-1
    u(il,1)=u(il+1,1);
    u(il,2)=u(il+1,2);
```

```
      end,
      save test2
   end,
%
%=====================================================
%
% Plot results
%
% ========================
%
%    Output
%
   subplot(121);
   plot(tp,yp1);
   grid;
   ylabel('Liquid Heigth');
   xlabel('Time (min)');
   plot(tp,yp2);
   grid;
   xlabel('Time (min)');
   ylabel('Concentration of B');
   clg
   subplot(121);
   plot(tp,up1);
   grid;
   ylabel('Flow Rate of Concentrated B');
   xlabel('Time (min)');
   plot(tp,up2);
   grid;
   xlabel('Time (min)');
   ylabel('Flow Rate of Dilute B');
%
%
%  ========================
%
%    Save Results
%
   save test2
%
%  ================================================
%
%     THE END
%
%  ================================================
%
% This subroutine calculates the cost J
%
%   J=INT(al^(t)((Cx(t)-r(t))'Q(Cx(t)-r(t))+u(t)'Ru(t))))
%  -al^(tf)((Cx(tf)-r(tf))'Q(Cx(tf)-r(tf))+u(tf)'Ru(tf)))/ln(al)
%
% global Q,R,xi,tv,us,uu,n,al,r,umax,ks,umaxim,umin
%
   function J=cost(x)
   J=0;
   dt=tv(2)-tv(1);
   nf=vn-1;
   for i=2:nf,
       xd=[x(i,1);x(i,2)];
```

```
        ud=[uu(i,1);uu(i,2)];
        J=J+2*al^(tv(i))*((xd-rs)'*Q*(xd-rs)+ud'*R*ud);
    end,
    xd=[x(1,1);x(1,2)];
    ud=[uu(1,1);uu(1,2)];
    J=J+al^(tv(1))*((xd-rs)'*Q*(xd-rs)+ud'*R*ud);
    xd=[x(vn,1);x(vn,2)];
    ud=[uu(vn,1);uu(vn,2)];
    tmp=al^(tv(vn))*((xd-rs)'*Q*(xd-rs)+ud'*R*ud);
    J=J+tmp;
    J=J*dt/2-tmp/log(al);
%
%   ================================================
%
      function FX=fmimo(t,X)
      k1=1.0;
      k2=1.0;
      cb1=24.9;
      cb2=0.1;
%
%     Loop to find the value of U that should be use in the integration
%
      U(1)=spline(tv,uu(:,1),t);
      U(2)=spline(tv,uu(:,2),t);
 FX(1) = U(1) + U(2) - 0.2*X(1)^0.5;
 FX(2)=(cb1-X(2))*U(1)/X(1)+(cb2-X(2))*U(2)/X(1);
 FX(2)=FX(2)-k1*X(2)/((1+k2*X(2))^2);
%
%   ================================================
%
      function Fl=flambda(tx,ld)
%
%     Costate equations
%
      CB1=24.9;
      CB2=0.1;
      k1=1;
      k2=1;
      U(1)=spline(tv,uu(:,1),tx);
      U(2)=spline(tv,uu(:,2),tx);
      x(1)=spline(tv,xr(k,1),tx);
      x(2)=spline(tv,xr(k,2),tx);
      dr1=-0.1*x(1)^(-0.5);
      dr2=0;
dr3=-(CB1-x(2))*U(1)/(x(1)*x(1))-(CB2-x(2))*U(2)/(x(1)*x(1));
dr4=-U(1)/x(1)-U(2)/x(1)+(k1*k2*x(2)-k1)/((1+k2*x(2))^3);
df=[dr1 dr2;dr3 dr4];
dl=2*al^(tx)*(Q*x'-Q*rs)
Fl=-df*ld-dl;
%
%   ================================================
%
function g=fg(ld,x,u)
cb1=24.9;
cb2=0.1;
for kk=1:vn
      xd=[x(kk,1);x(kk,2)];
      ud=[uu(kk,1);uu(kk,2)];
```

```
      ldd=[ld(kk,1);ld(kk,2)];
gd=[1 1;(cb1-xd(2))/xd(1) (cb2-xd(2))/xd(1)]'*ldd+2*al^(tv(kk))*R*ud;
    g(kk,1)=gd(1);
    g(kk,2)=gd(2);
end,
%
%  ================================================
%
function hv=falf(afg)
%
%    The following is assumed to be global:
%
%          Q,R,xi,tv,xr,us,uu,n,al,r,umaxim,umin,pp
%
%    Calculate the new input
%
%       check if u+alfa is in the feasible set otherwise set
%       alfa to zero
%
%
    fcv=0.0001;
    for kk=1:vn
          ppd=[pp(kk,1);pp(kk,2)];
          ud=[uu(kk,1);uu(kk,2)];
          udnew=ud+afg*fcv*ppd;
          if udnew(1)>umax1
                udnew(1)=umax1;
          end,
          if udnew(2)>umax2
                udnew(2)=umax2;
          end,
          if udnew(1)<umin1
                 udnew(1)=umin1;
          end,
          if udnew(2)<umin2
                udnew(2)=umin2;
          end,
          unew(kk,1)=udnew(1);
          unew(kk,2)=udnew(2);
    end,
    uold=uu;
    uu=unew;
    [tr,xx]=lsodar('fmimo',tv,xi);
    hv=cost(xx),
    uu=uold;
%
%  ================================================
%
  function hv=fbeta(gn,g)
%
%    The following is assumed to be global:
%
%          Q,R,xi,tv,xr,us,uu,n,al,r,umax,ks,umaxim,umin,p
%
%    Calculate the new input
%
%       check if u+alfa is in the feasible set otherwise set
%       alfa to zero
```

```
%
%
   dt=tv(2)-tv(1);
   nf=vn-1;
   I1=0;
   I2=0;
   I3=0;
   for i=2:nf,
       gd=[g(i,1);g(i,2)];
       gnd=[gn(i,1);gn(i,2)];
       ppd=[pp(i,1);pp(i,2)];
       I1=I1+2*gnd'*gnd;
       I2=I2+2*gd'*gd;
       I3=I3+2*gnd'*ppd;
   end,
   gd1=[g(1,1);g(1,2)];
   gnd1=[gn(1,1);gn(1,2)];
   ppd1=[pp(1,1);pp(1,2)];
   gdn=[g(vn,1);g(vn,2)];
   gndn=[gn(vn,1);gn(vn,2)];
   ppdn=[pp(vn,1);pp(vn,2)];
   I1=I1+gnd1'*gnd1+gndn'*gndn;
   I1=dt*I1/2;
   I2=I2+gd1'*gd1+gdn'*gdn;
   I2=dt*I2/2;
   I3=I3+gnd1'*ppd1+gndn'*ppdn;
   I3=dt*I3/2;
   if I2 ~= 0
         Bo=I1/I2;
   end,
   if I3 ~= 0
         Bm=I1/I3;
   end,
   if I2 == 0
         Bo=0;
         hv=0;
   end,
   if I3 == 0
         Bm=0;
         hv=0;
   end,
   if Bo<Bm
      if I2~=0
         hv=Bo;
      end,
   end,
   if Bo>=Bm
      if I3 ~=0
         hv=Bm/2;
      end,
   end,
%
%  ================================================
%
```

APPENDIX D

Simulation Results

D.1 CSTBR

D.1.1 ELCC: Nominal Performance

Nominal system response for a change in the desired set point by 50%

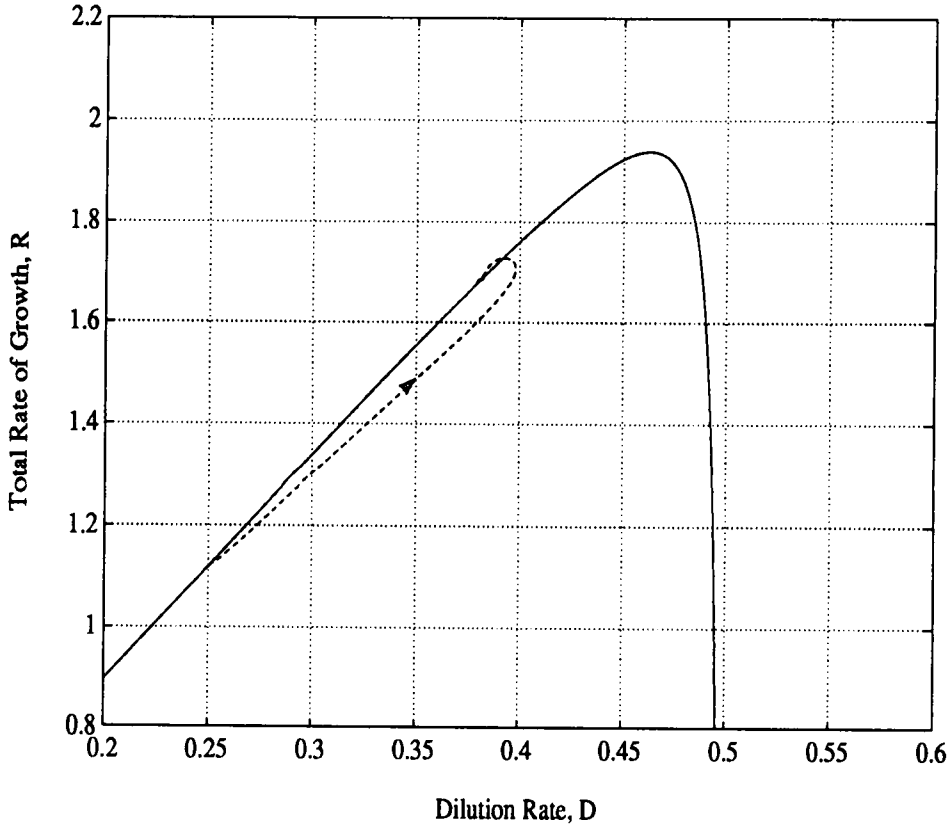

Figure D.1: Set point change in the total rate of growth by 50%. System trajectory

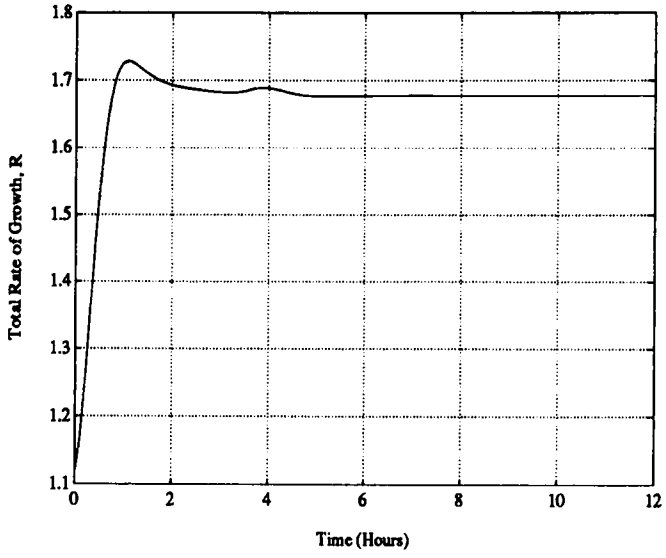

Figure D.2: Set point change in the total rate of growth by 50%. Output versus time

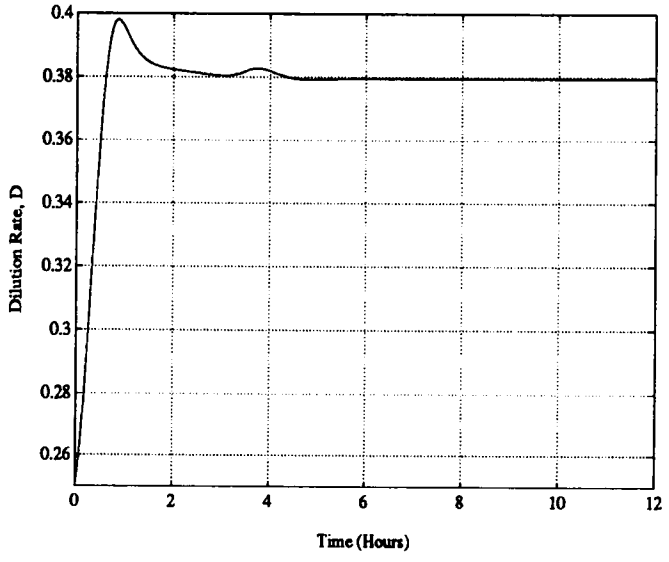

Figure D.3: Set point change in the total rate of growth by 50%. Input versus time

Nominal system response for a change in the desired set point by 50%

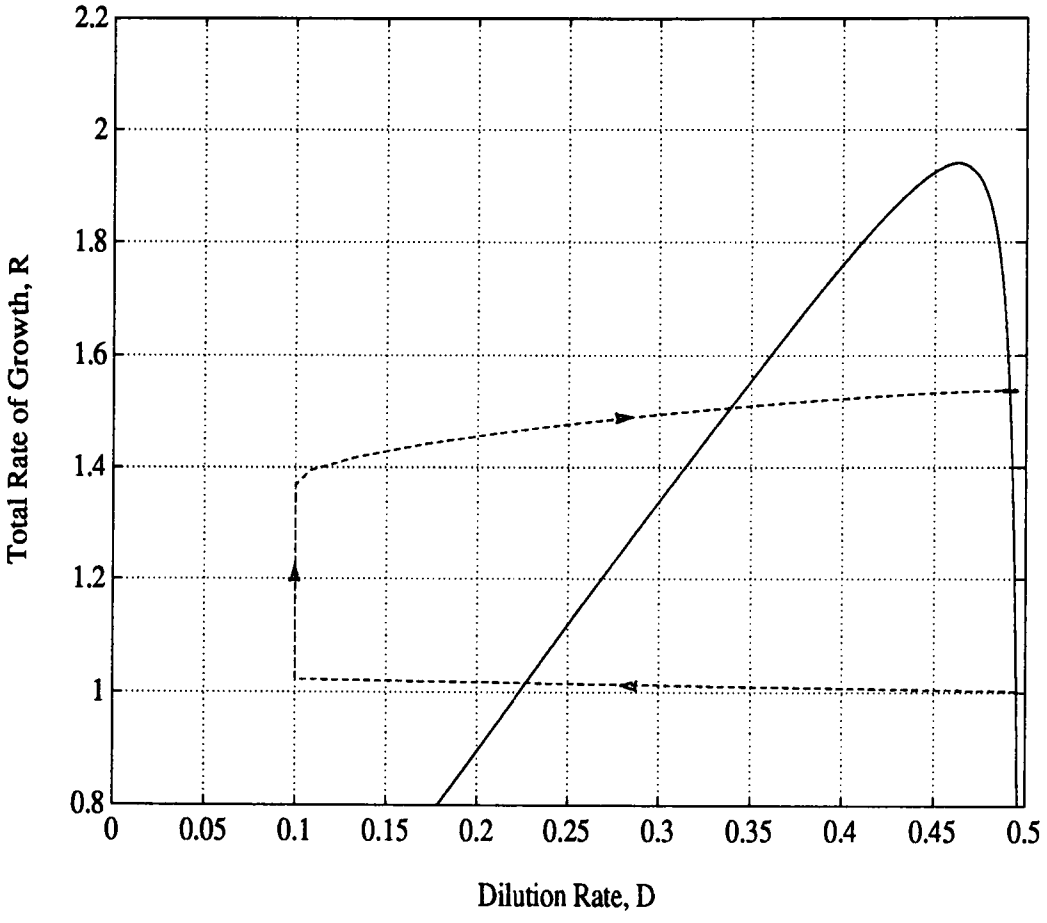

Figure D.4: Set point change in the total rate of growth by 50% with initial dilution rate higher than D_{opt}. System trajectory.

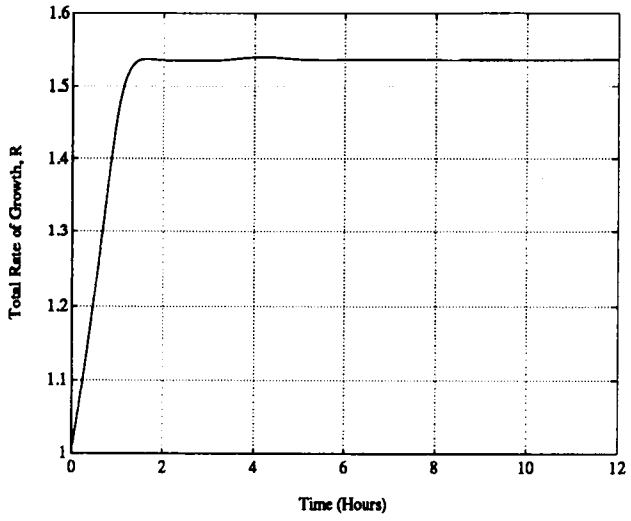

Figure D.5: Set point change in the total rate of growth by 50% with initial dilution rate higher than D_{opt}. Output versus time

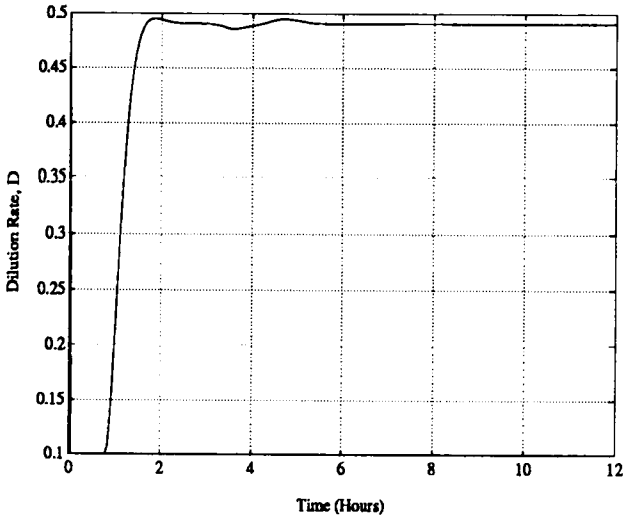

Figure D.6: Set point change in the total rate of growth by 50% with initial dilution rate higher than D_{opt}. Input versus time

D.1.2 ELCC: Parameter Variations

System response for a change in y_s by -12% with an initial dilution rate higher than D_{opt}.

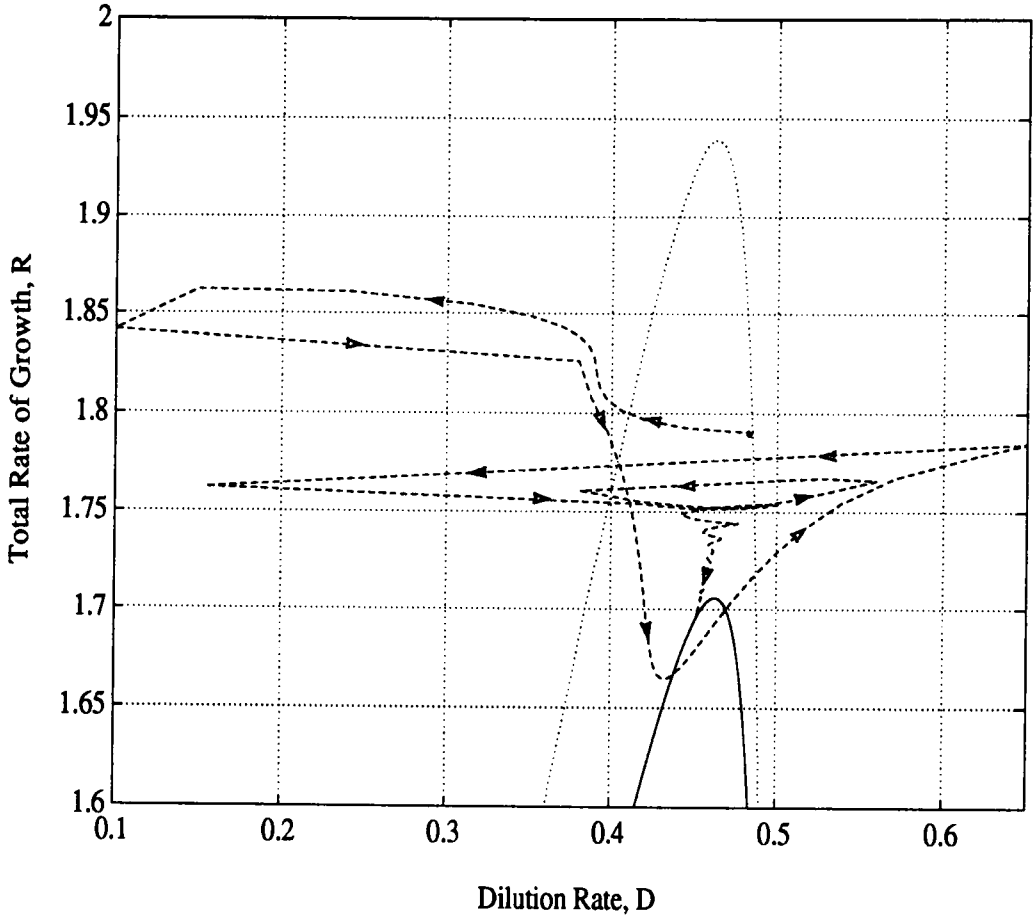

Figure D.7: Change in y_s by -12%. Unachievable set point. System trajectory

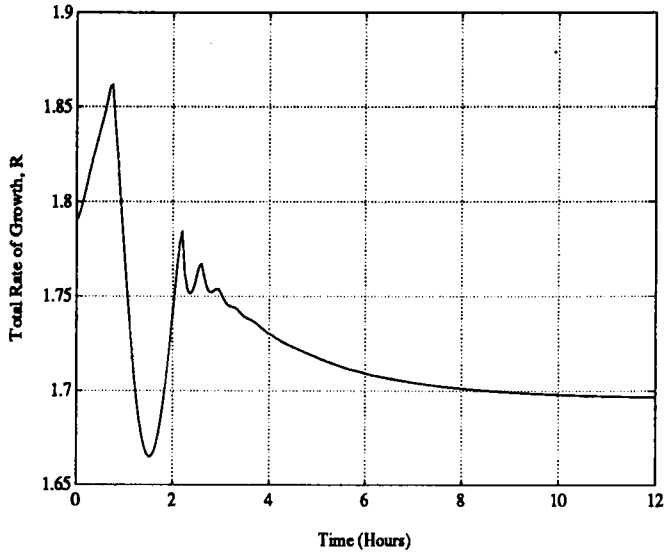

Figure D.8: Change in y_s by -12%. Unachievable set point. Output versus time

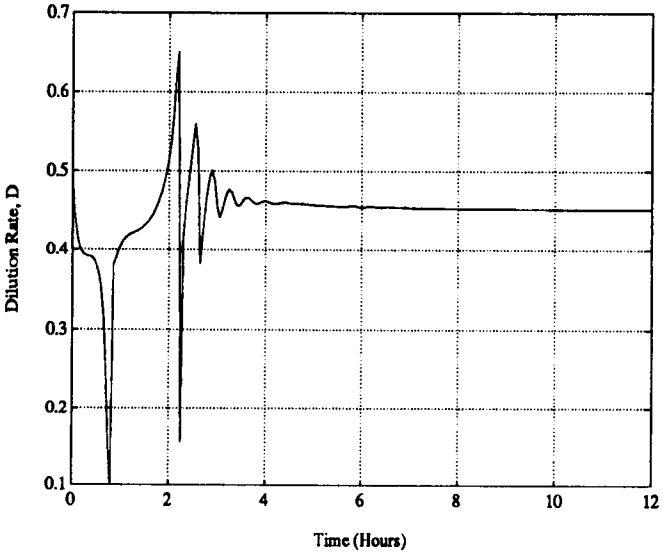

Figure D.9: Change in y_s by -12%. Unachievable set point. Input versus time

System response for a change in y_s by +10% with an initial dilution rate lower than D_{opt}.

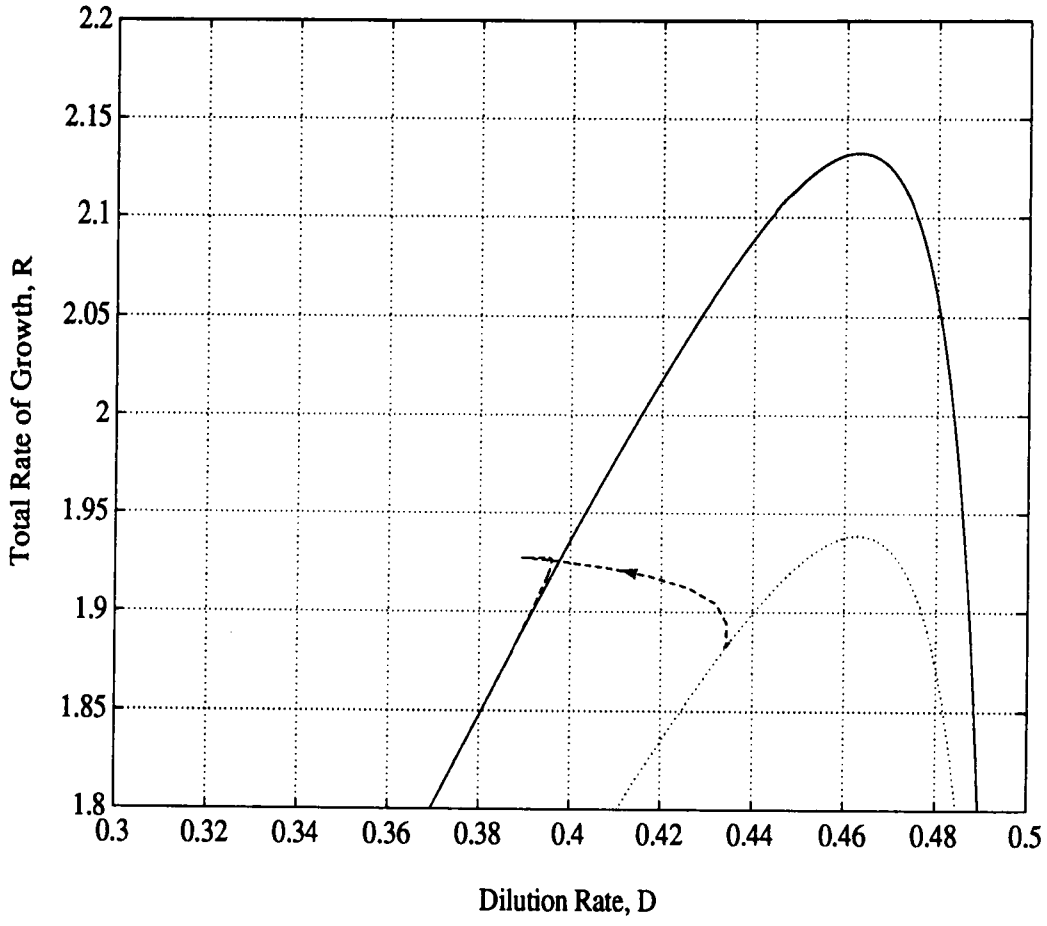

Figure D.10: Change in y_s by +10%. Achievable set point. System trajectory

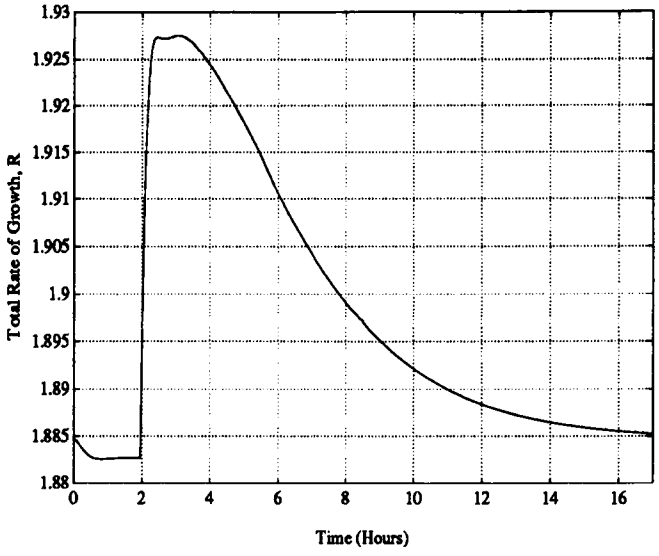

Figure D.11: Change in y_s by +10%. Achievable set point. Output versus time

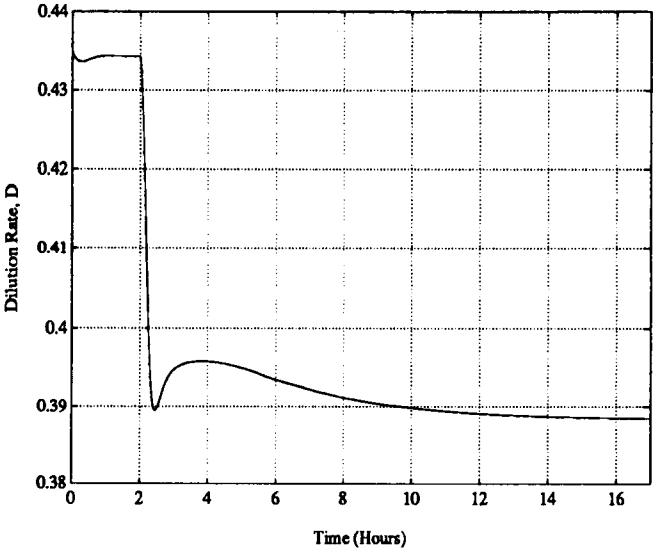

Figure D.12: Change in y_s by +10%. Achievable set point. Input versus time

System response for a change in μ_{max} by -12%

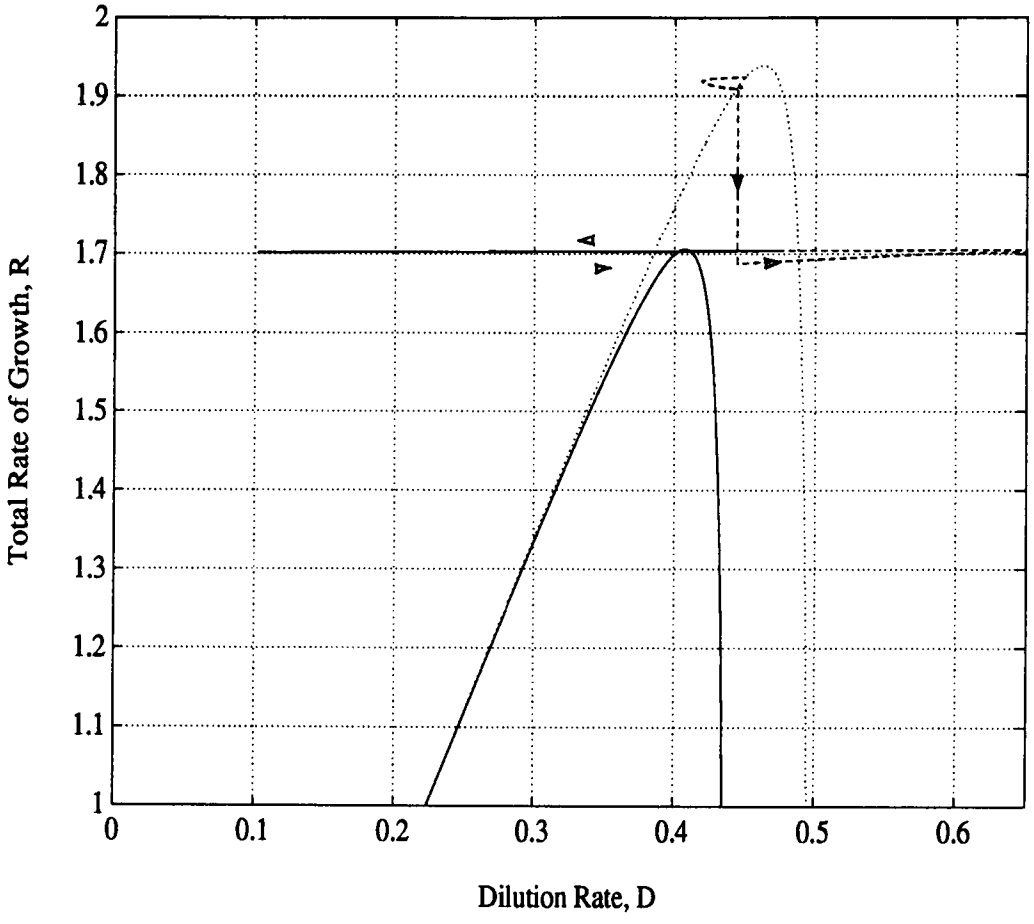

Figure D.13: Change in μ_{max} by -12%. Unachievable set point. System trajectory

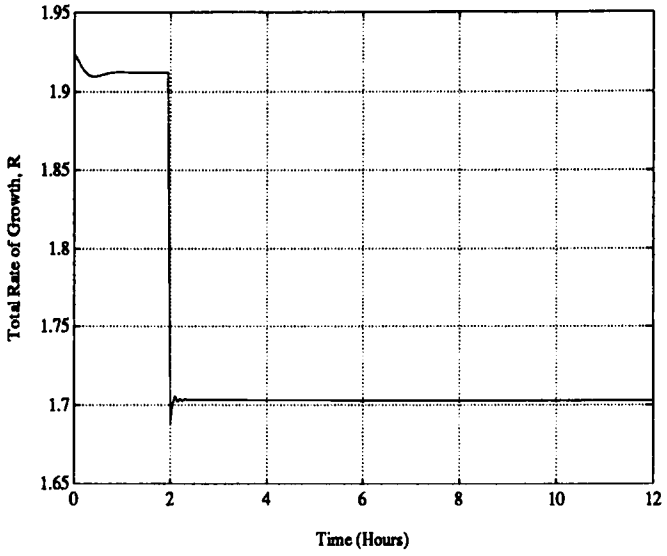

Figure D.14: Change in μ_{max} by -12%. Unachievable set point. Output versus time

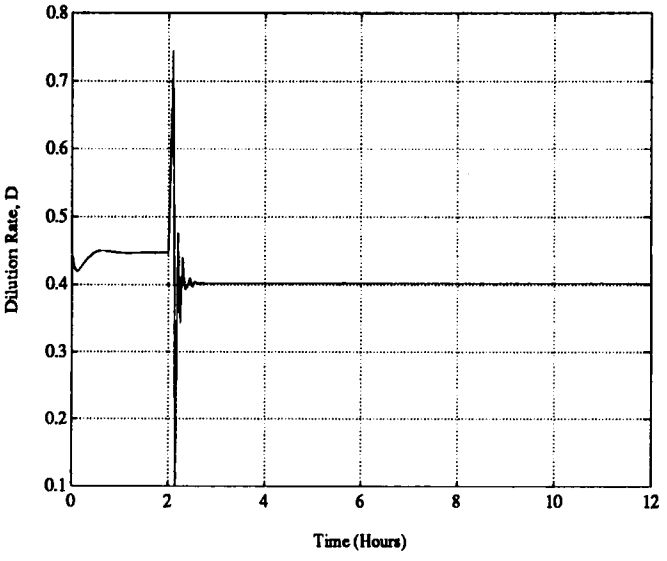

Figure D.15: Change in μ_{max} by -12%. Unachievable set point. Input versus time

D.1.3 Global Optimization: Nominal Performance

Nominal system response for a change in the desired set point by 5%

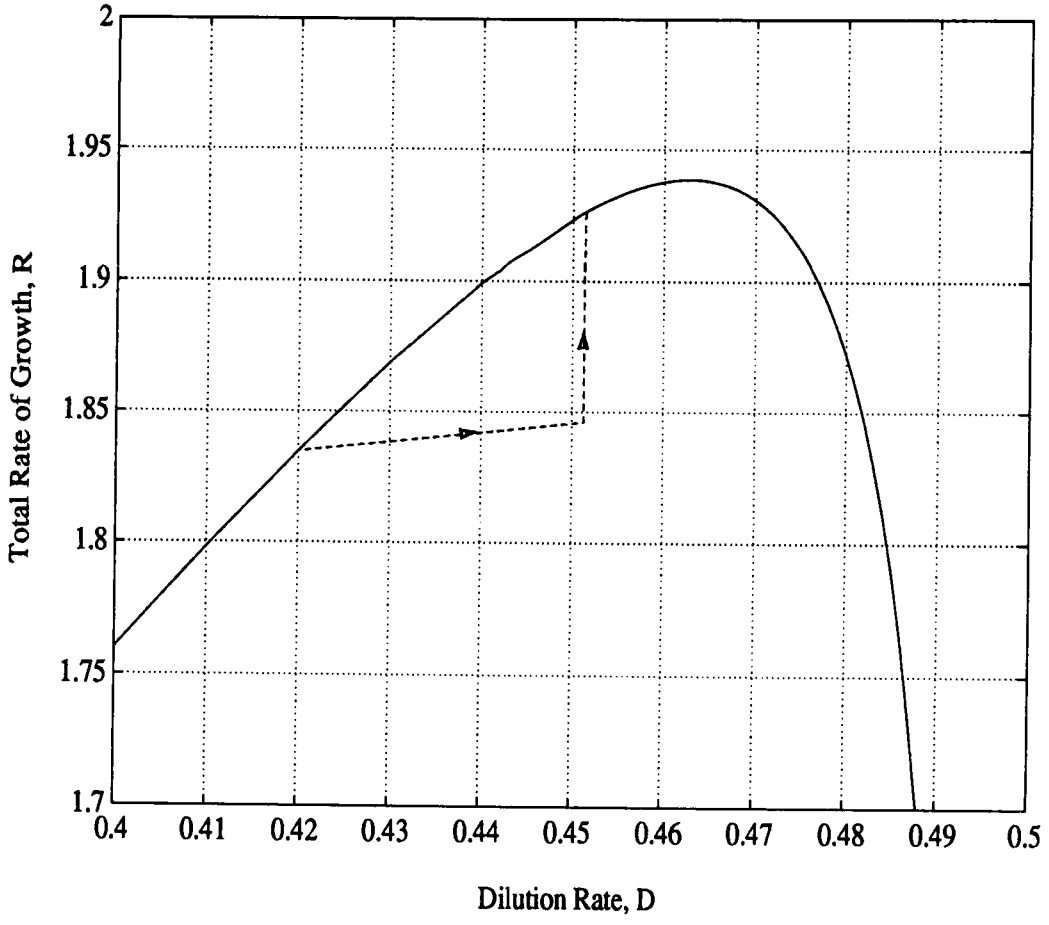

Figure D.16: Set point change in the total rate of growth by 5%. System trajectory

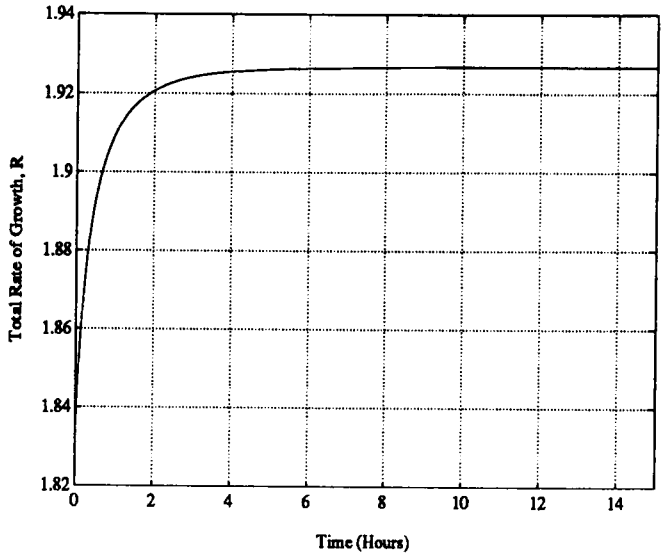

Figure D.17: Set point change in the total rate of growth by 5%. Output versus time

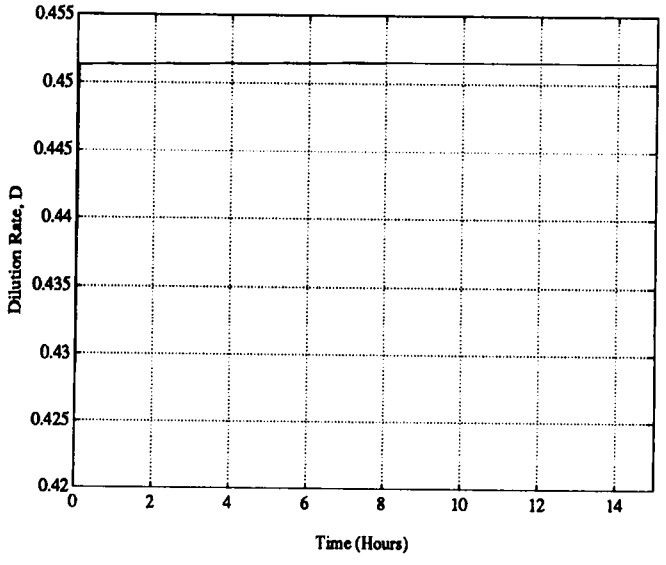

Figure D.18: Set point change in the total rate of growth by 5%. Input versus time

Nominal system response for a change in the desired set point by 30%

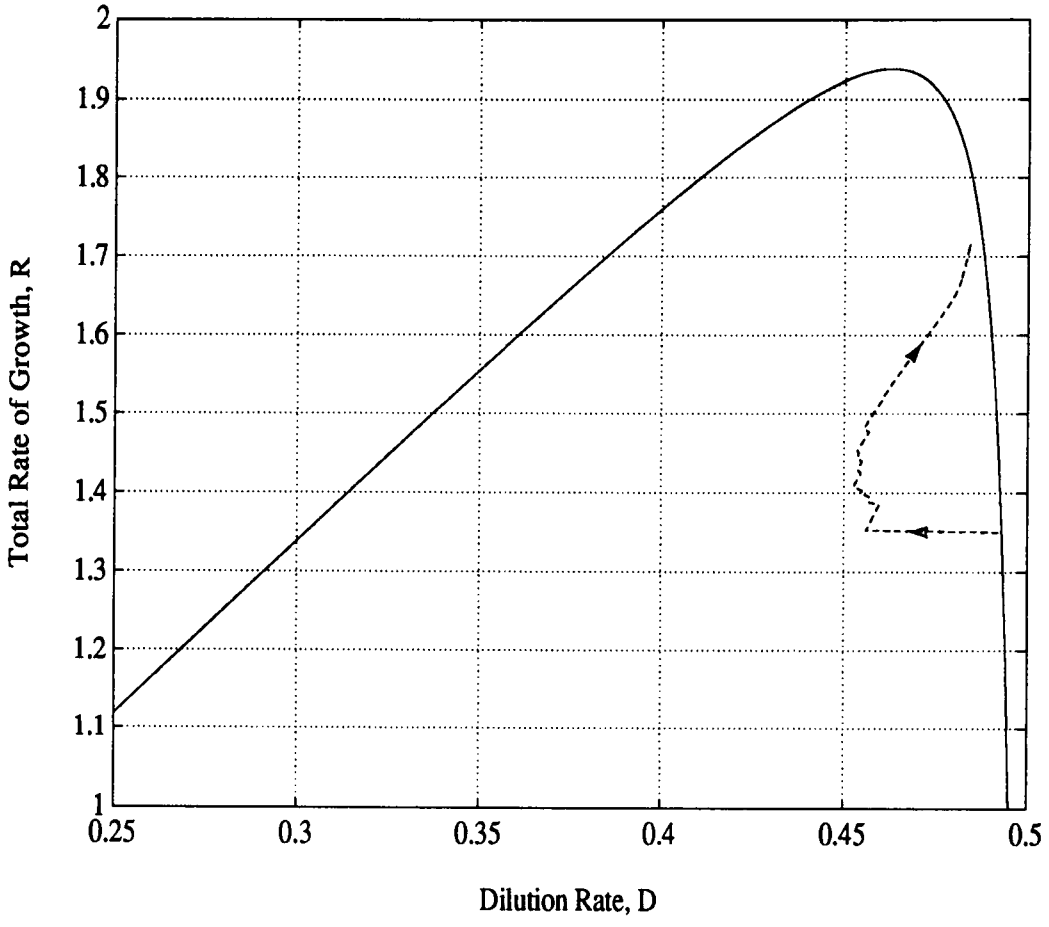

Figure D.19: Set point change in the total rate of growth by 30%. System trajectory

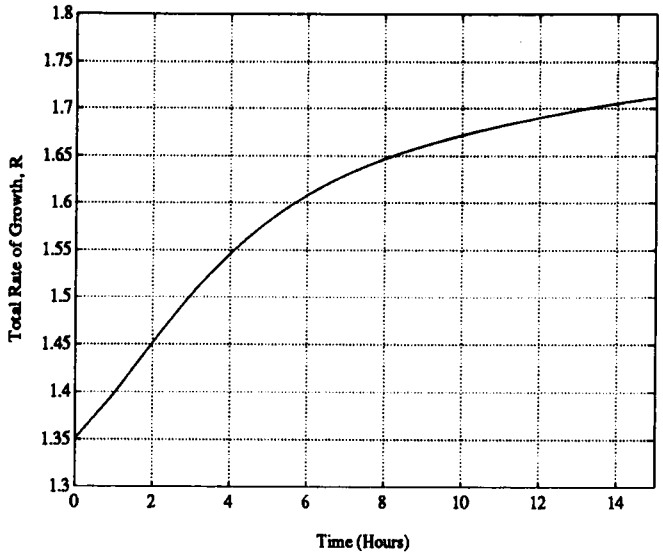

Figure D.20: Set point change in the total rate of growth by 30%. Output versus time

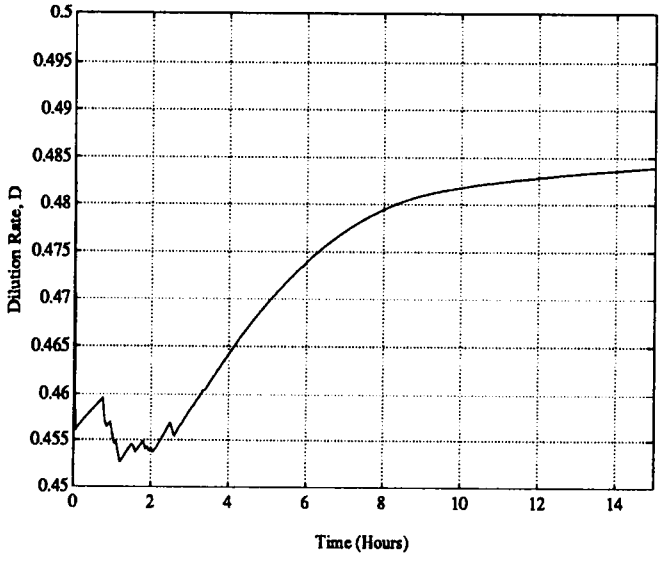

Figure D.21: Set point change in the total rate of growth by 30%. Input versus time

D.1.4 Global Optimization: Parameter Variations

System response for a change in μ_{max} by -10%

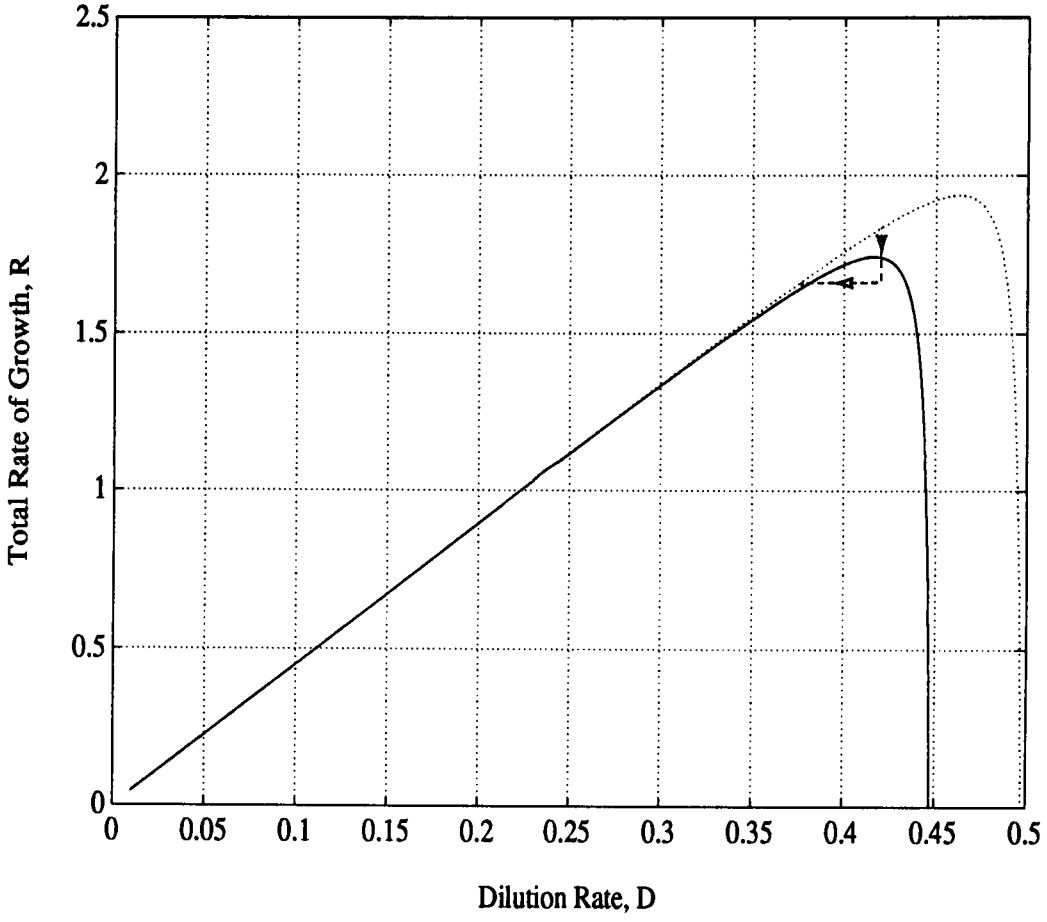

Figure D.22: Change in μ_{max} by -10%. Unachievable set point. System trajectory

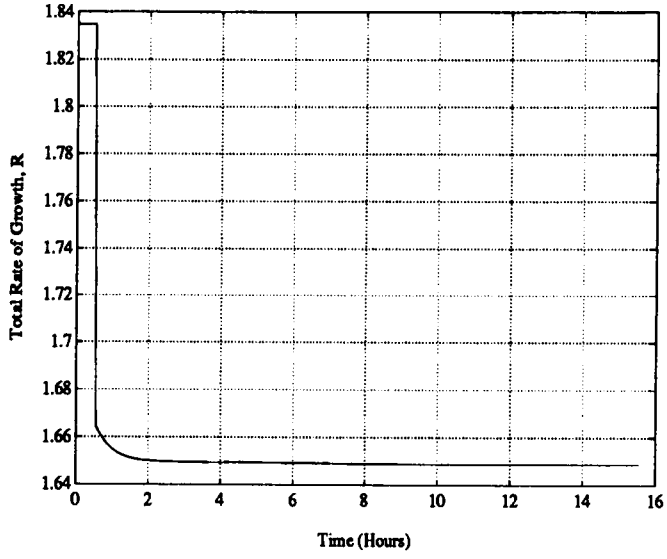

Figure D.23: Change in μ_{max} by -10%. Unachievable set point. Output versus time

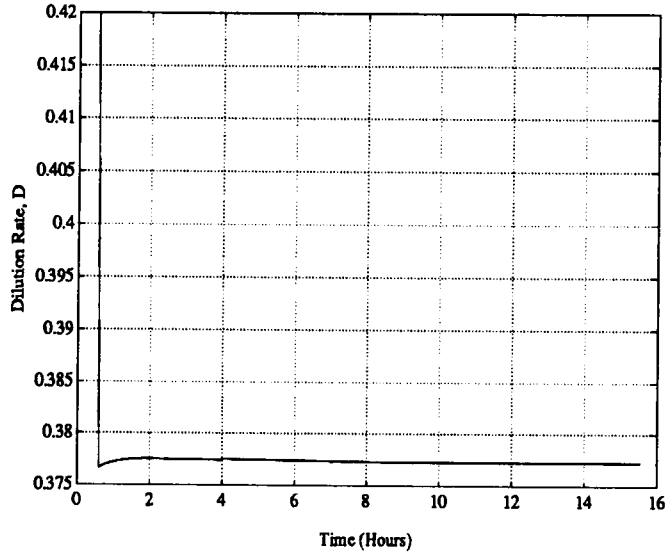

Figure D.24: Change in μ_{max} by -10%. Unachievable set point. Input versus time

D.2 CSTR

D.2.1 ELCC: Nominal Performance

Nominal system response for a change in the desired set point by 50%

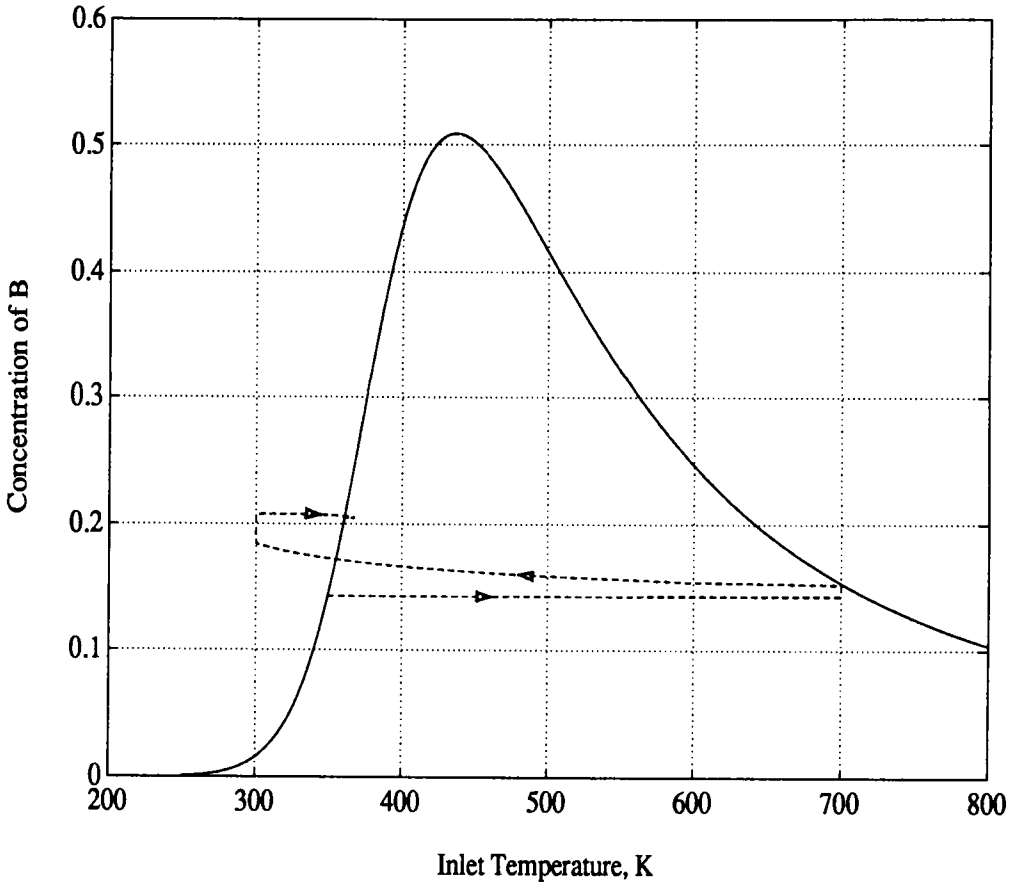

Figure D.25: Set point change in the concentration of B by 50%. System trajectory

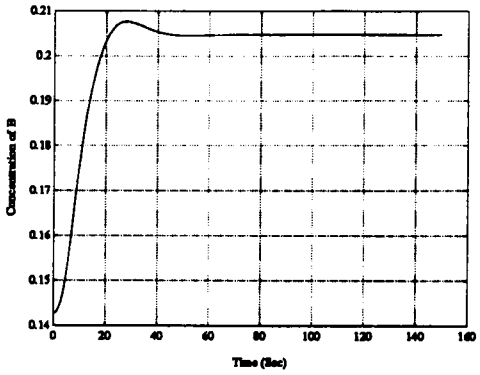

Figure D.26: Set point change in the concentration of B by 50%. Output versus time

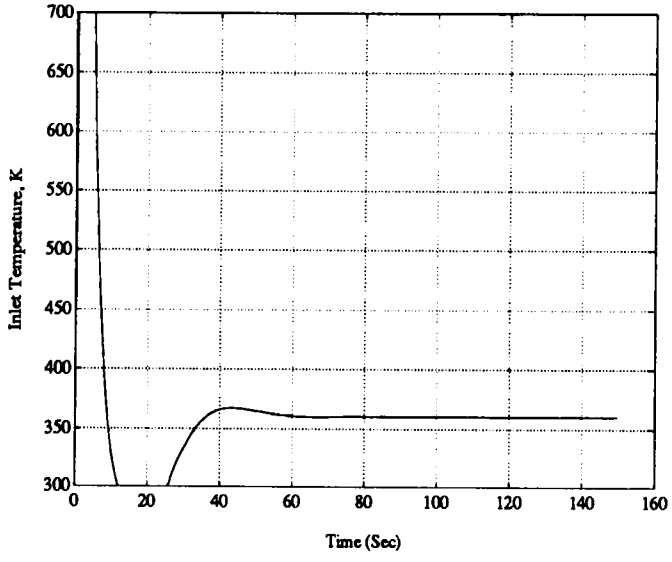

Figure D.27: Set point change in the concentration of B by 50%. Input versus time

Nominal system response for a change in the desired set point by 10%

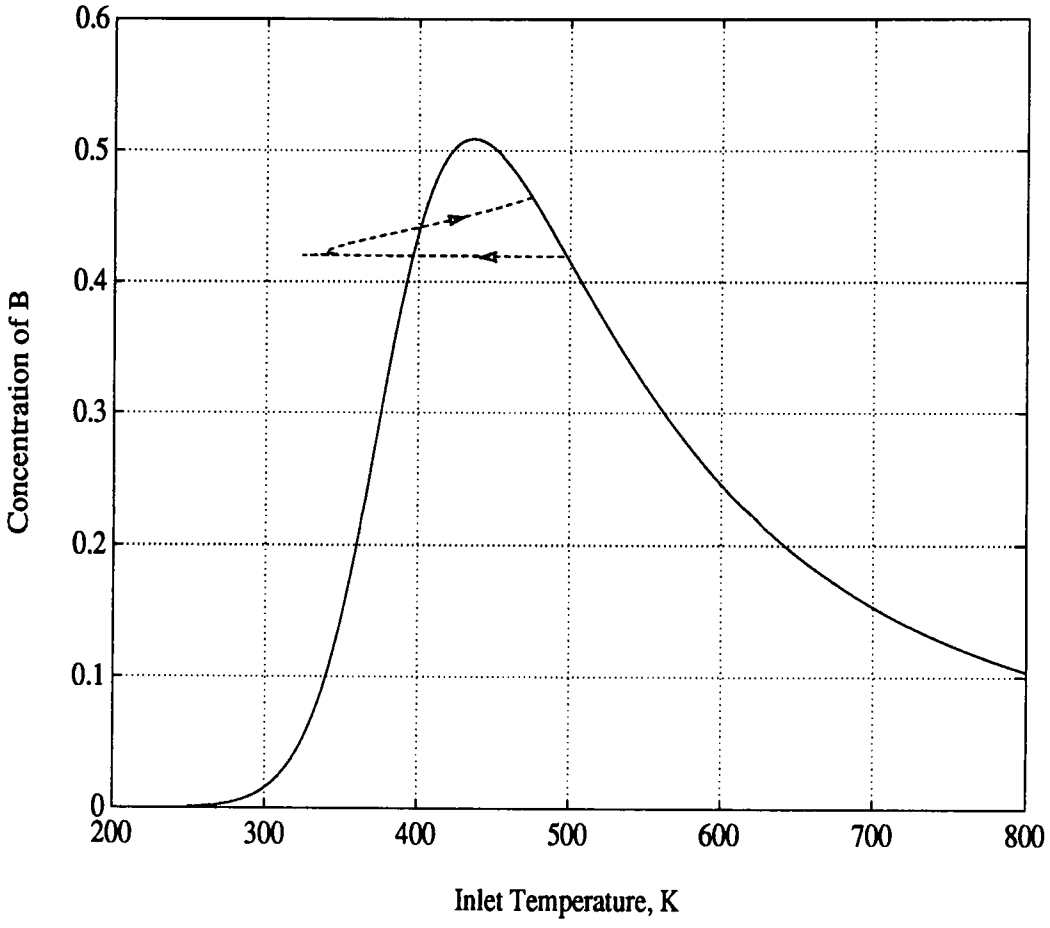

Figure D.28: Set point change in the concentration of B by 10% with initial temperature higher than $T_{i,opt}$. System trajectory.

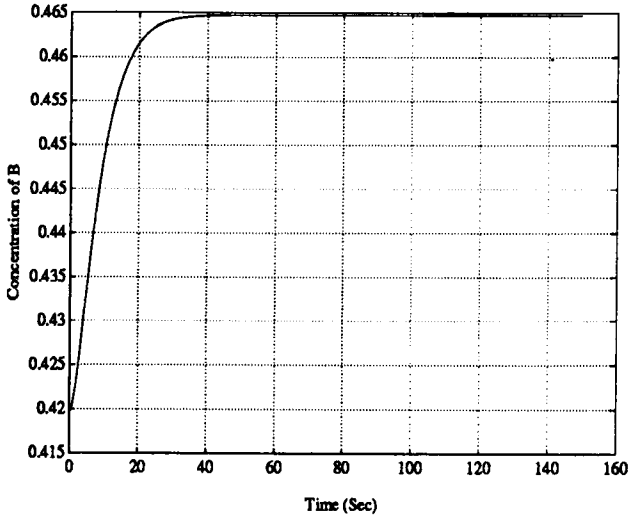

Figure D.29: Set point change in the concentration of B by 10% with initial temperature higher than $T_{i,opt}$. Output versus time

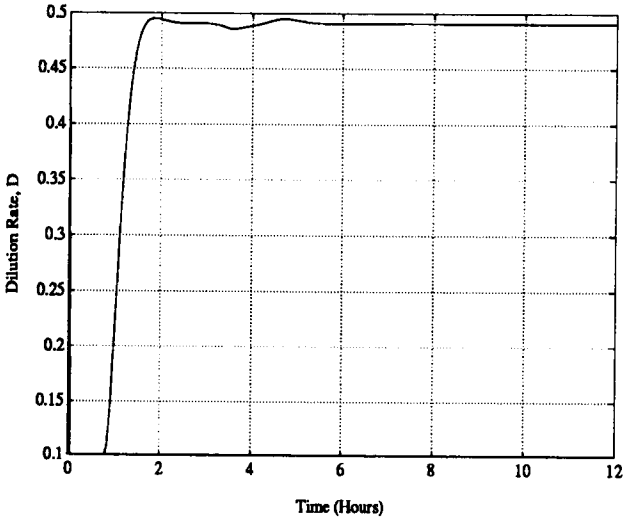

Figure D.30: Set point change in the concentration of B by 10% with initial temperature higher than $T_{i,opt}$. Input versus time

D.2.2 ELCC: Parameter Variations

System response for a change in k_1 by -15%

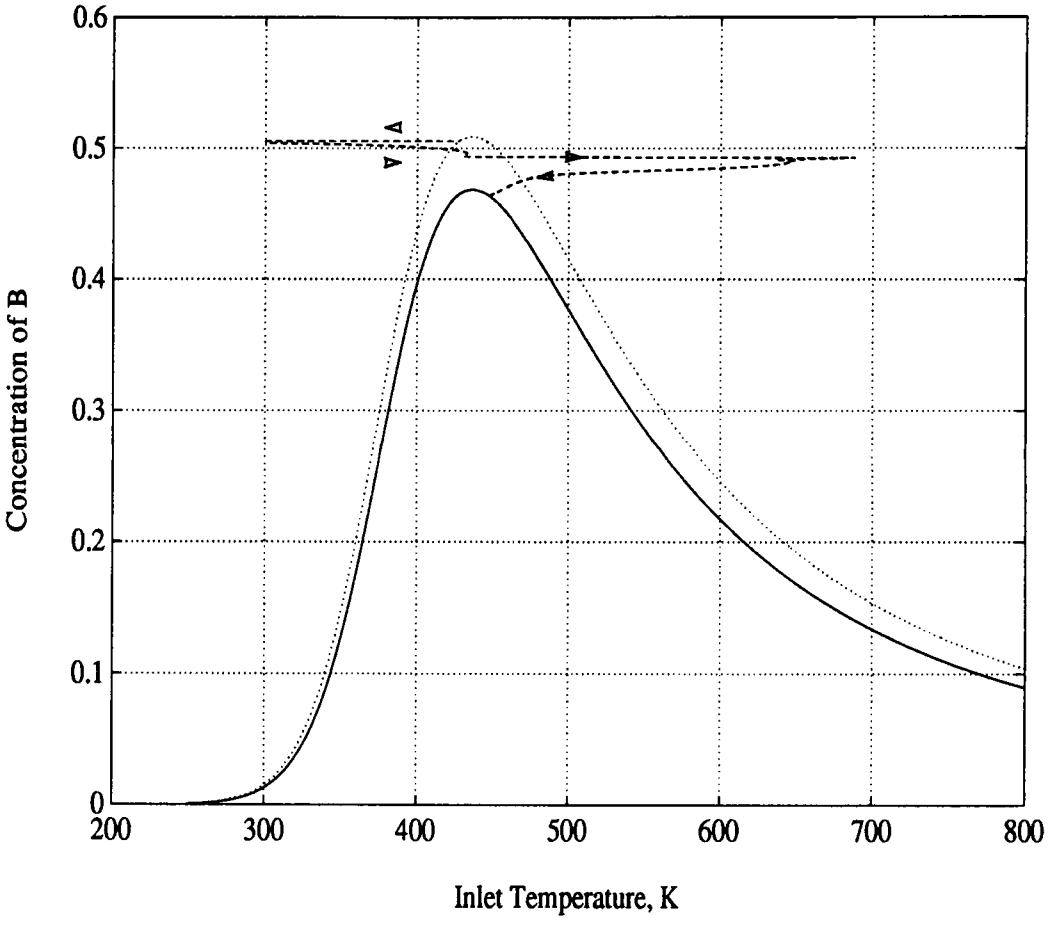

Figure D.31: Change in k_1 by -15%. Unachievable set point. System trajectory

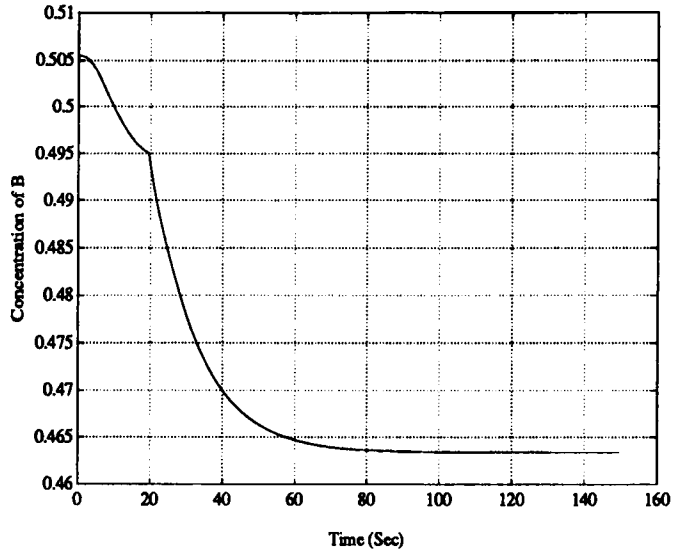

Figure D.32: Change in k_1 by -15%. Unachievable set point. Output versus time

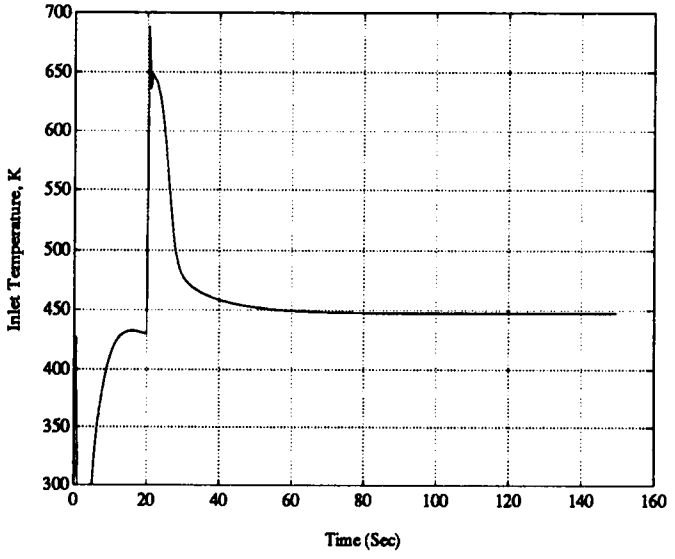

Figure D.33: Change in k_1 by -15%. Unachievable set point. Input versus time

System response for a change in k_1 by -20% and A_i by -18%

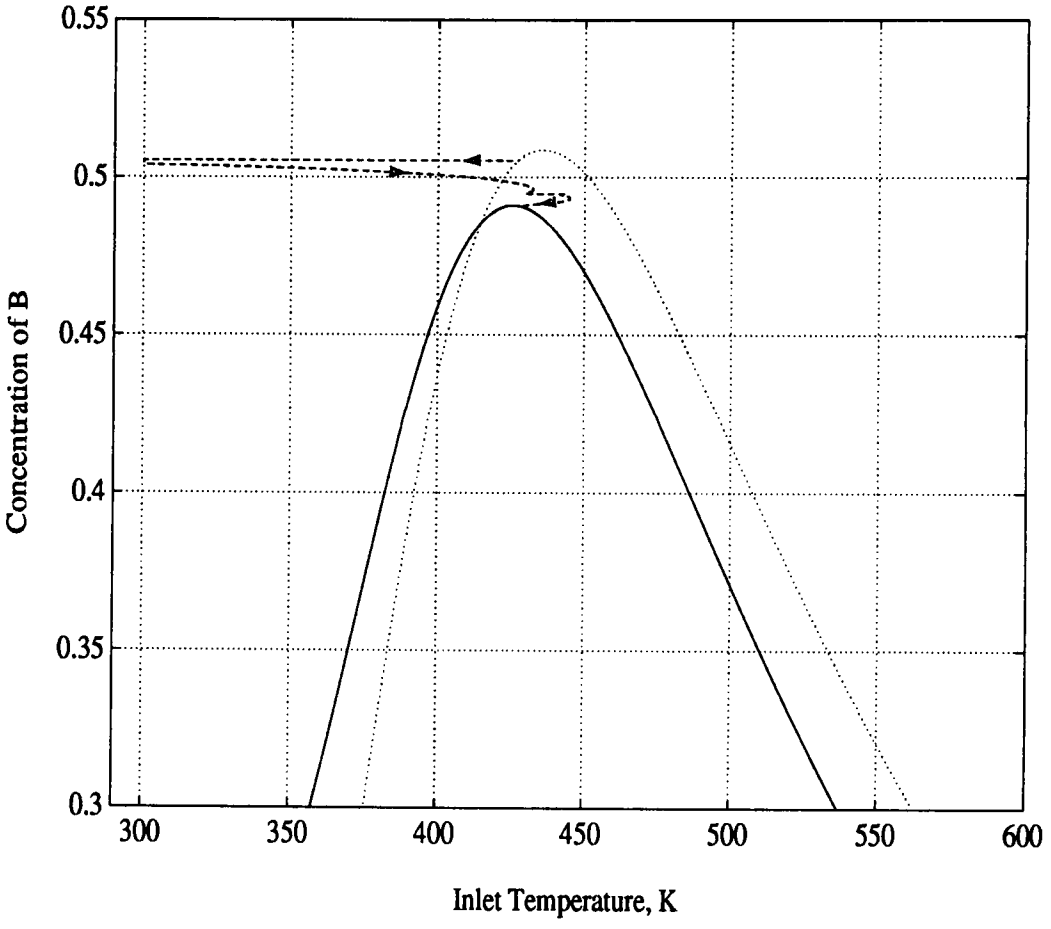

Figure D.34: Change in k_1 by -20% and A_i by -18%. Unachievable set point. System trajectory

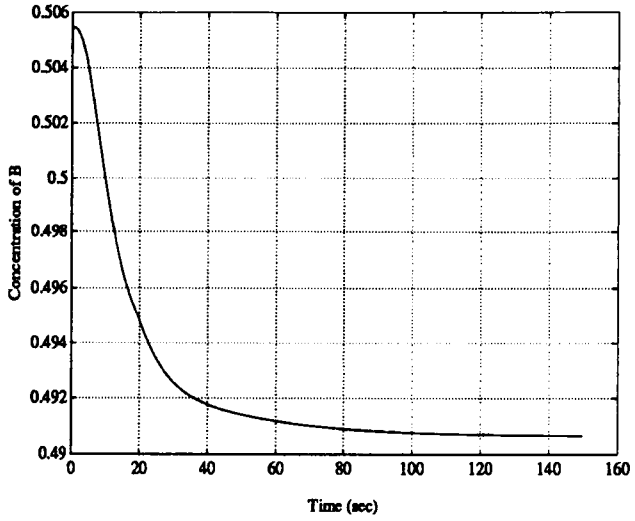

Figure D.35: Change in k_1 by -20% and A_i by -18%. Unachievable set point. Output versus time

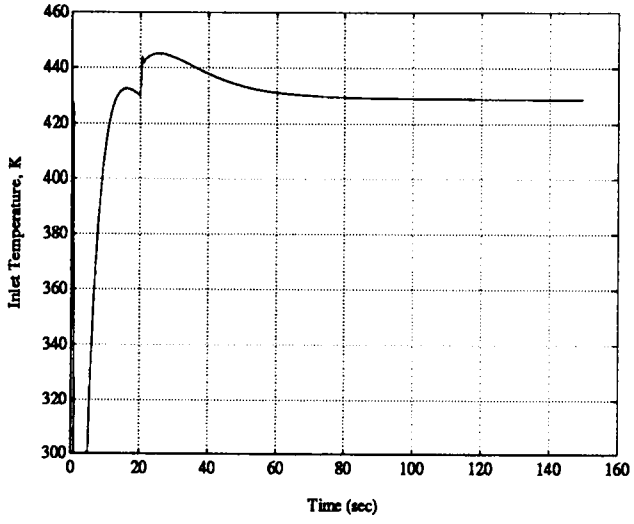

Figure D.36: Change in k_1 by -20% and A_i by -18%. Unachievable set point. Input versus time

D.3 Catalytic CSTR

D.3.1 ELCC: Nominal Performance

Nominal system response for $P = 10$ and $M = 5$

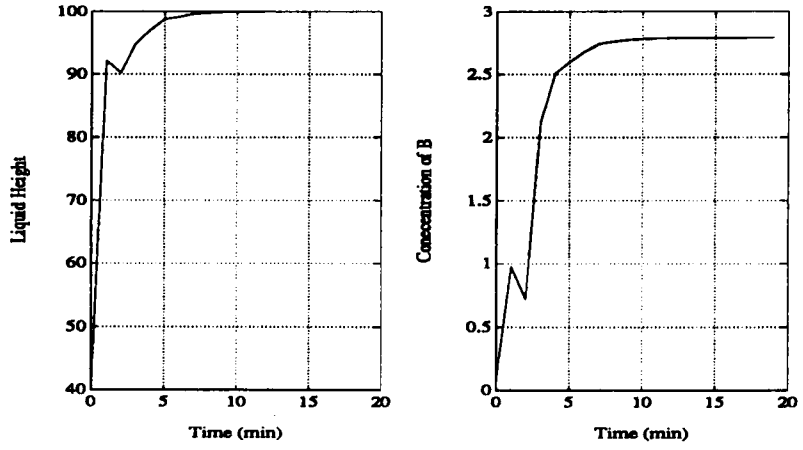

Figure D.37: Marginally unstable set point. Output versus time

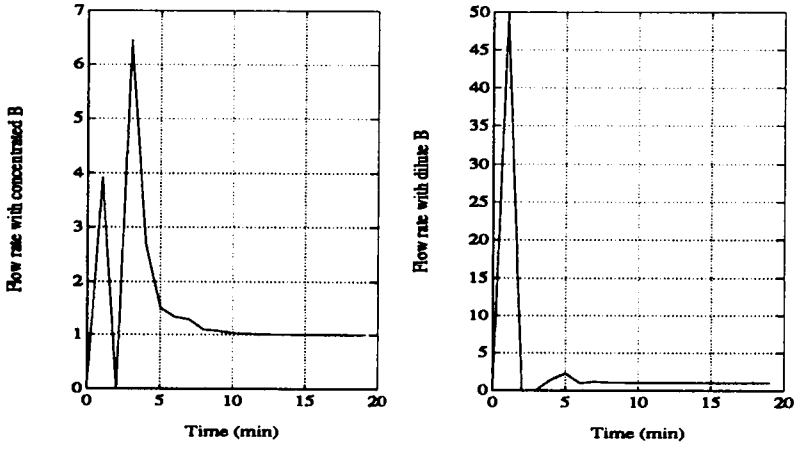

Figure D.38: Marginally unstable set point. Input versus time

Nominal system response for $P = 2$ and $M = 2$

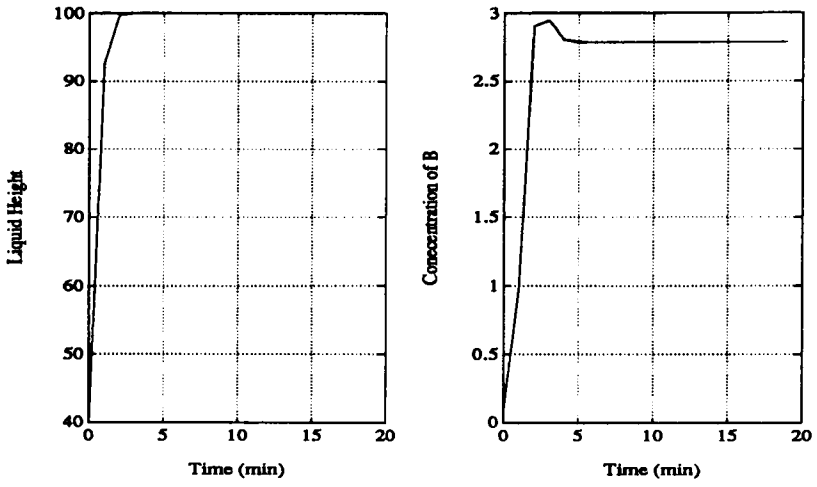

Figure D.39: Marginally unstable set point. Output versus time

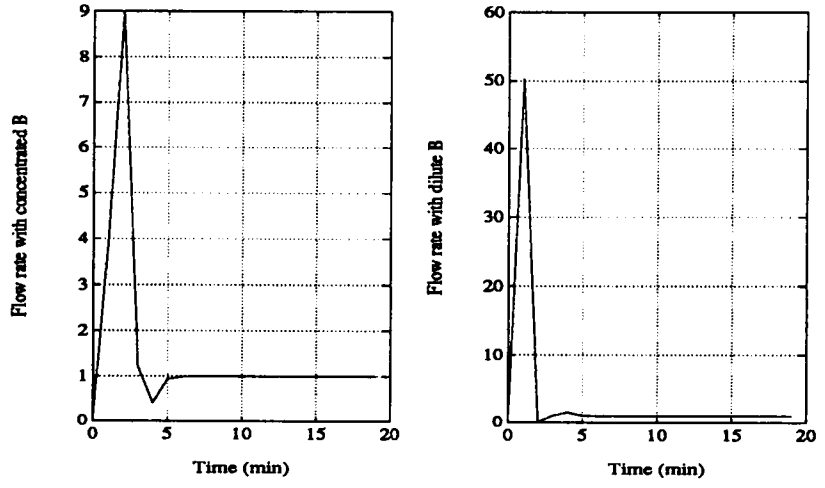

Figure D.40: Marginally unstable set point. Input versus time

D.3.2 ELCC: Parameter Variations

System response for a change in k_2 by -30%

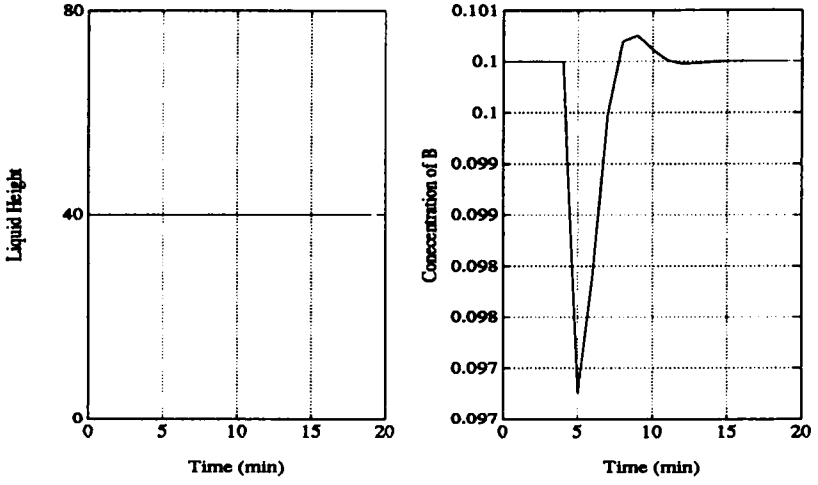

Figure D.41: System response for a change in k_2 of -30%. Output versus time

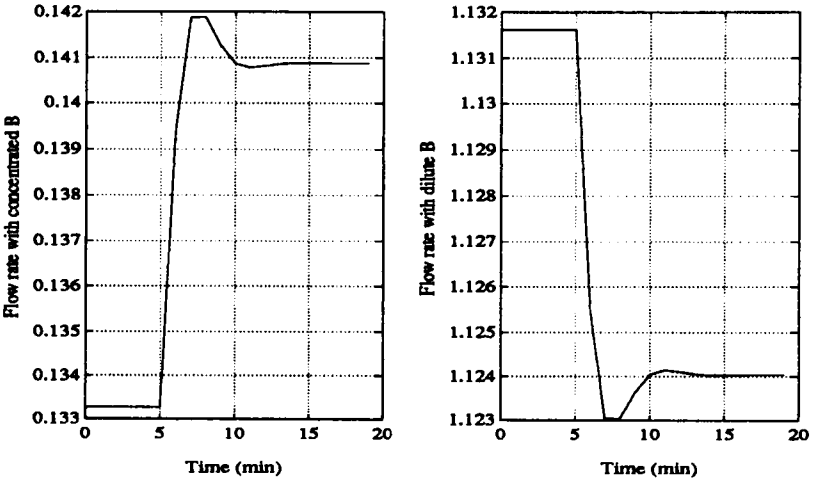

Figure D.42: System response for a change in k_2 of -30%. Input versus time

System response for a change in k_1 by -20% and C_{B1} by +10%

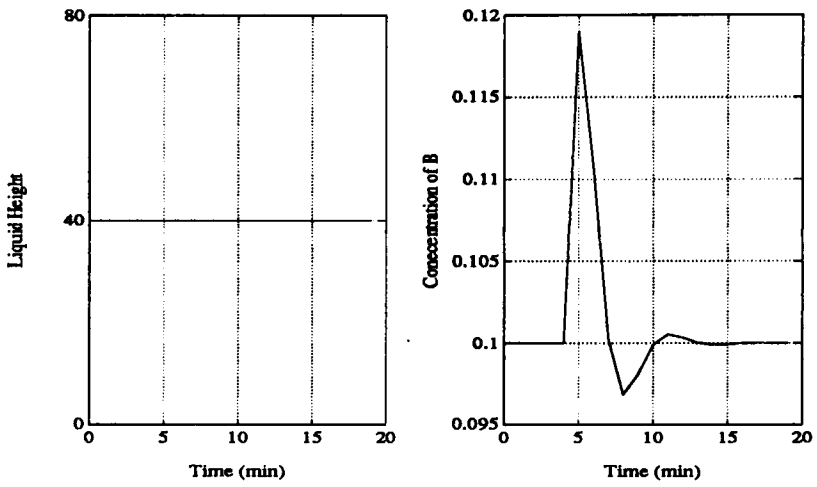

Figure D.43: System response for a change in k_1 by -20% and C_{B1} by +10%. Output versus time

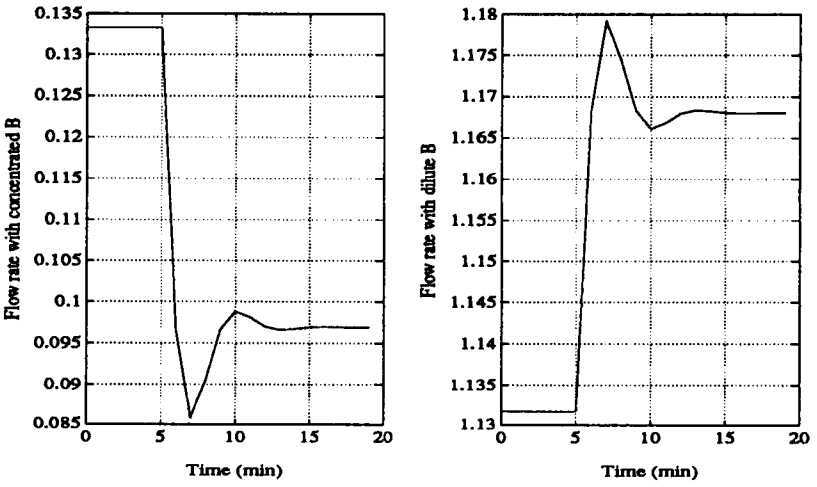

Figure D.44: System response for a change in k_1 by -20% and C_{B1} by +10%. Input versus time

VITA

Daniel Rangel was born on October 22, 1962 in Santa Fe de Bogota, Colombia. He studied Physics simultaneously with Chemical Engineering for two years and later decided that there was no hope for a Nuclear Physicist in Colombia. Thus, hunting for something more practical and, being undecided, and the son of a career Chemical Engineer and a Chemist he chose Chemical Engineering as his life. He received a Bachelor of Science degree in Chemical Engineering from the Universidad de America at Santa Fe de Bogota in 1986. After assiduously avoiding English in Colombia, he travelled to the United States on October of 1986.

In the Fall of 1987, he started graduate studies at the University of Tennessee at Chattanooga, where, in April 1989 he obtained a Master of Science degree with a major in Chemical Engineering. This was the beginning of his focus on the area of Chemical Process Control. He continued with his graduate studies in Advanced Process Control at The University of Tennessee, Knoxville, and was awarded the Doctor of Philosophy degree in Chemical Engineering, in May, 1992.