NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Previously copyrighted materials (journal articles, published tests, etc.) are not filmed.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

Les documents qui font déjà l'objet d'un droit d'auteur (articles de revue, tests publiés, etc.) ne sont pas microfilmés.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30.

THE UNIVERSITY OF ALBERTA

MULTIVARIABLE OPTIMAL CONSTRAINED CONTROL ALGORITHM (MOCCA)

BY

KIAN YAP LIM

A THESIS

SUBMITTED TO THE FACULTY OF GRADUATE STUDIES AND RESEARCH

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE

OF MASTER OF SCIENCE

IN

PROCESS CONTROL

DEPARTMENT OF CHEMICAL ENGINEERING

EDMONTON, ALBERTA

SPRING, 1988

ISBN 0-315-42815-5

THE UNIVERSITY OF ALBERTA

RELEASE FORM

NAME OF AUTHOR: KIAN YAP LIM

TITLE OF THESIS: MULTIVARIABLE OPTIMAL CONSTRAINED CONTROL ALGORITHM (MOCCA)

DEGREE: MASTER OF SCIENCE

YEAR THIS DEGREE GRANTED: SPRING, 1988

Permission is hereby granted to THE UNIVERSITY OF ALBERTA LIBRARY to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves other publication rights, and neither the thesis nor extensive extracts from it may be printed or otherwise reproduced without the author's written permission.

(SIGNED) .....

PERMANENT ADDRESS:

4115 - 37 Street
Edmonton, Alberta
Canada. T6L 5M7

DATE: April 18, 1988

THE UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled Multivariable Optimal Constrained Control Algorithm (MOCCA) submitted by Kian Yap Lim in partial fulfillment of the requirements for the degree of Master of Science in Process Control.

.....

Supervisor

.....

.....

Date: April 25, 1988

To J & J & the next one

ABSTRACT

Model Predictive Control (MPC) is a computer based control algorithm that originated in industry. The industrial success of MPC became a driving force for further investigation and implementation of MPC in industry and academia. In this thesis, a MPC algorithm, MOCCA (Multivariable Optimal Constrained Control Algorithm), is developed and implemented as a software package.

MOCCA is a computer based algorithm developed to meet practical control objectives such as multivariable applications, constraints, servo and regulatory control, feedforward control and optimization of a user specified performance index. Future output predictions are obtained from a convolution of step response data and past values of input variables. The predictive control problem is to calculate the control action required to minimize the user specified performance index, e.g. minimize the error between process and setpoint values subject to constraints. This control problem can be transformed into an optimization problem, e.g. a linear or quadratic programming problem, which can be solved using well established and documented numerical techniques which are not application specific.

The implementation of MOCCA in the form a "translator/simulator" software package makes the design and implementation of the algorithms straightforward. The

translator program includes three controller design options: weighted least squares, linear programming and quadratic programming. The controller design is accomplished by entering the design parameters, constraints and objective functions in "control terminology" and the translator generates the data required for on-line control calculations. Evaluation of the controller using the simulator provides an immediate feedback mechanism in the controller design. The step response model used in the controller design is normally used to simulate the process. Load disturbances and model process mismatch can also be investigated. Software engineering and management techniques were used in the software design to provide the necessary portability needed for industrial implementation i.e. only the database and process operator communication interface of the on-line control program requires modification if MOCCA is installed on a new process control computer.

Evaluation of the control algorithms using simulated processes indicates that MOCCA is a robust controller that meets industrial objectives. Industrial implementation of MOCCA is currently being undertaken by two major Canadian companies.

ACKNOWLEDGMENT

The author would like to express his gratitude to his thesis supervisor, Dr. D. Grant Fisher, for his guidance and assistance provided during the course of this research. Appreciation is also extended to the DAC center staff for their assistance in using the computing facilities and the workshop staff for their assistance in the operations of the evaporator pilot plant.

Appreciation is also extended to my fellow graduate students for their friendship and support.

Finally, financial support provided by the Department of Chemical Engineering at the University of Alberta and the National Sciences and Engineering Research Council is greatly appreciated.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Nomenclature

Abbreviations

DMC	Dynamic Matrix Control
IMC	Internal Model Control
LDMC	Linear Programming DMC
LP	Linear Programming
MAC	Model Algorithmic Control
MPC	Model Predictive Control
MPHC	Model Predictive Heuristic Control
MOCCA	Multivariable Optimal Constrained Control Algorithm
QDMC	Quadratic Programming DMC
QIMC	Quadratic Programming IMC
QP	Quadratic Programming
WLSQ	Weighted Least Square

Alphabetic

a	step response coefficients
A_1	step response coefficient matrix associated with past inputs
A_2	step response coefficient matrix associated with future inputs
d	disturbance variable
e	error signal
E	error signal vector notation
h	impulse response coefficients
K	observer or (Kalman) filter gain
M	control horizon
N	number of step response coefficients
P	prediction horizon
u	input variable
U	input variable vector notation
y	output variable
Y	output variable vector notation

Greek

α	feedback filter constant
β	setpoint filter constant
Δ	change or difference value
Γ	output weighting matrix
Γ_m	input weighting matrix
γ	output weighting vector
γ_m	input weighting vector
λ	time delay
	disturbance variable

Superscripts

*	for process variables, prediction based on past inputs; for matrix, pseudo inverse
^	estimated values
~	residuals
T	vector or matrix transpose

Subscripts

d	desired value
f	filtered value
F	future value
FF	feedforward
m	model value
p	process value
sp	setpoint value
ss	steady state value

Introduction

1.1 Introduction to Model Based Predictive Control

The development and rapid growth of digital computers has changed the practice of chemical process control. More and more chemical plants, whether new or old, are being brought under digital computer control leading to better process control and reduced operating costs.

Frequently the applications of advanced control concepts requires a mathematical description of the process to be controlled. In most cases the development of a parametric model such as a transfer function or state space model proves to be costly and time consuming. The problem becomes more difficult when processes with unusual dynamics are encountered. This problem can be avoided by the use of non-parametric models (step or impulse response), which are easily obtained and understood.

In addition to the model, advanced control concepts frequently require the optimization of the chemical process, that is the chemical plant is operated according to market conditions and controlled so that operating costs are minimized and profit maximized. Optimization also includes meeting various operational constraints encountered in a

chemical plant. The advanced control algorithm must also be able to meet servo and regulatory requirements while ensuring the stability of the chemical process.

Model predictive control (MPC) is an advanced control algorithm that was introduced in recent years and has enjoyed a considerable amount of success. The success of MPC is mainly due to several key features that meet the design requirements discussed above:

1. The multivariable process is characterized by a set of discrete step or impulse response coefficients which are used to predict current and future process outputs.
2. A "correction term" to account for modeling error and/or unmeasured disturbances is usually calculated based on the difference between the estimated value and measured plant output. If the step responses is unknown, then this correction term can be calculated either by identifying parameters on-line to permit forecasting of future values; or through estimation techniques such as Kalman filtering.
3. A predictive control strategy is used to calculate the control action which minimizes a user specified performance index. The optimization is performed over a trajectory (multi-step optimization).

4. The control calculation is usually formulated as an optimization problem: linear or non-linear; weighted or unweighted; constrained or unconstrained. Depending on the characteristic of the optimization problem, the solution may require anything from simple off-line calculations to on-line, non-linear optimization.

This thesis documents the development of a MPC software package and its application at the University of Alberta. The algorithm is known as Multivariable Optimal Constrained Control Algorithm (MOCCA).

1.2 Objective of Thesis

The objectives of this work are to extend and implement the original work on MOCCA by Sripada and Fisher (1985). Specifically, the extensions include:

1) some theoretical analysis and a formal definition of the formulation,
2) the design and implementation of a "translator/simulator" software package to aid in the translation of the control problem into a standard optimization problem and simulation of the resulting closed loop performance. The formulation and implementation of MOCCA were done with the objective of transferring the algorithm to an industrial application.

3) on-line control algorithms to implement MOCCA on a process control computer such as the Hewlett Packard/1000. Portability of the control algorithm was considered during the design so that it can be implemented in any process control computer with minimum effort,
4) evaluation of the algorithms by simulation.

1.3 Structure of Thesis

Following the introduction, chapter two of the thesis provides a functional overview of MOCCA. Chapter three discusses the model formulation using the direct and state space approaches. Feedback correction, feedforward contributions, supervisory systems and time varying features of MOCCA are also discussed. The transformation of a standard control problem into a standard optimization problem for both the constrained and unconstrained cases is discussed in chapter four. A literature survey presented in chapter five examines the differences and similarities of different model predictive control (MPC) schemes. The overall software design is discussed in chapter six, using functional specifications and high level program structure diagrams. In chapter seven, simulation results using single input single output (SISO) and multi-input multi-output (MIMO)

models are presented. Comparisons with published results are also included. Conclusions and recommendations for future work are included in chapter eight.

Multivariable Optimal Constrained Control Algorithm (MOCCA)

2.1 Introduction

In this chapter, an overview of MOCCA will be presented. MOCCA was developed to meet two major objectives: first, a practical, robust, model-based control algorithm that could be applied to multivariable industrial servo and regulatory control problems and second, an algorithm that could be implemented effectively on an industrial digital control computer.

To be practical for industrial applications it was concluded that the control algorithm should:

1. utilize easily obtained input/output information rather than complicated or theoretical process models.
2. be able to follow trajectories generated by higher levels of control e.g. optimization based on economic specifications
3. regulate the process despite measured or unmeasured disturbances
4. handle hard constraints on the process variables
5. optimize a user specified performance index and
6. provide a means to accommodate practical factors such as transducer and/or sensor failures.

To be effectively implemented on a control computer the algorithm should:

1. be simple computationally
2. not require excessive memory for storage
3. be robust to round-off errors i.e. short word length
4. use solution techniques that are explicit or at least have guaranteed convergence, as opposed to open ended, iterative calculation and
5. use established numerical techniques that are documented, tested and not application specific.

MOCCA meets these and a number of other objectives.

2.2 Functional Overview of Model Predictive Control

MOCCA is a computer based algorithm developed to meet practical control objectives using effective numerical and software engineering techniques. As indicated by Figure 2.1, MOCCA can be subdivided into four main parts:

1. model based predictions of the process output trajectory, $Y_m(k)$
2. supervisory system to generate the desired trajectory, $Y_d(k)$ and constraints
3. feedback correction trajectory, $Y_f(k)$ for regulatory control and model process mismatch and

4. a predictive control algorithm that generates the control action u(k) which minimizes a user specified objective function and meets its constraints.

Each of the subdivisions will be described functionally in the following sections.

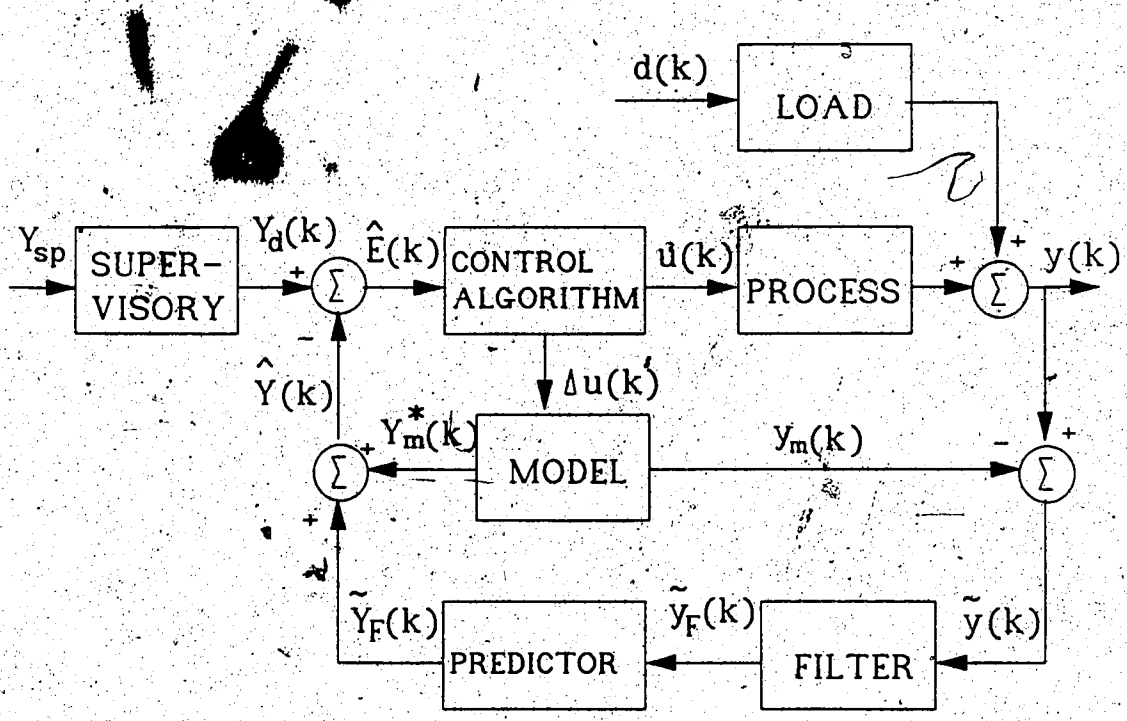

Figure 2.1: Schematic Representation of MOCCA

2.2.1 Model Based Predictions

Any type of model can be used for the prediction step in MOCCA provided that the fundamental function of modeling is insured, i.e. the ability to predict what the behavior of the modeled process will be when subjected to a known input.

MOCCA uses discrete step response data rather than a parametric model to predict the current output, $y_m(k)$ and the future output trajectory, $Y_m(k)=\{y_m(k+i), i=1,2,\ldots,P\}$ of the process, where P, the prediction horizon, is a user specified design parameter. The advantages of using step response data are that the coefficients are easily obtained from input/output data, no model structure needs to be assumed, and processes with unusual dynamic behaviors can be modeled easily. The model formulation can be accomplished by either the direct or the state space approach.

In the direct method, using the principles of linear superposition and multiplicative scaling, the process output is predicted as a function of known past inputs and the step response coefficients. In the state space approach, the prediction equations are also obtained from basic principles and then cast into state space forms. Due to the recursive nature of the state space model, model predictions are computationally more efficient. In addition, state space theorems and techniques e.g. filters and observers can be used to design better prediction schemes. The direct model can be used to generate the initial conditions (predicted output trajectory) required by the state space model.

2.2.2 Supervisory System

The supervisory portion of MOCCA is used to generate the appropriate desired trajectory $Y_d(k) = (y_d(k+i), i=1,2...,P)$. For example it can accept a step setpoint change from the process operator and generate the signal $Y_d(k)$ using filters. It can also be used to set or change the constraints on the process variables. Higher level optimization (e.g. optimal setpoint selection in the economic sense) can also be included. The supervisory system is optional in the sense that the desired trajectory could be set equal to the setpoint. A feedback path can also be incorporated into the supervisory system as shown by Richalet et al. (1978). (This will be discussed in a later section of the thesis.) The supervisory system can be used to introduce considerable design flexibility and provides a mean of achieving several practical industrial objectives. However, since it can be separated from the main MOCCA controller, it will be considered as a separate problem and will not be fully developed.

2.2.3 Feedback Paths and Corrections

The feedback path consists of two parts, a filter and a disturbance predictor. The feedback path corrects for the effects of unmeasured disturbances d(k) and model process mismatch. It will be shown in a later section, that the

filter (exponential) not only reduces process noise but can also be used to ensure stability of the closed loop system. The signal $y(k)=y(k)-y_m(k)$ (cf. Figure 2.1) is filtered and then sent to a predictor which produces an estimate of the future effect of the disturbance, $\bar{Y}_f(k)=\{y_f(k+i), i=1,2,\ldots,P\}$. An estimate of the future output trajectory $\bar{Y}(k)=\{y(k+i), i=1,2,\ldots,P\}$ based on past (known) values of the process input and output is then produced by adding $\bar{Y}_f(k)$ to the model based predictions $\bar{Y}_m(k)$. The estimated trajectory is then fed back and subtracted from the desired trajectory to produce an error trajectory $\bar{E}(k)=\{e(k+i), i=1,2,\ldots,P\}$ that is the input to the predictive control algorithm. The implementation of the feedback path is similar to classical predictive control systems except that MOCCA uses trajectories rather than a scalar value. Note that estimates of the future disturbances and/or model process mismatch $\bar{Y}_f(k)$ could be generated by a number of filtering or estimation schemes since a state space model is available. These details will be discussed in later sections of the thesis.

As mentioned earlier, an alternative feedback path can be formulated via the supervisory system. An analysis of this alternative will be presented later.

2.2.4 Predictive Control Algorithm

The predictive control algorithm is used to generate an incremental control action, $\Delta u(k)$ at each control interval, k, such that the estimated process output $\hat{Y}(k)$ follows the desired trajectory $Y_d(k)$ and does not violate any of the user specified constraints. If the process representation is accurate, then the actual process output trajectory will equal the desired trajectory, i.e. servo control is perfect. The incremental action, $\Delta u(k)$ is summed to produce $u(k)$ before being sent to the process. The predictive controller is designed based on an user specified performance index, constraints, weighting and available process control computer power. The solution techniques (algorithms) will be discussed in later sections.

MOCCA Formulation

3.1 Introduction

MOCCA is based on a straightforward, time domain representation and intuitive control objectives. In this chapter, the formulation of the model used for prediction will be discussed, first using the direct approach and then using the state space approach. The handling of unmeasurable and measurable disturbances is discussed in the feedback and feedforward section along with feedback schemes which uses an alternative feedback path via the supervisory system. Finally some of the features for handling time varying parameters are presented.

3.2 Direct Formulation

The relationship between the input and output variables of a process can be expressed in many ways such as state space and ARMA models. In the next section the model for a single input single output (SISO) system will be formulated using the direct or classical approach. Extensions to handle multi-input multi-output (MIMO) systems will then be considered.

3.2.1 SISO Model Formulation

Assume that the relationship between the input and output variables of the process to be controlled is available in discrete step response form as illustrated in Figure 3.1.

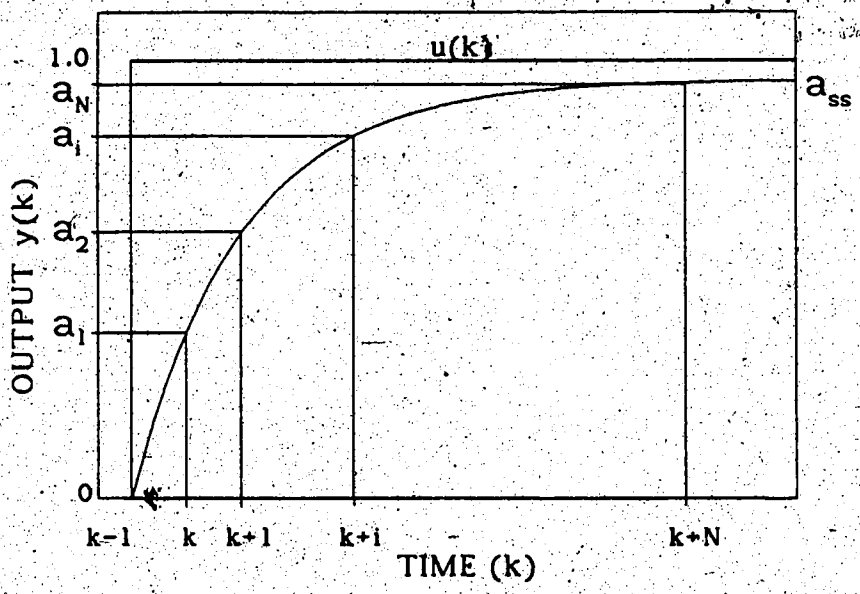

Figure 3.1: Process Response for a Unit Step in Input

Then for a unit step change in input at time k-1, the process response can be written as:

$$y_m(k+i-1|k) = a_i \quad i = 1,2,\ldots,N \qquad (3\text{-}1)$$
$$= a_N \quad i > N$$

where y_m is the model output

a_i are the step response coefficients

N and the sampling time, T_s are chosen such that a_N is sufficiently close to a_{ss}

a_{ss} is the steady state value of the step response. The notation $(k+i|k)$ is used to signify a value at time k+i, based on input values, $\Delta u(\cdot)$, up to and including time k-1. In writing equation (3-1), the process is assumed to be open loop stable so that N and a_{ss} have finite values. An unstable process would have to be stabilized e.g. by proportional feedback control, before MOCCA could be applied. An alternative to this for simple integrating processes is to differentiate the unstable step response data (see Appendix A). Note that the MOCCA formulation also assumes perturbation variables so that both input and output variables are zero at the normal steady states. The step response coefficients are normalized coefficients to describe a unit step response. In the case where the input is not a unit step (usually small step changes are made in the plant), then the step response coefficients have to be scaled to that of unit step change by dividing the coefficients by the magnitude of the input step.

The principles of linear superposition and multiplicative scaling are used in the development of the model. In the case of multiplicative scaling, the response of the output variable for any size step change in the input variable is obtained by multiplying

the unit step response data for the output variable by the magnitude of the actual input step as illustrated in Figure 3.2.

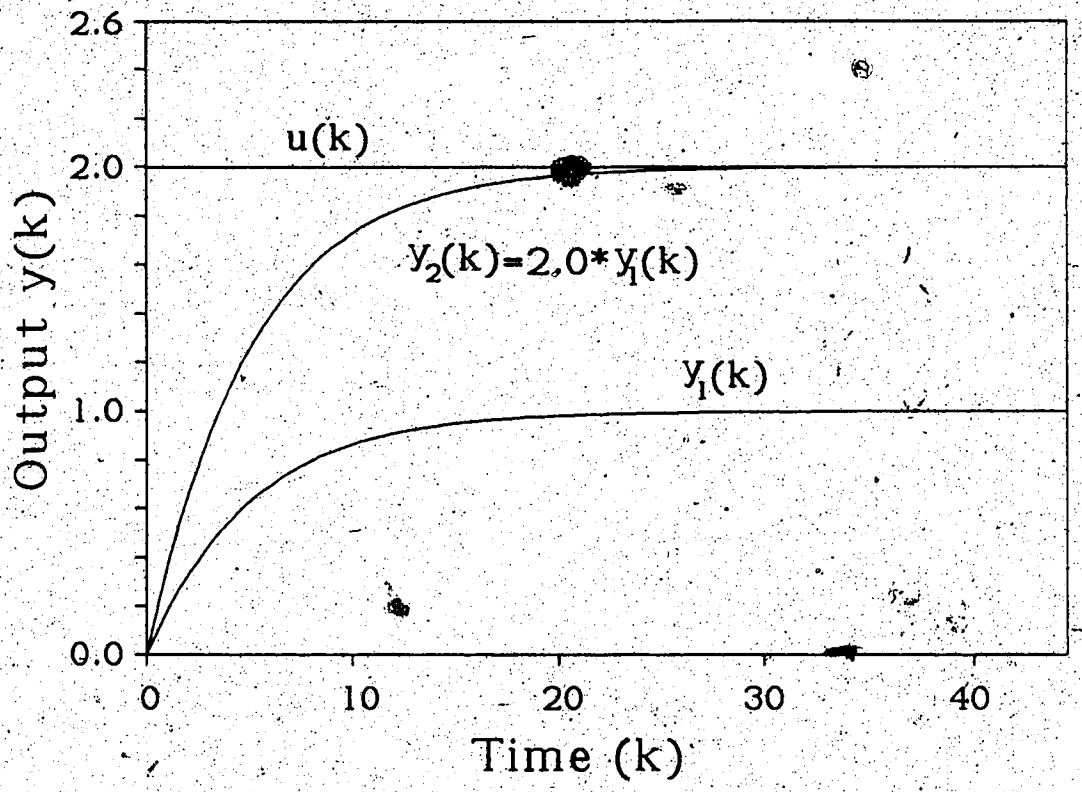

Figure 3.2: Principle of Multiplicative Scaling

Note that in Figure 3.2, the response to a step input with magnitude two is twice the magnitude of the unit change. If the unit curve is discretized into a set of numbers then the output response to an input step of magnitude M is obtained by multiplying this set of numbers by M. The principle of linear superposition is illustrated in Figure 3.3 where the responses of the output variable to a unit change in two different input

variables is shown. Also shown is the response due to simultaneous unit changes in both input variables. It is assumed that the output responses can be obtained by summing the individual unit response curves for each input variables. Superposition also applies when calculating the output response to two separate step changes in the same input variable.

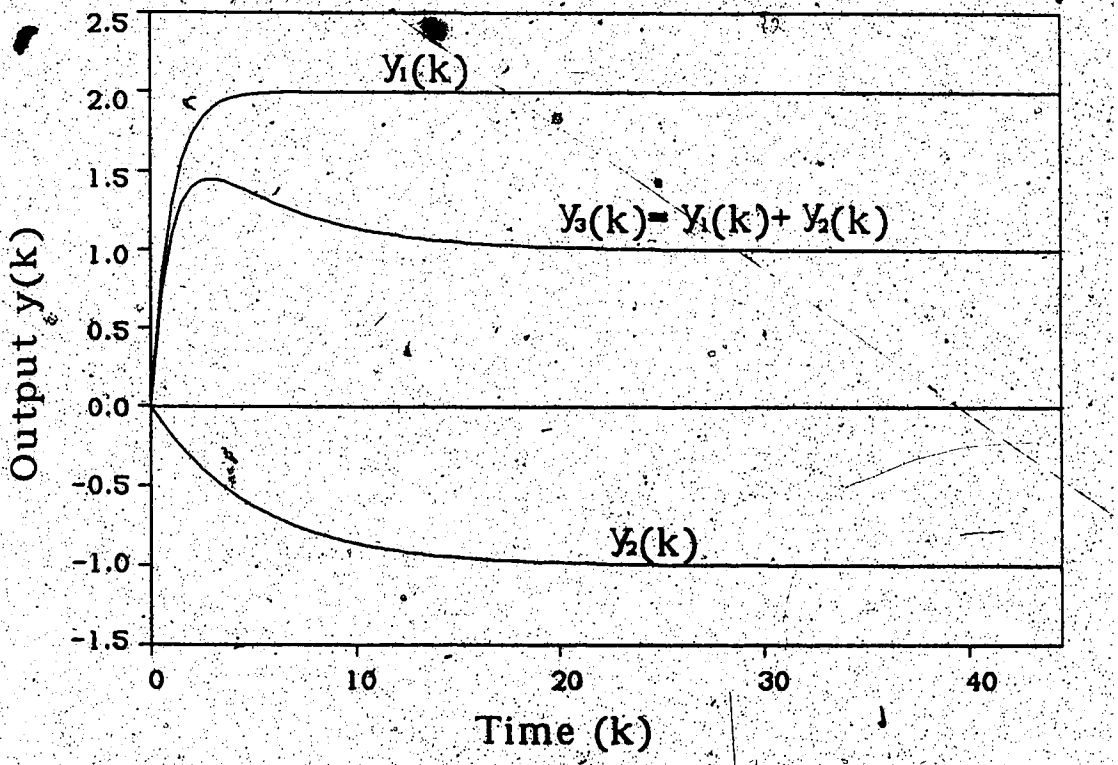

Figure 3.3: Principle of Linear Superposition

For a single step change in the input at time (k-i), i.e. $\Delta u(k-i)$, the change in the output response at time k, (assuming that at some time in the past the process was at steady state) will be:

$$\Delta y_m(k|k) = y_m(k|k) - y(-\infty|-\infty) = y_m(k|k) \tag{3-2}$$

In view of equation (3-2) and assuming multiplicative scaling and the step response data of equation (3-1). The output at time k can be rewritten as:

$$\begin{aligned} y_m(k|k) &= a_i \Delta u(k-i) \quad i \le N \\ &= a_{ss} \Delta u(k-i) \quad i > N \end{aligned} \tag{3-3}$$

If the input is changed at every previous time interval, k-i, i=1, 2, ..., ∞ by $\Delta u(k-i)$ where

$$\Delta u(k-i) = u(k-i) - u(k-i-1) \tag{3-4}$$

then assuming linear superposition, the total change in the output at time k as a result of changes in input up to and including time (k-1) is given by the summation of the right hand side of equation (3-3).

$$y_m(k|k) = \sum_{j=1}^{N} a_j \Delta u(k-j) + a_{ss} \sum_{j=N+1}^{\infty} \Delta u(k-j) \tag{3-5}$$

This follows since, by assumption, the output at any time interval k is a function only of changes in input over the past N intervals. Since, $u(-\infty) = 0$ by assumption, equation (3-5) can be rewritten as:

$$y_m(k|k) = \sum_{j=1}^{N} a_j \Delta u(k-j) + a_{ss} u(k-N-1) \tag{3-6}$$

To be able to predict a future output trajectory, $\{y_m(k+i),\ i=1,2\ldots,P;\ 1 \le P \le N\}$, the time indices in equation (3-6) can be shifted by i. Therefore replacing any occurrence of k by k+i in equation (3-6) yields:

$$y_m(k+i|k+i) = \sum_{j=1}^{N} \alpha_j \Delta u(k+i-j) + \alpha_N u(k+i-N-1) \qquad (3\text{-}7)$$

At time k, the past changes in input, $\Delta u(k-1), i=1,2\ldots,N$ are known and the present and future changes in input $\{\Delta u(k+i-1), i=1,2\ldots,M; M \le P\}$ can be chosen such that the future output trajectory tracks to the desired trajectory. The control horizon, M, can be less than the prediction horizon P, to provide greater flexibility in the design. The value of M also influences the response and robustness of the control system. The right hand side of equation (3-7) can be partitioned into terms involving past changes in input (known) and present/future changes in input (unknown):

$$\begin{bmatrix} y_m(k+1|k+1) \\ y_m(k+2|k+2) \\ \vdots \\ y_m(k+P|k+P) \end{bmatrix} = \begin{bmatrix} y_m^*(k+1|k) \\ y_m^*(k+2|k) \\ \vdots \\ y_m^*(k+P|k) \end{bmatrix} + A_2 \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+M-1) \end{bmatrix} \quad (3-8)$$

where the matrix A_2 is given by:

$$A_2 = \begin{bmatrix} a_1 & 0 & 0 & \dots & 0 \\ a_2 & a_1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_M & a_{M-1} & \vdots & \vdots & a_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_P & a_{P-1} & \dots & \dots & a_{P-M+1} \end{bmatrix}_{P\times M}$$

The contribution due to past changes $\{y_m^*(k+i|k), i=1,2\dots,P\}$ is given by:

$$\begin{bmatrix} y_m^*(k+1|k) \\ y_m^*(k+2|k) \\ \vdots \\ y_m^*(k+P|k) \end{bmatrix} = a_{ss} \begin{bmatrix} u(k-N) \\ u(k-N+1) \\ \vdots \\ u(k-N+P-1) \end{bmatrix} + A_1 \begin{bmatrix} \Delta u(k-N+1) \\ \Delta u(k-N+2) \\ \vdots \\ \Delta u(k-1) \end{bmatrix} \quad (3-9)$$

where the matrix A_1 is given by:

$$A_1 = \begin{bmatrix} a_N & a_{N-1} & \cdots & \cdots & \cdots & a_2 \\ 0 & a_N & a_{N-1} & \cdots & \cdots & a_3 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdots & 0 & a_N & \cdots & a_{P+1} \end{bmatrix}_{P\times(N-1)}$$

Equation (3-9) represents the future output prediction based on known information. Equation (3-6) is used to predict the process output at time k and equation (3-8) is used in the design of the predictive controller. The extension of SISO to MIMO system is discussed next.

3.2.2 Extension to MIMO Processes

The multivariable model can be derived by simple extension of the SISO case. Consider a r x s system where r is the number of output variables and s is the number of input variables (s ≥ r for the system to be output controllable). The multivariable model may be written as an extension of the single variable case using partitioned vectors and matrices. The current model output, equation (3-6) may be written as:

$$y_m^l(k|k) = \sum_{n=1}^{s}\left[\sum_{j=1}^{N_{ln}}\left[a_j^{ln}\Delta u^n(k-j)\right] + a_{ss}^{ln}u^n\left(k-N_{ln}-1\right)\right], l=1,\ldots,r \qquad (3\text{-}10)$$

where N_{1n} is the number of step response coefficients for each input/output pair. There are r*s values of N_{1n}. a_i^n are the step response coefficients for each input/output variable pair (including the steady state coefficient, a_{ss}). There are r*s sets of step response data.

Similarly equations (3-8) and (3-9) can be extended to include MIMO processes. After defining y and Δu as vectors, equation (3-8) becomes:

$$\begin{bmatrix} \{y_m^1(k+i|k+i)\} \\ \{y_m^2(k+i|k+i)\} \\ \vdots \\ \{y_m^r(k+i|k+i)\} \end{bmatrix} = \begin{bmatrix} \{y_m^{1*}(k+i|k)\} \\ \{y_m^{2*}(k+i|k)\} \\ \vdots \\ \{y_m^{r*}(k+i|k)\} \end{bmatrix} + A_2 \begin{bmatrix} \{\Delta u^1(k+j-1)\} \\ \{\Delta u^2(k+j-1)\} \\ \vdots \\ \{\Delta u^s(k+j-1)\} \end{bmatrix} \tag{3-11}$$

where i=1,..P_1; l=1,..,r

j=1,..M_1; l=1,...,s

P_l, l=1,...,r is the prediction horizon for each output variable

M_l, l=1,...,s is the control horizon for each input variable

and the A_2 matrix is given by

$$A_2 = \begin{bmatrix} A_2^{11} & A_2^{12} & \cdots & A_2^{1s} \\ A_2^{21} & A_2^{22} & \cdots & A_2^{2s} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ A_2^{r1} & A_2^{r2} & \cdots & A_2^{rs} \end{bmatrix}$$

where $A_2^{ij}, i=1,\ldots,r; j=1,\ldots,s$; is the A_2 matrix for each input/output pair.

Instead of extending equation (3-9) to its multivariable equivalent, a more convenient (and efficient for implementation) way is to write the MIMO system as r MISO systems:

$$\left(y_m^{le}(k+i|k)\right) = \sum_{n=1}^{s}\left[a_{ss}^{ln} I_{Pl}\left(u^n(k+i-N_{ln}-1)\right)\right. \\ \left. + A_1^{ln}\left(\Delta u^n(k+j-N_{ln})\right)\right] \qquad (3\text{-}12)$$

where i=1,..,P_1, l=1,...,r, I_{pl} = identity matrix of dimension P_1 x P_1 and j=1,..,N_{1n}-1. The use of equation (3-12) also provides additional computational flexibility since N_{1n} can be specified individually. It is important to note, at this point, that proper shifting of the Δu and u vectors must be performed during implementation of equation (3-12) to ensure proper prediction. This is because the implementation of equation (3-12) requires that Δu and u be saved in memory. The shifting of the data at every interval should be: shift the two vectors by (N_n-1) where N_n = max [(N_{1n}, l=1,...,r)]; n=1,...,s.

3.2.3 Equivalent MOCCA Model

The following sections will show that the state space, ARMA and impulse models are equivalent to the step response model, equation (3-7) used in MOCCA.

3.2.3.1 Impulse Response Model

From equation (3-7), the model prediction can be rewritten as:

$$y_m(k+i|k+i) = \sum_{j=1}^{N} a_j \Delta u(k+i-j) + a_{ss} u(k+i-N-1)$$

Replacing $\Delta u(k)$ by u(k)-u(k-1) in the above equation yields:

$$y_m^e(k+i|k+i) = \left[\sum_{j=1}^{N} a_j [u(k+i-j) - u(k+i-j-1)]\right] + a_{ss} u(k+i-N-1) \quad (3\text{-}13)$$

Expanding and collecting all the similar terms in equation (3-13) yields:

$$y_m(k+i|k+i) = a_1 u(k+i-1) + (a_2 - a_1) u(k+i-2) + \ldots + (a_N - a_{N-1}) u(k+i-N) + (a_{ss} - a_N) u(k+i-N-1) \quad (3\text{-}14)$$

The impulse and step response coefficients are related by:

$$h_i = a_i - a_{i-1} \quad (3\text{-}15)$$

with $h_1 = a_1$ since $a_0 = 0.0$ when deviation variables are used. Defining a_{ss} as a_{N+1} in equation (3-14) and substituting equation (3-15) into (3-14) the impulse response model is defined as:

$$y_m(k+i|k+i) = \sum_{j=1}^{N+1} h_j u(k+i-j) \qquad (3\text{-}16)$$

Note that there is no steady state term in equation (3-16) and that the impulse coefficients, h_j decreases as j increases.

3.2.3.1.1 Impulse Response and Convolution Integral

The impulse response of a linear system is defined as the output response of the system when the input is a unit impulse function $\delta(t)$. The use of impulse responses in control can be found in standard textbooks (e.g. Kuo, 1982). The following discussion shows how the output, y(t) of a linear system can be computed from the input, u(t) and impulse response, h(t) of the system. Assume that the input, u(t) is approximated by a sequence of pulses of pulse width $\Delta\tau$ as shown in Figure 3.4.

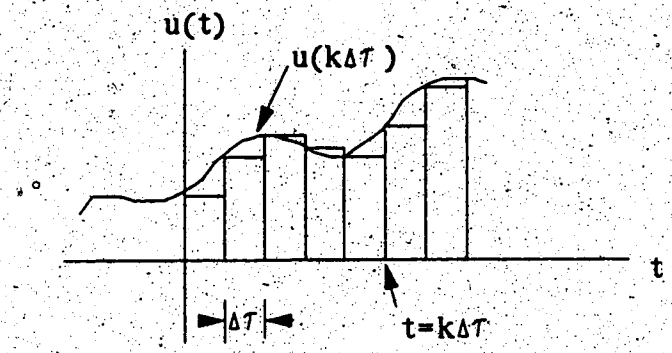

Figure 3.4: Input Signal Represented by Sum of Pulses

In the limit as $\Delta\tau$ approached zero, these pulses become impulses which have a strength of $\Delta\tau \cdot u(k\Delta\tau)$ (i.e. area of the pulse) at time $k\Delta\tau$. The output response of the linear system can now be calculated. When only the impulse at time $t=k\Delta\tau$ is considered the output response is given by:

$$y(k\Delta\tau) = \Delta\tau * u(k\Delta\tau)h(t-k\Delta\tau) \tag{3-17}$$

which is the system impulse response multiplied by the impulse strength. By the use of the superposition principle, the total response due to $u(\tau)$ is obtained by adding up the responses due to each of the impulses from $-\infty$ to $+\infty$. Therefore,

$$y(t) = \lim_{\substack{\Delta\tau \to 0 \\ N \to \infty}} \sum_{k=-N}^{N} u(k\Delta\tau)h(t-k\Delta\tau)\Delta\tau \tag{3-18}$$

or

$$y(t) = \int_{-\infty}^{\infty} u(\tau)h(t-\tau)d\tau \tag{3-19}$$

For all physical systems the output response does not precede excitation, thus $h(t-\tau)=0$ for $t<\tau$ or y(t) = 0 for t < 0 if the impulse function is applied at t=0. Equation (3-19) becomes:

$$y(t)=\int_{-\infty}^{t} u(\tau)h(t-\tau)d\tau \tag{3-20}$$

If $u(\tau)=0$ for $\tau<0$ then equation (3-20) becomes:

$$y(t)=\int_{0}^{t} u(\tau)h(t-\tau)d\tau \tag{3-21}$$

The expression (3-20) and (3-21) are called convolution integrals. Equation (3-21) represents the Laplace real convolution theorem:

$$y(t)=L^{-1}\{F_1(s)F_2(s)\}=\int_{0}^{t} f_1(t)f_2(t-\tau)d\tau \tag{3-22}$$

where $F_1(s)$ is the process transfer function and $F_2(s)$ is the Laplace transfer function of the input signal. For direct comparison to MOCCA, the discrete equivalent of equation (3-22), is given by:

$$y(k)=Z^{-1}\{E_1(z)E_2(z)\}=\left[\sum_{i=0}^{\infty} e_1(i)z^{-i}\right]\left[\sum_{j=0}^{\infty} e_2(j)z^{-j}\right] \tag{3-23}$$

where $E_1(z)$ and $E_2(z)$, are the process and input signal power series. In the case of MOCCA,

$$e_1(i) = h_1 + h_2 + \ldots + h_i + \ldots$$

$$e_2(j) = u_{(k)} + u_{(k-1)} + \ldots + u_{(k-j)} + \ldots \quad (3\text{-}24)$$

Direct multiplication of the power series involves a lot of computation as compared to the use of step response or impulse response models. However, the objective here is to show the equivalence of impulse (step) response and convolution models.

3.2.3.2 ARMA Model

The ARMA model for a SISO system can be written as:

$$A(z^{-1})y(k) = B(z^{-1})u(k) \quad (3\text{-}25)$$

where $A(z^{-1}) = 1 + a_1 z^{-1} + \ldots\ldots + a_{na} z^{-1}$

$B(z^{-1}) = b_0 + b_1 z^{-1} + \ldots\ldots + b_{nb} z^{-1}$

z^{-1} is the backward shift operator.

Equation (3-25) can be rewritten as:

$$y(k) = \frac{B(z^{-1})}{A(z^{-1})} u(k) \quad (3\text{-}26)$$

where the long division between the two polynomials yields:

$$\frac{B(z^{-1})}{A(z^{-1})} = h_0 + h_1 z^{-1} + \ldots\ldots + h_\infty z^{-\infty} \quad (3\text{-}27)$$

Combining equations (3-26) and (3-27) and using the summation notation yields:

$$y(k) = \left[\sum_{j=0}^{\infty} h_j z^{-j}\right] u(k) \qquad (3\text{-}28)$$

$$= \sum_{j=0}^{\infty} h_j u(k-j)$$

However the term b_0 in the polynomial $B(z^{-1})$ is zero due to zero order hold assumption. It can be shown that the term $h_0' = b_0$, therefore equation (3-28) can be rewritten as (by omitting the index j=0):

$$y(k|k) = \sum_{j=1}^{\infty} h_j u(k-j) \qquad (3\text{-}29)$$

where the y term is written as a prediction based on information up to and including interval (k-1). To predict i steps ahead where i=1,...,P, replace k by k+i in equation (3-29) to give:

$$y(k+i|k+i) = \sum_{j=1}^{\infty} h_j u(k+i-j) \qquad (3\text{-}30)$$

Assuming that the series is truncated after N+1 terms to give:

$$y(k+i|k+i) \sim \sum_{j=1}^{N+1} h_j u(k+i-j) \qquad (3\text{-}31)$$

which is the same as equation (3-16), which has been shown to be equivalent to the step response model. Therefore the truncated ARMA model is equal to by the truncated impulse response model.

3.2.3.3 State Space Model

Consider the following discrete state space model:

$$x(k+1) = \Phi x(k) + \Gamma u(k) \tag{3-32}$$

$$y(k) = Cx(k) + Du(k) \tag{3-33}$$

Equation (3-32) can be rewritten as:

$$zx(k) = \Phi x(k) + \Gamma u(k)$$

or expressing x(k) in terms of u(k):

$$x(k) = [zI - \Phi]^{-1} \Gamma u(k) \tag{3-34}$$

Combining equations (3-34) and (3-33) gives:

$$y(k) = \left[C(zI - \Phi)^{-1}\Gamma + D\right]u(k) \tag{3-35}$$

or in terms or backward shift operator

$$y(k) = \left[C\left(I - z^{-1}\Phi\right)^{-1} z^{-1}\Gamma + D\right]u(k) \tag{3-36}$$

Comparing equation (3-36) and (3-26) shows that if the input/output behavior is identical, then:

$$\frac{y(k)}{u(k)} = \frac{B\left(z^{-1}\right)}{A\left(z^{-1}\right)} = \left[C\left(I - z^{-1}\Phi\right)^{-1} z^{-1}\Gamma + D\right]$$

which was shown in the preceding section to be equivalent to the impulse or step response models.

3.3 State Space Formulation

A state space form of the step response model formulated earlier can be also be derived, which reduces the computation time and permits the use of state space theorems and techniques. The original idea for the state space approach is presented in Li et al. (1986). In the following sections, the full order and reduced order state space model formulations will be discussed followed by a discussion of some of their properties.

3.3.1 Full Order Formulation of Recursive Estimator

A recursive relationship for estimating current and future values of the process output $\{y(k+i),\ i=0,1,\ldots,N\}$ can be developed as follows. Assume that at some arbitrary point in time k the output $\{y_m(k+i|k-1),\ i=0,1,\ldots,N\}$ is known. Although the future trajectory is not measurable an estimate can be obtained, for example, by assuming the process is at steady state; using the regular step response model; or the future values can be set equal to the current measurement. Now assume that a change in the input variable $\Delta u(k-1)$ is made at time (k-1) and that it is desired to estimate the new output trajectory i.e. to add the effect of

$\Delta u(k-1)$ to the known trajectory. From the definition of the step response data (and noting that k always defines the current computational interval) it follows that

$$y_m(k|k) = y_m(k+1|k-1) + a_1 \Delta u(k-1) \tag{3-37}$$

Extending equation (3-37) to the entire trajectory, i=0,1,...,N, gives

$$y_m(k+i|k) = y_m(k+i+1|k-1) + a_{i-1} \Delta u(k-1) \tag{3-38}$$

with $a_{i+1} = a_{ss}$ for $i \geq N$. However, by definition the effect of $\{\Delta u(k-i), i=1,...\}$ on the output predictions is constant for $i \geq N$. Therefore for i=N,

$$y_m(k+N+1|k-1) = y_m(k+N|k-1) \tag{3-39}$$

Equations (3-38) and (3-39) can be put in a more compact vector/matrix notation, using the second part of the time index used for $y_m(\cdot|k)$ as the time index k in X(k) (i.e. the computational interval):

$$X(k) = \Phi X(k-1) + \Theta \Delta u(k-1) \tag{3-40}$$

where

$$X(k) = [y_m(k|k) \quad y_m(k+1|k) \quad \dots \quad y_m(k+N|k)]^T_{(N+1)\times 1}$$

$$\Phi = \begin{bmatrix} 0 & 1 & 0 & \dots & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \dots & \dots & 0 & 0 & 1 \\ 0 & \dots & \dots & 0 & 0 & 1 \end{bmatrix}_{(N+1)\times(N+1)}$$

$$\theta = [a_1 \quad a_2 \quad \cdots \quad a_N \quad a_{ss}]^T_{(N+1)\times 1}$$

The output equation, Y(k) = $y_m(k|k)$ (i.e. the current estimate) can be obtained by using:

$$Y(k) = HX(k) \tag{3-41}$$

where

$$H = [1 \quad 0 \quad 0 \quad \cdots \quad 0]_{1\times(N+1)}$$

Note that equations (3-40) and (3-41) has the same form as a state space model and that ϕ, θ and H have very special (sparse) forms.

3.3.2 Reduced Order Formulation

The dimension of the state vector in equation (3-40) can be reduced from N+1 to P+1 as shown in Appendix B. Equation (3-40) is still valid if the vectors and matrices are defined as:

$$X(k) = [y_m(k|k) \quad y_m(k+1|k) \quad \cdots \quad y_m(k+P|k)]^T_{(P+1)\times 1}$$

$$\Delta U(k-1) = [\Delta u(k-N+P-1) \quad \cdots \quad \Delta u(k-1)]^T_{(N-P+1)\times 1}$$

$$\phi = \begin{bmatrix} 0 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \cdot & \cdot & \cdot & & & \cdot \\ \cdot & \cdot & \cdot & & & \cdot \\ 0 & \cdots & \cdots & 0 & 0 & 1 \\ 0 & \cdots & \cdots & 0 & 0 & 1 \end{bmatrix}_{(P+1)\times(P+1)} \tag{3-42}$$

$$\Theta = \begin{bmatrix} 0 & \dots & \dots & 0 & a_1 \\ 0 & \dots & \dots & 0 & a_2 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ h_{ss} & h_N & \dots & h_{P+2} & a_{P+1} \end{bmatrix}_{(P+1)\times(N-P+1)}$$

Note that the input term in equation (3-40) becomes a matrix/vector product rather than a vector/scalar product and the output equation remains the same. The lower dimension of the state vector simplifies the observers and filters formulated in later sections.

3.3.3 MIMO Extensions

The multivariable state space model with r output and s input variables can be constructed by simply redefining the vectors and matrices as block vector and matrices:

$$X(k) = \left[\{y_m^1(k+i|k)\} \quad \dots \quad \{y_m^r(k+i|k)\} \right]^T$$

$$\Phi = \begin{bmatrix} \Phi_{11} & 0 & \dots & \dots & 0 \\ 0 & \Phi_{22} & 0 & \dots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \dots & \dots & 0 & \Phi_{rr} \end{bmatrix}$$

$$\Theta = \begin{bmatrix} \Theta_{11} & \Theta_{12} & \dots & \dots & \Theta_{1s} \\ \Theta_{21} & \Theta_{22} & \dots & \dots & \Theta_{2s} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \Theta_{r1} & \Theta_{r2} & \dots & \dots & \Theta_{rs} \end{bmatrix} \qquad (3\text{-}43)$$

$$H = \begin{bmatrix} H_{11} & 0 & \dots & \dots & 0 \\ 0 & H_{22} & 0 & \dots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \dots & \dots & 0 & H_{rr} \end{bmatrix}$$

The contents of the block matrices depend on the type of model used, i.e. full or reduced order. The MIMO input vector for the full and reduced order model is:

$$\Delta U(k-1) = [\Delta u^1(k-1) \quad \ldots \quad \Delta u^s(k-1)]^T \tag{3-44}$$

and

$$\Delta U(k-1) = [\Delta U^1(k-1) \quad \ldots \quad \Delta U^s(k-1)]^T \tag{3-45}$$

respectively.

3.3.4 State Space Properties

The process model defined by equations (3-40) and (3-41) or (3-42) is a (non-minimal) state space model. Thus many of the theorems and results from the state space area can be applied to interpret and improve model predictive control schemes. In the following sections we will look at the computational efficiency and observability of the state space formulation.

Computational Efficiency of Prediction: Control schemes such as MOCCA (Sripada and Fisher, 1985) use equation (3-9) to predict the effect of past inputs on the future output trajectory. The recursive equations represented by equations (3-40) and (3-42) offer a much more efficient alternative. For a SISO system with N=40 and P=40, equation (3-9) requires 396 multiplications and 385 additions. The recursive relationship in equation (3-40)

requires 41 multiplications and 41 additions. The reduced order recursive relationship in equation (3-42) requires 41 multiplications and 51 additions. Since the prediction must be made at every control interval, the computational advantages of equations (3-40) and (3-42) over equation (3-9) is significant.

Observability: The observability matrix for the SISO system represented by equation (3-40) is

$$\left[H, H\Phi, \ldots, H\Phi^{N-1}\right]^T = I_N$$

Thus the system is completely observable and the state (cf. predicted output trajectory) can be reconstructed from the measured output $y_p(k)$ and the known input.

3.4 Feedback Correction and Feedforward Contribution

3.4.1 Introduction

The preceding section described the open loop predictions of the output variables using a step response model. A perfect model will result in perfect prediction and hence provides a basis for perfect servo/regulatory control. However, prediction error, and hence control offset will result when there is model process mismatch or unknown disturbances. In this section the use of different feedback paths to correct for unmeasured disturbances and model process mismatch will be discussed. The easiest type

of feedback correction to employ is to assume that future residuals are equal to the current residual (i.e. the difference between measured and predicted output as shown in Figure 2.1). A better method is to use disturbance predictors in the feedback path to predict the future effect of unmeasured disturbances. Observers and filters based on the state space model can also be used in the feedback path to provide a better prediction of the future effect of unmeasured disturbances and model process mismatch. An alternative feedback path (cf. Richalet et al., 1978) will also be discussed.

If the disturbances are measurable then feedforward control can be introduced to provide better control. Servo control can also be made more flexible by the introduction of a supervisory control system.

3.4.2 State Space Based Observers and Filters

The future outputs of the process cannot be measured directly but the state vector X(k) can be reconstructed by means of observers and filters based on the state space model. The use of an open loop predictor, closed loop observer and Kalman filter to estimate the future trajectory will be discussed in the following sections.

3.4.2.1 Open Loop Predictor

The simplest form of the linear predictor used in MOCCA (cf. Figure 2.1) is equivalent to a simple open loop predictor (cf. Figure 3.5) defined by:

$$\bar{X}(k) = \Phi \bar{X}(k-1) + \Theta \Delta U(k-1) \tag{3-46}$$

$$\hat{X}(k) = \left[\bar{X}(k) + K\left[y_p(k) - y_m^*(k|k)\right]\right] \tag{3-47}$$

$$\hat{Y}(k) = H_P \hat{X}(k) \tag{3-48}$$

where

$$y_m^*(k|k) = H\bar{X}(k)$$

$$H = [1,0,\ldots,0]_{1\times(P+1)}$$

is used to extract $y_m^*(k|k)$ from $\bar{X}(k)$.

$$H_P = \begin{bmatrix} 0 & 1 & 0 & \ldots & \ldots & 0 \\ 0 & 0 & 1 & 0 & \ldots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \ldots & \ldots & \ldots & 0 & 1 \end{bmatrix}_{P\times(P+1)}$$

is used to extract $\{y_m^*(k+i|k), i=1,\ldots,P\}$ from the vector $\hat{X}(k)$.

$$K = \left[k_1,\ldots,k_P,k_{P+1}\right]^T_{(P+1)\times 1}$$

and

$\hat{Y}(k)$ is the corrected, model-based, predicted trajectory of the process.

In equation (3-48) only the last P values are used. The first value of $\hat{Y}(k)$ says that the best estimate of the current process output is the measured output itself (i.e.

we know this intuitively). The "gains" k_i are set equal to one if the future estimates of the disturbance and/or model process mismatch are assumed to be equal to the current estimate. If non-unity values of k_i are used to weight the future estimates then there will be offset in the steady state predictions. Figure 3.5 shows a schematic diagram of the open loop predictor.

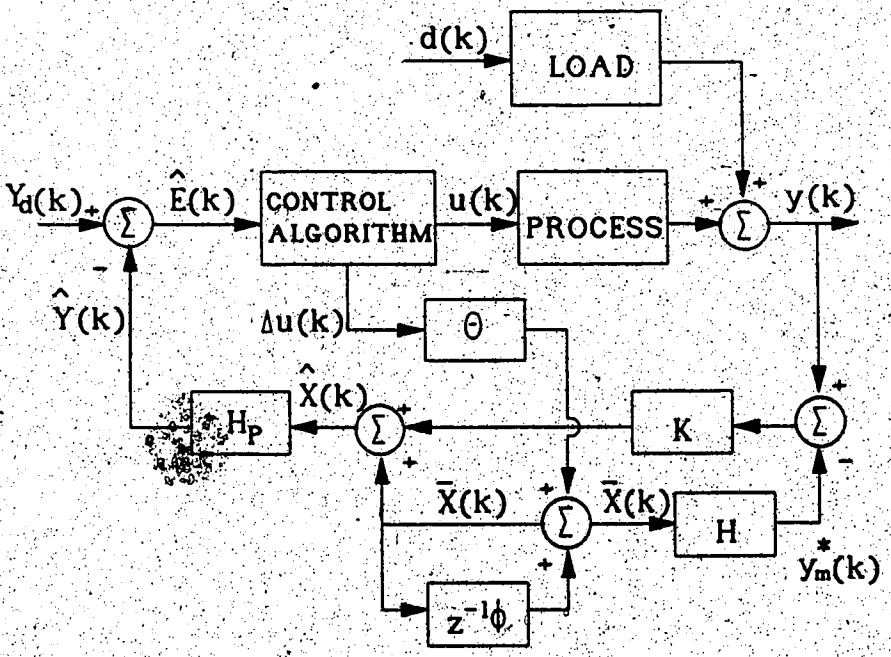

Figure 3.5: Schematic Diagram of Open Loop Observer

3.4.2.2 Closed Loop Observer

Although the observer represented by equation (3-46) to (3-48) is simple and efficient, the state space literature suggests that a closed loop, asymptotic observer would give

better performance (i.e. prevent the estimate $\hat{Y}(k)$ from drifting away from the actual values y(k) due to model process mismatch and/or disturbances). The derivation is presented in Appendix C. The closed loop observer can be defined as follows:

$$\bar{X}(k) = \Phi \hat{X}(k-1) + \Theta \Delta U(k-1) \tag{3-49}$$

$$\hat{X}(k) = \bar{X}(k) + K\left[y_p(k) - y_m^*(k|k)\right] \tag{3-50}$$

$$\hat{Y}(k) = H_p \hat{X}(k) \tag{3-51}$$

where K is chosen such that the observer is asymptotically stable (see Appendix C). The observer corresponding to equations (3-49) to (3-51) is shown schematically in Figure 3.6.

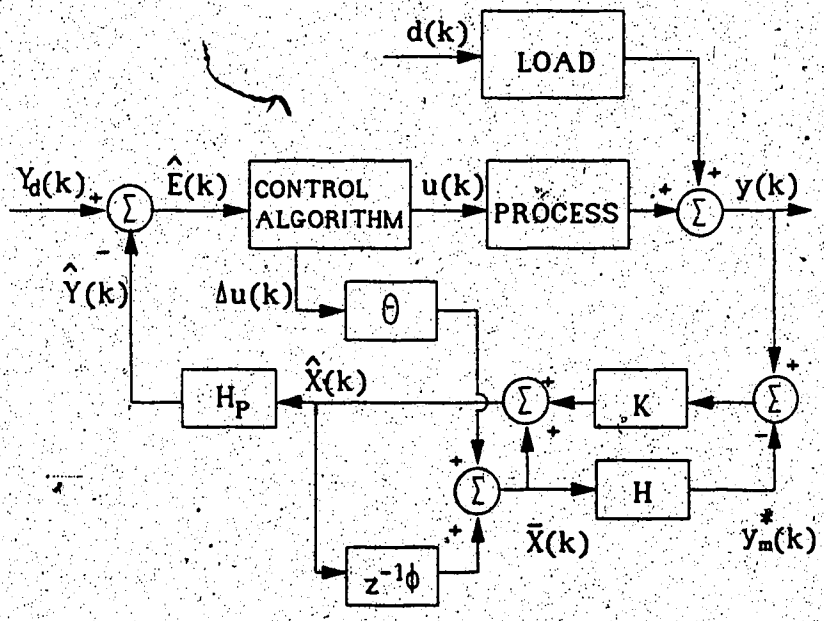

Figure 3.6: Schematic Diagram of Closed Loop Observer

It has the advantage over Figure 3.5 in that error between $\hat{X}(k)$ and X(k) tends to zero as time increases, even in the presence of sustained step type disturbances. (This is due to the "closed loop" in the observer in Figure 3.6.)

3.4.2.3 Kalman Filter

When process noise, w(k) and measurement noise, v(k) are present, then the errors in the state estimation will not tend to zero for a deterministic observer. An optimal stochastic filter, such as the Kalman filter will generally give better performance than a deterministic observer in this case. The stochastic state space model of the process can then be given by:

$$X(k) = \Phi X(k-1) + \Theta \Delta u(k-1) + Dw(k-1) \tag{3-52}$$

$$Y(k) = HX(k) + v(k) \tag{3-53}$$

If w(k) and v(k) are independent white noise sequences then the standard Kalman filter design procedure (Åström, 1970; Åström and Wittenmark, 1984; Goodwin and Sin, 1984) leads to

Time Update of the State

$$\bar{X}(k) = \Phi \hat{X}(k-1) + \Theta \Delta U(k-1) \tag{3-54}$$

Time Update of Covariance

$$M(k) = \Phi P(k-1) \Phi^T + D R_w D^T \tag{3-55}$$

Gain Calculation

$$K(k) = M(k)H^T[HM(k)H^T + R_v]^{-1} \quad (3\text{-}56)$$

Measurement Update of the State (and Predicted Output)

$$\hat{X}(k) = \bar{X}(k) + K(k)[y_p(k) - \bar{y}_m(k|k)] \quad (3\text{-}57)$$

$$\hat{Y}(k) = H_p\hat{X}(k) \quad (3\text{-}58)$$

Measurement Update of Covariance

$$P(k) = M(k) - K(k)HM(k) \quad (3\text{-}59)$$

where R_w = covariance of w(t)

R_v = covariance of v(t)

M(k) = a priori error covariance

P(k) = a posteriori error covariance

K(k) = Kalman gain

Note that the matrix D, which appears only in equation (3-55) is usually an unknown. In practice this is compensated by "tuning" using R_w. In most applications, a steady state Kalman filter will give satisfactory performance. In this case K(k) is constant and can be calculated off-line by integration of equations (3-55), (3-56) and (3-59). The state estimation is then reduced to equations (3-54), (3-57) and (3-55) as illustrated by Figure 3.7. (Note that the feedback path is the same as for the observer in Figure 3.6. The only difference is in the method of calculating K.)

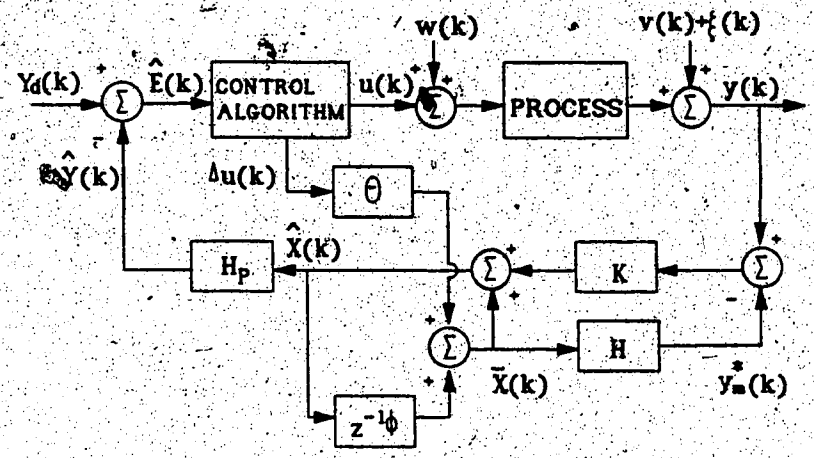

Figure 3.7: Schematic Diagram of a Kalman Filter Predictor

Walgama (1986) showed that a Kalman filter such as that defined by equations (3-54) to (3-59) exhibited prediction error when subjected to sustained disturbances. He showed that this prediction error could be eliminated by augmenting the Φ matrix to include an integrator. This modification of the Kalman filter is not required in the MOCCA application because the integration is inherent due to the use of step response type model.

3.4.3 Disturbance Prediction

The observers and filter discussed in the previous section can be interpreted as model based output prediction combined with disturbance prediction. The purpose of the predictor in the feedback path in Figure 2.1 is to model and predict the effect of disturbances and model process

mismatch. The use of single series forecasting and a Kalman filter as disturbance predictors are discussed in the following sections.

3.4.3.1 Single Series Forecasting

A unified approach to disturbance modeling and prediction can be developed (Åström, 1980a; Åström and Wittenmark, 1984) by using a "disturbance generator" of the form:

$$v(k) = \frac{C(q^{-1})}{A(q^{-1})} d(k) \qquad (3\text{-}60)$$

where q^{-1} is the backward shift operator. A and C are polynomials with degree of C < degree of A and C assumed stable. d(k) is a set of isolated pulses or an independent white noise sequence. Depending on the polynomials A and C, the disturbance v(k) may be a step, ramp or more complicated signal. Note that this representation is unique and is a convenient means of handling both deterministic and stochastic disturbances. The polynomials C and A can be determined in a number of ways including open loop response testing as used in MOCCA's process model.

Prediction of v(k) can be achieved by defining unique polynomials F and G such that:

$$\frac{C(q^{-1})}{A(q^{-1})}d(k+m) = F(q^{-1})d(k+m) + \frac{G(q^{-1})}{A(q^{-1})}d(k) \qquad (3\text{-}61)$$

where

$$F(q^{-1}) = 1 + f_1 q^{-1} + \ldots + f_{m-1} q^{-m+1}$$

$$G(q^{-1}) = 1 + g_1 q^{-1} + \ldots + g_{n-1} q^{-n+1}; n = deg(A)$$

Equation (3-60) can be rewritten for m-step prediction:

$$v(k+m) = \frac{C(q^{-1})}{A(q^{-1})}d(k+m) \qquad (3\text{-}62)$$

Substituting equation (3-62) into the left hand side of equation (3-61) yields:

$$v(k+m) = F(q^{-1})d(k+m) + \frac{G(q^{-1})}{A(q^{-1})}d(k) \qquad (3\text{-}63)$$

Since d(k) is an independent white noise sequence, we cannot predict d(k+m), hence setting d(k+m)=0 in equation (3-63) yields:

$$\hat{v}(k+m) = \frac{G(q^{-1})}{A(q^{-1})}d(k) \qquad (3\text{-}64)$$

Substituting d(k) from equation (3-60) into equation (3-64) yields:

$$\hat{v}(k+m) = \frac{G(q^{-1})A(q^{-1})}{A(q^{-1})C(q^{-1})}v(k) = \frac{G(q^{-1})}{C(q^{-1})}v(k) \qquad (3\text{-}65)$$

To obtain MOCCA notation, replace v by y, (cf. Figure 2.1):

$$y_F(k+m) = \frac{G(q^{-1})}{C(q^{-1})} y_F(k) = \left[\sum_{i=0}^{r} \omega_i q^{-i}\right] y_F(k) \qquad (3\text{-}66)$$

where the r^{th} order polynomial is the truncated polynomial after the long division. Therefore the disturbance predictor for both deterministic and stochastic systems is an r^{th} order polynomial. Note that the predicted residual includes model mismatch and unmeasured disturbances. Equation (3-66) can be described as a single series forecaster, to emphasize that it is not an input-output model. The coefficients ω_i are obtained from open loop response data or estimated on-line and the summation of $\omega_i, i=1,\ldots,r$ is unity to avoid steady state offset. Man (1984) in his thesis on single series forecasting (for one step ahead deadbeat controllers) suggested that since calculation is done at every sampling interval, a one step ahead predictor or forecaster is sufficient, i.e. m=1 in equation (3-66). In the case where a trajectory is needed then, equation (3-66) is used to calculate the value, $y_F(k+1)$ and the future values are set equal to the single step forecast. Alternatively equation (3-66) can be used recursively to generate the trajectory $\{y_F(k+i), i=1,\ldots,P\}$. For multivariable systems, equation (3-66) can be rewritten to accommodate multiple outputs:

$$\hat{y}_F^l(k+m) = \left[\sum_{i=0}^{r_l} \omega_i^l q^{-i}\right] y_F^l(k); l = 1,\ldots,n \qquad (3\text{-}67)$$

Man (1984) showed that the use of single series forecasting can significantly improve the performance of predictive control systems.

3.4.3.2 Disturbance Prediction Using Kalman Filter

The Kalman filter may be modified to handle disturbances by augmenting a disturbance model into the state space equation for the plant (Candy, 1986). Note that because MOCCA uses input/output approach the disturbances are assumed to add to the outputs only (i.e. they do not directly affect the process states, as shown in Figure 3.8). One way of modeling the disturbance is to use the "representation theorem" (Åström, 1970), that is any stationary correlated process can be modeled by driving a linear system with white noise, as shown in Figure 3.8. This implies that the disturbance system can be modeled by a Gauss-Markov representation:

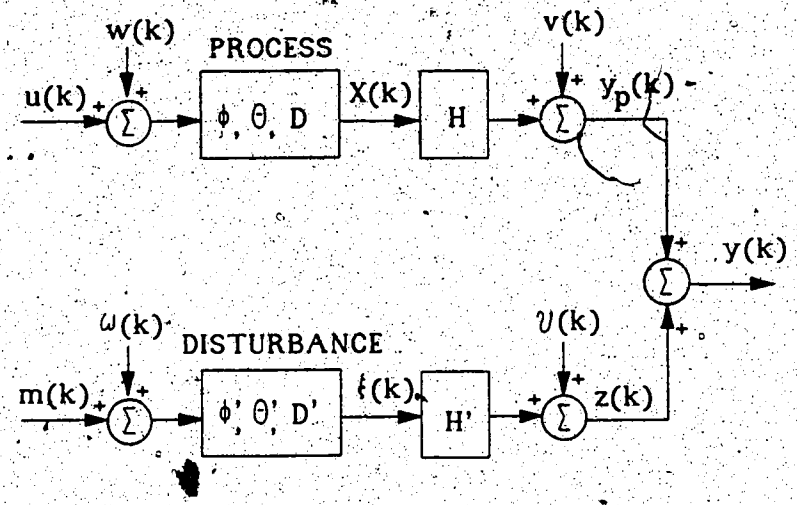

Figure 3.8: Schematic Diagram of a Disturbance Model

$$\xi(k) = \Phi\,\xi(k-1) + \Theta\,m(k-1) + D\,\omega(k-1) \tag{3-68}$$

$$z(k) = H\,\xi(k) + \upsilon(k) \tag{3-69}$$

where ω and υ are Gaussian white noise and m(k) is an external input variable that can drive the disturbance. Augmenting equations (3-68) and (3-69) into equations (3-52) and (3-61) respectively yields:

$$X_a(k) = \Phi_a X_a(k-1) + \Theta_a U_a(k-1) + D_a W_a(k-1) \tag{3-70}$$

$$Y_a(k) = H_a X_a(k) + V_a(k) \tag{3-71}$$

where

$$X_a(k) = \begin{bmatrix} \xi(k) \\ X(k) \end{bmatrix}$$

$$\Phi_a = \begin{bmatrix} \Phi & 0 \\ 0 & \Phi \end{bmatrix}$$

$$\theta_* = \begin{bmatrix} \theta \\ \theta \end{bmatrix}$$

$$U_*(k-1) = [m(k-1) \quad \Delta u(k-1)]$$

$$D_* = \begin{bmatrix} D & 0 \\ 0 & D \end{bmatrix}$$

$$W_*(k-1) = [\omega(k-1) \quad w(k-1)]$$

$$Y_*(k) = [z(k) \quad y_p(k)]$$

$$H_* = \begin{bmatrix} H & 0 \\ 0 & H \end{bmatrix}$$

$$V_*(k-1) = [v(k-1) \quad v(k-1)]$$

The Kalman filter for equations (3-70) and (3-71) is identical to the one discussed in equations (3-62) through (3-67) with the exception that the augmented model is used in the estimator. The corrected output trajectory, $\hat{Y}(k)$ can then be calculated by adding together the output predictions and disturbance predictions (both extracted from $\hat{X}_*(k)$

$$\hat{Y}(k) = [H_p \quad H_p]\hat{X}_*(k) \qquad (3\text{-}72)$$

where Hp is the matrix (defined earlier) used to extract the output predictions, $\{\hat{y}_*(k+i|k), i=1,\ldots,P\}$ and H_p is (same as Hp) used for extracting the disturbance predictions, $\hat{z}(k) = \{\hat{z}(k+i|k), i=0,\ldots,P\}$ from the state vector.

An equivalent way of interpreting the prediction of disturbances is to consider the design of a separate Kalman filter for the disturbance system described by equations (3-68) and (3-69), with the "measurement" of the disturbance calculated from the difference between current measured output and predicted output. The predicted disturbance values can then be added to the predicted process outputs from the second Kalman filter. Figure 3.9 illustrates this approach. Note that the two Kalman filters in Figure 3.9 are directly analogous in function to the two boxes in Figure 2.1, i.e. model based prediction of output trajectory $\hat{Y}(k)$ plus prediction of the residual trajectory, $\hat{Z}(k)$.

The first Kalman filter estimates the future output trajectory without considering the disturbance. The second Kalman filter, estimates the future effect of the disturbance based on the current and past residuals. The corrected estimate of the future output trajectory is then given by:

$$\hat{Y}(k) = H_p \hat{X}(k) + H_p \hat{\xi}(k) \qquad (3\text{-}73)$$

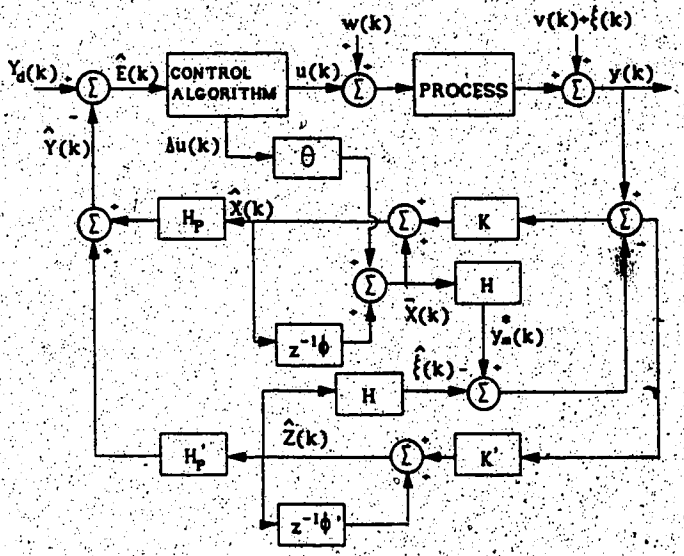

Figure 3.9: Schematic Diagram of Dual Kalman Filter Predictor

However, in Figure 3.9 the two calculations are not independent because the Kalman filters are driven by the same innovations sequence $(y_p(k)-y_m(k))$. The use of two Kalman filters is computationally more efficient than a single Kalman filter in this case.

3.4.4 Supervisory System and Alternate Feedback Path

In the most general case, the supervisory system calculations can be based on input from process operators or from higher level control programs plus feedback of information such as output variables, manipulated variables, residuals, constraints etc. Calculations can range from complex constrained optimization to simple filtering of the

setpoint. Typical applications are calculation of time varying gains and/or constraints used by the MOCCA controller.

One option, which resembles the approach used by Model Predictive Heuristic Control (MPHC) (Richalet et al., 1978), is to filter step inputs in setpoints such that $(y_d(k+i|k), i=1,...,P)$ exhibits the desired dynamics. In this case MOCCA tries to control the process such that the process outputs follows the desired dynamics (i.e. the filtered setpoint).

An alternative feedback path can be obtained by modifying the setpoint filter so that $y_d(k)$ is equal to the current measured output $y_p(k)$ at every control interval. For a first order filter then:

$$y_d^l(k+i) = \beta^l y_d^l(k+i-1) + (1.0-\beta^l) y_p^l(k) \qquad (3\text{-}74)$$

where $i=1,...,P_1$; $l=1,...,r$ and $y_d^l(k)=y_p^l(k)$. This feedback path was used by Richalet et al. (1978) in their MPHC algorithm. It can be shown that the MPHC filter is equivalent to a regular filter (i.e. no feedback) on the setpoint and a exponential discounting filter on the residuals between the desired and actual outputs, $y_p(k)-y_d(k+1|k-1)$ (see Appendix D). A similar feedback path can be implemented in MOCCA by replacing the feedback filter and disturbance predictor (cf. Figure 2.1) by the exponential discounting filter. The only difference is that

the exponential discounting filter in this modification of MOCCA would act on the residuals between the estimated and actual outputs, which includes modeling errors. (Including the modeling errors would appear to be advantageous but can introduce steady state prediction offset.) The choice of β influences the process response to both setpoint and load changes and influences the closed loop stability and robustness.

3.4.5 Feedforward Control

The concept of feedforward control in MOCCA is the same as for classical feedforward control. However the implementation is slightly different due to the fact that MOCCA uses trajectories and constraints. In order to facilitate the implementation of constraints the feedforward controller calculations are combined with the predictive controller. Appendix E shows that the desired feedforward control can still be achieved using the predictive controller. Figure 3.10 illustrates how MOCCA handles measurable disturbances using feedforward concepts.

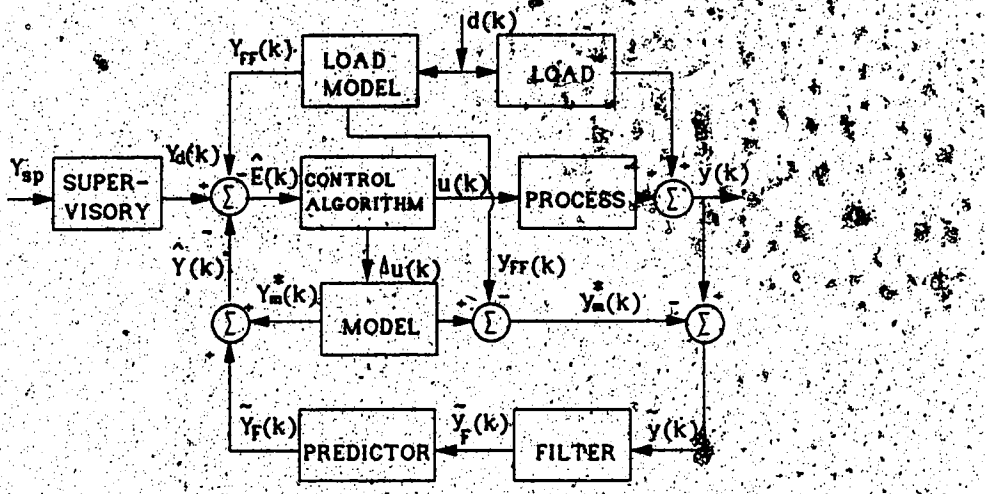

Figure 3.10: Schematic Diagram of MOCCA with Feedforward Control

The effect of measured disturbance, d(k), on the output variables can be represented using the step response relationship between the disturbance, d(k), and output, y(k), as discussed earlier. Specifically equations (3-8) and (3-9) are still valid, assuming that the step response coefficients due to a step change in the disturbance are available.

Since future values of the disturbance cannot be measured the simplest case is to set the future disturbances equal to the current (measured) disturbance or equivalently:

$$\Delta d(k+i) = 0 \quad \text{for} \quad i \geq 1 \tag{3-75}$$

Equations (3-8) and (3-9) can then be combined to give (using b_i as the disturbance step response coefficients):

$$\begin{bmatrix} \hat{y}_{FF}(k+1|k) \\ \hat{y}_{FF}(k+2|k) \\ \vdots \\ \hat{y}_{FF}(k+P|k) \end{bmatrix} = a_{ss} \begin{bmatrix} d(k-N_d+1) \\ d(k-N_d+2) \\ \vdots \\ d(k-N_d+P) \end{bmatrix} + B_1 \begin{bmatrix} \Delta d(k-N_d) \\ \Delta d(k-N_d+1) \\ \vdots \\ \Delta d(k) \end{bmatrix} \quad (3\text{-}76)$$

where N_d is number of disturbance step response coefficients and B_1 is given by:

$$B_1 = \begin{bmatrix} b_N & b_{N-1} & \cdots & \cdots & \cdots & b_2 & b_1 \\ 0 & b_N & b_{N-1} & \cdots & \cdots & b_3 & b_2 \\ \vdots & & & & & & \vdots \\ 0 & \cdots & 0 & b_N & \cdots & b_{P+1} & b_P \end{bmatrix}_{P\times(N-1)}$$

The MOCCA feedback path must also be modified when feedforward is active because the output of the controller $\Delta u(k)$ includes a feedforward component, $\Delta u_{FF}(k)$, as well as the control action normally generated by MOCCA for servo and regulatory control, $\Delta u_N(k)$, i.e. $\Delta u(k) = \Delta u_{FF}(k) + \Delta u_N(k)$. Note that $\Delta u_{FF}(k) + \Delta u_N(k)$ is fed to the process model block (see Figure 3.10), so that the predicted output $\hat{y}(k)$ includes the effect of feedforward control. The prediction of the current output from this block will also be affected by the feedforward action. When calculating $y_m(k)$ for use in

calculating the residual, it is necessary to remove the feedforward effect. Hence, as shown in Figure 3.10 the estimated effect of the disturbance, d(k) on y at time k is added to the output of the process model block.

Once the effect of the disturbance on the future output is calculated, a new error trajectory can be defined as shown in Figure 3.10:

$$\tilde{e}(k+i) = y_d(k+i) - \tilde{y}(k+i) - \tilde{y}_{FF}(k+i); \quad i = 1,\ldots,P \tag{3-77}$$

The presence of disturbance model mismatch will deteriorate the performance of the feedforward control but will not affect the implementation described above. Mismatch between the process and the process model will appear the in feedback residual, $\tilde{y}$ and in the controller calculation. Mismatch between the load step response and the actual load will also appear in the feedback residual.

MQCCA can handle known time delays in the process and/or the load. The step responses for the process and load are entered "as recorded" during an experimental step response, i.e. all the leading zeros due to delays are left in the data. Assume that the process and load time delays are τ_{DP} and τ_{DL}. There are two cases to consider:

1. $\tau_{DL} > \tau_{DP}$: in this case the trajectory $\tilde{y}_{FF}$ is delayed by $(\tau_{DL} - \tau_{DP} + 1)$ before being subtracted from the setpoint trajectory.

2. $\tau_{DL} < \tau_{DP}$: In this case the leading τ_{DP} terms (counting the leading zeros) are removed from the start of the trajectory g_{ij}.

An alternative to equation (3-75) for generating the trajectory of future disturbance effects $\{Y_{ij}(k+i), i = 1, \ldots, P\}$ would be to use a predictor (such as the single series forecaster described earlier for the feedback residuals) to predict the future disturbances and use equation (3-8) to calculate the effect of the measured (current) and estimated (future) disturbances on the output variables. Extension to multiple disturbances and multiple output and state space formulations can be done as discussed earlier.

3.4.6 Adaptive MOCCA Model

Time varying processes are frequently encountered in industry. MOCCA's step response model can be made partially time varying by factoring out the gains and the time delays. In the SISO case the dominant time constant can also be made time varying. These time varying features allow the model to be changed on-line without the need to redesign the controllers. These features will be discussed in this section.

3.4.6.1 Time Varying Gains

The steady state gains of a time varying process can be compensated by factoring the ratio of the current to

previous time interval steady state gains from the step response model. The MIMO current model output can be rewritten as:

$$y_m^l(k|k) = \sum_{n=1}^{s} K(k)_p^{ln} \left[\sum_{j=1}^{N_{ln}} \left[a_j^{ln} \Delta u^n(k-j) \right] + a_{ss}^{ln} u^n\left(k - N_{ln} - 1\right) \right], \quad l = 1, \ldots r \tag{3-78}$$

Similarly, equations (3-11) may be rewritten with the ratio of the steady state gains factored out:

$$\begin{bmatrix} \{y_m^1(k+i|k+i)\} \\ \{y_m^2(k+i|k+i)\} \\ \vdots \\ \{y_m^r(k+i|k+i)\} \end{bmatrix} = \begin{bmatrix} \{y_m^{1*}(k+i|k)\} \\ \{y_m^{2*}(k+i|k)\} \\ \vdots \\ \{y_m^{r*}(k+i|k)\} \end{bmatrix} + K(k)_p A_2 \begin{bmatrix} \{\Delta u^1(k+j-1)\} \\ \{\Delta u^2(k+j-1)\} \\ \vdots \\ \{\Delta u^s(k+j-1)\} \end{bmatrix} \tag{3-79}$$

where i=1,..P_l; l=1,..,r

j=1,..M_l; l=1,...,s

P_l, l=1,...,r is the prediction horizon for each output variable

M_l, l=1,...,s is the control horizon for each input variable

and $K(k)_p$ is given by

$$K(k)_p = \begin{bmatrix} K_p^{11} & K_p^{12} & \dots & K_p^{1s} \\ K_p^{21} & K_p^{22} & \dots & K_p^{2s} \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ K_p^{r1} & K_p^{r2} & \dots & K_p^{rs} \end{bmatrix}$$

where $K_p^{ij}, i=1,\dots,r; j=1,\dots,s;$ is the K(k)p matrix for each input/output pair. Similarly equation (3-12) becomes

$$\left\{y_m^{l*}(k+i\,|\,k)\right\} = \sum_{n=1}^{s} K_p^{ln}\left[a_{ss}^{ln} I_{pl}\left\{u^n\left(k+i-N_{ln}-1\right)\right\} + A_1^{ln}\left\{\Delta u^n\left(k+j-N_{ln}\right)\right\}\right] \qquad (3\text{-}80)$$

where i=1,..,P_1, l=1,...,r, I_{pl} = identity matrix of dimension P_1 x P_1 and j=1,...,N_{ln}-1. Equations (3-78) through (3-80) allow on-line changing of the gain without resetting the step response data if the dynamics of the system remain the same. Note that the calculated input variables u^i, i=1,...,s must be compensated if the ratio is non-unity, before it is applied to the final control element as implied by equation (3-79). This separation of the gain term from the dynamic (step response) compensation allows MOCCA to use the same dynamic controller even when the gains of the system change.

This feature of MOCCA is particularly attractive in "gain scheduling" applications where the process gains can be estimated based on a knowledge of the process conditions.

3.4.6.2 Time Varying Delays

If the time delay is known, then the model can be rewritten to accommodate variable time delays. For a SISO system, with a delay of λ, a change in the input at time k will only affect the output at time greater that $(k+\lambda)$. Equation (3-8) can then be rewritten as:

$$\begin{bmatrix} y_m(k+\lambda+1\,|\,k+1) \\ y_m(k+\lambda+2\,|\,k+2) \\ \vdots \\ y_m(k+\lambda+P\,|\,k+P) \end{bmatrix} = \begin{bmatrix} y_m^*(k+\lambda+1\,|\,k) \\ y_m^*(k+\lambda+2\,|\,k) \\ \vdots \\ y_m^*(k+\lambda+P\,|\,k) \end{bmatrix} + A_2 \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \vdots \\ \Delta u(k+M-1) \end{bmatrix} \tag{3-81}$$

and equation (3-9) becomes:

$$\begin{bmatrix} y_m^*(k+\lambda+1\,|\,k) \\ y_m^*(k+\lambda+2\,|\,k) \\ \vdots \\ y_m^*(k+\lambda+P\,|\,k) \end{bmatrix} = a_{ss} \begin{bmatrix} u(k-N) \\ u(k-N+1) \\ \vdots \\ u(k-N+P-1) \end{bmatrix} + A_1 \begin{bmatrix} \Delta u(k-N+1) \\ \Delta u(k-N+2) \\ \vdots \\ \Delta u(k-1) \end{bmatrix} \tag{3-82}$$

where A_1 and A_2 are now based on the N non-zero coefficients of the step response coefficients to ensure full rank. Similarly the current model output, equation (3-6), can be rewritten to include the known time delay:

$$y_m(k|k) = \sum_{j=1}^{N} a_j \Delta u(k-\lambda-j) + a_{\infty} u(k-\lambda-N-1) \qquad (3\text{-}83)$$

At this point, it should be noted that in the implementation of MOCCA the time delay is handled differently for the case of future predictions and current process estimates. Equations (3-81) and (3-82) represent the future predictions and the known time delay is handled via the left hand side of the equation, i.e. the trajectory calculated will constitute a prediction λ steps greater than the case with zero delay. Equation (3-83) which represents the current model output handles the delay via the right hand side of the equation, producing an actual delayed output. Therefore a SISO controller designed using the step response data for a process with no delay can be used for the same process with time delay, if the delay is known.

In the MIMO case, the known delays must be expressed in terms of delays in the output variables. Let $\lambda_i, i=1,\ldots,r$ be the known time delay of each output variable, then equations (3-11) and (3-12) can be rewritten with the following changes:

1. (y_i) has the argument $(k+i+\lambda_i|k+i), i=1,\ldots,p, j=1,\ldots,r$

output trajectory can be a different value. In the case where time varying weightings are used or time varying step response coefficients are used, then on-line calculation of equation (4-15) is necessary.

The extension to the multivariable case is straight forward. The derivation and solution is the same as in the SISO case with the matrices and vectors being replaced by block matrices and vectors with the following forms.

$$\hat{E}(k)=\begin{bmatrix}\hat{E}^1(k)\\ \vdots \\ \hat{E}^P(k)\end{bmatrix} \qquad Y_d(k)=\begin{bmatrix}Y_d^1(k)\\ \vdots \\ Y_d^P(k)\end{bmatrix} \qquad \Delta U_F(k)=\begin{bmatrix}\Delta U_F^1(k)\\ \vdots \\ \Delta U_F^P(k)\end{bmatrix}$$

$$Y_F(k)=\begin{bmatrix}Y_F^1(k)\\ \vdots \\ Y_F^P(k)\end{bmatrix} \qquad \hat{Y}(k)=\begin{bmatrix}\hat{Y}^1(k)\\ \vdots \\ \hat{Y}^P(k)\end{bmatrix}$$

$$\Gamma(k)=\begin{bmatrix}\Gamma^1(k) & 0 & \cdots & \cdots & 0\\ 0 & \Gamma^2(k) & 0 & \cdots & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & \cdots & \cdots & 0 & \Gamma^P(k)\end{bmatrix}$$

$$\Gamma_m(k)=\begin{bmatrix}\Gamma_m^1(k) & 0 & \cdots & \cdots & 0\\ 0 & \Gamma_m^2(k) & 0 & \cdots & 0\\ \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & \cdots & \cdots & 0 & \Gamma_m^P(k)\end{bmatrix}$$

Due to the fact that the dynamic matrix, A_2 is constructed of shifted values of the step response data, the matrix may be ill-conditioned, i.e. a small perturbation in the data of the problem can lead to a large change in the solution. Ill-conditioning is a property of the problem itself. Noisy or inaccurate measurements (in industrial applications) will also contribute to the ill-conditioning of the A_2 matrix. The conditioning of the dynamic matrix is discussed further in Appendix G. Ill-conditioned matrices can cause the solution, (in this case $\Delta\underline{u}(k)$ to be sensitive to small changes in the error vector. To avoid computational problems due to sensitivity, the data should be normalized or scaled. It has also been suggested that the calculation of the pseudo inverse (cf. equation (4-15)) using normal Gaussian elimination can caused stability problems with regard to propagation of data errors and uncertainty. To avoid this problem, an algorithm for finding the pseudo inverse using orthogonal or elimination matrices (e.g. QR factorization or singular value decomposition) should be used (Lawson & Hanson, 1974, Gill et. al., 1981).

Although the weighted least squares problem is considered to be unconstrained, the solution can handle soft constraints by proper use of the input weighting matrix (for the multivariable case, output weighting can also be used).

By soft constraints it is meant that there is no guarantee that the constraints will not be violated. By using the control input weighting matrix the solution, Δu can be constrained to a certain maximum change for a given error.

4.3 Constrained Optimization

Although the unconstrained optimization problems discussed above have explicit solutions that can be calculated a priori in most cases, they can not handle hard constraints on the process variables. Constraints are important in industrial applications since all process variables have physical limitations and frequently the optimum operating condition lies near a set of constraints. Control problems with hard constraints can be handle by MOCCA using mathematical programming routines. The constrained control problem can be transformed into different types of constrained optimization by proper specification of the objective function and constraints. In this section, several such transformations will be discussed.

4.3.1 Linear Objective Function and Constraints

In this section the transformation of the MOCCA control problem into a standard LP problem will be discussed. A very special case of LP for SISO systems with one set of constraints will also be discussed.

4.3.1.1 Linear Programming

Linear programming (LP) requires the use of a linear objective function and linear constraints. The following discussion will involve the transformation of the MOCCA control problem with constraints into a standard LP problem. For our purposes the standard LP problem is defined as:

$$\max \; J = c^T y$$

$$\text{subject to } Ay \begin{bmatrix} \leq \\ = \\ \geq \end{bmatrix} b \qquad (4\text{-}16')$$

$$y \geq 0$$

Other LP formulations exist but this only adds to the family of MOCCA algorithms since one standard LP problem can always be transformed into another standard LP problem by proper algebraic manipulation. The first set of constraints in the LP problem can be either equality or inequality constraints. The constraint $y \geq 0$ is called the non-negativity constraint in standard LP notation. A typical MOCCA problem can be stated as:

minimize J = a linear function of input and/or output variables

subject to linear constraints on input, changes in input, output, uncontrolled output, derivative or rate of change of output variables

The MOCCA linear objective function will minimize the sum of absolute error (IAE) and the sum of the absolute changes in inputs (IAΔu). Additional flexibility can be added by introducing weighting factors on the output and input changes. The objective function is stated mathematically as:

$$\min \; J = \gamma(k)\,|Y_d(k) - \hat{Y}_F(k)| + \gamma_m(k)\,|\Delta U_F(k)| \qquad (4\text{-}17)$$

where $\gamma(k)$ and $\gamma_m(k)$ are the output and input changes weighting vectors of dimension 1xP and 1xM respectively. Substituting equation (4-11) into (4-17) yield:

$$\min \; J = \Gamma(k)\,|\hat{E}(k) - A_2 \Delta U_F(k)| + \Gamma_m(k)\,|\Delta U_F(k)| \qquad (4\text{-}18)$$

Let

$$X(k) = \hat{E}(k) - A_2 \Delta U_F(k) \qquad (4\text{-}19)$$

then the objective function becomes

$$\min \; J = \gamma(k)\,|X(k)| + \gamma_m(k)\,|\Delta U_F(k)| \qquad (4\text{-}20)$$

$$\text{subject to } X(k) = \hat{E}(k) - A_2 \Delta U_F(k)$$

This problem is still not suitable for an LP computation since the objective function in equation (4-20) is of the absolute value type and hence not linear. To linearize the objective function, the variables in the absolute sign can be decomposed into pairs of non-negative variables (difference variables method, Tabak and Kuo, 1971):

$$X(k) = X(k)^{+} - X(k)^{-} \tag{4-21}$$

$$\Delta U_F(k) = \Delta U_F(k)^{+} - \Delta U_F(k)^{-}$$

$$X(k)^{+}, X(k)^{-}, \Delta U_F(k)^{+}, \Delta U_F(k)^{-} \geq 0$$

Note that by using this decomposition, the problem of non-negativity constraints in the standard LP problem is also solved. That is the variables in the original objective function to be optimized can be physically positive or negative and the difference variables method transforms them into strictly non-negative variables. The main disadvantage of this method is that the number of LP variables is effectively doubled. The variables in the absolute sign can now be written as

$$|X(k)| = |X(k)^{+} - X(k)^{-}| = X(k)^{+} + X(k)^{-} \tag{4-22}$$

$$|\Delta U_F(k)| = |\Delta U_F(k)^{+} - \Delta U_F(k)^{-}| = \Delta U_F(k)^{+} + \Delta U_F(k)^{-}$$

and the objective function is transformed into

$$\min \; J = \gamma(k)X(k)^{+} + \gamma(k)X(k)^{-} + \gamma_m(k)\Delta U_F(k)^{+} + \gamma_m(k)\Delta U_F(k)^{-} \tag{4-23}$$

$$\text{subject to } X(k)^{+} - X(k)^{-} + A_2\Delta U_F(k)^{+} - A_2\Delta U_F(k)^{-} = \hat{E}(k)$$

The problem can now be translated into the standard LP problem by

1) multiplying the objective function, J by -1 to convert the problem from minimization to maximization. Therefore the objective function becomes

$$\max J = -y(k)X(k)^{+} - y(k)X(k)^{-} \qquad (4\text{-}24)$$
$$-y_m(k)\Delta U_F(k)^{+} - y_m(k)\Delta U_F(k)^{-}$$

(2) the vector c^T in the standard LP problem can now be written as

$$c^T = [-y, -y, -y_m, -y_m] \qquad (4\text{-}25)$$

(3) define the y vector as

$$y = \begin{bmatrix} X(k)^{+} \\ X(k)^{-} \\ \Delta U_F(k)^{+} \\ \Delta U_F(k)^{-} \end{bmatrix} \qquad (4\text{-}26)$$

(4) the constraints that appear from the transformation can be put into an equality constraint where

$$A = [I_o \quad -I \quad A_2 \quad -A_2] \qquad (4\text{-}27)$$
$$b = [\hat{E}(k)]$$

The original MOCCA objective function has now been translated into a standard LP problem with an equality constraint. Equations (4-24) to (4-27) represent part of the standard LP problem. The dimension of the optimization problem is now represented by the total number of LP variables to be optimized. The total number of LP variables (not including the slack variables created by the LP algorithm) is given by

$$2*\left[\sum_{i=1}^{NO} P(i) + \sum_{i=1}^{NI} M(i)\right] \qquad (4\text{-}28)$$

where NI and NO are the total number of input and output variables in the problem. M(i) and P(i) are the prediction and control horizon for each of the input and output variables respectively. The total number of constraints will be calculated in later sections.

The objective function, equation (4-17), and its corresponding translation can be modified by proper specification of the weighting matrices. Table 4.3 shows several possible performance indices.

Table 4.3: Other Performance Indices For Linear Programming

Output Weighting Γ	Input Weighting Γ_m	Performance Index
Identity	Zero	Simple IAE
Non-identity	Zero	Weighted IAE
Function of time	Zero	Weighting function of time or ITAE
Non-identity	Non-identity	Weighted IAE and $IA\Delta u$
Function of time	Function of time	Weighted ITAE and $ITA\Delta u$ Weighting function of time

Constraints

Constraints on process variables are very common in an industrial processes since all process variables have a physical limit. Constraints on the input, input changes, controlled output and uncontrolled output variables will be discussed in this section.

Constraints On Changes In Input: Constraints on changes in input can be written as

$$\Delta U(k)_{min} \le \Delta U(k)_F \le \Delta U(k)_{max} \qquad (4\text{-}29)$$

or

$$\Delta U(k)_F \le \Delta U(k)_{max}$$

$$\Delta U(k)_F \ge \Delta U(k)_{min}$$

where the upper and lower bounds are vectors of the same dimension and meaning (i.e. trajectories into the future) as $\Delta U(k)_F$. The constraints can be put into the LP form by decomposing $\Delta U(k)_F$ into pairs of non-negative variables. Note that this procedure effectively doubles the number of constraints in the LP problem. Equation (4-29) can be rewritten as

$$\Delta U(k)_F^+ - \Delta U(k)_F^- \le \Delta U(k)_{max} \qquad (4\text{-}30)$$

$$\Delta U(k)_F^+ - \Delta U(k)_F^- \ge \Delta U(k)_{min}$$

Input Constraints: Constraints on the input variable can be handled in the same way as the constraints on input changes:

$$U(k)_{min} \le U(k)_F \le U(k)_{max} \qquad (4\text{-}31)$$

or

$$U(k)_F \le U(k)_{max}$$

$$U(k)_F \ge U(k)_{min}$$

where $U(k)_F = (u(k+i-1), i = 1,\ldots,M)$ is the future input trajectory. To convert the future input into future changes in input,

Consider the following SISO case. $\{\Delta u(k+i-1), i=1,\ldots,M\}$ can be written explicitly for i=1,...,M (using successive substitution) as

$$i=1 \quad u(k)=u(k-1)+\Delta u(k) \tag{4-32}$$

$$i=2 \quad u(k+1)=u(k)+\Delta u(k+1)$$
$$=u(k-1)+\Delta u(k)+\Delta u(k+1)$$

or in vector matrix notation

$$U(k)_F = L\Delta U(k)_F + U(k-1) \tag{4-33}$$

where L is a MxM lower triangular matrix of ones and U(k-1) is vector of dimension Mx1 with u(k-1) as its entries. Extension to the multivariable case results in a L matrix constructed from blocks of lower triangular matrices of ones in the diagonal and U(k-1) as a vector constructed from blocks of U(k-1) vector for each input variable.

Equations (4-31) and (4-33) are combined to give

$$L\Delta U(k)_F \le U(k)_{max} - U(k-1) \tag{4-34}$$

$$L\Delta U(k)_F \ge U(k)_{min} - U(k-1)$$

Once again, using the decomposition technique, equation (4-34) can be written in LP form,

$$L\Delta U(k)_F^+ - L\Delta U(k)_F^- \le U(k)_{max} - U(k-1) \tag{4-35}$$

$$L\Delta U(k)_F^+ - L\Delta U(k)_F^- \ge U(k)_{min} - U(k-1)$$

Output Constraints: Constraints on the controlled output variable can be stated as

$$\hat{y}(k)_{min} \le \hat{y}(k)_F \le \hat{y}(k)_{max} \qquad (4\text{-}36)$$

or

$$\hat{y}(k)_F \le \hat{y}(k)_{max}$$

$$\hat{y}(k)_F \ge \hat{y}(k)_{min}$$

Replacing $\hat{y}_F$ by equation (4-8) yields:

$$A_2 \Delta U(k)_F \le \hat{y}(k)_{max} - \hat{y}(k) \qquad (4\text{-}37)$$

$$A_2 \Delta U(k)_F \ge \hat{y}(k)_{min} - \hat{y}(k)$$

Using the decomposition technique, equation (4-37) can be written in the LP form as

$$A_2 \Delta U(k)_F^+ - A_2 \Delta U(k)_F^- \le \hat{y}(k)_{max} - \hat{y}(k) \qquad (4\text{-}38)$$

$$A_2 \Delta U(k)_F^+ - A_2 \Delta U(k)_F^- \ge \hat{y}(k)_{min} - \hat{y}(k)$$

The output constraints discussed above are mainly used to provide physical constraints e.g. upper bound on the pressure of a distillation column. For regulatory control, a better way of implementing the output constraints is to use constraints on the error trajectory, $E(k)$ since the output variable setpoints or trajectories are often changed (i.e. equation (4-38) does not take into account changes in the desired trajectory). The transformation of the error trajectory constraints can be achieved by using equation (4-10).

Output Constraints On Uncontrolled Variables: There could be intermediate variables in a process or output variables that are not controlled directly but can be subject to

constraints. To be able to include an uncontrolled output variable in the optimization problem, a relationship between the variable and the input (manipulated) variables must be established, i.e. as in the MOCCA step response model. Assuming that $\hat{z}(k)=\{\hat{z}(k+i|k-1), i=1,\ldots,P_z\}$, the uncontrolled variable and the input variable can be related by the dynamic step response matrix B_2, then equations (3-8) and (3-9) are still applicable with the variable changed. Note that the uncontrolled variable can have its own prediction horizon. Equation (3-8) can be rewritten as (using the same notation as before):

$$\hat{z}_F(k) = \hat{z}(k) + B_2 \Delta U_F(k) \qquad (4\text{-}39)$$

The constraint is in the same form as that of the output variable,

$$\hat{z}(k)_{min} \le \hat{z}(k)_F \le \hat{z}(k)_{max} \qquad (4\text{-}40)$$

or

$$-\hat{z}(k)_F \le \hat{z}(k)_{max}$$

$$\hat{z}(k)_F \ge \hat{z}(k)_{min}$$

The transformation in this case is the same as that of the controlled output variables. The translated constraints are given as:

$$B_2 \Delta U_F^+ - B_2 \Delta U_F^- \le \hat{z}(k)_{max} - \hat{z}(k) \qquad (4\text{-}41)$$

$$B_2 \Delta U_F^+ - B_2 \Delta U_F^- \ge \hat{z}(k)_{min} - \hat{z}(k)$$

Other Types of Constraints: Constraints on the rate of change of the output variable can also be implemented. In this case the rate of change has to be discretized (i.e. expressed as a difference equation). The discretized rate of change can then be related to the dynamic matrix and the input changes. Any other type of constraint can be handled as long as it can be translated into a linear function of the LP variables. Upper and lower bounds on the LP variables can also be used as part of the constraints if they have physical meaning.

The inequality constraints can now be combined with the equality constraints into the standard LP constraint matrix A:

$$A = \begin{bmatrix} 0 & 0 & I & -I \\ 0 & 0 & I & -I \\ 0 & 0 & L & -L \\ 0 & 0 & L & -L \\ 0 & 0 & A_2 & -A_2 \\ 0 & 0 & A_2 & -A_2 \\ 0 & 0 & B_2 & -B_2 \\ 0 & 0 & B_2 & -B_2 \\ I & -I & A_2 & -A_2 \end{bmatrix} \qquad (4\text{-}42)$$

$$X(k)^+, X(k)^-, \Delta U_f(k)^+, \Delta U_f(k)^- \geq 0$$

and the vector b along with the associated equality and inequality sign is given by:

$$b = \begin{bmatrix} \leq \\ \geq \\ \leq \\ \geq \\ \leq \\ \geq \\ \leq \\ \geq \\ = \end{bmatrix} \begin{bmatrix} \Delta U(k)_{max} \\ \Delta U(k)_{min} \\ U(k)_{max} - U(k-1) \\ U(k)_{min} - U(k-1) \\ Y(k)_{max} - \hat{Y}(k) \\ Y(k)_{min} - \hat{Y}(k) \\ Z(k)_{max} - \hat{Z}(k) \\ Z(k)_{min} - \hat{Z}(k) \\ \hat{E}(k) \end{bmatrix} \tag{4-43}$$

The LP problem to be solved is then equation (4-16) using equations (4-25), (4-26), (4-42) and (4-43). Note that only a subset of constraints is needed in an actual application. The total number of constraints (not including the upper and lower bounds on the LP variables) in this specific case can be calculated as:

$$4*\sum_{i=1}^{NI} M(i) + 3*\sum_{i=1}^{NO} P(i) + 2*\sum_{i=1}^{NZ} P_z(i) \tag{4-44}$$

where NZ is the total number of uncontrolled variables and P_Z is the corresponding prediction horizon.

Remarks: The proposed MOCCA LP can be solved using a standard commercial LP or special LP algorithm. Since the control algorithm has to be implemented on-line, some of the special characteristic of the MOCCA LP problem should be examined and taken advantage of. Some of the special characteristics of MOCCA relative to the LP problem are:

1. the constraint matrix, A (equation (4-42)) remain constant if the step response matrix is not adapted on-line,
2. only the right hand side vector, b changes at every control interval,
3. if the constraint matrix, A (equation (4-42)) and the objective function vector, c_T (equation (4-25)) remains constant, the LP problem can be transformed into a bounded LP problem which is simpler and can be solved faster on-line (Gill et. al., 1981) and
4. if an LP algorithm can be developed such that the initial solution set does not have to be feasible, then at every control interval the previous solution can be taken as the starting solution set for the next control problem since in most cases the new solution will be the same or only slightly different from the previous one.

Note that the objective function in this case was transformed into a linear objective function at the expense of increasing the number of optimization variables. This problem can be solved by going to the quadratic objective function as discussed in section 4.3.2.

4.3.1.2 Interval Programming

Interval programming is a very special case of the LP problem. It involves the use of upper and lower bound constraints. A standard problem can be mathematically written as:

$$\min \; J = c^T y \tag{4-45}$$
$$\text{subject to } l \le Ay \le u$$
$$y \ge 0$$

where A is mxn constraint matrix, l and u are the lower and upper bound constraint vectors of dimension mx1. The solution of equation (4-45) can be found as follow. Let z, a vector of dimension mx1 be

$$z = Ay \quad \text{or} \quad y = A^* z \tag{4-46}$$
$$\text{where } A^* = \left(A^T A\right)^{-1} A^T$$

Substituting equation (4-46) into equation (4-45) yields an equivalent problem

$$\min \; J = \left[c^T A^*\right] z \tag{4-47}$$
$$\text{subject to } l \le z \le u$$
$$z \ge 0$$

The solution to equation (4-47) (a sufficient condition when minimizing a univariate function, f(x) over a bounded interval $[l,u]$ is that if $f'(l) > 0$ then $x^* = l$ and if $f'(u) < 0$ then $x^* = u$ (Gill et. al., 1981)) may be stated explicitly as

$$z_i = u_i \text{ if } \left(c^T A^*\right)_i < 0, \qquad (4\text{-}48)$$

$$z_i = l_i \text{ if } \left(c^T A^*\right)_i > 0$$

and the solution to the original problem may be obtained from equation (4-46). Since the problem can be solved explicitly, on-line computational time can be reduced significahtly.

One of the simplest applications for interval programming would be the control of a SISO loop. Traditionally PID controllers have been used in most of the SISO loops in industry. However the PID controller performance deteriorates when there is time delay in the loop, process constraints cannot be handled directly and the performance is usually not optimal. SISO MOCCA using interval programming provides an alternative to the PID controller. In this case MOCCA can handle the known time delay and the use of interval programming provides optimal performance in the presence of process constraints. Furthermore, the SISO MOCCA can include gain scheduling plus a time-varying time delay and time constant (see section 3.4.6). Note that interval programming is restricted to SISO MOCCA and one set of constraints.

The MOCCA objective function to be minimized is the IAE (following previous notation) subject to one set of constraints (in general form):

$$\min \; J = |Y_d(k) - \hat{Y}_F(k)| \tag{4-49}$$

$$\text{subject to } l \le G\Delta U_F(k) \le u$$

where the matrix G depends on the type of constraints used (see discussion on constraints in the linear programming section). For example if the constraint is on input changes then G is an identity matrix, if the constraint is on the input variable then G is a lower triangular matrix.

Substituting equation (4-11) into the objective function,

$$\min \; J = |\hat{E}(k) - A_2 \Delta U_F(k)| \tag{4-50}$$

$$\text{subject to } l \le G\Delta U_F(k) \le u$$

From equation (4-19),

$$\Delta U_F(k) = A_2^{\dagger}[\hat{E}(k) - X(k)] \tag{4-51}$$

Using equations (4-19) and (4-51), equation (4-50) may be rewritten as

$$\min \; J = |X(k)| \tag{4-52}$$

$$\text{subject to } l \le GA_2^{\dagger}[\hat{E}(k) - X(k)] \le u$$

Letting $H = GA_2^{\dagger}$, then equation (4-52) can be rewritten as:

$$\min \; J = |X(k)| \tag{4-53}$$

$$\text{subject to } l - H\hat{E}(k) \le -HX(k) \le u - H\hat{E}(k)$$

Decomposition of X(k) into two non-negative variables (cf. equation (4-21)) yield

$$\min \quad J = X(k)^{+} + X(k)^{-} \tag{4-54}$$

$$\text{subject to} \quad l - H\hat{E}(k) \le -HX(k)^{+} + HX(k)^{-} \le u - H\hat{E}(k)$$

The transformation is completed by

1) letting the vector

$$y = \begin{bmatrix} X(k)_1^{+} \\ \vdots \\ X(k)_P^{+} \\ X(k)_1^{-} \\ \vdots \\ X(k)_P^{-} \end{bmatrix}_{2P\times 1} \tag{4-55}$$

2) letting the vector

$$c^T = [1 \quad 1 \quad \cdots \quad 1 \quad 1]_{1\times 2P} \tag{4-56}$$

3) letting the matrix

$$A = \begin{bmatrix} -H & 0 \\ 0 & H \end{bmatrix}_{2M\times 2P} \tag{4-57}$$

4) letting the lower and upper bound vectors

$$l = \begin{bmatrix} l - H\hat{E}(k) \\ l - H\hat{E}(k) \end{bmatrix}_{2M\times 1} \tag{4-58}$$

$$\bar{u} = \begin{bmatrix} u - H\hat{E}(k) \\ u - H\hat{E}(k) \end{bmatrix}_{2M\times 1}$$

The control problem can now be solved explicitly as described earlier.

4.3.2 Quadratic Objective Function and Linear Constraints

The use of a quadratic objective function in process control is also popular. With the addition of linear constraints, the problem can be transformed into a quadratic programming (QP) problem or least squares with linear inequality constraints (LSI) problem (Lawson and Hanson, 1974).

4.3.2.1 Quadratic Programming

Quadratic programming (QP) involves the use of a quadratic performance index and linear constraints. We can defined a standard QP problem as:

$$\max \quad J = p^T y + \frac{1}{2} y^T D y \tag{4-59}$$

$$\text{subject to} \quad Ay \begin{bmatrix} \leq \\ = \\ \geq \end{bmatrix} b$$

$$y \geq 0$$

The above QP problem was chosen to suit an available QP algorithm. The non-negativity constraints can be removed if an appropriate algorithm can be found. This would reduces the dimension of the problem to be solved. The quadratic performance index used by MOCCA in this case is the same as that for the weighted least squares, equation (4-7). Equation (4-12) will be the starting point for the QP problem.

$$J = \frac{1}{2}[[\hat{E}(k) - A_2 \Delta U_F(k)]^T \Gamma(k)[\hat{E}(k) - A_2 \Delta U_F(k)] + \Delta U_F(k)^T \Gamma_m(k) \Delta U_F(k)] \quad (4\text{-}60)$$

Multiplying the first term out yields:

$$[\hat{E}(k) - A_2 \Delta U_F(k)]^T \Gamma(k)[\hat{E}(k) - A_2 \Delta U_F(k)] \quad (4\text{-}61)$$
$$= \hat{E}(k)^T \Gamma(k) \hat{E}(k) - 2\hat{E}(k)^T \Gamma(k) A_2 \Delta U_F(k) + \Delta U_F(k)^T A_2^T \Gamma(k) A_2 \Delta U_F(k)$$

Rearranging equation (4-61) yields:

$$J = \frac{1}{2}[\hat{E}(k)^T \Gamma(k) \hat{E}(k) - \hat{E}(k)^T \Gamma(k) A_2 \Delta U_F(k) + \Delta U_F(k)^T [A_2^T \Gamma(k) A_2 + \Gamma_m(k)] \Delta U_F(k)] \quad (4\text{-}62)$$

Now define the matrix D as,

$$D(k) = [A_2^T \Gamma(k) A_2 + \Gamma_m(k)] \quad (4\text{-}63)$$

and the vector p^T as:

$$p(k)^T = -A_2^T \Gamma(k) \hat{E}(k) \quad (4\text{-}64)$$

The objective function can then be rewritten as:

$$J = \frac{1}{2}\hat{E}(k)^T \Gamma(k) \hat{E}(k) - p(k)^T \Delta U_F(k) + \frac{1}{2} \Delta U_F(k)^T D(k) \Delta U_F(k) \quad (4\text{-}65)$$

Since the term $\frac{1}{2}\hat{E}(k)^T \Gamma(k) \hat{E}(k)$ is constant at every sampling interval (i.e. it is not a function of the QP variables) it can be dropped from the performance index to give

$$\min \; J = p(k)^T \Delta U_F(k) + \frac{1}{2} \Delta U_F(k)^T D(k) \Delta U_F(k) \tag{4-66}$$

If a QP algorithm (routine) that does not require non-negativity constraints is used, then the optimization problem is, minimize equation (4-66) subject to equations (4-29), (4-34), (4-37) and other types of constraints (without the decomposition) as discussed earlier. However the QP algorithm used for this work requires that the non-negativity constraints be satisfied, therefore the decomposition is necessary. After decomposing $\Delta U_F(k)$ into a pair of non-negative variables, equation (4-66) can be rewritten as:

$$\min \; J = \begin{bmatrix} p(k)^T & -p(k)^T \end{bmatrix} \begin{bmatrix} \Delta U_F(k)^+ \\ \Delta U_F(k)^- \end{bmatrix} \tag{4-67}$$
$$+ \frac{1}{2} \begin{bmatrix} \Delta U_F(k)^+ \\ \Delta U_F(k)^- \end{bmatrix}^T \begin{bmatrix} D(k) & 0 \\ 0 & -D(k) \end{bmatrix} \begin{bmatrix} \Delta U_F(k)^+ \\ \Delta U_F(k)^- \end{bmatrix}$$

The decomposed constraints discussed in the LP section can now be used in the QP problem. Transformation into the standard QP problem is completed by

1) multiplying the objective function, J by -1 to convert the problem from minimization to maximization. Therefore the objective function becomes

$$\max \; J = \begin{bmatrix} -p(k)^T & p(k)^T \end{bmatrix} \begin{bmatrix} \Delta U_F(k)^+ \\ \Delta U_F(k)^- \end{bmatrix} \tag{4-68}$$
$$+ \frac{1}{2} \begin{bmatrix} \Delta U_F(k)^+ \\ \Delta U_F(k)^- \end{bmatrix}^T \begin{bmatrix} -D(k) & 0 \\ 0 & +D(k) \end{bmatrix} \begin{bmatrix} \Delta U_F(k)^+ \\ \Delta U_F(k)^- \end{bmatrix}.$$

2) The vector p^T in the standard QP problem can now be written as

$$\begin{bmatrix} -p(k)^T & p(k)^T \end{bmatrix} \tag{4-69}$$

where p(k) in the bracket is given by equation (4-64).

3) let the vector y be defined by

$$y = \begin{bmatrix} \Delta U_F(k)^+ \\ \Delta U_F(k)^- \end{bmatrix} \tag{4-70}$$

4) let the matrix, D in the standard QP problem be

$$\begin{bmatrix} -D(k) & 0 \\ 0 & +D(k) \end{bmatrix} \tag{4-71}$$

where the matrix D inside the brackets is given by equation (4-63).

5) the constraint matrix A in the standard QP problem is given by

$$A = \begin{bmatrix} I & -I \\ I & -I \\ L & -L \\ L & -L \\ A_2 & -A_2 \\ A_2 & -A_2 \\ B_2 & -B_2 \\ B_2 & -B_2 \end{bmatrix} \tag{4-72}$$

$$\Delta U_F(k)^+, \Delta U_F(k)^- \geq 0$$

6) the vector b along with the associated equality and inequality signs is given by

$$b = \begin{bmatrix} \leq \\ \geq \\ \leq \\ \geq \\ \leq \\ \geq \\ \leq \\ \geq \end{bmatrix} \begin{bmatrix} -\Delta U(k)_{max} \\ \Delta U(k)_{min} \\ U(k)_{max} - U(k-1) \\ U(k)_{min} - U(k-1) \\ \hat{Y}(k)_{max} - \hat{Y}(k) \\ \hat{Y}(k)_{min} - \hat{Y}(k) \\ \hat{Z}(k)_{max} - \hat{Z}(k) \\ \hat{Z}(k)_{min} - \hat{Z}(k) \end{bmatrix} \qquad (4\text{-}73)$$

Once again as in the LP problem only a subset of constraints is needed in actual application. The total number of constraints (not including the upper and lower bounds on the QP variables) in this specific case can be calculated as:

$$4*\sum_{i=1}^{NI} M(i) + 2*\sum_{i=1}^{NQ} P(i) + 2*\sum_{i=1}^{NZ} P_z(i) \qquad (4\text{-}74)$$

and the total number of QP variables to be optimized is given by

$$2*\left[\sum_{i=1}^{NI} M(i)\right] \qquad (4\text{-}75)$$

As in the weighted least squares case, proper selection of the weighting matrices yields different objective functions. Table 4.2 is still valid for the QP problem.

Remarks: The MOCCA QP problem described above can be solved using a commercial QP program or a special QP algorithms can be developed to take advantage of the following characteristics of the MOCCA QP problem:

1. QP algorithms without the non-negative variable constraints can be developed so that the dimension of the QP problem is reduced by half
2. the constraint matrix, A is constant if the step response data are not updated on-line
3. the right hand side vector b, changes at every control interval
4. an algorithm should be developed such that the initial solution set can be infeasible so that the previous control solution can be used for the next optimization (e.g. gradient type algorithm)

4.3.2.2 Least Squares With Linear Inequality Constraints

This section on least squares with linear inequality constraints (LSI) was motivated by the fact that the solution, in the form of Fortran subroutines is available in the book on least squares by Lawson and Hanson (1974). Lawson and Hanson (1974) defined the standard LSI problem as

$$\min \; J = [Ex - f]^T [Ex - f] \qquad (4\text{-}76)$$

$$\text{subject to } Gx \geq h$$

First it will be shown that the MOCCA least squares objective function can be easily converted into the standard LSI objective function. The simple least squares objective function is minimize the ISE

$$\min \; J = [Y_d(k) - \hat{Y}_F(K)]^T [Y_d(k) - \hat{Y}_F(K)] \qquad (4\text{-}77)$$

Substituting equation (4-11) into equation (4-77) yields

$$\min \; J = [\hat{E}(k) - A_2 \Delta U_F(k)]^T [\hat{E}(k) - A_2 \Delta U_F(k)] \qquad (4\text{-}78)$$

The objective functions are equivalent if we let $Ex = -A_2 \Delta U_F(k)$ and $f = -\hat{E}(k)$.

The transformation of the constraints was described earlier in the LP section. The problem now is to group together the constraints as inequality constraints of the type greater than or equal to as required by the algorithm. Any less than or equal to constraints can be converted to a greater than or equal to constraint by multiplying by negative one. Therefore the basic constraints, equations (4-29), (4-34) and (4-37) can be rewritten as

$$\begin{aligned}
-\Delta U(k)_F &\geq -\Delta U(k)_{max} \qquad (4\text{-}79)\\
\Delta U(k)_F &\geq \Delta U(k)_{min}\\
-L\Delta U(k)_F &\geq -U(k)_{max} + U(k-1)\\
L\Delta U(k)_F &\geq U(k)_{min} - U(k-1)\\
-A_2 \Delta U(k)_F &\geq -\hat{Y}(k)_{max} + \hat{Y}(k)\\
A_2 \Delta U(k)_F &\geq \hat{Y}(k)_{min} - \hat{Y}(k)
\end{aligned}$$

or in matrix vector notation corresponding to the constraints presented in the standard LSI problem as

$$\begin{bmatrix} -I \\ I \\ -L \\ L \\ -A_2 \\ A_2 \end{bmatrix} \Delta U_F(k) \geq \begin{bmatrix} -\Delta U(k)_{max} \\ \Delta U(k)_{min} \\ -U(k)_{max} + U(k-1) \\ U(k)_{min} - U(k-1) \\ -\hat{Y}(k)_{max} + \hat{Y}(k) \\ \hat{Y}(k)_{min} - \hat{Y}(k) \end{bmatrix} \qquad (4\text{-}80)$$

The solution of the LSI problem is described in detail by Lawson and Hanson. Basically the algorithm used by Lawson and Hanson involves the use of the Kuhn-Tucker theorem. The algorithm is described as a least distance programming (LDP) and it does not require an initial feasible vector (i.e. the initial guess can be infeasible) to start the algorithm. This implies that the solution for one control interval can be used as the initial condition for the next control interval which in effect reduces the computation time if there is no significant change in the original problem.

4.3.3 Non-linear Programming

In general, the objective function and constraints used in MOCCA can be of any kind, linear or non-linear. The solution method depends heavily on the type of objective function and constraints. It can vary from simple off-line calculation of controller gains (weighted least squares without constraints) to general on-line, non-linear optimization. As long as the non-linear objective function and constraints can be transformed into a function of MOCCA

variables (specifically error, change in input variable, past input etc.) then the control problem can be solved using non-linear optimization. A general description of such a problem was given by Morshedi (1986).

5

Literature Review

5.1 Introduction

In this chapter, model predictive control (MPC) algorithms such as Dynamic Matrix Control (DMC), Model Algorithmic Control (MAC), Internal Model Control (IMC), Generalized Predictive Control (GPC) etc. will be examined and compared to MOCCA. MPC has been reviewed by Richalet (1980), Garcia and Prett (1986), Prett and Garcia (1987), Garcia (1987) and Richalet (1987). Prett and Garcia (1987) also pointed out the limitations of MPC in terms of stability, robustness and performance analysis in the presence of constraints and model uncertainties. A brief review and comparison between DMC and MPHC was given by Martin (1981). Proposals for future research in these areas were also given. The following review and comparison of each method will be in the same format as the structure of this thesis: process modeling; control algorithm; and applications.

5.2 Dynamic Matrix Control (DMC)

Dynamic Matrix Control (DMC) was developed and applied successfully in industry by Shell Oil Ltd., Houston in the early nineteen seventies. It was first reported in a paper by Cutler and Ramaker (1979). Since then a number of DMC

papers (Prett and Gillete, 1980; Cutler, 1982; Ogunnaike, 1983; Asbjornsen, 1984; Cutler and Johnston, 1985; Morshedi et al., 1985; Morshedi, 1986; Ogunnaike and Adewale, 1986; Cutler and Hawkins, 1987) have been published in the literature including a doctoral thesis by Cutler (1983). Garcia and Prett (1986) gave an concise introduction to the use of optimization methods in process control in the context of DMC.

5.2.1 DMC Process Model

The DMC process model evolved from a technique of representing the process dynamics by a set of numerical coefficients (step response data) which is described each of the DMC papers. The method used for modeling the process is the same as used in MOCCA. The output prediction vector is obtained from the process model as described below.

1) Initialize all prediction vectors and manipulated input values with the current output and input measurements respectively. The output prediction horizon for DMC is equal to the number of step response coefficients, S plus the control horizon, C so that the steady state effect of the future most change in manipulated input (i.e. the C^{th} change in input value) shows in the prediction vector (Cutler, 1983; Morshedi et al., 1985).

2) Time is advanced by one interval and a cumulative change in output, δO_i, is calculated based on the change in the input variable.

3) The cumulative change in output variable is added to the prediction vector, O_i. In the first interval after initialization, O_i is the measured output. After the first interval, O_i is the prediction vector from the previous interval.

4) The output prediction vector is shifted forward except for the last element, which does not change if the steady state is one in which the output is a constant (i.e. the second element becomes the first element, the third becomes the second and so on). If the steady state is one in which the rate of change of the output is constant then the last element is calculated from preceding values: $O_n = O_{n-1} + (O_{n-1} - O_{n-2})$, i.e. an integrating process (Cutler, 1982).

5) The current output prediction is compared to the observed value and the difference is added to all the elements of the prediction vector. This provides a feedback mechanism for model process mismatch, unmeasured disturbances and slight non-linearities.

6) Repeat 2 to 5 after the new manipulated input has been calculated and implemented.

Using the above method the transients that existed in the system prior to the initialization will die out after a time equal to the prediction horizon. In the original MOCCA the rigorous calculation of the prediction vector is performed at each control interval. This allows the prediction of transients that originated before MOCCA was initialized, provided that the changes in inputs for the past P intervals are stored in the computer memory. In later papers, Morshedi et al. (1985), Garcia and Prett (1986) and Garcia and Morshedi (1986), DMC models similar to that of the original MOCCA were also presented. The full order MOCCA state space formulation is closer to the DMC method. A key difference between DMC and MOCCA is that the reduced order MOCCA formulation makes it possible to have the prediction horizon, P less than or equal to the number of step response coefficients (see step 1 of above).

DMC for SISO process systems with time varying parameters was described by Ogunnaike and Adewale (1986). Their paper described how the time delay and steady state gain of the process model may be made time varying. The varying time delay was handled by extending the error prediction vector to include λ periods. A new error vector is extracted from this extended error vector by ignoring the first λ predictions and the control calculation proceeds as normal. In contrast MOCCA handles the time delay in the

prediction vector calculations without manipulating anything else. The handling of the time varying steady state gain for DMC is the same as MOCCA. The use of a time varying time constant for SISO system, which was included in MOCCA was not considered in their paper. The factoring of the time delays and steady state gains as described by Ogunnaike and Adewele (1986) was also briefly mentioned by Cutler in his thesis (1983).

Cutler (1983) in his thesis also described the possibility of updating of the dynamic matrix on-line by using a least squares algorithm. The proposed parameter estimation method is essentially the opposite of the DMC control calculations, that is the outputs and inputs are known and a set of numerical coefficients describing the step response are calculated using weighted least squares. This is essentially a deconvolution problem as described in chapter three.

Feedforward capabilities within DMC, which provide significantly better control for measured disturbances, were discussed by Cutler and Ramaker (1979), Cutler (1982, 1983) and Asbjornsen (1984). The future output contribution due to measured disturbances is calculated by multiplying the measured disturbance by the disturbance step response data. This contribution vector is then added to the output prediction vector, hence there is no need for a separate

feedforward controller (see MOCCA feedforward section). No details on implementation issues as described in MOCCA feedforward were provided.

5.2.2 DMC Control Calculation

The DMC control calculations are also described as optimization problems. The first DMC algorithm described by Cutler and Ramaker (1979) and Cutler (1982, 1983) used the least squares method to minimize the sum of the squares of the output error over the prediction horizon. Using the least squares method to minimize the objective function, an equation for calculating the change in manipulated inputs was obtained

$$\Delta I = \left(A^T A\right)^{-1} A^T e \qquad (5\text{-}1)$$

where ΔI is the change in input vector, e is the error vector and A is the dynamic matrix of dimension (SxC) by C which is considerably larger than that of MOCCA. The change in input obtained from equation (5-1) may be too large for some applications so a control move suppression factor, k (k > 0) was implemented empirically by augmenting the dynamic matrix with a diagonal matrix with diagonal k elements. This produced a new control law where the magnitude of the calculated manipulated input could be reduced by increasing the suppression parameter,

$$\Delta I = \left(A^T A + k^2 1\right)^{-1} A^T a \qquad (5\text{-}2)$$

where 1 is the identity matrix. The same solution was obtained more formally by modifying the objective function to include weighting on the changes in input variable as shown by Sripada and Fisher (1985). Ogunnaike (1983) also pointed out the statistical parallel between the suppression parameter and weighted least squares. The weighted least squares optimization provides the theoretical explanation for the suppression parameter method.

Constrained control in DMC is handled using an optimization method such as linear programming or quadratic programming. One of the earliest DMC constrained control formulations was that described by Prett and Gillette (1980), where a catalytic cracking unit was brought under constrained control by augmenting DMC with the active constraints. Equality constraints defined as time invariant constraints that must be satisfied at each control interval can be augmented onto the dynamic matrix and solved off-line. However, the time invariant constraint is actually a soft constraint since it allows temporary moves away from the optimization set point (it only requires that the optimization set point be satisfied at the termination of control sequence).

5.2.2.1 DMC Using Mathematical Programming

The solution of the constrained control problem using linear programming in DMC was discussed by Cutler (1983) in his doctoral thesis. Cutler concluded in his thesis that the on-line time requirement for LP solution is excessive and describes an alternative method (called a partitioning algorithm) of handling problem with constraints on the input or manipulated variables (constrained control on manipulated inputs problem) which will be discussed later. The use of linear programming in DMC (LDMC) was further documented by Morshedi et al. (1985). A USA patent application involving the DMC method and the linear/quadratic programming method of optimization has also been submitted by Morshedi, Cutler, Fitzpatrick and Skrovanek. LDMC minimizes the sum of the absolute error as its objective function subject to linear constraints. Linear transformation of the objective function allows the absolute error objective function to be used here (see MOCCA LP formulation). The LDMC problem is essentially the same as that of MOCCA LP problem with the following exceptions: 1) the LDMC dimension is considerably larger that that of MOCCA and 2) the suppression parameter is still implemented empirically although control weighting can be implemented more formally via the objective (see MOCCA LP formulation). The handling of constraints is the same as that of MOCCA with the exception of the treatment of

economic optimum for systems with more inputs than outputs. The economic optimum problem is actually a two stage process (Morshedi et al., 1985):

1) an optimization subproblem (usually steady state) with an economic objective function and constraints is solved so that an optimum final value of the input variables can be determined. The input variables then can be regarded as controlled variables with their calculated optimum values as their steady state setpoints.
2) the solution to the optimization subproblem becomes the steady state setpoint to the DMC problem. This is performed by augmenting the dynamic matrix and error vector. The dynamic matrix is augmented with a block identity matrix for each input variable. The error vector is also augmented correspondingly with the difference between the steady state setpoint (from 1) and the present value of the input. The new dynamic matrix equation is then used in the LDMC problem.

The use of quadratic programming to solve the DMC problem was described by Garcia and Morshedi (1986). The transformation is straight forward and is essentially the same as MOCCA. The only difference is again the dimension of the problem if the original DMC formulation is used. They also recommended the use of a parametric QP algorithm to reduce on-line computation time.

A natural extension of the linear/quadratic programming problem is the use of non-linear programming to solve the control problem. Morshedi (1986) described the universal DMC (UDMC) algorithm for the optimization and control of non-linear processes where the process models are described by non-linear differential equations. The resulting algorithm can be solved by any standard non-linear optimization routine although the use of successive quadratic programming was highly recommended. Other types of optimization methods such as a penalty function algorithm have been used by Little and Edgar (1986). Garcia and Prett (1986) pointed out the use of LP and QP involves the lumping of all the objectives into a single objective function where weightings are used to influence the importance of each objective. Such an approach, although computationally convenient, has some difficulties. Some of the difficulties are 1) selection of the weights is dependent on the scaling of the variables, 2) the relative importance of objectives is difficult to quantify, 3) since all the objectives are lumped together there is no guarantee that increasing the corresponding weight of one objective will not affect the others due to an inherent interaction through lumping and 4) the weights are highly dependent on the operating conditions of the process (i.e. the loop is no longer maintenance

free). Multi-objective optimization was suggested as an alternative so that weighting in the objective function can be eliminated.

5.2.2.2 Partitioning Algorithm

Cutler (1983) described a partitioning algorithm that can handle constraints on manipulated inputs. The algorithm identifies constraints on the manipulated variables serially as they occur in time. When the manipulated variable becomes constrained in time, its allowable changes are imbedded in the prediction vectors for the outputs and the dimension of the problem is reduced. The control calculation still uses the standard least squares solution which is calculated off-line. However, in this case a set of constant matrices results from the permutations that evolve from the reduction in the dimension of the problem. For example, a reduction of a three input by two output control problem to a two input by two output (i.e. one of the inputs is constrained) generates three possible permutations of the input variables: 1 and 2, 1 and 3, 2 and 3. The heuristic argument for the algorithm assumes that all allowable changes in the "first input variable" to become constrained in time are optimal and should be made. The algorithm is therefore problem specific, that is for a given problem all possible outcomes [illegible] form of active

constraints) have to be considered and checked on-line before calculations are performed. Another restriction is that only constraints on the manipulated inputs are considered. Additional constraints on the controlled and uncontrolled output variables result in a more complex algorithm. The procedure may not produce globally optimal solutions. This is the price paid in exchange for faster on-line execution (most of the control calculations are performed off-line). Simulations using the partitioning algorithm were performed on a fractionator model and showed that it was possible to maintain the manipulated input constraints.

5.2.3 DMC Applications

DMC was applied to a preheat furnace as described by Cutler and Ramaker (1979) where the furnace outlet temperature is controlled by manipulating the fuel with feedforward from the inlet temperature. The use of the suppression parameter was demonstrated using an example simulation. Compared to conventional analog and computer PID/feedforward control, DMC performed substantially better. Robustness of the DMC algorithm was also shown for a large temperature disturbance. A comparison using simulations was performed by Cutler and Johnston (1985) and showed the superiority of DMC (or in general model predictive control)

over the conventional PID controller.

Prett and Gillette (1980) described the application of a modified DMC algorithm for non-square multivariable system (more inputs than outputs) constrained control of a catalytic cracking unit where the setpoints of the controlled variables were maintained while holding the manipulated variables as close as possible to the optimal values prescribed by a higher level optimization program (LP). No results were presented but the algorithm is in operation on four Shell Oil catalytic cracking units and projects are underway to install it on two others.

Cutler (1982) described the control of imbalanced systems (integrating processes) using the setpoint trajectory to create a target area for the control of the level in a fractionator bottom. In addition to modifying the process model (see section 5.2.1), the DMC algorithm is modified so that if the error trajectory falls within the target area, no control action is taken by resetting the error vector to zero. Simulation results were presented and compared to a conventional PI controller and to a computer calculated proportional/integral error-squared algorithm. Both feedback controllers yield changes in the flow out that follow the pattern of the disturbance which upsets downstream processes. The DMC with its predictive model is able to minimize the movement of the flow out while letting the

controlled output (level) change within the target area. No industrial applications of the algorithm to averaging level control has been reported. McDonald et al. (1986) presented a survey of the level control problem in the literature as well as presenting a new optimal predictive level controller which was compared to traditional methods of level control and DMC. In general McDonald et al. found that their algorithm performed as well as the DMC controller with the exception that their formulation does not take into account the time delay of the process.

Ogunnaike and Adewale, (1986) applied a modified DMC algorithm to a distillation column and pre-heat furnace models with time-varying gain and time delay. Simulation results were presented and showed significant improvement over the standard DMC for the same process. McDonald and McAvoy (1987) described the use of gain and time constant scheduling in DMC for the control of non-linear binary distillation towers. The gains and time constants were evaluated using simple analytical models to update key process model parameters (process gains and time constants) on-line. Simulation results showed significant improvement in regulatory control performance over the standard DMC.

Morshedi et al. (1985) presented simulation results for LDMC using a three input two output system. A LP subproblem using a steady state model, an economic objective function

and constraints was solved for the optimum steady state values of the manipulated variables. LDMC with a modified dynamic matrix and error vector was then used to solve the control problem subject to constraints on output and input variables such that the optimal steady state values of the manipulated variables were implemented.

Garcia and Morshedi (1986) applied QDMC successfully to a pyrolysis furnace with three manipulated inputs and three controlled output variables. Two measured disturbance variables were also fed forward to the control algorithm. Results showed that the constraints on the manipulated inputs were maintained and the algorithm provided a smooth return to the operating region in the presence of unmeasurable disturbances. Numerous other applications on continuous and batch processes within Shell using QDMC were emphasized by the authors.

Cutler and Hawkins (1987) described the application of a constrained DMC algorithm to the first stage of the Suncor Hydrocracker reactor in Sarnia, Ontario. The algorithm was used to control four output variables using five manipulated variables and two feedforward variables with an on stream factor of greater than 90 percent, resulting in a reduction of 55 percent (approximately $1000 per day) in the energy used by the reactor preheat furnace.

Kiparissides et al. (1987) performed a comparative study of linear quadratic control (LQC), DMC and an extended self-tuning regulator (ESTR). The algorithms were used to control a simulated polymerization batch reactor. The results show that DMC and LQC produced almost the same results. ESTR also produced the same results after identification was completed. The main point in this study was that ESTR could reproduce comparable control as DMC and LQC without a priori knowledge of the system.

5.3 Model Algorithmic Control (MAC)

Model Algorithmic Control (MAC) was developed in France by Richalet et al. (1978) as Model Predictive Heuristic Control (MPHC) whose commercial software is called IDCOM (Identification-Command). MPHC was first conceived by Adersas/Gerbios in the late 60's (Mehra et al., 1981, Richalet, 1987). A general introduction to MPHC is given by Richalet (1980, 1987), where the basic principles of model predictive control are described.

5.3.1 MAC Process Model

As in MOCCA and DMC, MAC relies on an internal model for on-line, long range prediction. The process to be controlled is represented by its impulse response. Mehra et al. (1979) divided MAC long range prediction into open and closed loop MAC. In the first case, the prediction is made

independent of the measured output and in the second case the measured output is used as the starting point (initial condition) for the future prediction. The equations used in open loop and closed loop MAC prediction are theoretically equivalent to MOCCA and DMC models since step response and impulse response are related. In fact, a step response model can be converted to an impulse response model by replacing the step response coefficients by its impulse response coefficients and the change in inputs by input values. In any case both the step response and impulse response models are non-minimal.

Initially model process mismatch and unmeasured disturbances were not handled explicitly as in MOCCA and DMC, however it was included in later papers by Mehra et al. (1979, 1980), Mehra and Rouhani (1980) and Rouhani and Mehra (1982). MAC also included another feedback mechanism in their formulation by feeding back the current measured output to the reference trajectory calculations where it was used as an initial condition for the trajectory. A simple first order filter was given as an example, where the filter constant is the main tuning parameter, Richalet et al. (1978). (cf. section 3.4.4)

One of the main criticisms of non-minimal modeling is the ill-posed nature of the impulse response identification involved. However with proper conditioning (using a

relaxation factor) this can be avoided (Richalet et al., 1978). An identification algorithm for the impulse response coefficients was also given (Richalet et al., 1978) and is described as a dual to the control problem which in effect is the convolution/deconvolution idea discussed in Chapter 3. The identification algorithm is an orthogonal projection algorithm, (Goodwin and Sin, 1984) modified to include a relaxation factor.

5.3.2 MAC Control Calculation

The control calculation was initially described as a dual to the identification problem and constraints were introduced by limiting the calculated values (Richalet et al., 1978). A projection type algorithm (an iterative optimization procedure) was used to minimize the weighted least squares objective function. Mehra et al. (1980) and Rouhani and Mehra (1982) suggested the use of various optimization algorithms to solve the weighted least squares problem.

Mehra et al. (1980) and Rouhani and Mehra (1982) described a linear quadratic (LQC) control type problem applied to non-minimum phase systems. The MAC problem is first transformed into its state space equivalent where the state vector is made up of present (unknown) and past inputs before the LQC control design is applied. The resulting

solution (optimal input sequence) is linearly dependent on the state vector. The gain for the LQC problem is calculated off-line from the matrix Ricatti equation and is relatively simple compared the the standard LQC design. Once the gain is calculated, the optimal control law can be calculated and depends uniquely on the impulse response and reference trajectory filter constants. Mehra et al. (1980) also describe a MAC controller using direct pole placement for the control of non-minimum phase systems. The basic idea is to choose the controller such that it retains as much as possible of the dynamic properties of the process while stabilizing the characteristic polynomial. This is done by maintaining the stable roots of the process and altering the unstable ones. The only difficulty pointed out was to decide where the optimal location of the poles should be.

Mehra et al. (1981) describe an inversion type optimization routine (in the absence of constraints) which finds inputs to zero the error between the reference and zero-input trajectories. To make the on-line problem tractable, the error is only considered at a few points in the future and the future controls are required to be constant over intervals ("blocks") between the output matching points, that is an input may be held constant for more than one control interval. For example, three

("block") input values may be computed from three ("block") future errors for the next five steps into the future. The advantage of this blocking method is that it permits reasonably long optimization horizons with low dimensional calculations and is justified by the fact that the controls will generally be recalculated at every control interval. Mehra et al. (1981) also described the development of a gradient-projection algorithm to improve the control calculation for general systems. The algorithm retains the input blocking features but does not block the outputs. It minimizes the sum of the squares of the output errors at each step in the future. For example, the gradient projection algorithm would compute three control inputs to minimize five future errors. The algorithm is capable of weighting both input and output variables in its cost function.

Mehra et al. (1979, 1980), Mehra and Rouhani (1980) and Rouhani and Mehra (1982) presented some theoretical results regarding MAC's robustness, stability, application in stochastic environment and non-minimum phase systems. An excellent summary of the results was given by Mehra et al. (1981) and is reproduced here for convenience:

STABILITY AND ROBUSTNESS PROPERTIES: The following notation will be used: 1) an actual plant, h_o; 2) an internal (plant) model used for prediction, h and 3) a reference model used

to calculate the reference trajectory. Due to identification and modeling errors, the internal and actual model will differ. The following three equations represent the plant (actual model), closed loop prediction model and reference model respectively.

$$y_o(t+1) = h_o u(t) \tag{5-3}$$

$$y(t+1) = hu(t) + [y_o(t) - y(t)] \tag{5-4}$$

$$y_r(t+1) = \alpha y_o(t) + (1-\alpha)C \tag{5-5}$$

where C is the desired setpoint and α is a parameter related inversely to the trajectory time constant (small α implies fast approach to the set point and $0 < \alpha < 1$). For the closed loop system to be stable, it is necessary and sufficient that the polynomial

$$z^{-1}[H_o(z) - H(z)] + H(z) - \alpha z^{-1} H_o(z) \tag{5-6}$$

has all its roots within the unit circle, where $H_o(z)$ and $H(z)$ are the z transforms of $h_o(z)$ and $h(z)$ respectively. It has been shown that if the identification is: 1) nearly perfect, then the stability of the plant implies the stability of the closed loop controller (besides the output of the plant is of first order with a pole at α) 2) not perfect, then the designer has to choose both α and $H(z)$ in a way that equation (5-6) has all its roots within the unit circle. In most cases, if the plant is stable and minimum

phase then relatively simple choices of h stabilize the closed loop system. The closer h is to h_o, the shorter the convergence time of the output to its desired setpoint.

One of the most useful properties of MAC is its robustness, which appears to result at least partially from the use of an impulse response model as opposed to a parametric model for plant representation. It can be shown that when the mismatch between the plant and the internal model is a pure gain, as long as the gain factor, q is less than the inverse of $1-\alpha$ the system is stable and the output converges to its desired setpoint. In particular, a slow reference model (α close to 1) enhances the robustness of the systems.

STOCHASTIC ENVIRONMENT: The properties of MAC have been studied in the presence of both white and colored additive output noise. In particular it is shown that the variance on the output of the controlled system decreases as the reference model becomes slower. For colored noise the results are more complex depending on the correlation properties of the noise. For instance with a highly positively correlated noise, it is necessary to speed up the reference model to decrease the variance on the output. In general it is often advantageous to perform output filtering particularly when the output noise level is large. The

proper choice of an output filter decreases the output variance while allowing the selection of a faster reference trajectory.

NON-MINIMUM PHASE PLANT: When the plant is non-minimum phase, difficulties arise in the sense that the convergence of the output to its desired value may require inputs of infinite magnitude. Within the framework of MAC, it is relatively simple to come up with a solution which consists of selecting an optimum internal model such that the Euclidean distance of the output to the reference trajectory is minimized while the input energy remains finite. This requires the off-line solution of a Ricatti equation which determines the internal model for control calculations. The important concept here is that the internal model used for control calculations is different from the internal model used for prediction calculations.

The following summaries describe published but unattainable (i.e. contract) reports that were briefly described by Mehra et al. (1981).

HIERARCHICAL MAC AND SINGULAR PERTURBATION THEORY: It was shown that the optimization of systems with multiple time scales leads naturally to a hierarchical control structure, in which the reference trajectories for the lower (faster) levels are specified by higher (slower) levels. This eliminates the arbitrariness regarding the specification of

trajectories and constraints. A combination of MAC and singular perturbation theory appears to provide a powerful tool for overall optimization of plant performance through proper coordination and control of different levels.

ALTERNATIVE PREDICTION SCHEMES: For linear systems, the predicted output response can be written as the sum of predictions based on past inputs and predictions based on future inputs. Current implementations of MAC use the impulse response model for computations of both the output predictions and control calculations. However if a state model of the system is available the predictions based on past inputs can be obtained from a state estimator (Kalman type) or a Luenberger observer (see also Li et al., 1986; Navratil, 1988). This is particularly attractive for continuous time and/or non-linear systems. If a state representation is not available, then an ARMA model may be used for prediction. It may even be more suitable due to its lower order. The main advantage of the impulse response representation is in the control computation. Thus it is possible to use two different models, one for prediction (any type of model) and an impulse model for control calculations. The two models need not necessarily have the same input output properties since the objective of the prediction model is to produce accurate predictions whereas the objective of the control model is to produce

satisfactory, stabilizing and robust control values. In particular, for non-minimum phase systems, the two models are different since the true plant model does not have a stable inverse.

SAMPLE RATE SELECTION: For continuous time systems with discrete observations, the selection of sampling rate is important. A very high sampling rate is expensive and wasteful of system resources whereas a very slow sampling rate would lead to large excursions of the controlled variable over the sampling interval. A procedure for selecting the sample rate in MAC was described. The procedure involves comparing the reference trajectories and the predicted plant outputs on a continuous time basis even though the observations and control actions are discrete. The optimal sampling rate is determined by solving two Ricatti equations, one representing estimation cost and the other representing control cost for a given sampling scheme.

Foigel and Richalet (1979) proposed a self-adapting procedure for MPHC. The procedure uses indirect adaptive control to perform on-line identification of the closed loop process. The identification routine used was the modified orthogonal projection algorithm. Additive extra signals are used to improve the identification and stabilize the self-adaptation. The bias induced by feedback is analyzed and removed through a restoration procedure which is performed

by a simple filter in the particular case of gain disadaptation. Stability analysis and robustness are performed and an example is described. Although the self-adapting procedure is shown to be feasible, there are many things that still need to be considered. Mahmood and Larimore (1986) describe the integration of MAC with an identification routine, Canonical Variate Analysis (CVA). The new result here is the identification routine which can be used to identify the plant on-line. CVA can determine the state order and identify a plant or part of it in closed loop configuration. No initialization is needed and it perform near maximum likelihood.

Bruijn et al. (1979) extended the cost function used by MAC (ISE) to include weighting on the inputs. This represents a weighted least squares solution to the control problem. Simulation results were used to show the effect of the weighting.

5.3.3 MAC Applications

A number of industrial applications using MAC*(IDCOM) were summarized by Richalet et al. (1978), Froisy and Richalet (1986) and Mehra et al. (1981). The first application involved control of an entire PVC plant in France, where four distillation columns and three cracking furnaces were controlled. The units were controlled using a

master and slave procedure (defined as transparent control by MAC) where IDCOM is the master, calculating optimal setpoints to the slave (PID) loops. No results were presented for the PVC plant except economic savings. The total economic saving for the PVC plant was estimated at $370 000 per year. Lebourgeois (1980) provides more details on the process and the control schemes.

IDCOM was also applied to a fluid catalytic cracking fractionator column, where two key column temperatures related to product quality were placed under multivariable control. The internal pan level in the column is affected by various disturbances and also by the manipulated variables (side draw flow rates). It is constrained so that the level does not fall below a safe minimum. Here the predictive nature of IDCOM was used to allow the level to vary within bounds but also to override the product quality control objective if predictions indicated that the level limit would be violated. Results relative to analog control indicated tighter control and were verified by histograms and variance calculations. Engrand (1980) described the application of multivariable control (IDCOM) in a refinery. The control of a distillation unit (2 input 1 output) was described and the results (graphs and histograms) indicate tighter control compared to conventional control. Economic

evaluations estimated a $2 million annual savings compared to an investment of $40 000 for equipment (computer, instruments etc.) for the unit.

A control system for a power plant steam generator was also built using multivariable IDCOM where three output variables are controlled using three manipulated variables under conditions of measurable load changes. This was the first application described that involved non-minimum phase responses in the process. Here, the process non-linearity was taken into account using gain scheduling and changing the sampling time to reflect changing time constants. IDCOM was compared to an adaptive analog controller and the results indicate a reduction of the controlled variable deviations (Mehra et al., 1981). Mehra et al. also reported a case where IDCOM was applied to the control of an utility boiler attached to a large crude distillation column. The objective was to maintain steam pressure as close as possible to 4 bars while maintaining the water level. Application of IDCOM to the unit resulted in a reduction of the pressure from 7 bars (conventional control) to 5 bars.

Mereau et al. (1978) applied MAC using IDCOM to the identification and control of the fast mode of an aircraft controlled by its autopilot to keep zero pitch angle (i.e. the system is closed loop where the process input is the autopilot output and the process output is the pitch angle).

The aircraft was simulated using an analog computer and the result showed better control of the pitch angle and smoother aircraft response compared to classical methods.

Daclin (1980), described the application of IDCOM to a simulated mine hunter boat. No results were given. The main conclusions was that multivariable control using a process model was necessary and that computers will have to play an important role in control systems.

Martin et al. (1984) described the application of IDCOM to a first stage hydrotreater reactor and fractionation unit at Gulf Canada's lube oil plant at Clarkson, Ontario. The main objective in the control of the reactor was its weight average bed temperature (WABT). WABT is controlled by a three level hierarchy: the first level (microprocessor control) sets each of the reactor bed inlet temperatures; level two includes several IDCOM controllers that maintain bed outlet temperature; and at level three IDCOM computer control adjusts bed one inlet temperature and the outlet temperature targets. In addition to controlling the WABT, IDCOM was used to enforce the reactor bed outlet temperature profile which is important in maintaining catalyst life. Feedforward from the feedrate and hydrogen partial pressure was also included. The fractionation control includes the computer control of two vacuum columns and an atmospheric column, where the main objectives are to stabilize and

regulate the distillation processes and control the product quality parameters (e.g. viscosity). The results shows substantial reduction in variation of product quality, fuel (25%) and steam consumption (50%) and also a reduction in catalyst deactivation.

A few other applications were also mentioned by Mehra et al. (1981) and Froisy and Richalet (1986) but no details were given or available (i.e. no referenced papers). The applications includes: Mirage (aircraft) control, F-100 jet engine control, adaptive missile control, glass furnace control, cryogenic wind tunnel control, green temperature/humidity control, fluid catalytic cracker regenerator control and catalytic reformer control.

Usoro and Mehra (1984) presented simulation results for the control of a non-linear three phase electric arc furnace. Simulation results for MAC compared favorably to the simulated conventional control (proportional and logic feedback control) results. A partially adaptive MAC, where the gain of the internal model was changed on-line was also implemented and compared to MAC. Due to the non-linearity of the process, the gain scheduled MAC gave better results. Mahmood and Larimore (1986) presented simulation results for the adaptive control of a missile. It was demonstrated that

the robustness of MAC and the accuracy of the CVA identification even in closed loop has resulted in a very reliable and satisfactory adaptive controller.

5.4 Internal Model Control (IMC)

Internal Model Control (IMC), proposed by Garcia and Morari (1982, 1985a, 1985b) is an attempt to unify several different control schemes (optimal control, Smith predictor, inferential control, MAC, DMC). Since then a number of IMC papers have been published, Morari et al. (1984), Rivera et al. (1986), Economou et al. (1986), Economou and Morari (1986), Zaffirou and Morari (1987a, 1987b), Kozub and MacGregor (1987), Bergh and MacGregor (1987), Levien and Morari (1987), Wassick and Tummala (1987), Laughlin and Morari (1987) and Palazoglu (1987). The IMC results are not used directly in the MPC work in this theses and are referred to here only for completeness and future work. IMC was partly stimulated by the success of MPC algorithms such as MAC and DMC which were developed heuristically in the industry. The development of IMC was also motivated by Brosilow's work (1979, note that no details were available for this work) on the inferential Smith predictor. Man (1984) provided a brief review of IMC in his thesis and incorporated the use of single series forecasting into IMC.

A detailed evaluation of IMC involving single and multivariable applications was performed by Smillie (1984) for his thesis work.

Garcia and Morari (1982, 1985a, 1985b) postulated four criteria used to evaluate the quality of a controller:

1) Regulatory behavior: the output variable is to be kept at its setpoint despite unmeasured disturbances affecting the process.
2) Servo behavior: changes in the setpoint should be tracked fast and smoothly.
3) Robustness: stability and acceptable control performance should be maintained in the face of structural and parametric changes in the underlying process model, that is a controller should be designed with a minimum of information about the process.
4) Ability to deal with constraints: major economic returns from process control arise from the optimization of the operating conditions (determination of optimal set-points). A well functioning regulatory control layer is needed to implement the optimal operating conditions which often occur at the intersection of constraints. Therefore the ability of the regulatory control layer to deal with constraints on both the inputs and outputs is very important.

Note that an error in Garcia and Morari (1985b) was pointed out by Mijares and Holland (1987) and was acknowledged and corrected by Morari (1987). The following sections provide a brief summary of IMC.

5.4.1 IMC Process Model

Garcia and Morari (1982, 1985a) show how both single and multivariable process models for IMC can be derived from a discrete state space model arriving at an impulse (step) response model (see Chapter 3). The impulse response model is exactly the same as that of MAC with the exception of a disturbance term. The advantages of the impulse response model were described as 1) the model is structure free, that is no assumptions have to be made about the number of poles and zeros for the system; 2) robustness is partly attributed to the non-minimal form of the model and 3) the impulse response model is very convenient for a theoretical investigation of robustness. The main disadvantage discussed was the large number of parameters that have to be determined because of the non-minimal form of the model.

Garcia and Morari (1982, 1985a, 1985b) also proved some theoretical properties regarding stability, perfect control and zero offset. The use of a feedback filter (exponential) to improve stability and robustness was also discussed. Walgama (1986) pointed out the ad hoc nature of the IMC

filter and discussed the use of Kalman filter to provide minimum variance estimates by taking into account the statistical properties of the noise and the model of the process.

5.4.2 IMC Control Calculations

Garcia and Morari (1980, 1985a, 1985b) showed the similarity between the linear quadratic optimal control and IMC. They also formulated a predictive control problem using IMC. Essentially the problem formulation is the same as that of MOCCA's weighted least squares case with the following exceptions:

1) Both the error and input trajectory are optimized over a prediction horizon, P (as opposed to P and M in MOCCA). Note that the input trajectory is optimized as oppose to MOCCA where the input changes trajectory is optimized.
2) M, optimal input values were calculated, even though P values were specified in the objective function. This was done by 1) setting the input weighting for all future input variables greater than M equal to zero and 2) setting future input variables equal to the input variables at the M^{th} value as constraints in the optimization.

3) The control law, although is in the same form as the MOCCA off-line weighted least squares solution was unnecessarily complicated due to use of partitioned matrices.

4) An offset compensator is needed in IMC for the case where the input penalty (input weighting) parameter is non-zero. It was shown that if the change in input variable was optimized, then there is no need for the offset compensator (cf. MOCCA).

5) Constraints were not taken into account explicitly. Only soft constraints, i.e. via weighting, were considered although they suggest the use of linear programming and gradient type algorithm (cf. Mehra et al., 1981).

Theoretical proofs of stability were obtained for the predictive control problem, resulting in tuning procedures for non-minimum phase and minimum phase systems (SISO, Garcia and Morari, 1982; MIMO, Garcia and Morari, 1985b). The guidelines for both systems are summarized here. For the first case, the non-minimum phase system, the first goal is stability and then performance and in the second case stability is not an issue and hence only the quality of performance is considered.

IMC Tuning Guidelines for Non-minimum Phase (NMP) Systems

1) Sampling Time. Frequent sampling of a continuous process with right half plane (RHP) zeroes produces an H(z) with roots outside the unit circle. If the sampling time, T is increased these roots are made stable. An alternative is to increase T just enough to ensure $h_i>0$ (i=1,...,N) and to obtain a stable approximation by reducing the control horizon, M.
2) Control Horizon, M (IMC defines this as input suppression parameter). Any system with $h_i>0$ (i=1,...,N) can be stabilized by choosing M sufficiently small.
3) Input Penalty Parameter (β_i). This is the input weighting. A stable approximation of $H(z)^{-1}$ can always be obtained by making β_i sufficiently large.
4) Output Penalty Parameter (γ_i). In the SISO case the use of γ_i is only meaningful when the parameters are time varying. Otherwise one should choose $\gamma_i=1$. It has been shown that the controller $G_C(z) = H(z)^{-1}$ leads to large excursions of the input and strong oscillations of the output between sampling intervals. The reason for this undesirable behavior is that the controller drives only the output to zero without taking care of the other system states. It is well known that a general n^{th} order system can be brought precisely to rest in no fewer than n discrete steps. IMC suggest the following choice of tuning parameters for a system of known order, n: P=2n-1;

M=n+1; $\beta_i = 0, i = 1,..n; \beta_{n+1} = large$ $y_i = 0, i = 1,..n-1; y_i, i = n,...,2n-1 = 1$.
These values were tested in simulations and showed superb performance (Garcia and Morari, 1982).

5) Optimization Horizon, P. Any NMP system can be stabilized by selecting M sufficiently small and P sufficiently large.

IMC Tuning Guidelines for Minimum Phase Systems

1) Sampling Time: Contrary to conventional feedback control the stability of IMC is not affected by T (provided it is not unreasonably large). Larger T's generally lead to less extreme excursions of the manipulated variable, but they have a detrimental effect on the ability of the system to handle frequent disturbance changes.

2) Control Horizon, M. Perfect control (P=M=N) requires severe variations in the input variable and usually leads to strong oscillations of the output between the sampling times. Reducing M reduces the extreme excursion of the manipulated variable and thus leads to a more desirable response of the plant.

3) Input Penalty Parameter (β_i). Increasing β_i will decrease the actions taken by the manipulated variables and make the system more sluggish. Further more, $\beta_i \neq 0$ will generally lead to offset which can be corrected using an offset compensator. An alternative procedure is to reformulate the problem by penalizing the change in

manipulated variable (cf. MOCCA). However, it was shown by example that by penalizing the change in manipulated variable only it is generally not possible to obtain a stable approximation of $H(z)^{-1}$ for NMP systems. This statement is misleading in the sense that the NMP example chosen was unstable in the first place. The system would have to be stabilize before MOCCA could be applied.

4) Optimization Horizon, P. The length of the optimization horizon has virtually no effect on the performance as soon as P exceeds about twice the system order. It was proven that if P is chosen such that it is greater than N, the stability of the controller is ensured (cf. DMC).

Garcia and Morari (1982) also showed the equivalence of IMC to MAC and DMC.

Ricker (1985) formulated a constrained IMC which was solved using quadratic programming. The objective function and model prediction used was very similar to IMC. Only constraints on the manipulated inputs and process outputs were considered. The initial transformation is similar to that of DMC and MOCCA although no details were given. The transformed problem (QP with inequality constraints) is then converted to equality constraints via slack variables. The objective function was further converted into Lagrangian form and the Kuhn-Tucker conditions were applied. The resulting equation (Ax=b type) can be solved for the optimal

solution using iterative methods. A comparison of three types of QP algorithms was presented, although only two were implemented. The major difference in the QP algorithms is the starting (initial) point for the QP iterations. Details of the algorithms were provided by Ricker (1985). Simulations were performed to determine the "best" QP algorithm to use.

Campo and Morari (1986) presented a general mathematical programming framework for multivariable model predictive control. They discussed the translation of control objectives, which result from the definition of norms (which define the size of a vector valued error) into an appropriate objective function and constraints. By choosing an infinity norm, a new formulation was derived and cast as a linear programming problem. Additional details were given in Campo and Morari (1987). The formulation does not attempt to minimize errors at all future times, but only at the future sample time where the error is maximum. It generally does not require extreme control action and was shown to be have a computational advantage over previous algorithms (e.g. LDMC).

5.4.3 IMC Applications

Simulation results for three SISO systems; minimum phase, monotonic non-minimum phase; and nonmonotonic

non-minimum phase, were presented by Garcia and Morari (1982) to illustrate the effect of tuning parameters in model predictive IMC and the use of the tuning guidelines. Simulation results were presented for the multivariable case by Garcia and Morari (1985a, 1985b). The control of a simulated methanol/water distillation column using IMC with diagonal time delay factorization and feedback filter was compared to SISO PI loops and Ogunnaike and Ray's multivariable dead time compensator plus PI. The results show that IMC performed considerably better with regard to interactions and disturbance rejections. A simulated fixed bed reactor was controlled using IMC and PI plus steady state compensator. The results show better control by IMC with less severe output oscillations. When input saturation constraints were applied to the PI controller, large oscillations were observed when one of the inputs was constrained. In contrast, IMC remained stable but offset was observed in one of the controlled outputs.

Arkun et al. (1986) tested IMC experimentally for the temperature control of a heat exchanger (SISO) and the level/temperature control of a stirred tank (MIMO). In the laboratory experiment, IMC showed superior performance, was easier to tune and was sufficiently robust to deal with significant gain and time delay errors, process non-linearities and valve hysteresis. In the SISO experiment the main

tuning parameter used was the feedback filter time constant which proved to be the main advantage over PI control. No other controllers were tested for this case. In the multivariable case, IMC was compared to decoupled PI. Results were generally in favor of IMC although no tuning was performed for this case.

Matsko (1985) described the industrial application of IMC to a pulp and paper mill. Three chemical recovery processes, a slaker reactor where reclaimed sodium carbonate is converted to sodium hydroxide; evaporators where spent liquor from pulp digesting is evaporated to a required concentration before being sent to a chemical recovery furnace and a lime kiln where lime and calcium dioxide is reclaimed from calcium carbonate were all placed under various IMC controllers. No results (data) other than satisfactory performance were reported.

The use of quadratic programming for constrained IMC was reported by Ricker (1985). Simulation results on a methanol/water distillation column were used to illustrate the use of input parameter blocking as discussed in IMC. It was shown blocking could be use to provide smoother control action and better performance. The only disadvantage of input blocking was that it can result in increased interaction in multivariable systems. Two methods of QP were used to solved the constrained IMC problem on a

simulated multieffect evaporator (for black liquor recovery in kraft pulping). The QP/IMC controller was then compared to optimal PI control (tuned using a parameter optimization program that minimized the ISE for the servo response to a step change in the setpoint) in an unconstrained control problem (QP/IMC was made unconstrained by artificially inflating the constraints). Results show almost perfect setpoint tracking for QP/IMC whereas for the PI control, the response was extremely sluggish. Simulation results also showed that QP/IMC was able to handle hard constraints. No comparison to any other controller was given.

Ricker et al. (1986) applied IMC to a six effect evaporator process in a kraft pulp mill. The control objective was to maintain the final product concentration using steam and the feed throughput as the manipulated variables, subject to constraints on both manipulated variables. A secondary control objective was to hold the average feed flowrate at a target value. There are surge tanks preceding and following the evaporator that allow short term deviations from the target. The maximum steam flowrate varies with time and since no measurement of the steam flowrate was available, approximate maximum steam constraints were used. The feed throughput was subjected to a minimum value to prevent rapid fouling of heat transfer surfaces and other problems. Weightings and input blocking

were used in this application. When the controller was put into closed loop operation, QP/IMC held the steam flowrate at its specified maximum and manipulated the feed flow rate since the throughput was always below the target value. The concentration was brought to the setpoint with a deviation of ±0.2%. It was noted that given the constraint on the steam flowrate, it was impossible to achieve the targets on both the product concentration and throughput. However since the concentration was given a higher output weighting, the algorithm allows deviation of the throughput so that the product concentration target can be achieved. The algorithm was also tested for setpoint changes in the concentration with the results showing a deviation of ±0.5% which was due to an unexpected disturbance (not planned). Pre-programmed throughput-target changes were also tested and the results showed a deviation of less than ±0.4% in the concentration. Compared to the performance of a PI controller which is capable of maintaining the concentration with a deviation of ±1.0%, QP/IMC performed considerably better.

Campo and Morari (1986) applied an infinity norm formulation of MPC to a simulated methanol/water distillation column, using linear programming. Results showed the ability to handle constraints and provide smooth setpoint tracking with non-excessive control action (without weighting). No comparison to other methods was given.

5.5 Other Methods and Applications

A detailed analysis of predictive control techniques which are based on discrete convolution models (step or impulse) was performed by Marchetti et al. (1983a). The advantages of MPC were outlined and MPC was shown to reduce to deadbeat control and Dahlin's algorithm for special cases. The effect of various tuning parameters on the performance of a predictive control scheme was examined via simulation. A major conclusion is that a significant reduction in the dimension of the dynamic matrix (cf. MOCCA) does not significantly degrade the control system performance thus allowing reduction in computation time. Marchetti et al. (1983b) also presented a new design technique for predictive controllers based on closed loop pole assignment rather than minimization of a quadratic performance index. The controller gain can be calculated analytically from the system model parameters and the desired pole locations. The analytical solution does not require a matrix inversion. It was demonstrated that it may not be possible to arbitrarily assign all the closed loop poles using discrete convolution and a predictive control law (change in input equals gain vector times error vector). However, satisfactory control system performance was obtained by attempting to place the dominant poles in the

desired locations with the remainder at the origin. Stability is ensured as long as the dominant poles are located well within the unit circle.

Maurath et al. (1985a) reviewed the development of the predictive controller for SISO systems. A stability analysis for this controller was developed which allows the computation of closed loop system stability in the presence of model process mismatch. The effect of controller design parameters on the closed loop response and robustness were examined by simulation of several representative systems leading to recommendations for controller design parameters based on process dynamics. Another method of predictive controller design was proposed by Maurath et al. (1985b). The method employs singular value analysis of the process dynamics. The calculation of the gain matrix for model predictive control (weighted least squares with no constraints) requires the use of the pseudo inverse which can be ill-conditioned. Singular value decomposition allows the calculation of a pseudo inverse with good numerical properties. It was shown that the model predictive control problem can be transformed into orthogonal variables and solved using singular value analysis. The design method involves finding the pseudo inverse and then, for a unit step change in error, constructing a table of principal components displaying the associated singular value; the

contribution of that component towards reducing the residual; the remaining residual assuming that current and previous components are retained and the first change in input value. Using this table, the designer is able to trade off performance and robustness in designing the controller. Examples for the design of two distillation columns were presented and simulated. Experimental evaluations on a multivariable heat exchange network of this method were performed by Callaghan and Lee (1986b) and Wolfgang et al. (1986). No comparisons to other methods were performed. The results mainly served to confirm the theoretical predictions. Maurath et al. (1986) also introduced a method for achieving decoupling with predictive controllers. The technique uses a particular time varying setpoint trajectory to enforce decoupled setpoint changes in the closed loop system, while retaining the original load disturbances rejection properties of the controller. The time varying trajectories are calculated off-line by simulating setpoint changes with a weighted predictive controller such that heavier weighting is given to variables not having the setpoint change. The trajectory obtained from this simulation can then be scaled (since a linear model is assumed) according to the required setpoint change. Simulation results show significant improvement in the interaction at the expense of slower response. Yuan and

Seborg (1986) described the derivation of an observer for load estimation and a predictive control law with feedforward action based on the estimated load disturbance (cf. MOCCA disturbance predictors). Simulation was used to compare the modified predictive controller with PID and the basic model predictive controller. For setpoint changes with no modeling error, both predictive controllers performed considerably better than PID. However for load disturbances the modified controller performed much better than the basic predictive controller which performed marginally better than the PID. For the case with model process mismatch, the modified predictive controller gave better performance for both setpoint changes and load disturbances.

Chang and Seborg (1983) describe the development of a multivariable control scheme which includes explicit constraints on the state, control and output variables using linear programming (LP). A state space model was used here to design the controller. The design also uses the predictive control concepts of output and control horizons. Some theoretical results on the LP problem were also presented. Simulation studies on a continuous stirred tank reactor with two state variables and two control variables were performed for the LP controller and an optimal feedback controller based on a quadratic performance index. Needless

to say the LP controller was able to handle constraints where as the optimal controller could not. Simulation was also performed on a fifth order state space model of a pilot plant evaporator (Fisher and Seborg, 1976). The LP controller was compared to an optimal PI controller based on a quadratic performance index. The LP performed considerably better since it was able to handle constraints. However, the LP controller also exhibited undesirable oscillations in the control variable, which were eliminated by increasing the prediction horizon or by passing the control signal through a scalar gain.

Rotea and Marchetti (1987) transformed the impulse response model (SISO) in IMC into a state space model and applied linear quadratic regulator theory to obtained a controller whose gain is calculated from the matrix Ricatti equation. Constraints on control variables were implemented by calculating the control weighting factor and optimal gain vector such that the constraints were not violated. The solution requires automatic tuning using the control weighting factor and on-line computation of the matrix Ricatti equation. Simulation results confirm the ability to handle control variable constraints optimally. Wong et al. (1986) implemented a number of heuristic saturation algorithms to constrain the control variable and compared them to quadratic IMC. Experimental evaluations of the

algorithms were performed. Results indicate that a suboptimal performance is obtained using one of the saturation algorithms. QIMC was described as the most flexible algorithm for implementation of constraints.

Graham et al. (1985), compared DMC and a model based controller (which is basically a decision making mechanism with predefined control actions) using a simulated multivariable cement kiln. The results using these two methods were comparable although the model based technique does required longer execution time. No tuning was performed. Lee and Newell (1985) and Callaghan and Lee (1986a) compared multivariable control of a grinding circuit using DMC and multiloop PI controllers (with decoupler) and found that DMC performs considerably better and is easier to design.

Svoronos (1986) introduced model dependent control (MDC) which is a combination of IMC and pole placement. The algorithm gives improved disturbance rejection and allows application to open loop unstable processes. Simulated examples were given to demonstrate the disturbance rejection.

Åstrom (1980b) described a simplified form of MAC for linear SISO systems with positive impulse responses, where the model is a gain and derived a controller based on the model. The primary tuning parameters are the gain and the

sampling time. It was shown that the regulator works for a certain class of problem as long as a stability condition is satisfied. The main disadvantage is that the settling time of the resulting closed loop system is larger than the open loop system. A simple simulation was presented. A simplified model predictive control algorithm was presented by Arulalan and Deshpande (1987) where the one step control law was derived as a function of a tuning parameter, error and past predictions in the form of convolution. The tuning parameters were determined by off-line optimization which satisfies a user specified performance index. Simulation results for a distillation column were presented. No comparisons were made. Tung (1983) described how a state space model can be transformed into an impulse response model and derived a control law based on a quadratic performance index. It was shown for a special case (output weighting equal to identity and input weighting equal to a scalar times identity, prediction horizon equal to control horizon), the controller gain matrix i.e. $(A^TA+\Gamma_1)^{-1}A^T$ can be calculated on-line. A recursive formula was used to calculate the matrix inverse before the gain matrix is calculated. On-line adaptation of the model was also described. Simulation examples were presented with no comparison to other methods.

Adaptive model predictive control schemes have also been introduced lately (Clarke et al., 1984a, 1984b; Clarke and Zhang, 1985, 1987; Clarke, 1987; Clarke and Mohtadi, 1987; Favier, 1987; Freedman and Bhatia, 1985; Huang et al., 1985; Foigel and Richalet, 1979; Mahmood and Larimore, 1986; Hmood and Prasad, 1987; Morningred et al., 1987; Papadoulis and Svoronos, 1987). The algorithms in general use the same concepts as the basic model predictive control. Model adaptation is performed using a parametric model (e.g. CARIMA model) or a non-parametric model (impulse or step) and the control calculations are performed on-line. A review of adaptive model predictive control is beyond the scope of this thesis.

More recently, improvements in load estimation using predictors and observers (Wellons and Edgar, 1987; Huang et al. 1985; Stephanopoulos and Huang, 1987; Zafirou and Morari, 1987a) were emphasized as one method of improving predictive control and interpreted as an additional degree of freedom for the controller design. A review of this work is also beyond the scope of this thesis.

6

Software Design

6.1 Introduction

One of the objectives of this thesis was to design and implement a software package to translate the standard model predictive control problem (i.e. the engineer specified objective function, design parameters, step response data etc.) into a standard optimization problem that can be readily solved off-line and/or on-line control algorithms. Figure 6.1 represents a general idea of the off-line software package.

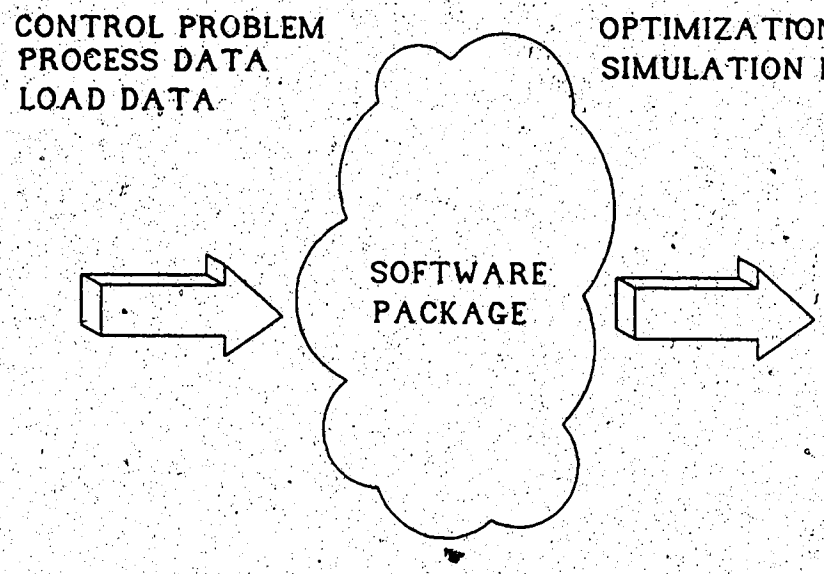

FIGURE 6.1: Schematic Representation of Software Package

The design and implementation of a "translator and simulator" software package will be discussed in this section using high level functional specifications and program structure diagrams. A first level program structure diagram is shown in Figure 6.2. The resulting software package can be used by control or application engineers to design, simulate and implement MOCCA in industry.

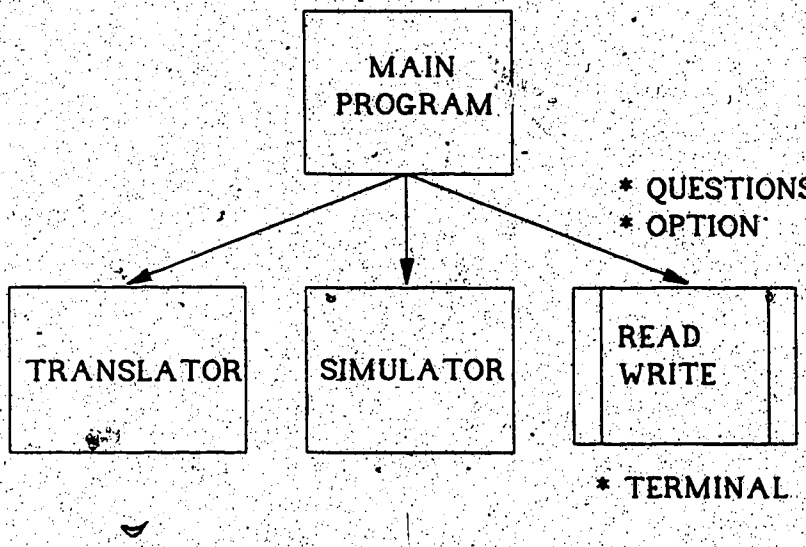

Figure 6.2: Program Structure Diagram of Software Package

6.2 Translator

The objective of the translator program is to translate a standard model predictive control problem into a standard optimization problem. Figure 6.3 showed a schematic representation of the translator program. The translator consists of 46 modules (subroutines) with a total 4070 lines

of Fortran code. Extensive documentation for each module is included in the routines as comment statements (2430 comment statements).

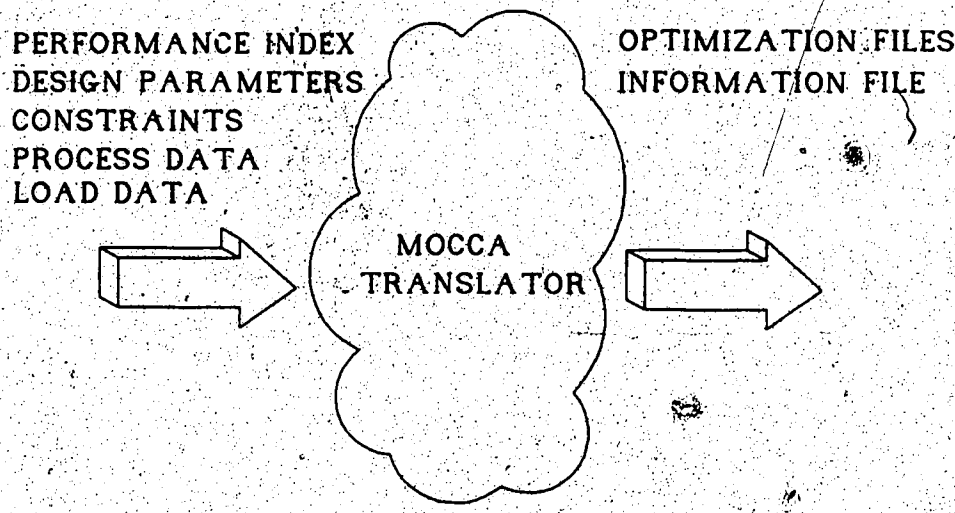

Figure 6.3: Schematic Representation of Translator Program

Using the translator program, the engineer specifies the objective function, constraints (if any), process step response, load step response (if any), design parameters (e.g. prediction horizon, P; control horizon, M; input and output weightings etc.) and a translated problem is written to output files, which are later used by the advanced control algorithm routines in the simulator or in an actual implementation. The current translator program includes translation for the following optimization problems: weighted least squares, linear programming and quadratic programming. Figure 6.4 shows the program structure diagram for the translator program.

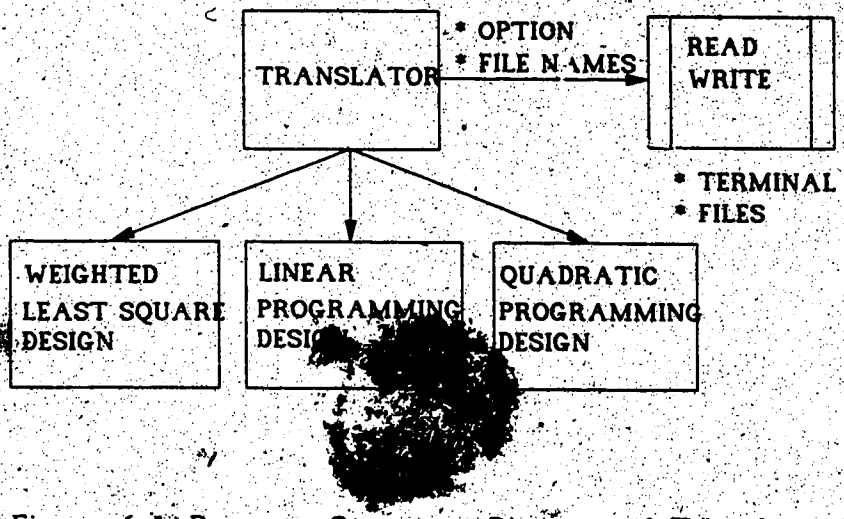

Figure 6.4: Program Structure Diagram of Translator Program

Each of the blocks under the translator block represent a separate controller design. In the weighted least squares case, a controller file and a design information file will be produced at the end of the translation. For the linear and quadratic programming translation, a controller file, mathematical programming file (optimization routine,data file), constraint file (constraints specified for the problem) and design information file will be produced. In each of the designs, the engineer has to provide the step response files for the process and load disturbance (if any). Information on the design parameters such as prediction horizon, control horizon, weighting etc. is obtained from the engineer through a series of questions. Default values are provided for all the design parameters but an opportunity is given to the engineer to change the default value or answer given.

example for M=10, a set of 10 calculated input changes are implemented for the next 10 control intervals before another 10 input changes are calculated.

In the presence of model process mismatch (in this case the presence of an unknown delay, $\tau_p - \tau_m = 5$) increasing the control horizon, M tends to deteriorate the performance as shown in Figure 7.9. This is because for M approaching P, the controller approaches a one step ahead predictive controller (equivalent to P=M=1). The controller therefore overcompensates the model process mismatch.

The ratio of M to P can be used as an additional guideline in choosing P and M. The ratio should be small if the objective is stability and robustness and large if performance is important bearing in mind the inevitable trade-off between the two criteria. In general, higher performance (M/P approaches one) assumes an accurate process model plus feedback prediction. The choice of P and M for system with unknown delays should be such that the difference between P and M is greater than or equal to the maximum estimated delays.

7.2.4 Effect of Input Weighting

Γ is the diagonal input weighting matrix in the performance index. The weighting matrix is usually a identity matrix multiplied by a scalar value. However,

different weighting can be assigned for each step in the future control signal (e.g. future control signals can be penalized heavier than the current control signals). Increasing the input weighting would decrease the magnitude of the change in the input variable, i.e. it penalizes the movement of the input variable.

For constrained optimization algorithms (i.e. linear and quadratic programming) hard constraints on the changes in input variables can be used to obtain some of the same objectives produced by input weighting. In the SISO case the input weighting is of limited use when LP is used. The weighting comes into use in the multivariable case where the input weighting can be used to change the relative economic cost of one input variable relative to another (e.g. on the evaporator since steam is more expensive than pumped flowrate, larger weighting can be assigned to the steam variables than the flowrate variable in the cost function, thereby penalizing the excessive use of the higher cost (steam) variable).

Figure 7.10 and Figure 7.11 shows the effect of input weighting on system 1 and 2 respectively. Generally increasing input weighting results in smaller changes in the input variable and a more sluggish output response for system 1 and a more oscillatory response for system 2. Notice that the calculated changes in input variable,

particularly the first one, decrease in value when the input weighting is increased. Therefore the input weighting can be chosen such that the initial change in input due to a step change in error is within physical limits thus providing "soft constraints" for the changes in input variable.

Figure 7.12 illustrates the use of hard constraints on the change in input variable when a constrained optimization algorithm is used. The linear programming algorithm was designed using the same parameters as in Figure 7.10 with the exception of the input weighting which was replaced by constraints on the change in the input variable. The value of the constraint was made equal to the initial change in the input variable when subject to a step change in error for input weighting of five in the weighted least squares criterion (changes in input variable are limited to and upper and lower bound of 0.6635). The response for both systems is almost the same with the difference attributed to the cost function being optimized, i.e. weighted least squares optimizes a quadratic cost function whereas the linear programming optimizes the sum of the absolute error. A more direct comparison could be made using a quadratic programming algorithm.

Input weighting can be used to penalize the control action resulting in smaller changes in input for a give change in error. The weighting can be used as a soft constraint to prevent the excessive use of the control action. Hard constraints on the input variable can be used in place of input weighting when constrained optimization is used. In multivariable systems, the input weighting can be used to penalize the more "expensive or important" input variable(s) so that minimal control movements are obtained. This will be shown in later examples.

7.2.5 Effect of Output Weighting

Γ is the output weighting matrix which is usually a scalar multiplied by an identity matrix. Output weighting can be varied for each output value in the prediction horizon. Thus any subset of the predicted output values can be selected for use in the controller optimization. A limiting case would be to select a single point $y(k+\lambda)$ by setting $\gamma(k+i)=0 \forall i$ except $i=\lambda$. In the multivariable case, relative output weighting can be use to place more importance on one output variable over another.

Figure 7.13 to Figure 7.15 shows the effect of individual output prediction weighting on the performance of systems 1, 2 and 3 respectively. For system 1, heavy weighting on the first five output predictions resulted in a

slightly faster response (Figure 7.13). For the underdamped process, system 2, heavy weighting on the first five output predictions resulted in less overshoot and oscillation in the response (Figure 7.14). Heavy weighting on the first five output predictions for the non-minimum phase system resulted in a sluggish response in exchange for a smaller inverse response (Figure 7.15). Output limiting can be achieved when using LP by placing upper and lower bounds on the output (i.e. deviation around the setpoint). Figure 7.16 shows the result of constraining the output variable for system 2 where a deviation of 0.1 was placed around the output variable during a setpoint change. Notice that the output variable was held to the upper bound and that the movement in the manipulated variable is significantly different. The output response using LP is comparable to the one obtained using output weighting in the weighted least squares case (cf. Figure 7.14).

Output weighting can be used to place more importance on the individual output predictions in both SISO and MIMO systems. The response of the system can be "shaped" by the use of individual output weightings. The output weightings can also be used to place more importance on one variable relative to another in MIMO systems as shown in later

examples. Hard output constraints can be interpreted as output weighting (i.e. infinite weighting on the output variable) when constrained optimization is used.

7.2.6 Effect of Setpoint Filtering

The setpoint trajectory can be generated in several ways, each with a different level of complexity e.g. from simple step changes to using high level optimization programs. A simple way of introducing model-following in MOCCA is to use setpoint filtering. For example, a first order filter can be used to generate an exponential setpoint trajectory once the step in setpoint is specified. Depending on the filter constant, setpoint filtering slows down the response to the speed of the first order filter as can be seen from Figure 7.17. Robustness is improved at the expense of performance. A one step ahead predictive controller can be used to achieve perfect model follow[illegible]s was shown in Appendix F.

7.2.7 Feedforward Control

If the disturbance is measurable then the use of feedforward will improve regulatory control for the system. Feedforward can be achieved by predicting the future effect of the disturbance on the output variables using a disturbance step response model (i.e. same as the output predictions). Figure 7.18 and Figure 7.19 show the response

without and with feedforward control for two types of load disturbances. A step disturbance added directly to the process output (i.e. a disturbance with fast dynamics) represents the worst case scenario whereas a filtered disturbance (with dynamics slower than the process) represents a typical disturbance in industry. Good disturbance handling can be achieved when the dynamics of the load are approximately the same or slower than the process.

When the time delays of the process and load disturbance are different and known, then time delay compensation (in addition to compensation built into the predictive models) can be implemented. Figure 7.20 shows the effect of additional time delay compensation when the load delay is greater than the process. In this case without the additional compensation, the controller reacted to early resulting in poor performance compared to the compensated case. Figure 7.21 shows the performance when the load delay is less than the process delays. (This means that the disturbance will affect the output before the feedforward compensation and hence the initial error will be large.) Although the effect is not as dramatic as the earlier case, significant improvement was obtained.

A perfect feedforward controller would contain the inverse of the process model. In MOCCA this could be approximated by setting P=M=1 and Γ_m=0 in the predictive controller. However in these examples, the controller has been detuned by using P=10, M=1 and Γ_m=5 to provide smoother control and greater robustness to modeling errors. Therefore the feedforward compensation is not perfect. Perfect feedforward control in MOCCA can then be achieved if the process model, disturbance model and predictive controller are perfect.

7.2.8 Effect of MPM on Performance

If properly designed (assuming a system without delay) the MOCCA control algorithm still performs reasonably well when small (unknown) time delays are introduced into the system as can be seen from Figure 7.22. As the number of unknown time delays increases, the performance becomes more oscillatory (deteriorates). For known output delays compensation equivalent to a Smith predictors is built into the MOCCA model predictions. Additional compensation is also implemented to handle known and different delays between the process and load disturbance when feedforward control is used (see section 1.2.7).

Figure 7.23 and Figure 7.24 show the performance of the controller when the actual process gain and time constant

are different from the assumed model values. The gain of the process model was changed (from true value of 1.0) to reflect a mismatch in the gain for Figure 7.23 and step response data with a different time constant were used in the simulator for Figure 7.24. In both cases, the system is robust to small changes. If the gain is known, then on-line compensation can be introduced by using the varying gain option. For SISO applications, the varying time constant can be compensated by changing the sampling time.

7.2.9 Effect of Feedback Filtering

A first order filter in the feedback path is suggested in the IMC literatures as a way to detune the control system for cases where excessive model process mismatch or disturbances are present. This provides robustness at the expense of performance. By filtering the residual which contains the model process mismatch and/or disturbances, the residuals fedback to the controller are smaller than the actual residual and thus the control action is less vigorous. Figure 7.25 illustrate the use of a first order filter to detune the controller when there is model process mismatch (unknown time delays) in the system.

7.2.10 MOCCA Design Procedure for SISO Systems

The MOCCA controller can be designed using the translator/simulator package. The only requirement is that

the step response data files be available. The following procedure outlines the necessary steps in designing a controller.

1. Choose a sampling interval, T_s and N such that the response is within five percent of the steady state value at time T_s*N, i.e. settling or response time of the system. In the case where there is an inverse response, the inverse response part must be included in the step response and in the optimization horizon, $P*T_s$. Another criterion for choosing N is the truncation errors in the prediction should be of the same magnitude as the measurement errors.
2. Choose a prediction horizon, P and control horizon, M. (Reasonable default values are P=10, M=3). Vary the input weighting, $r_m = k$, and calculate the sum of the absolute errors and changes in input which can then be plotted against the weighting as in Figure 7.26. (P and M are equal to 10 and 3 respectively). In Figure 7.26, a negligible change in the sum of the absolute errors and changes in input was obtained for $0 \leq k < 0.01$. The weighting should be chosen such that the sum of the absolute changes in input variable decreases rapidly in exchange for a slight increase in the sum of the absolute errors. Ideally then the weighting should be 0.012 for this

example. However one can make an engineering judgement on the trade-off between output errors and changes in input. The final value was chosen to be 0.02.

3. With a fixed P, and input weighting, the parameter M can now be increased or decreased and the response recorded. Figure 7.27 shows that for M=1 the response became slower with no overshoot and that any value greater than M=1 results in faster response with slight overshoot (for $M \geq 3$ the responses were virtually the identical). The first change in input value for a change in error remains constant for M greater than 2 indicating that the responses would be almost identical.
4. Use the feedback filter to detune the final control system on-line or the procedure can be repeated with different parameters.

This procedure can also be performed for the multivariable case but the procedure is more complicated.

7.3 Evaporator Applications

The evaporator used in this study is the double effect pilot plant evaporator located in the Department of Chemical Engineering, University of Alberta. A detailed description of the evaporator can be found in Fisher and Seborg (1976). A schematic diagram of the evaporator is shown in Figure 7.28.

7.3.1 Description of Evaporator

The first effect of the evaporator is a natural circulation calandria type unit with thirty two 3/4 inch OD tubes, 18 inches long. Triethylene glycol feed of fixed concentration is fed to the first effect and process steam is used to heat the solution. The concentrated product solution is then fed to the second effect. The second effect is a forced circulation evaporator with three 6 foot long, 1 inch OD tubes. It is operated under vacuum and is heated by the overhead vapor from the first effect.

The primary controlled variable is the final product concentration, C_2, coming out of the second effect. With SISO controllers, it is controlled by manipulating the steam flow rate, S. Other important controlled variables are the first effect holdup, W_1 and second effect holdup, W_2. The manipulated variables used to controlled the holdup are the first effect bottom flow rate, B_1 and the second effect bottom flow rate, B_2, respectively.

The evaporator is fully instrumented with industrial electronic instrumentation and can be controlled either by conventional PID electronic controllers (as in Figure 7.28) or under direct digital control (DDC) using a Hewlett Packard 1000/A700 computer. A process control software package, PMC (Process Monitoring and Control)/1000 supplied by Hewlett Packard is used to provide the DDC. The PMC/1000

database can be accessed by advanced control routines through an interface program. As a result advance control techniques can be easily applied on the evaporator pilot plant.

7.3.2 Simulated Evaporator Applications

The application of MOCCA to a fifth order non-linear model of the double effect evaporator is simulated in this section. A description of the model is presented and some simulation results are produced and discussed.

7.3.2.1 Description of Evaporator Model

Several models of the double effect evaporator ranging from a tenth order non-linear model to a first order transfer function model have been developed (Newell, 1971; Wilson, 1974). The evaporator model used in this simulation is the fifth order non-linear model developed by Newell (1971). The model consists of five ordinary differential equations and a set of algebraic equations given by:

Steam Chest:

$$S\lambda_s = U_1 A_2 (T_s - T_1) \tag{7-4}$$

First effect:

$$\frac{dW_1}{dt} = F - B_1 - O_1 \tag{7-5}$$

$$W_1 \frac{dC_1}{dt} = F(C_0 - C_1) + O_1 C_L \tag{7-6}$$

$$W_1 \frac{dh_1}{dt} = F(h_0 - h_1) + O_1(H_{v1} - H_1) + Q_1 - L_1 \tag{7-7}$$

where

$$Q_1 = U_1 A_1 (T_s - T_1) \tag{7-8}$$

$$O_1 = \frac{(Q_2 + L_2)}{(H_{v1} - h_{c2})} \tag{7-9}$$

$$Q_2 = U_2 A_2 (T_1 - T_2) \tag{7-10}$$

Second effect:

$$\frac{dW_2}{dt} = B_1 - B_2 - O_2 \tag{7-11}$$

$$W_2 \frac{dC_2}{dt} = B_1(C_1 - C_2) + O_2 C_2 \tag{7-12}$$

where

$$Q_2\left(H_{v2} - h_2 + \frac{\partial h_2}{\partial C_2} C_2\right) = \dot{Q}_2 - E_2 + B_1(h_1 - h_2) + \frac{\partial h_2}{\partial C_2} B_1 (C_2 - C_1) \tag{7-13}$$

Property relations are given by

$$H_v = 0.4T_v + 1066.0 \tag{7-14}$$

$$h_c = T_v - 32.0 \tag{7-15}$$

$$h_{sol} = T_{sol}(1.0 - 0.16C_{sol}) - 32.1 \tag{7-16}$$

The differential equations were solved using a fourth order Runge-Kutta method. Steady state operating data were obtained from Wilson (1974) and are tabulated in Appendix J.

7.3.2.2 Simulation Results.

The step responses for the fifth order non-linear evaporator model are presented in Figure 7.29. The evaporator model contains two open loop unstable (integrator) modes corresponding to the two levels. These were stabilized by low gain (-1.0 and -2.75 for level 1 and 2 respectively) proportional feedback control before MOCCA was applied. MOCCA in this MIMO application calculates the steam flow rate and the level setpoints for the proportional level control loops used to control the second effect concentration plus the first and second effect levels. Note that in this case, MOCCA uses the stabilized system and can be interpreted as the master controller with the two proportional level controllers as slave loops. If a system has local controllers that give good performance, then MOCCA can be added as a second level control to optimize the overall control performance by manipulating the setpoints to the local controllers. (However, the local controllers should be tuned for stabilization only and not for "tight control".)

Figure 7.30 shows the result of using the MOCCA weighted least squares algorithm when the total number of step response coefficients for all the input output pairs is restricted to 80. Significant truncation error occurs in this case and its effect can be seen by the "bump" in the evaporator responses at time $t=N*T_S$. The truncation error can be minimized by: 1) increasing the sampling interval so that the value of N=80 covers more of the output responses, 2) increasing N to cover more response data and 3) modifying the available step response data such that the final few values approaches the steady state. When computing power is available, option 2 is an attractive choice. When N was increased, the truncation error was reduced as shown in Figure 7.31. (Figure 7.31 and Figure 7.32 uses the same controller). The setpoint change in Figure 7.32 (20% increase) in C_2 results in a good C_2 response with minimal deviation in the levels. Similarly, the response to a step feed flow disturbance at t=101 is also quite good. The good control comes at the expense of relatively large deviations in the manipulated variables (no input weighting was used). Note that B_2 is less than -1.0 which indicates a negative flow rate. A preview of the controller performance can be obtained using the MOCCA simulator rather than a special purpose simulation program. The simulator can use the same step response data in the process simulation as for the

controller calculation (i.e. zero model process mismatch) or separate step response data can be supplied to study the effect of model process mismatch. Figure 7.33 (the step response data used in the controller was used in the process simulation) shows that similar results can be obtained using the simulator. Hence the simulator can provide a powerful tool to tune the controller.

Input weighting can be used to penalize the control action. In Figure 7.34, equal input weighting was used to penalize all the manipulated variables, resulting in a slight overshoot in C_2 but smaller excursions in the manipulated variables. Note that the manipulated variables are all within physical limits, i.e. no negative flows. The input weightings can be decreased to obtain better performance in C_2 as shown in Figure 7.35. In this case better performance was achieved at the expense of the manipulated variables. (The same is true when comparing Figure 7.35 versus Figure 7.32). Output weighting can also be used to achieved better performance as shown in Figure 7.36, where the variable C_2 was weighted 10 times more than the levels. Here slightly better C_2 performance was achieved at the expense of first effect level, W1. These simulations show that both input and output weightings have a significant effect on the response of the closed loop system.

The 5th order non-linear and linear responses of the evaporator for a step in the feed flowrate are shown in Figure 7.37 (note the difference in the steady state gains). The linear and non-linear responses are used in the design of the feedforward control portion of MOCCA. Figure 7.38 shows the responses of a "perfect" (i.e. P=M=1) controller to a 20% increase in feed flowrate without and with feedforward control. The weighted least squares controller and the feedforward control control for Figure 7.38 were designed using linear step and disturbance responses respectively. The simulations were performed using a fifth order linear evaporator model. When feedforward control is used the total absolute error is significantly smaller compared to the case when no feedforward is used. As stated earlier, perfect modeling and perfect control will give perfect disturbance rejection. In the case where the controller is an approximation (i.e. P not equal to M and $\Gamma_m \neq 0.0$) then perfect disturbance rejection cannot be achieved even when the models are perfect. Figure 7.39 shows the non-linear evaporator responses using MOCCA for a 20% increase in feed flowrate without feedforward control (the weighted least squares controller and feedforward control for Figure 7.39 and Figure 7.40 were designed using the non-linear step and disturbance responses respectively). Figure 7.40 shows the responses of the non-linear evaporator

when feedforward is used. Compared to Figure 7.39, significant improvement is obtained by using feedforward control. In the non-linear run the problem of using a non-perfect model and controller is more noticeable (compare to Figure 7.38).

Linear programming can be used to constrained the manipulated variables so that physical limits are not violated, thus eliminating the use of input weighting to "tune" the closed loop system. The input weighting can then be used to provide different economic weights to the manipulated variables as well as to improve the numerical conditioning of the LP problem. Figure 7.41 and Figure 7.42 show the use of input weighting to improve the numerical conditioning of the LP problem. In Figure 7.41, no input weighting was used, i.e. the objective function is IAE. Notice the unexpected sudden changes in the manipulated variables. Figure 7.42 is designed as in Figure 7.41 except input weighting was used to provide better numerical conditioning. In addition to providing better numerical properties, the input weighting can be used to penalize the manipulated variables, i.e. the more expensive or important variables can be penalized more. Figure 7.43 shows the effect of using the input weighting to penalize the manipulated variables. The steam, S and the product flow rate, B_2 were penalized more than the intermediate product

flowrate, B_1 since steam is more expensive and big fluctuations in the product flowrate are often not desirable especially when it is a feed stream for downstream units. Output weighting can still be used to place more importance of one output variable over another. Figure 7.44 shows the result of output constraints to minimize interactions in a multivariable processes. In Figure 7.44 (deviational) constraints of 0.01, 0.001 and 0.001 on the upper bound of C_2, W_{1s} and W_{2s} respectively were specified (compare versus Figure 7.42). Figure 7.45 shows the effect of using constraints on the input variables. The steam was constrained to an upper bound of 0.5 and a lower bound of -1.0 (zero steam flowrate). The setpoint to the first effect level feedback loop was set to an upper bound of 0.5 and a lower bound of -0.5 and the setpoint to the second effect level feedback loop was set to an upper bound of 0.182 and a lower bound of -0.182. The main difficulty here is converting "level setpoints" constraints into bottom flowrates constraints which are more familiar.

7.4 Other Simulated Applications

In this section, MOCCA is applied to some published examples so that direct comparison to some of the other algorithms can be performed. The results obtained using

other algorithms are not reproduced here because of copy right restrictions. However, references are given to widely available publications.

7.4.1 Cutler's Furnace Problem

Cutler and Ramaker (1980) and Ogunnaike and Adewele (1985) described the use of DMC to control a crude oil furnace. Figure 7.46 shows the step responses relating the fuel flow rate and a disturbance variable (inlet temperature of the crude oil) to the outlet temperature of the furnace. Table 7.1 shows the numerical values of the step response coefficients as published by Ogunnaike and Adewele (1985).

Table 7.1 Step Response Coefficients for Furnace Problem

Outlet temperature due to step in fuel flow rate	0.050, 0.092, 0.220, 0.415, 0.600, 0.740, 0.835, 0.905, 0.950, 0.975, 0.985, 0.988, 0.990, 0.990
Outlet temperature due to step in inlet temperature	0.000, 0.250, 0.345, 0.465, 0.550, 0.600, 0.635, 0.655, 0.670, 0.680, 0.685, 0.685

Figure 7.47 shows a typical run for the furnace problem. The objective here is to point out that no significant improvement can observed when P, the prediction horizon is chosen equal to N, the number of step response coefficients for both servo and regulatory control. The DMC formulation

as pointed out in Chapter 5 uses a prediction horizon equal to N+M (number of step response coefficients plus the control horizon).

7.4.2 Cement Kiln

Graham et al. (1985) applied DMC and MBC (model based control using fuzzy sets) to the multivariable control of a linear cement kiln model. The cement kiln model (obtained by fitting experimental data) was given as:

$$\begin{bmatrix} \Delta BZ \\ \Delta BE \\ \Delta OX \end{bmatrix} = \begin{bmatrix} \frac{-0.049}{70s+1} & \frac{0.024}{70s+1} & \frac{-0.111}{70s+1} \\ \frac{-0.037}{70s+1} & \frac{0.020}{70s+1} & \frac{0.089}{70s+1} \\ 0.0 & \frac{-0.0026}{40s+1} & \frac{0.010}{40s+1} \end{bmatrix} \begin{bmatrix} \Delta KF \\ \Delta BF \\ \Delta FS \end{bmatrix} \qquad (7\text{-}17)$$

where the measured variables BZ, BE and OX are the burning zone temperature, back end temperature and percentage oxygen in the exit gas respectively. The manipulated variables are KF, BF and FS which are the kiln feedrate, fuel flowrate to the burner and speed of the induced draft fan. The unit step responses are plotted in Figure 7.48.

In all of the simulations, three simultaneous setpoint changes were introduced, $\Delta BZ = -24.0$, $\Delta BE = -8.0$ and $\Delta OX = -0.95$. Figure 7.49 shows the responses of MOCCA weighted least squares controller with a sampling time of 2 minutes. Input weightings were chosen such that the initial changes in inputs are approximately within the prescribed limits. In

Figure 7.49, the variable OX was slow in settling back into the new steady state. Heavier output weighting can be used to place more importance on this variable and thereby speed up the settling time as shown in Figure 7.50. Compared to DMC and MBC, the results were comparatively better (see Table 7.2 for a summary). Figure 7.51 was performed using a sampling rate of 10 minutes.

Table 7.2

Average Absolute Error Per Minute For DMC, MBC and MOCCA

Method (Sample Time)	BZ	BE	OX	Total
DMC (2 Mins.)	4.5	0.8	0.15	5.5
MBC (2 Mins.)	5.1	1.5	0.05	6.6
MOCCA WLSQ (2 Mins.)	0.483	0.242	0.172	0.897
MOCCA LP (2 Mins.)	0.493	0.169	0.013	0.675
DMC (10 Mins.)	5.0	1.0	0.9	6.9
MBC (10 Mins.)	3.6	1.1	0.04	4.7
MOCCA WLSQ (10 Mins.)	0.105	0.046	0.025	0.176
MOCCA LP (10 Mins.)	0.093	0.027	0.002	0.122
MBC (60 Mins.)	3.8	2.6	0.05	6.5
MOCCA LP (60 Mins.)	0.039	[illegible]	0.002	0.052

The change in sampling time does not affect the performance of MOCCA, although the input weightings need to be re-specified so that the initial changes in inputs are within the prescribed limit. Figure 7.52, Figure 7.53 and Figure 7.54 (sample time of 2, 10, 60 minutes respectively)

illustrate the use of LP with change-in-input constraints. Note that the time axis is in sampling intervals rather than absolute time. The performance in all the LP and weighted least squares runs were much better than the ones presented by Graham et al. (1985). Table 7.2 summarizes the absolute error per minute for all the runs compared to that of Graham et al. (1985).

7.4.3 Simulated Distillation Column Using Industrial Data

Step response data relating the top and bottom composition of a distillation column to the temperature and reflux were obtained from an industrial column for preliminary testing of MOCCA. The coefficients are shown in Table 7.3 and the responses are plotted in Figure 7.55.

A weighted least squares controller without input weighting was designed and simulated using the simulator for 20% setpoint changes in the top and bottom composition. Figure 7.56 and Figure 7.57 show almost perfect setpoint tracking with little interaction achieved at the expense of large moves in the input variables, especially the reflux. Obviously the large moves in the input variables is not acceptable in industry, so the reflux movements were penalized (temperature movement was not penalized since its movement was relatively small). The simulated results are shown in Figure 7.58 and Figure 7.59. The setpoint tracking

is almost as good but with a slight overshoot. Notice that the reflux changes were reduced considerably. The initial reflux changes are still relatively large, so a heavier weighting can be used to further penalize the input variable or the reflux change could be ramped over a period of time to provide a smoother and slower change in the reflux. Another alternative would be to pass the step change in setpoint through a first order filter.

Table 7.3 Step Response Coefficients for Distillation Column

Top composition response due to step in temperature	0.0, 0.0, 0.0, 0.0, 0.273, 1.218, 1.678, 1.911, 2.037, 2.1, 2.1, 2.142, 2.142, 2.142, 2.142, 2.142
Bottom composition response due to step in temperature	0.0, 0.0, -3.665, -7.213, -8.988, -9.870, -10.311, -10.521, -10.637, -10.689, -10.721, -10.731, -10.731, -10.742, -10.742, -10.742
Top composition response due to step in reflux	0.0, 0.0, 0.0, 0.0, -0.252, -0.504, -0.672, -0.756, -0.784, -0.812, -0.84, -0.84, -0.84, -0.84, -0.84, -0.84
Bottom composition response due to step in reflux	0.0, 0.0, 0.0, 0.0, 0.154, 0.28, 0.378, 0.462, 0.531, 0.588, 0.644, 0.672, 0.7, 0.728, 0.756, 0.77

7.5 Summary

1. N, the number of step response coefficients should be chosen so that NT_s corresponds to the settling time of the process and truncation errors in the predicted values are of the same order as the measurement errors.

2. P, the prediction horizon and M, the control horizon are highly dependent on the type of system being controlled. Values of M between 1 and 3 and P between 5 and 10 appear to give reasonable responses for most systems. P should be chosen such that the desired closed loop response (i.e. rise time, overshoot etc.) is achieved. A good rule of thumb for choosing P in minimum phase systems is $P*T_s$ should be greater than the time constant. For non-minimum phase system, P should be chosen such that the period of inverse response is added to the prediction horizon calculated based on the balance of the response. Once P is chosen, M can be increased or decreased from its initial value to get the minimum value that does not result in a significant loss in performance. Another rule of thumb is that the ratio of M to P should be made sufficiently small, especially in non minimum phase systems. Controller robustness for handling model process mismatch can be achieved at the expense of performance by choosing a large value of P and a smaller value of M.
3. Input weighting should be chosen such that the first calculated change in input variable for a step change in error is reasonable, i.e. within physical limits, providing a "soft constraint" for the change in input

variable. In multivariable processes, the input weighting is used to penalize the input variable with higher costs, e.g. steam versus flowrate.

4. Individual output prediction weightings can be used to shape the closed loop response. Heavier weighting (large value) should be placed where the error between the setpoint and output prediction is to be reduced e.g. heavier weighting at the beginning of the output prediction reduces the oscillation for an underdamped system. In multivariable systems, the output weighting is used to place more importance (heavier weighting) of one output variable over another.
5. Setpoint filtering using a first order filter can be used to provide a smoother and slower response.
6. Feedback filtering using a first order filter can be used to detune the system when excessive model process mismatch and/or disturbances are encountered.
7. Feedforward control of measurable disturbance in MOCCA can be achieved by predicting the effect of the disturbances on the output variable. If the dynamics of the disturbance are approximately the same or slower than the process, good feedforward control can be achieved. Time delay compensations in addition to the one built

into the predictive model must be used to maintain good feedforward control when there are known and different delays in the process and disturbance.

8. Linear programming was used as a study case to show that constrained optimization can be used effectively with predictive control to provide hard constraints on the process variables. Input and output weightings are also used to provide additional on-line tuning. Input and output weightings are still used to penalize or place more importance of one variable over another. The input weighting is also used to provide better numerical stability in the LP controller. Output constraints were shown to be effective for minimizing interaction in multivariable system.
9. Overall, MOCCA performed better or equal to the other methods presented in the literature.

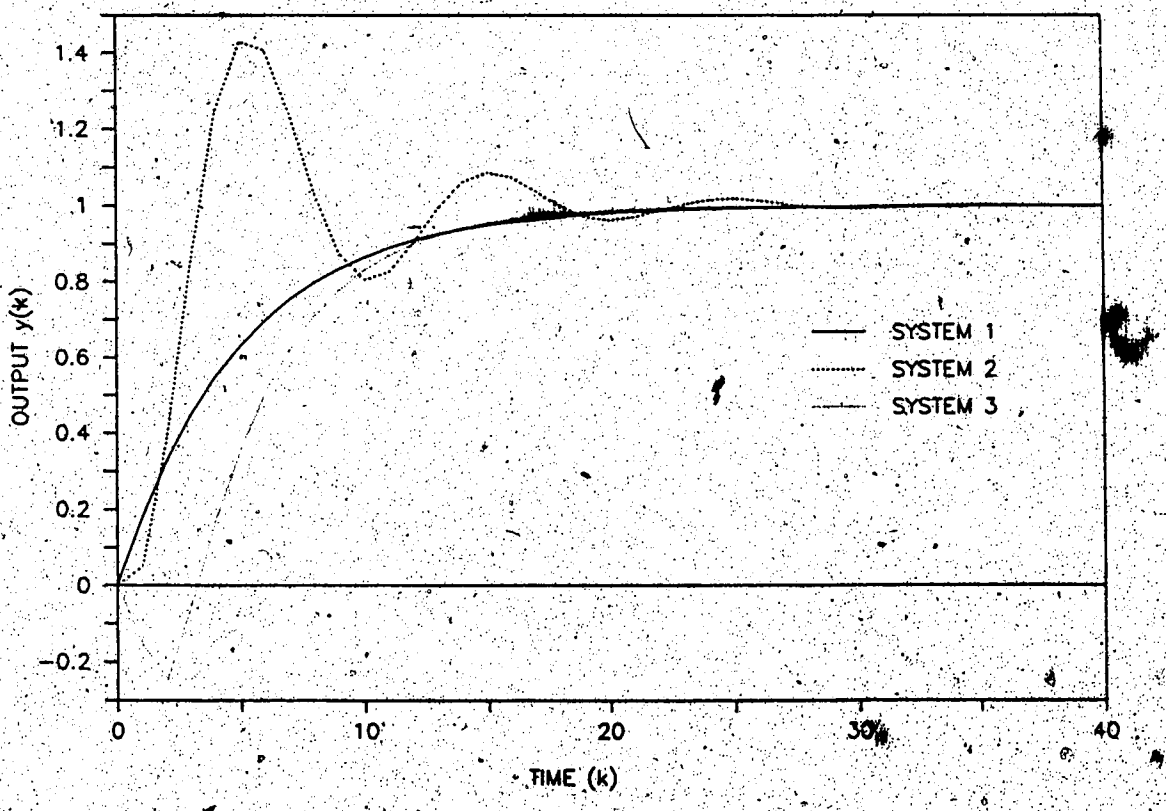

Figure 7.1: System Responses for Unit Step Change in Input.

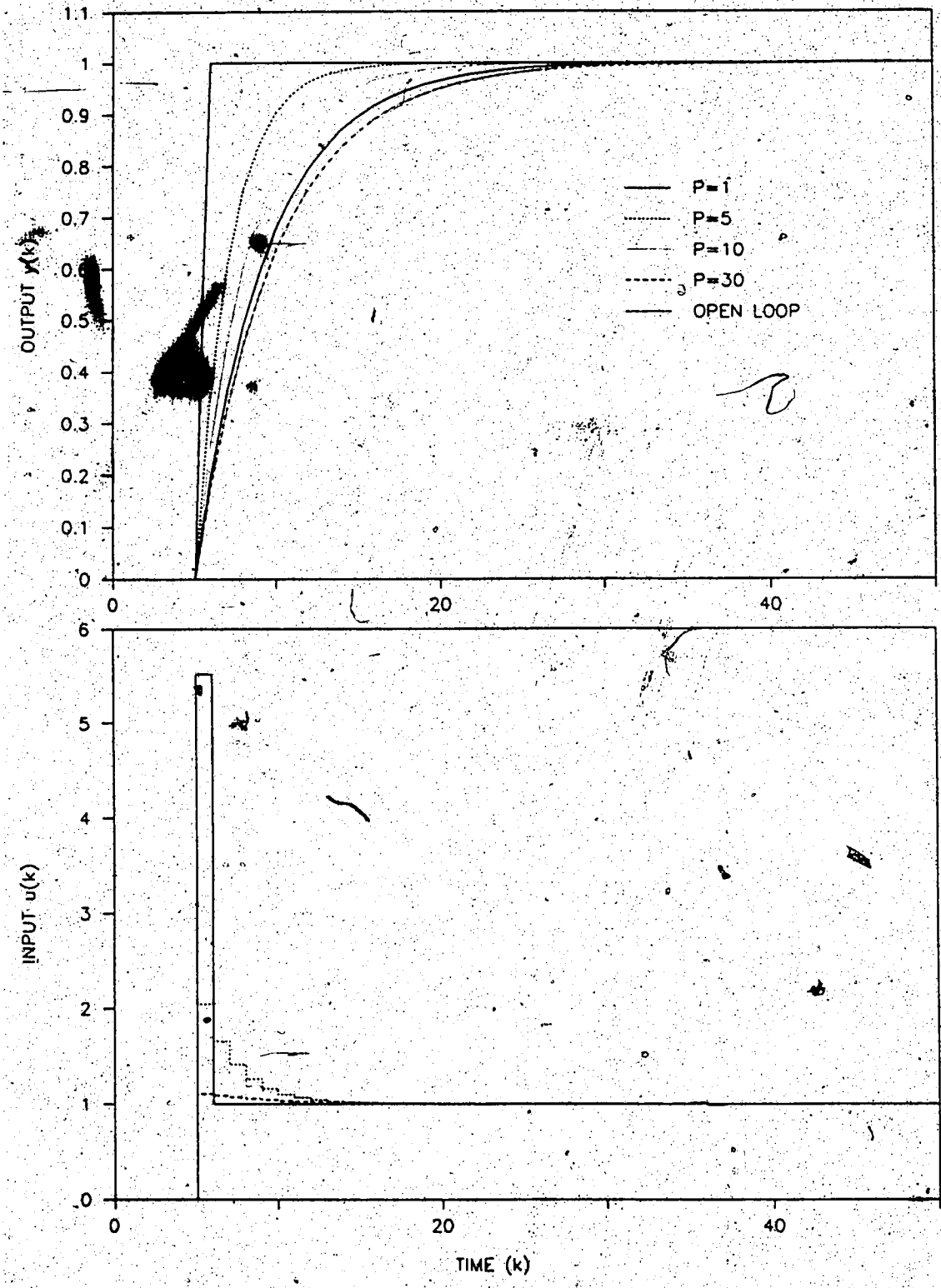

Figure 7.2: Effect of Prediction Horizon, P, on System 1.
N=30, M=1, $\Gamma = 1.0, \Gamma_u = 0.0$.

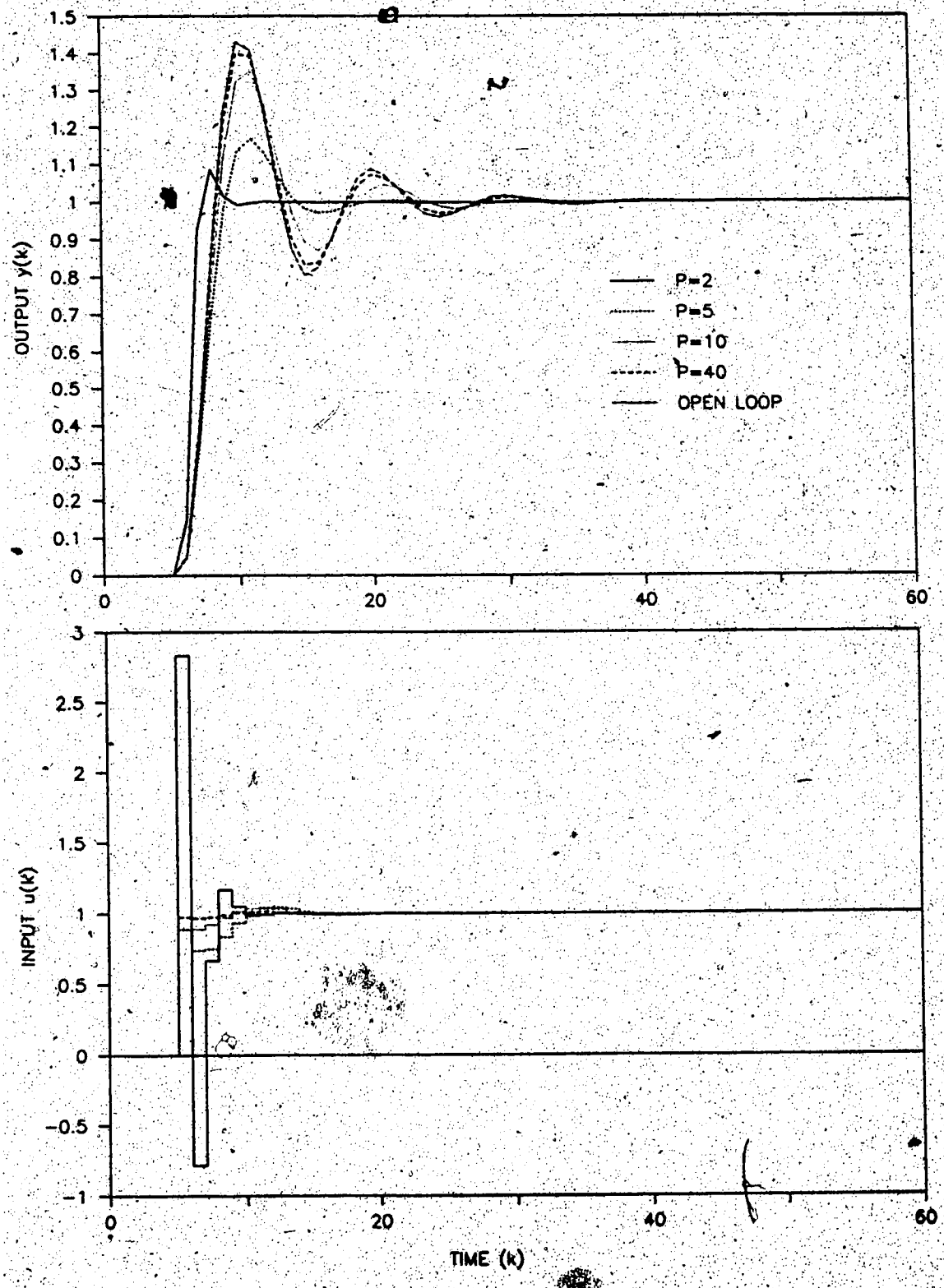

Figure 7.3: Effect of Prediction Horizon, P, on System 2. N=40, M=1, Γ=1.0, Γ_u=0.0.

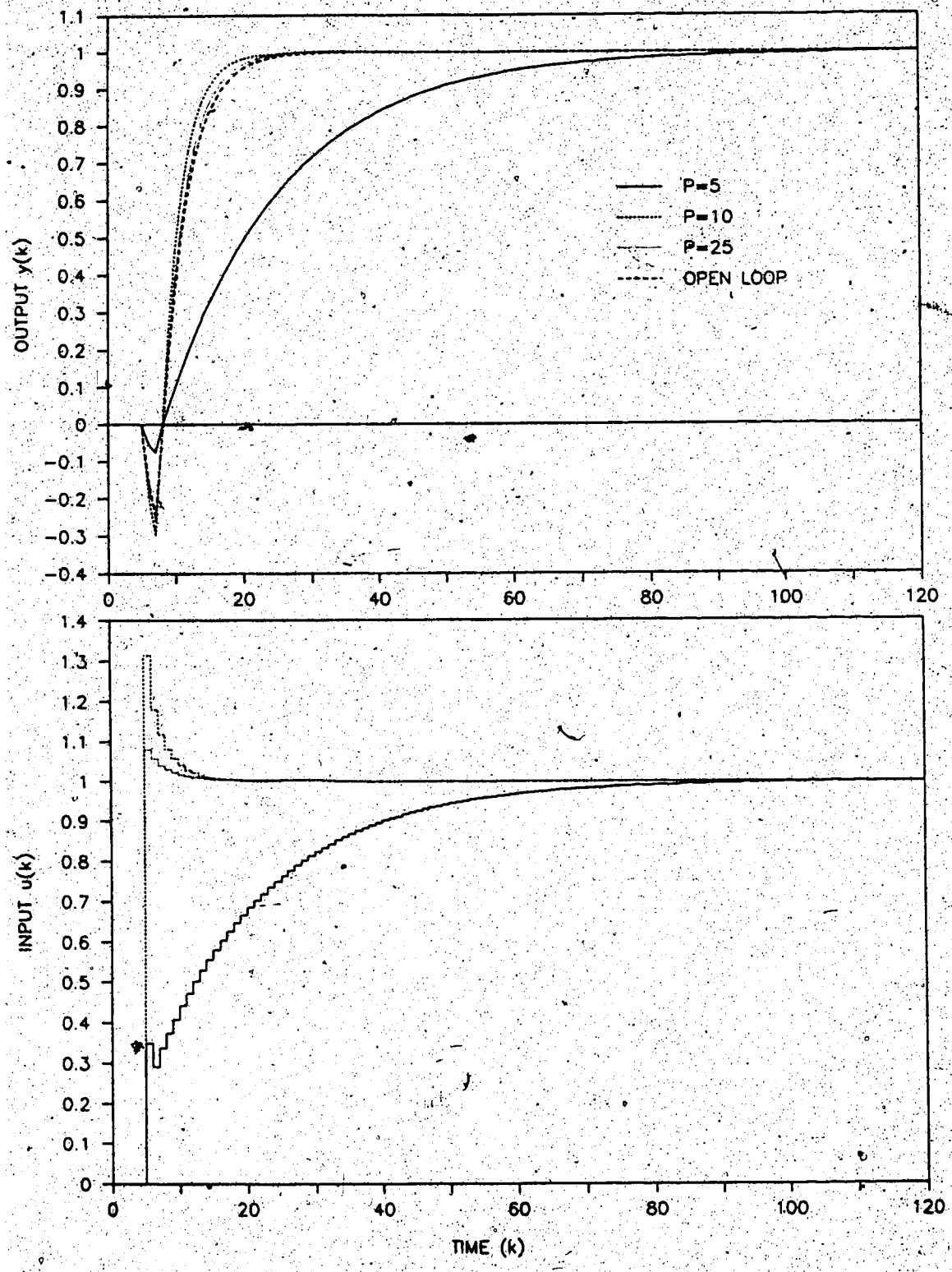

Figure 7.4: Effect of Prediction Horizon, P, on System 3.
N=25, M=1, $\Gamma = 1.0, \Gamma_u = 0.0$.

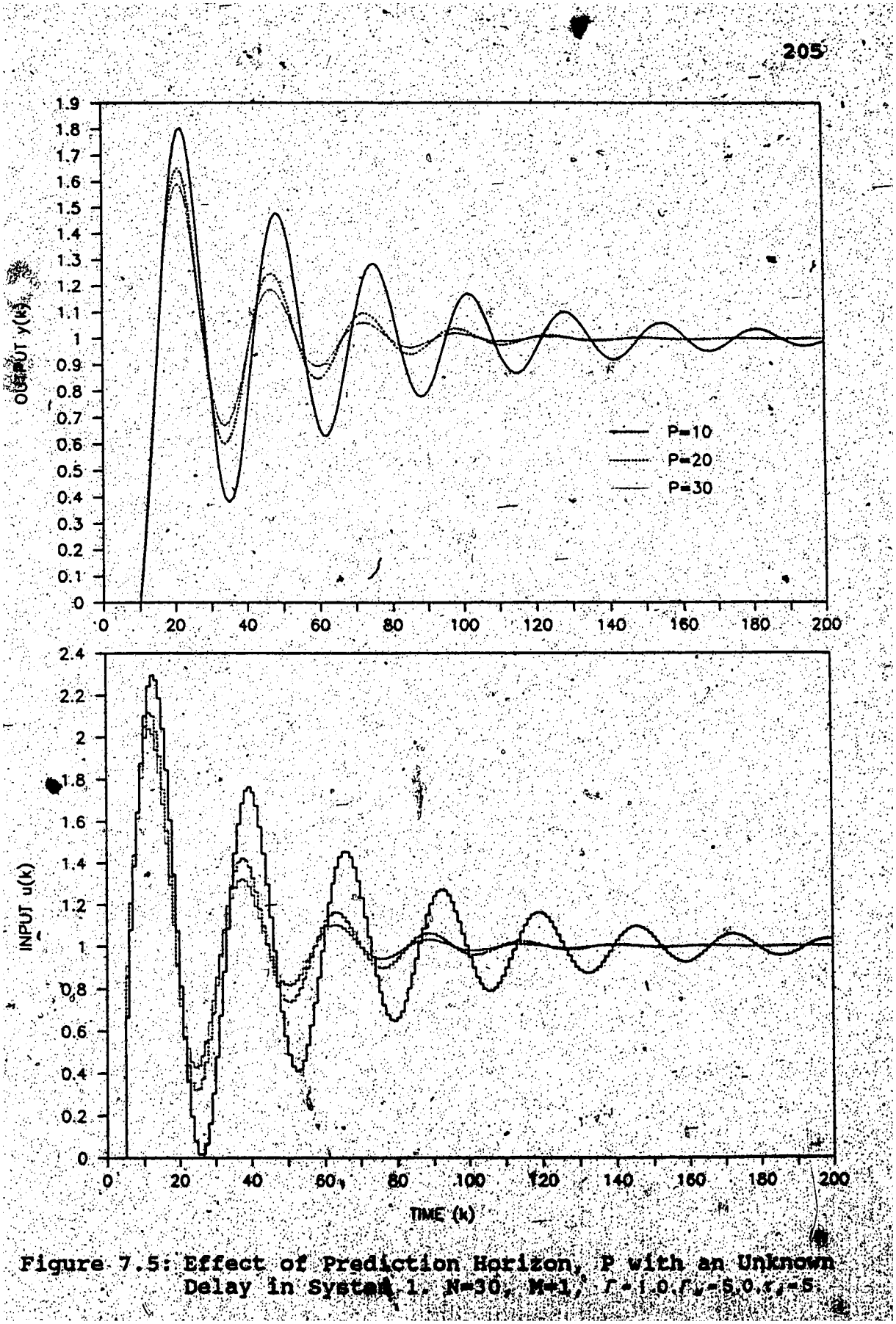

Figure 7.5: Effect of Prediction Horizon, P with an Unknown Delay in System 1, N=30, M=1, λ=1.0, r_u=5.0, r_y=5.

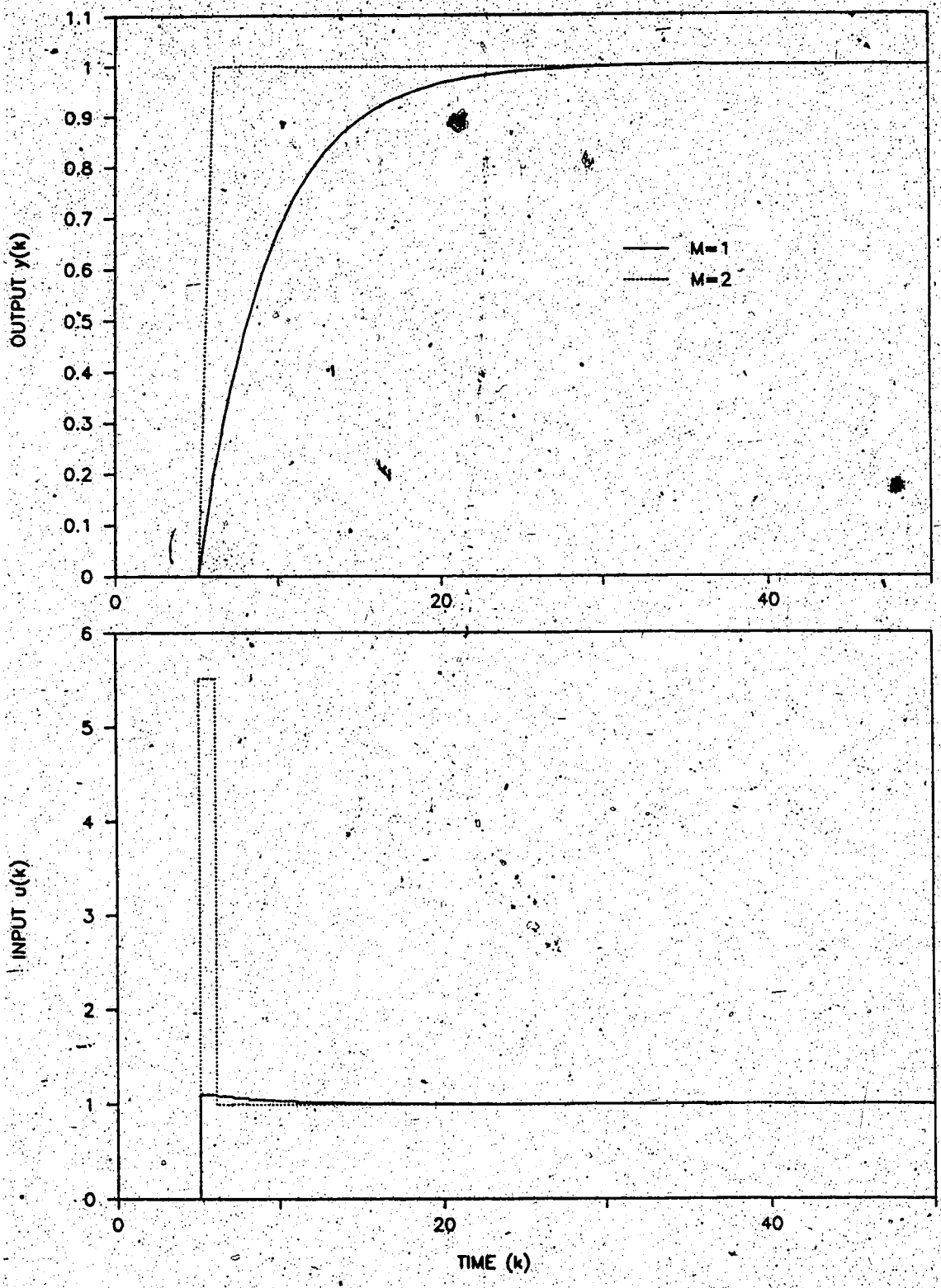

Figure 7.6: Effect of Control Horizon, M, on System 1.
N=30, P=30, $\Gamma = 1.0, \Gamma_M = 0.0$.

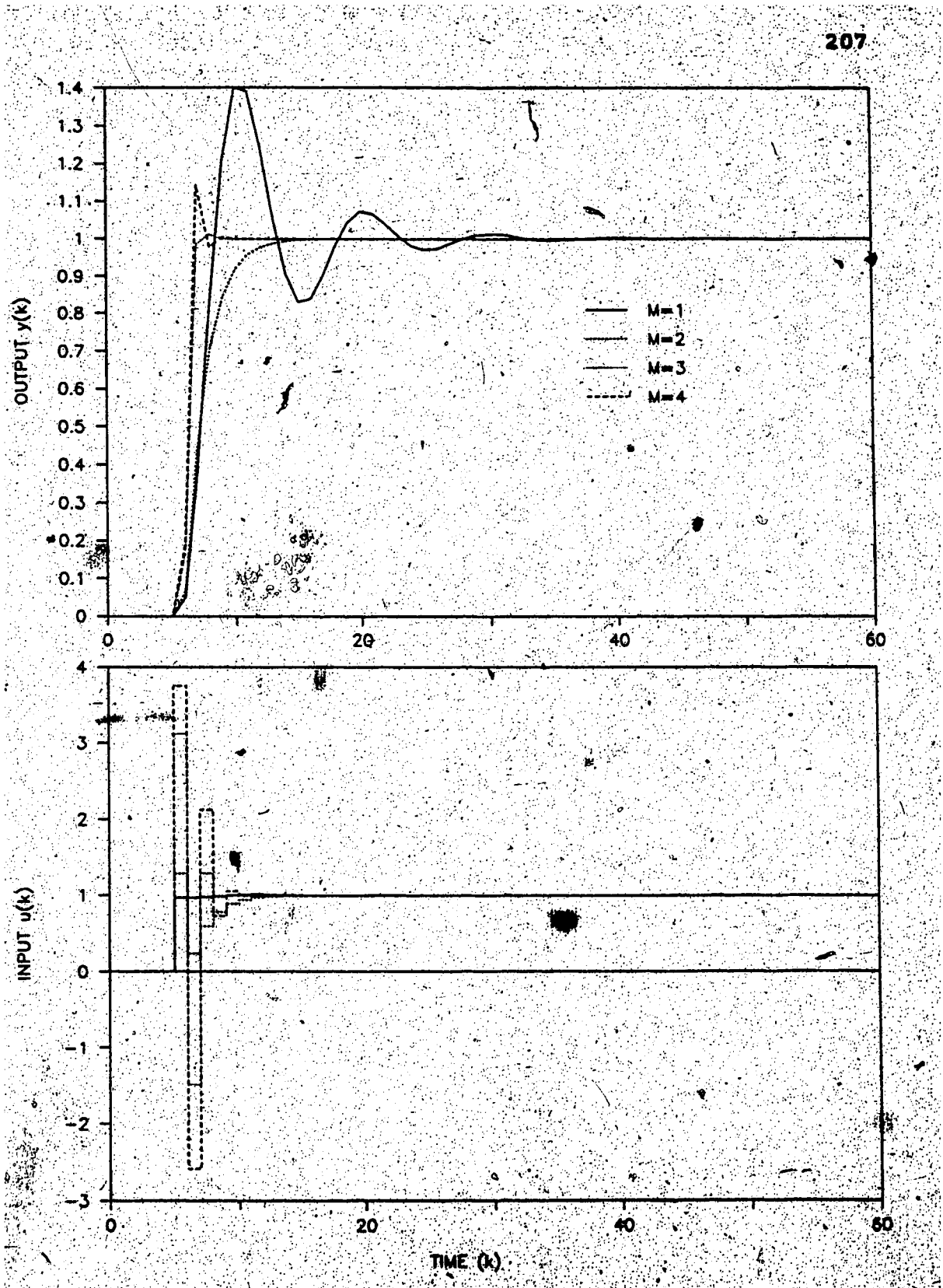

Figure 7.7: Effect of Control Horizon, M, on System 2.
N=40, P=40, Γ=1.0, λ=0.0.

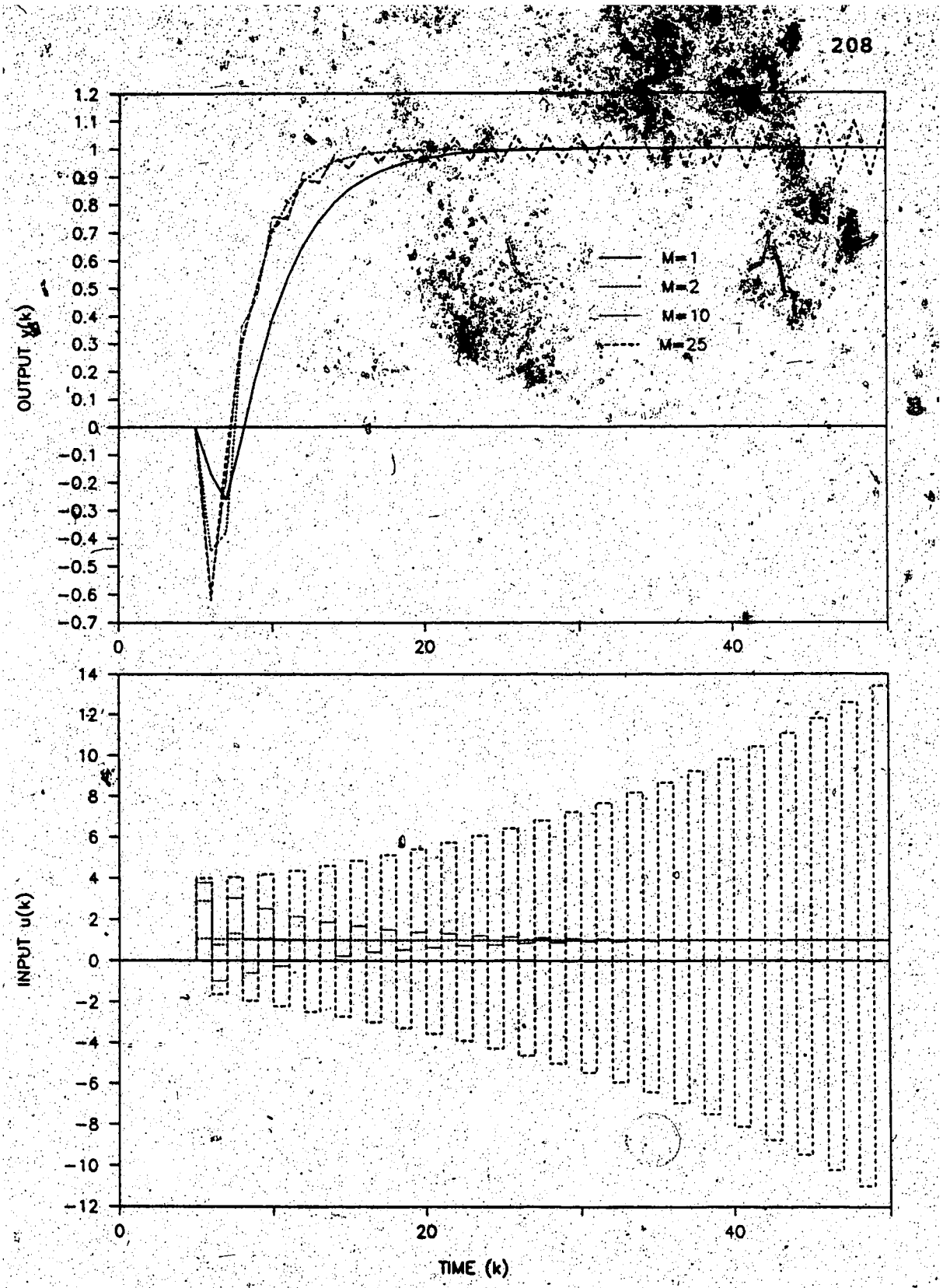

Figure 7.8: Effect of Control Horizon, M, on System 3.
N=25, P=25, $\Gamma = 1.0, \Gamma_u = 0.0$.

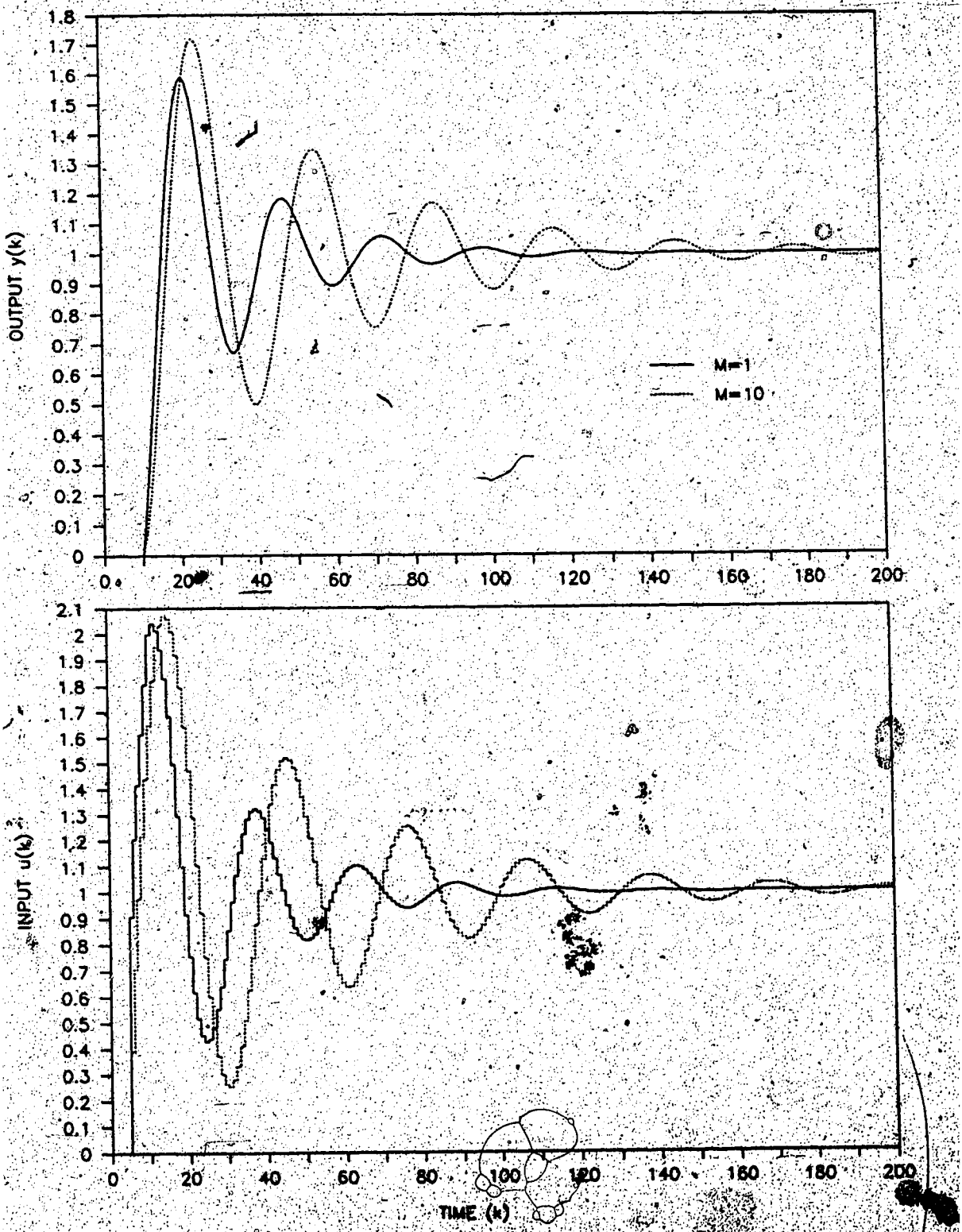

Figure 7.9: Effect of Control Horizon, M on Unknown Delay in System 1. N=30, P=30, $\Gamma=1.0, \Gamma_1=5.0, \Gamma_2=5.$

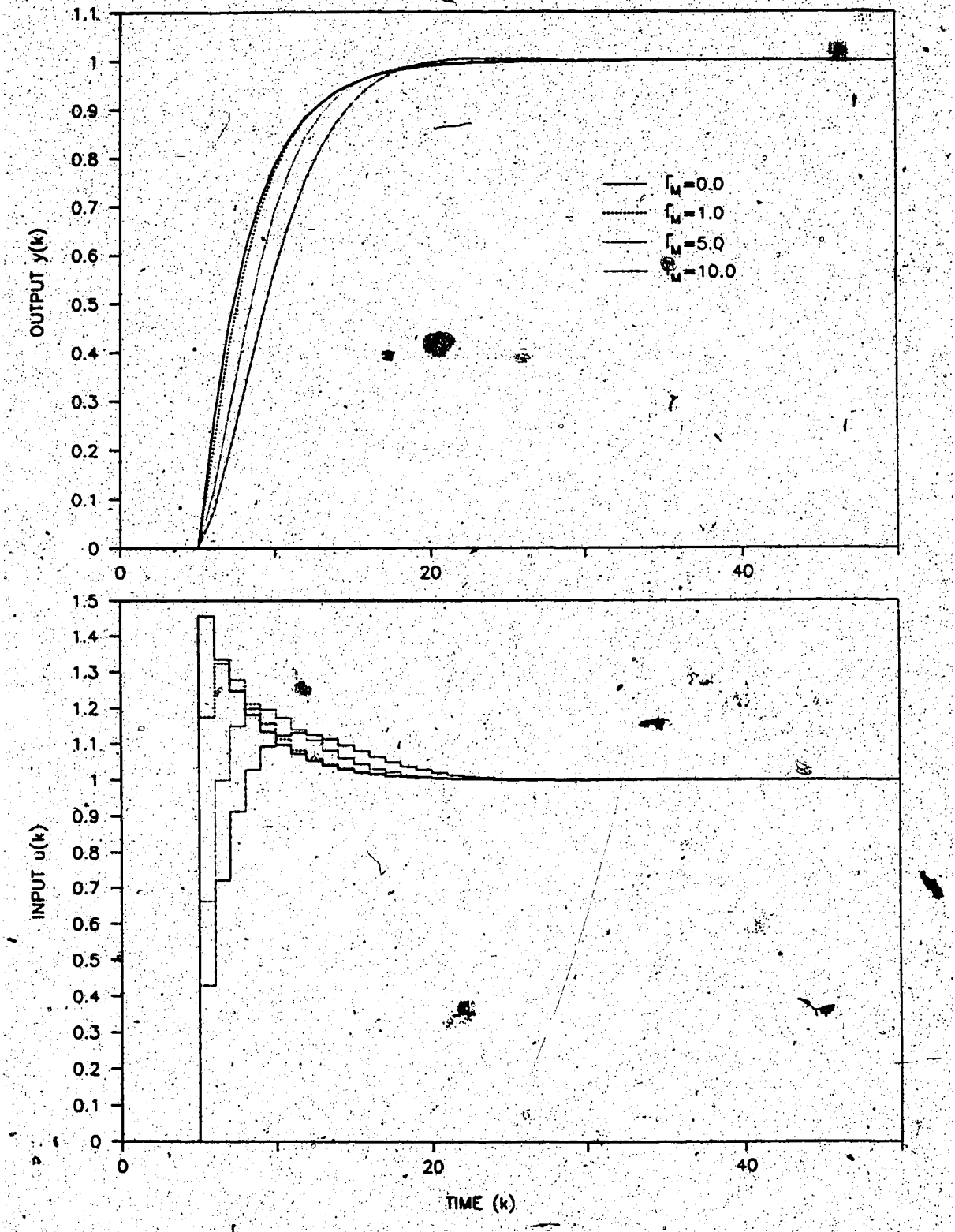

Figure 7.10: Effect of Input Weighting on System 1. N=30, P=10, M=1, Γ=1.0.

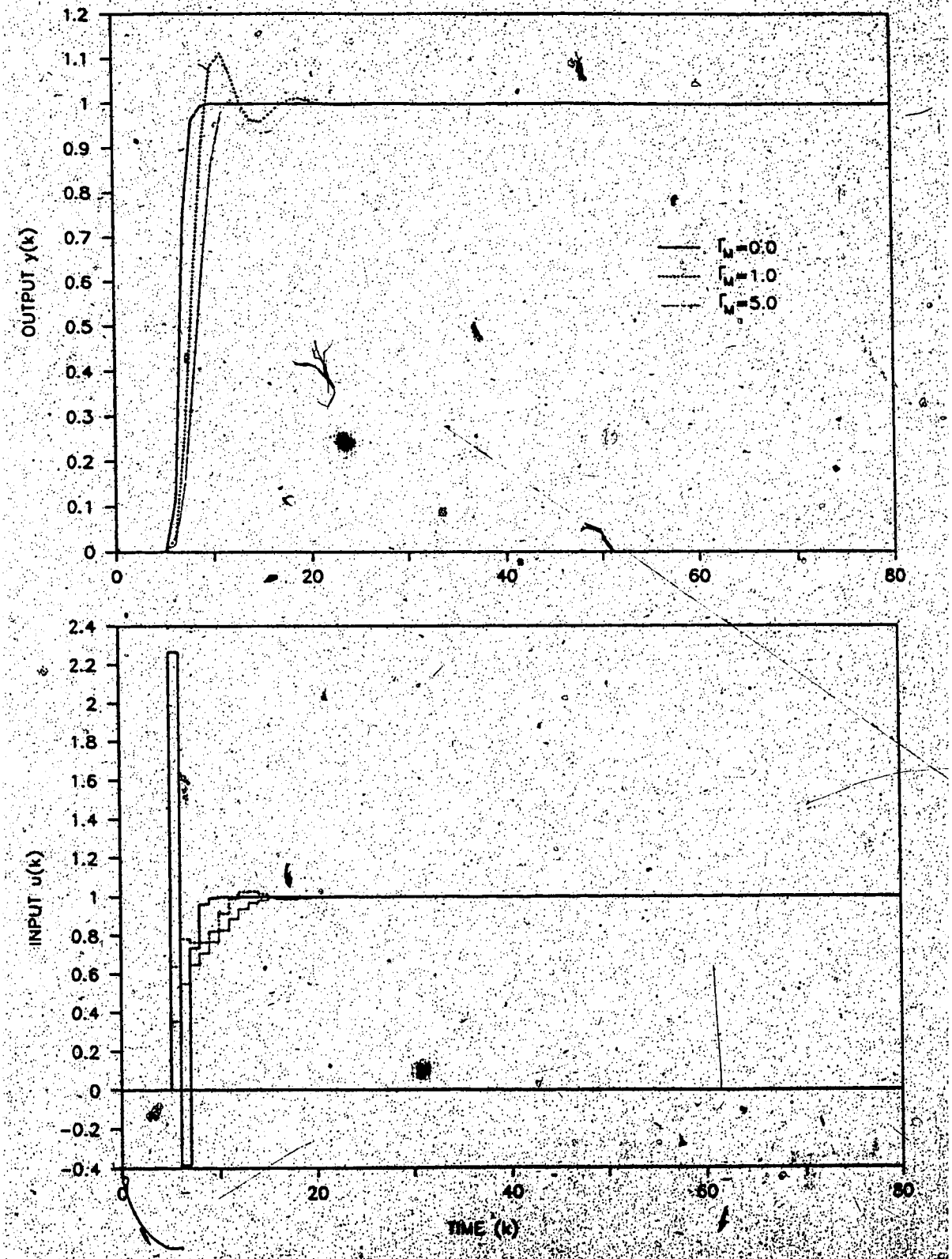

Figure 7.11: Effect of Input Weighting on System 2. N=40, P=5, M=2, Γ=1.0.

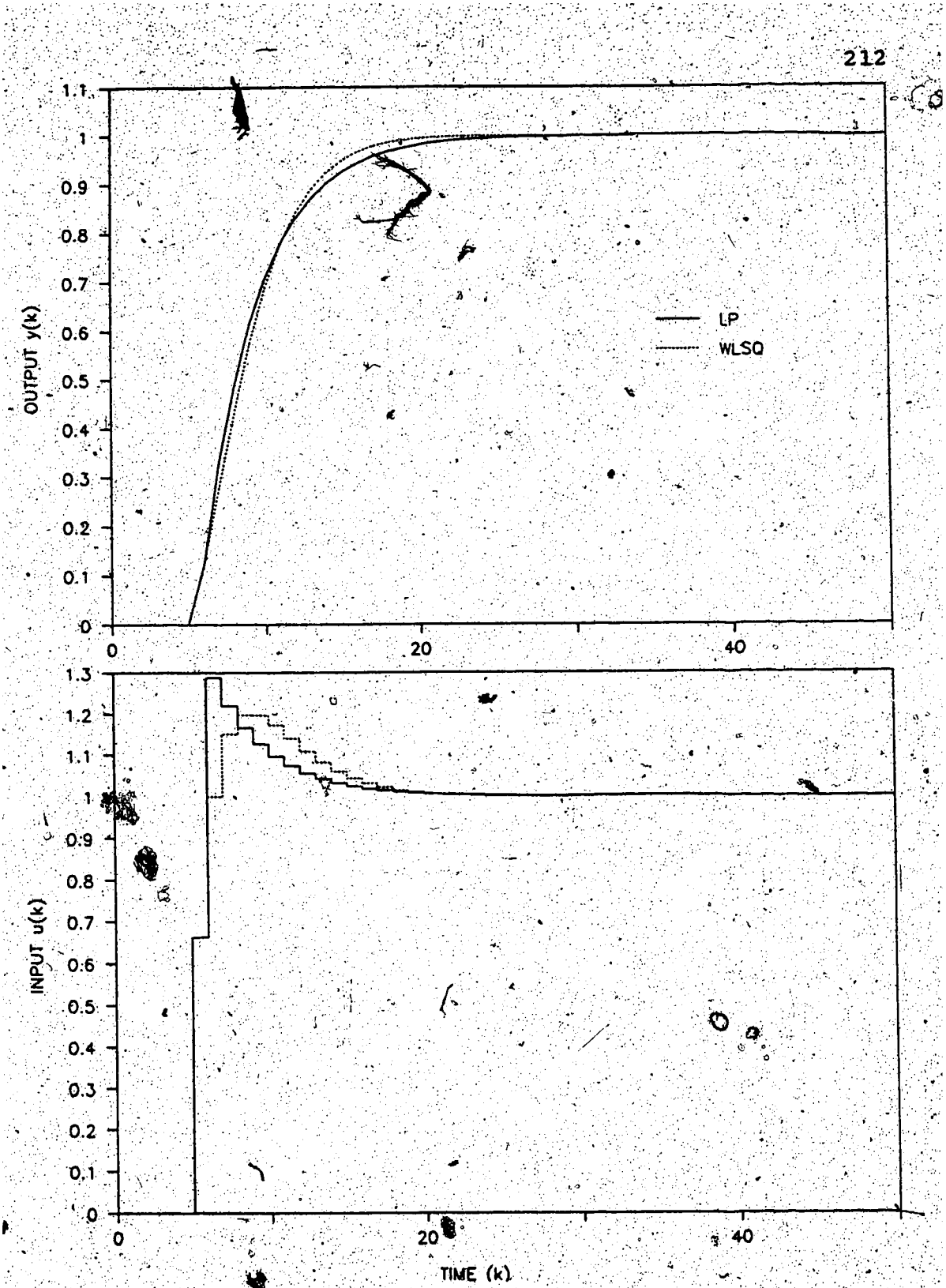

Figure 7.12: Linear Programming with Constraints on Input Changes for System 1. N=40, P=10, M=1; $\Gamma = 1.0; \Gamma_u = 5.0$ for WLSQ and $-0.6635 \leq \Delta U_i(k) \leq 0.6635$ for LP.

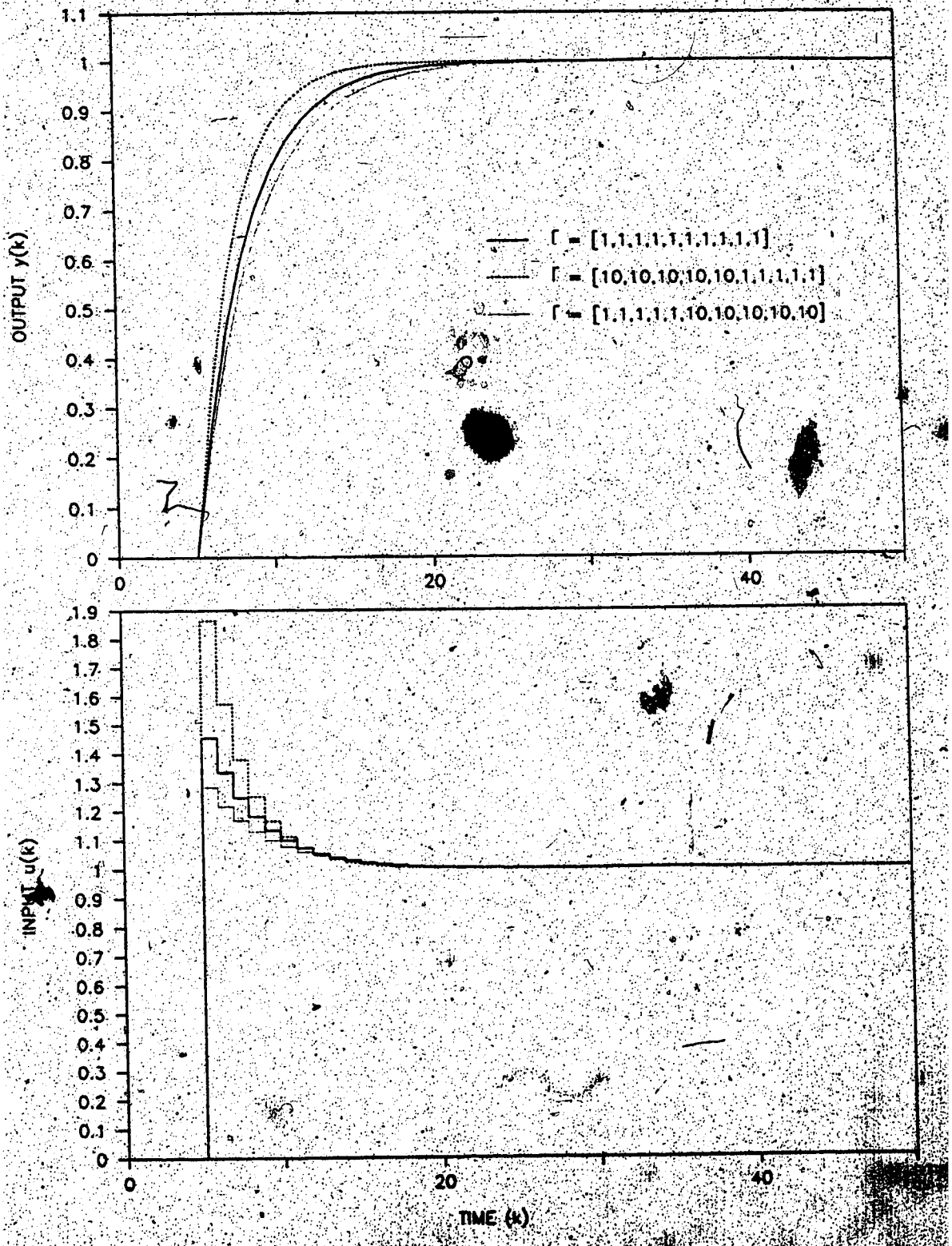

Figure 7.13: Effect of Output Weighting on System 1. N=30, P=10, M=1, Γ_u=0.0.

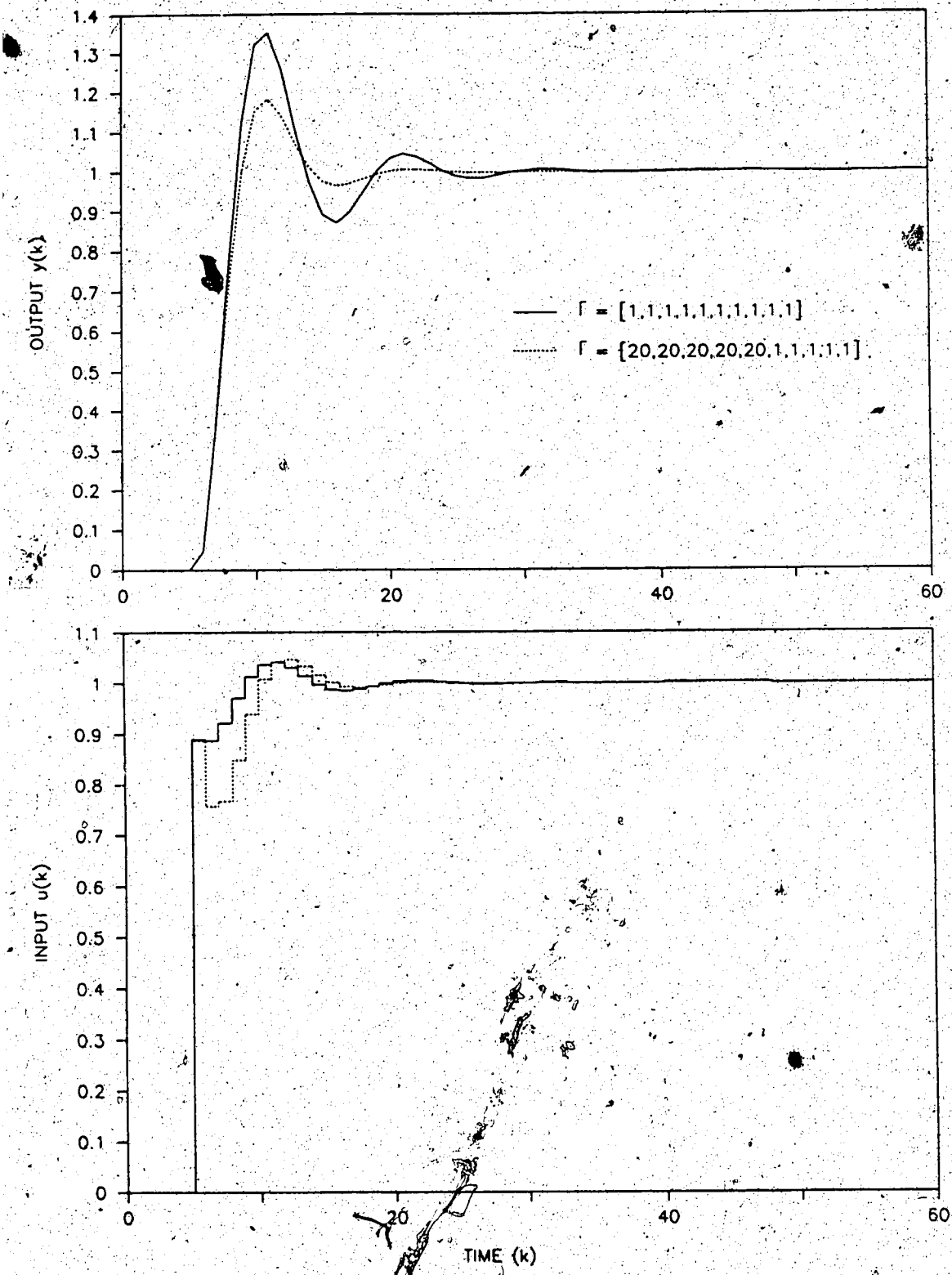

Figure 7.14: Effect of Output Weighting on System 2. N=40, P=10, M=1 r_u = 0.0.

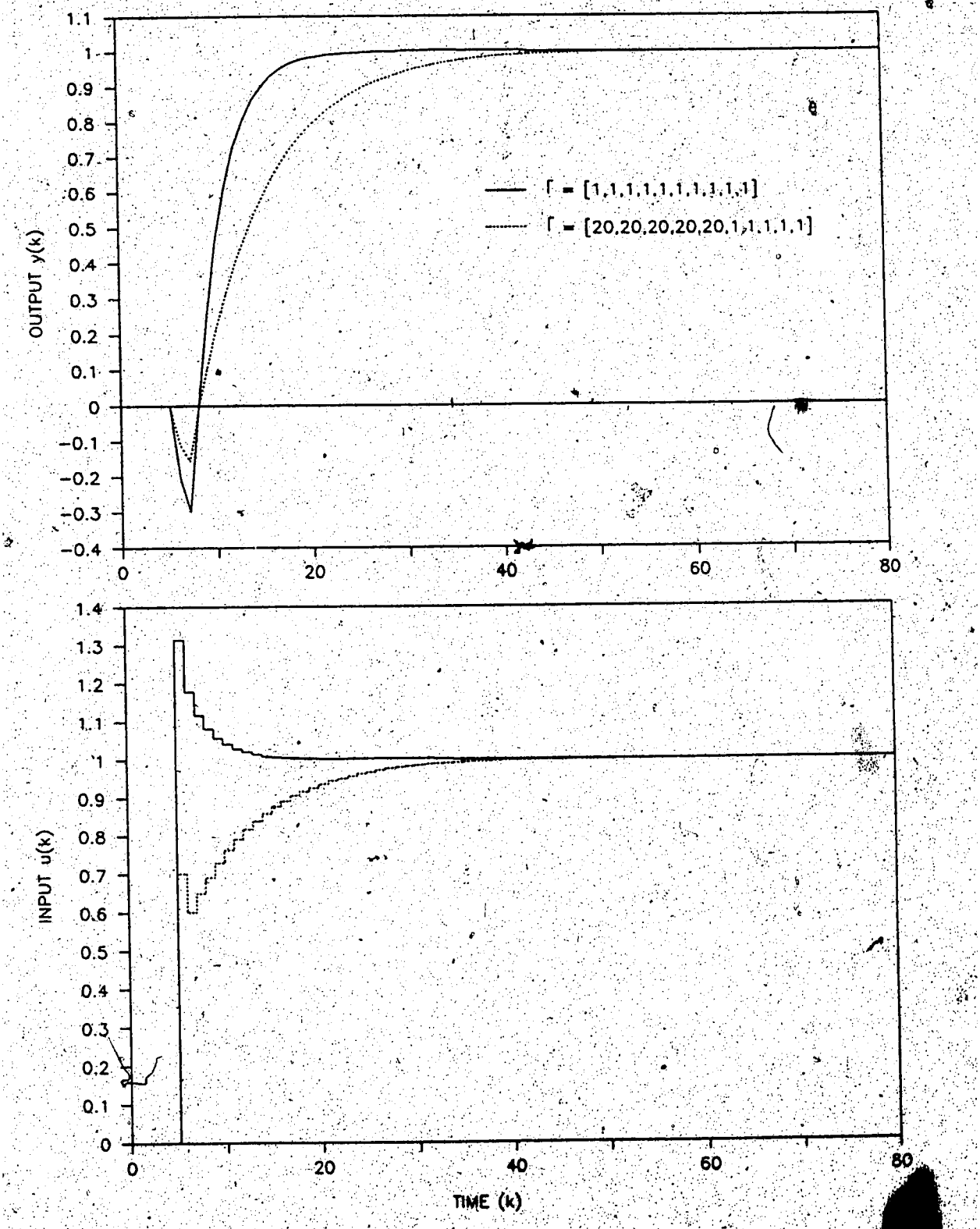

Figure 7.15: Effect of Output Weighting on System 3. N=25, P=10, M=1, Γ_u=0.0.

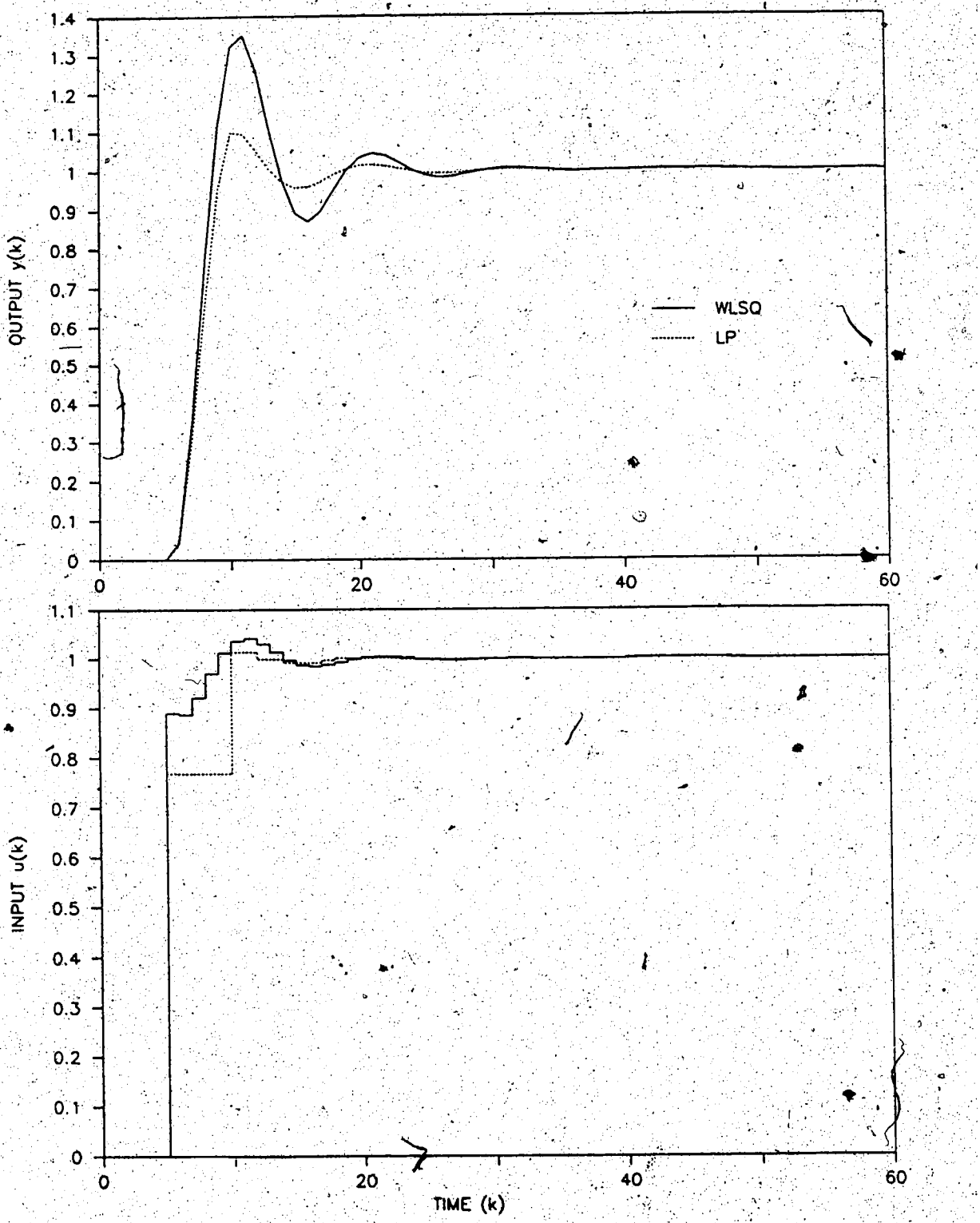

Figure 7.16: Linear Programming with Output Constraints on System 2. N=40, P=10, M=1, $\Gamma = 1.0, \Gamma_u = 0.0$ for WLSQ and Output is Constrained to a Deviation of 0.1 for LP.

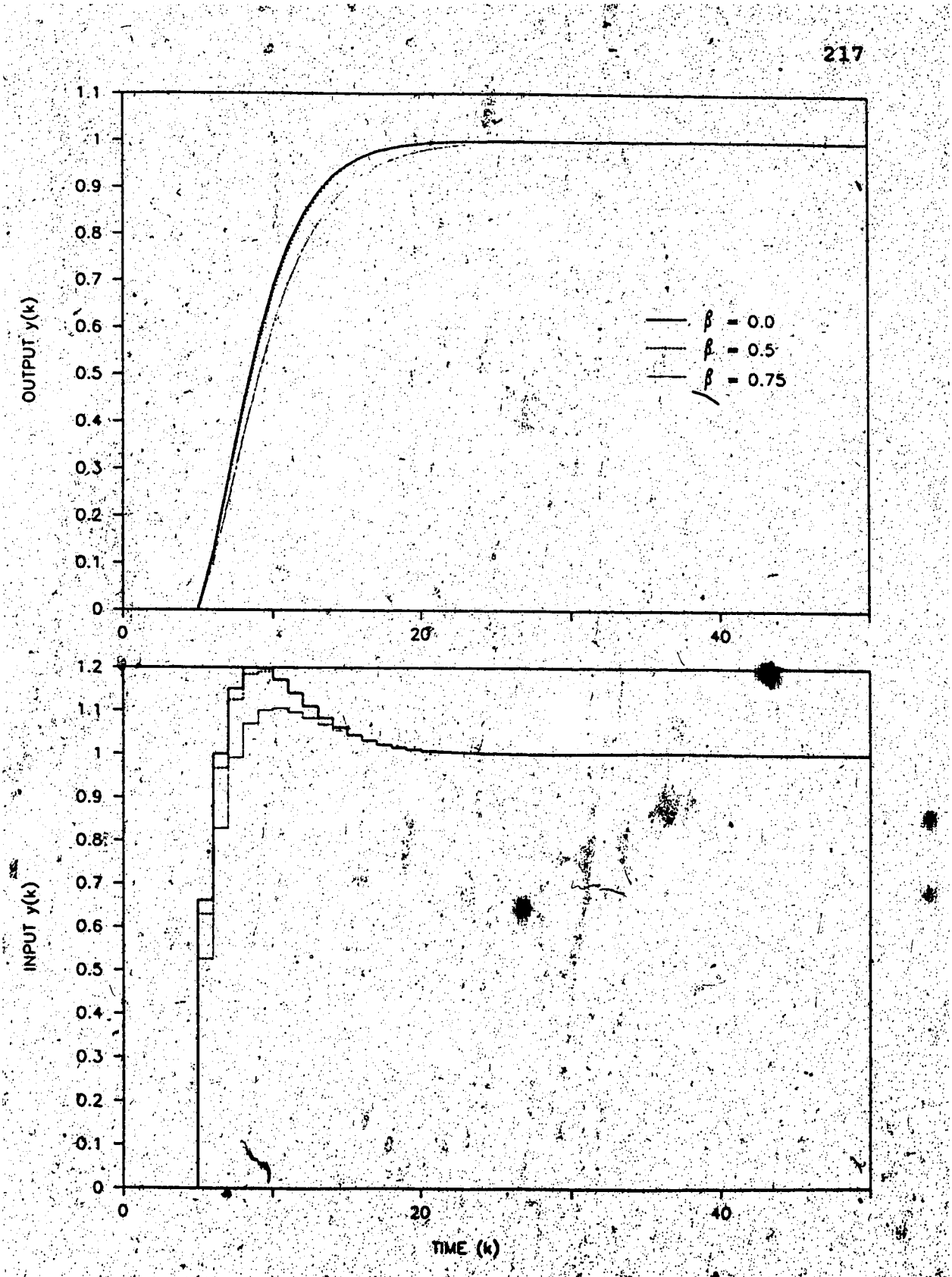

Figure 7.17: Effect of Setpoint Filtering on System 1.
N=30, P=10, M=1, γ = 1.0, γ_u = 5.0.

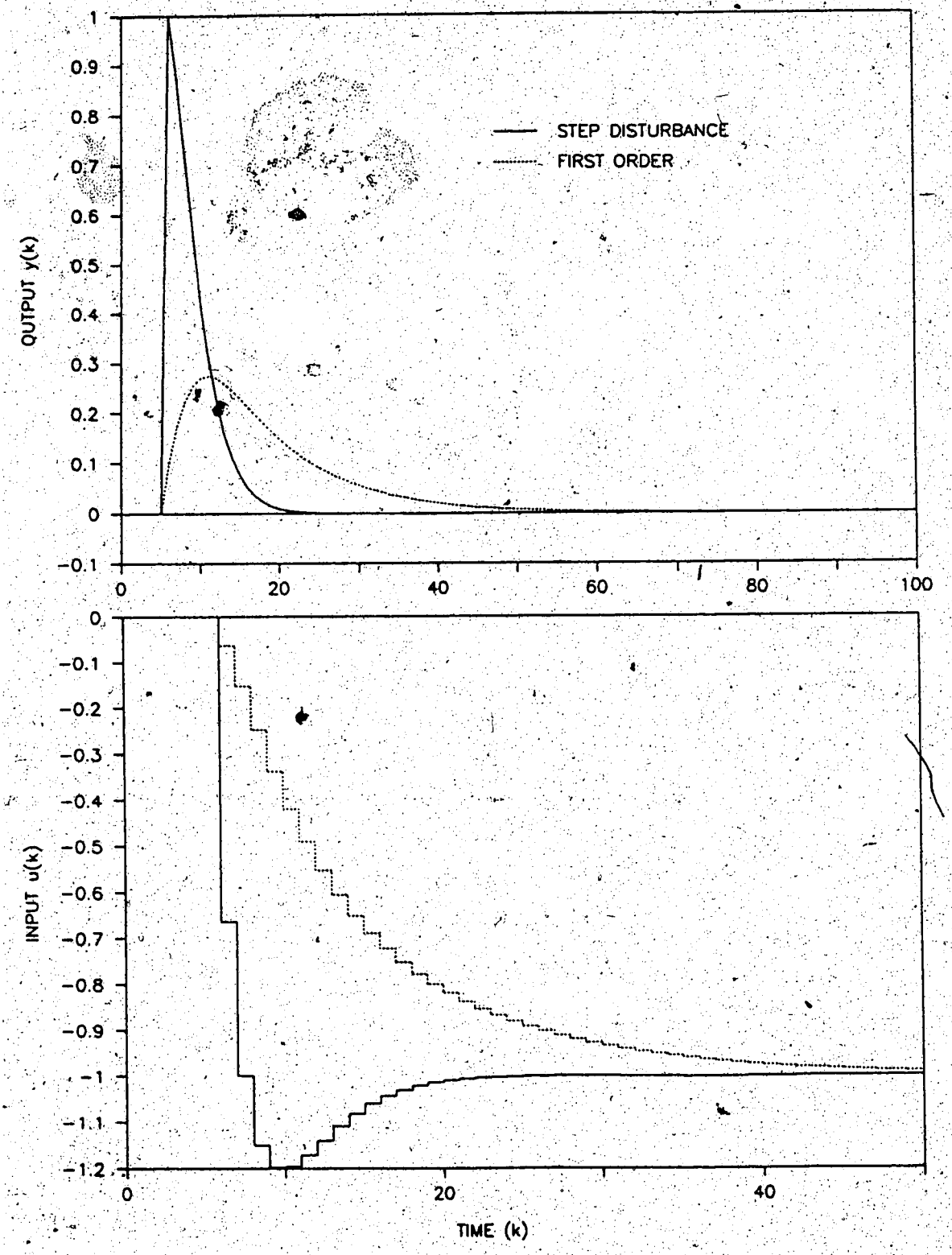

Figure 7.18: Disturbance Rejection Without Feedfoward for System 1. N=30, P=10, M=1, $\Gamma = 1.0, \Gamma_u = 5.0$.

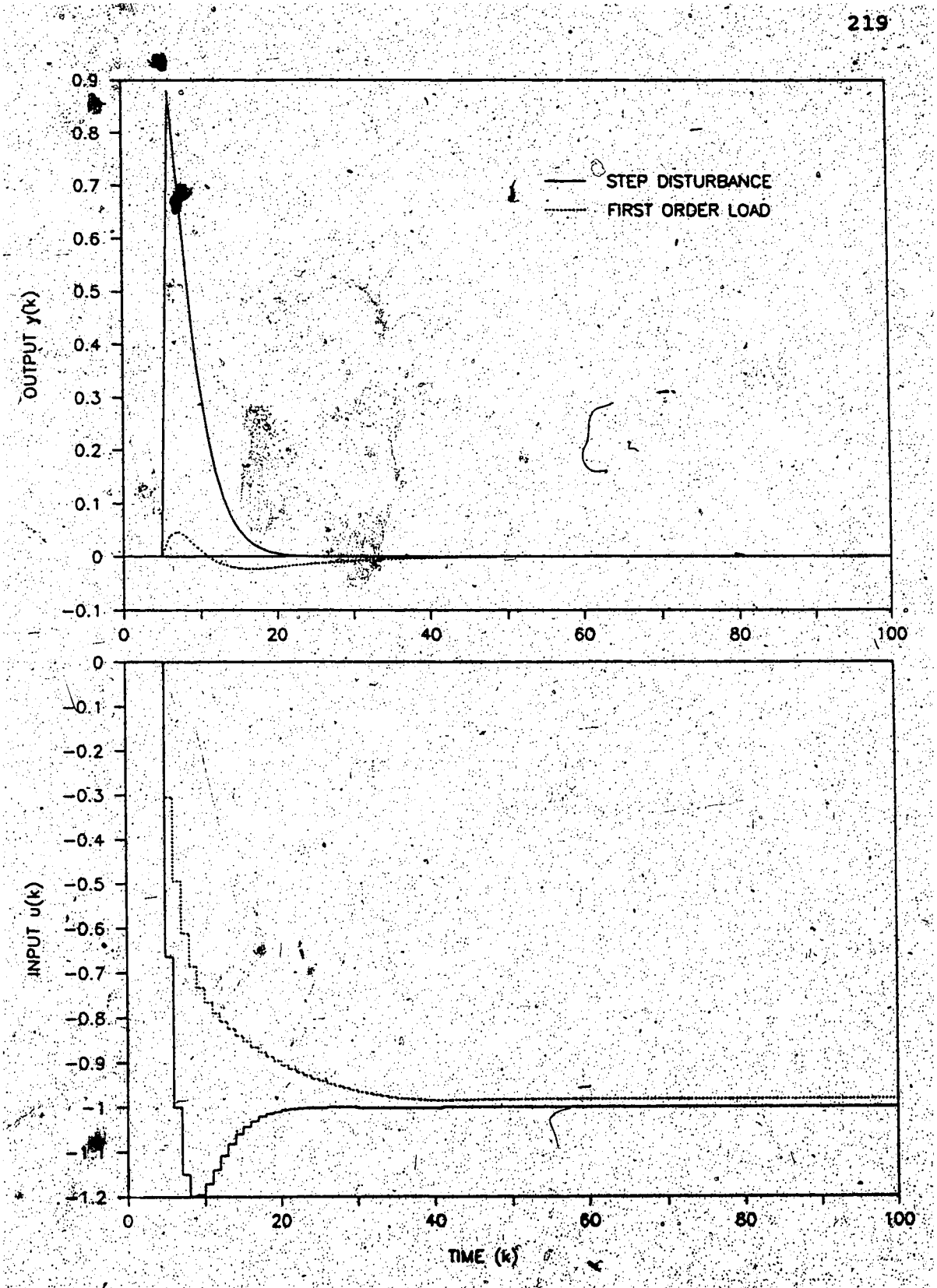

Figure 7.19: Disturbance Rejection With Feedfoward for System 1. N=30, P=10, M=1, Γ = 1.0, Γ_f = 5.0.

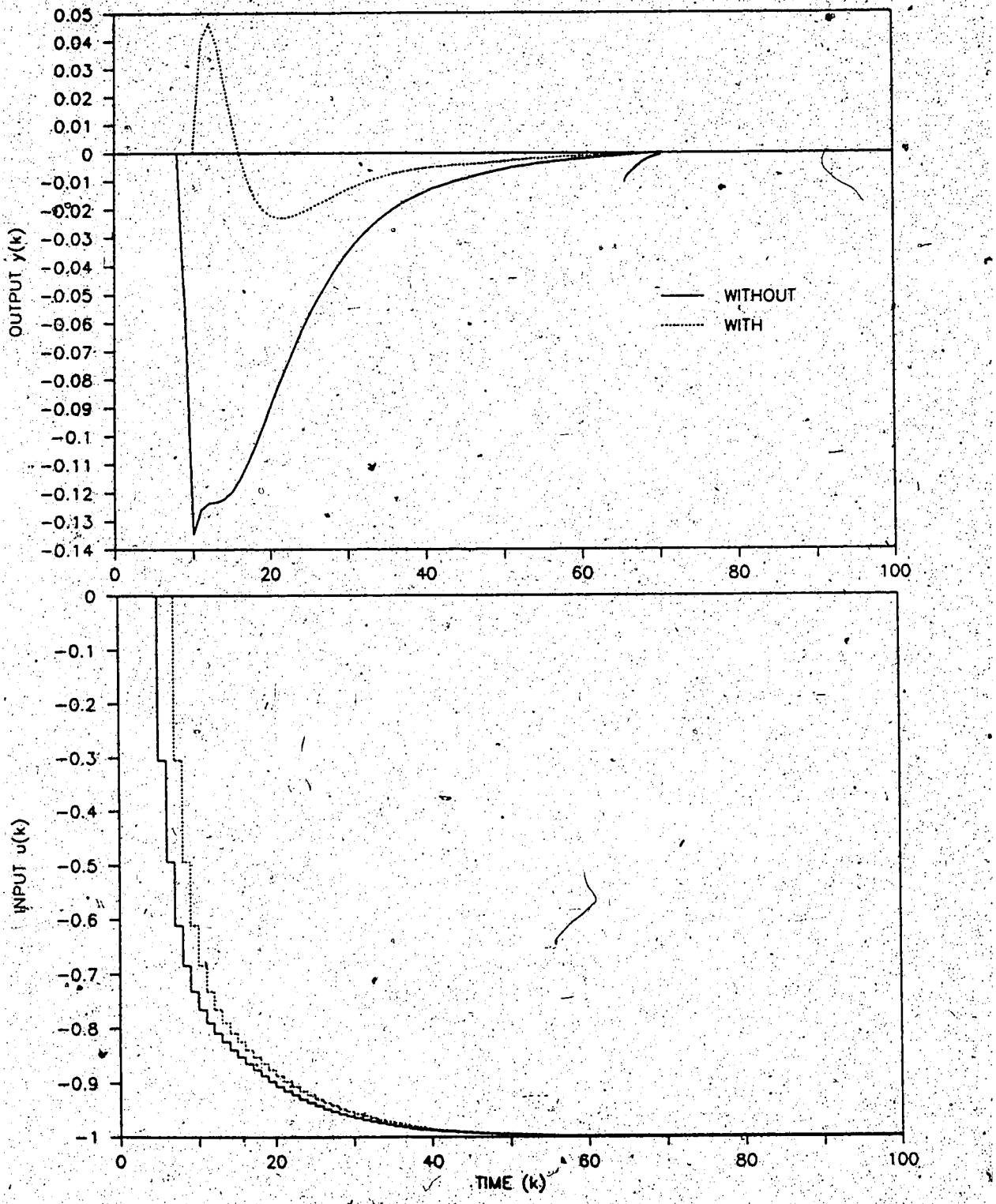

Figure 7.20: Effect of Additional Time Delay Compensation When Delay of Load Disturbance (5*Ts) is Greater Than Process (3*Ts). N=30, P=10, M=1, $\Gamma = 1.0, \Gamma_M = 5.0$.

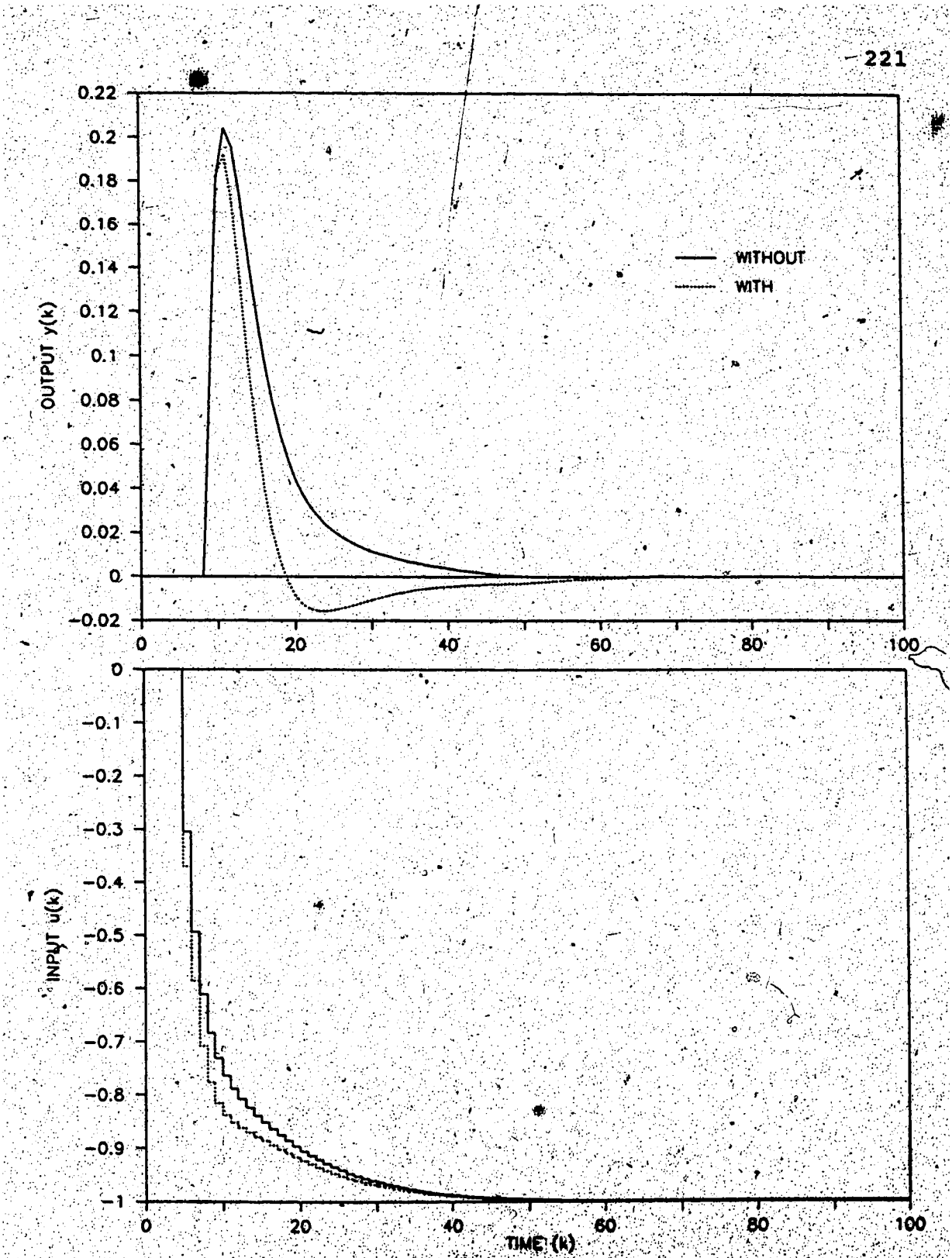

Figure 7.21: Effect of Additional Time Delay Compensation When Delay of Load Disturbance (3*Ts) is Less Than Process (5*Ts). N=30, P=10, M=1, $\Gamma = 1.0, \Gamma_u = 5.0$.

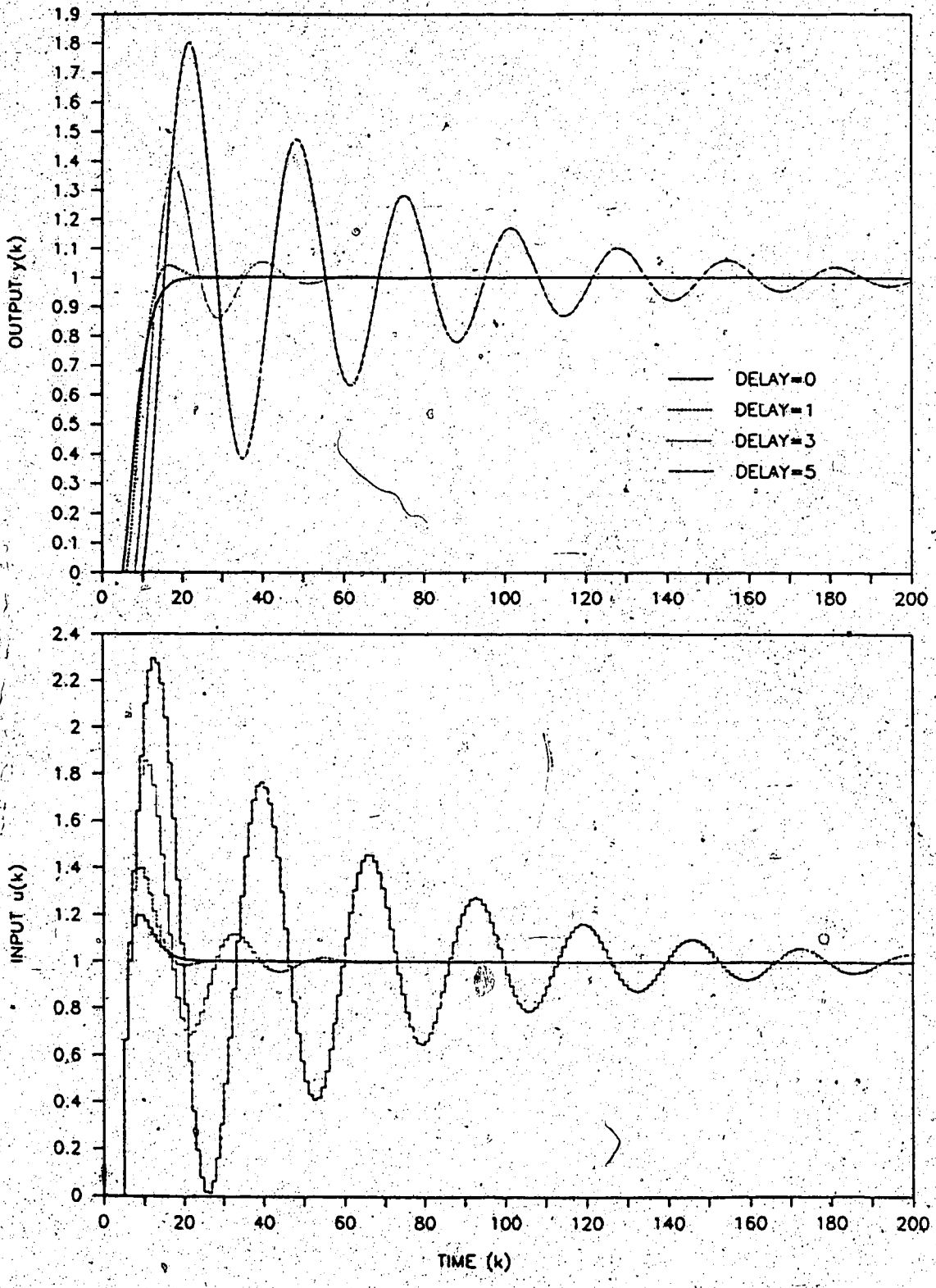

Figure 7.22: Effect of Unknown Time Delay on MPC in System 1. N=30, P=10, M=1, $\Gamma = 1.0, \Gamma_u = 5.0$.

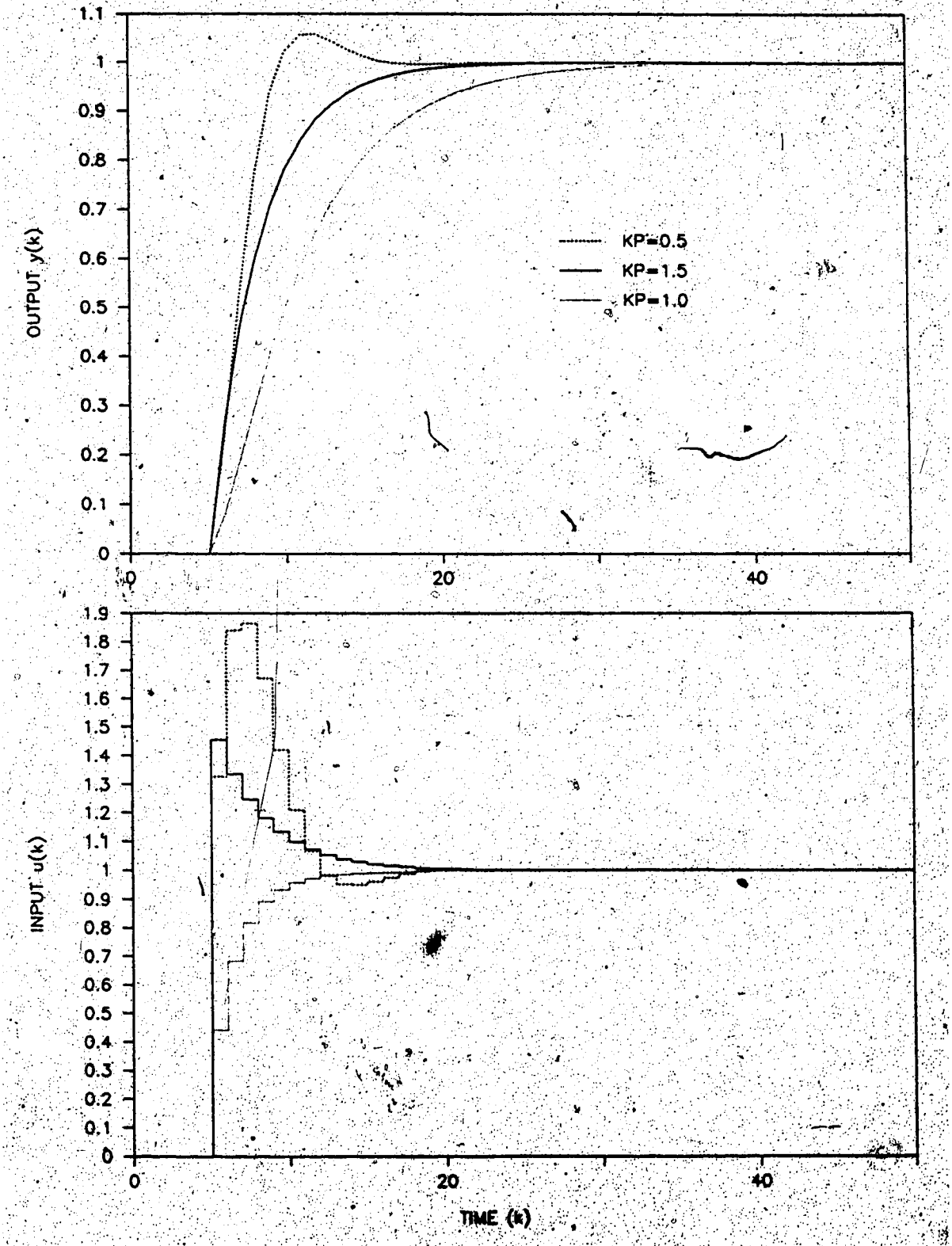

Figure 7.23: Effect of Unknown Process Gain on MPC in System 1. N=30, P=10, M=1, γ=1.0, γ_u=5.0.

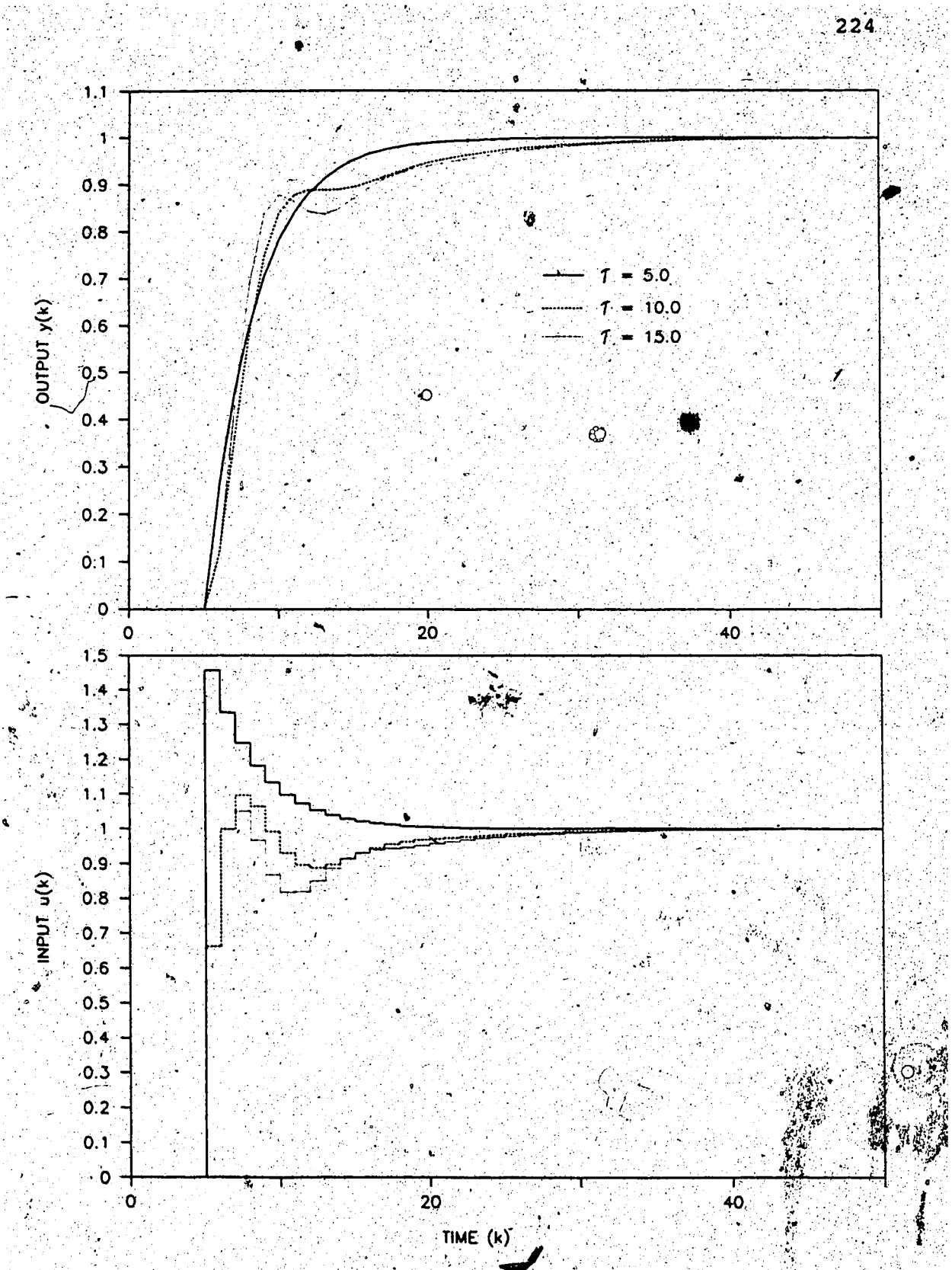

Figure 7.24: Effect of Unknown Process Time Constant on MPC in System 1. N=30, P=10, M=1, $\Gamma = 1.0, \Gamma_u = 5.0$.

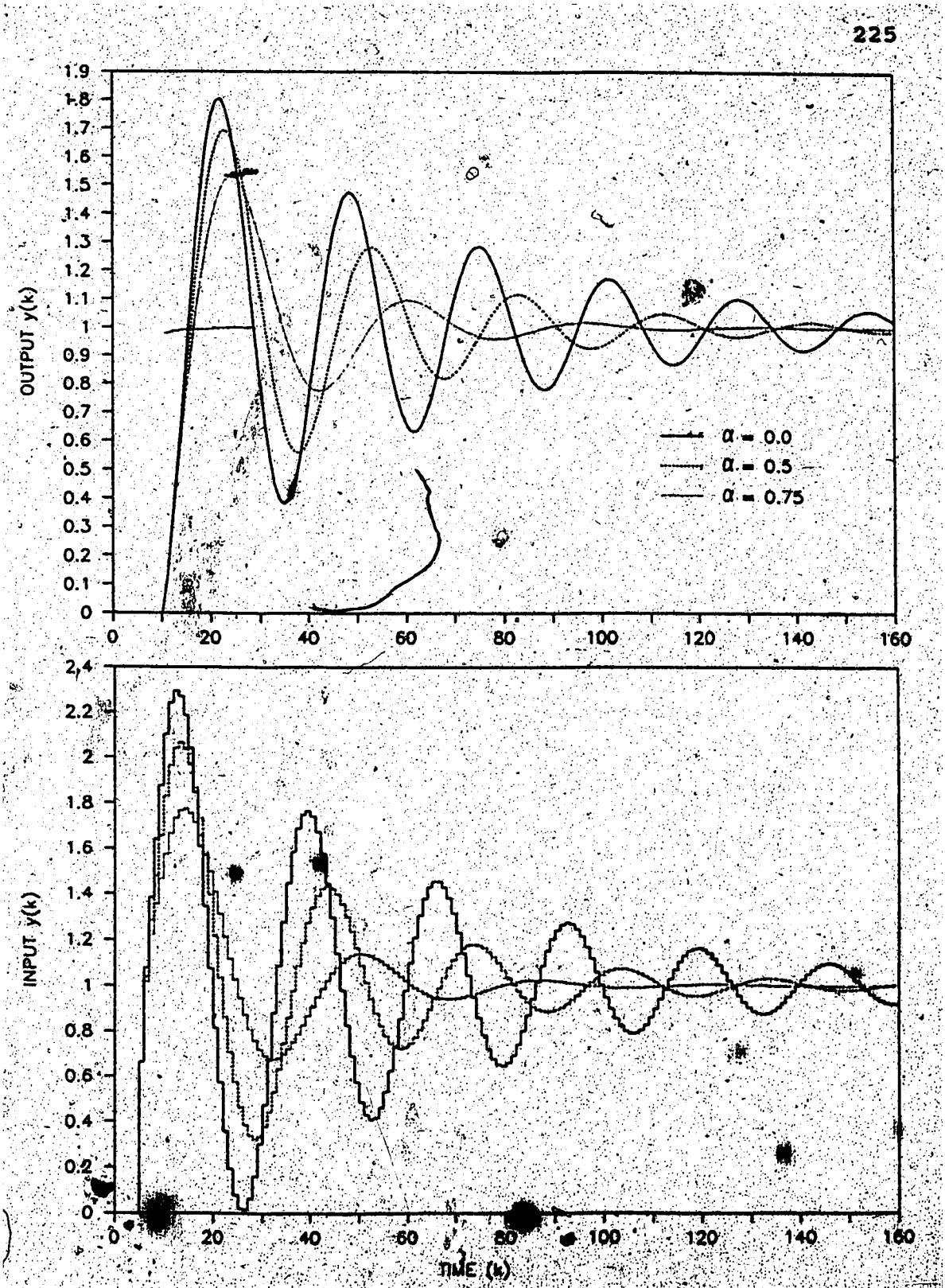

Figure 7.25: Effect of Feedback Filtering on System 1 with Unknown Time Delay of 5. N=30, P=10, M=1, $\Gamma = 1.0, \Gamma_\alpha = 5.0$.

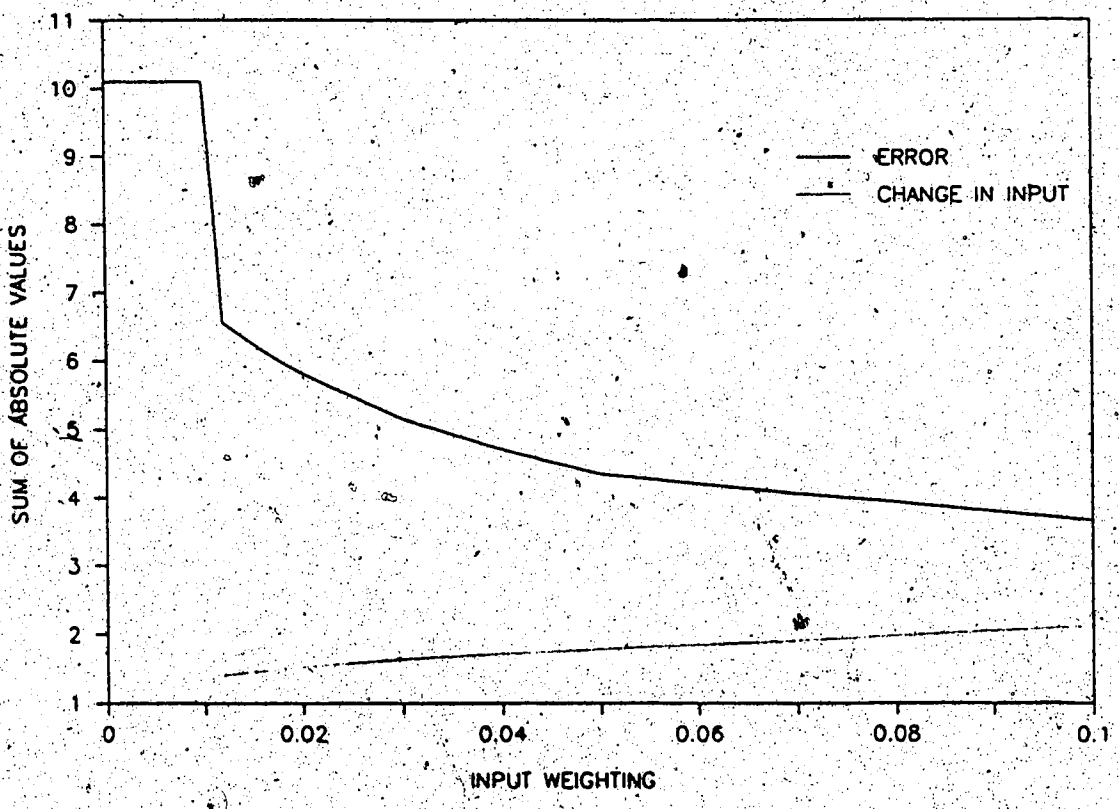

Figure 7.26: Plot of Sum of Absolute Error and Change in Input Against Input Weighting for System 1. N=30, P=10, M=1, Γ = 1.0.

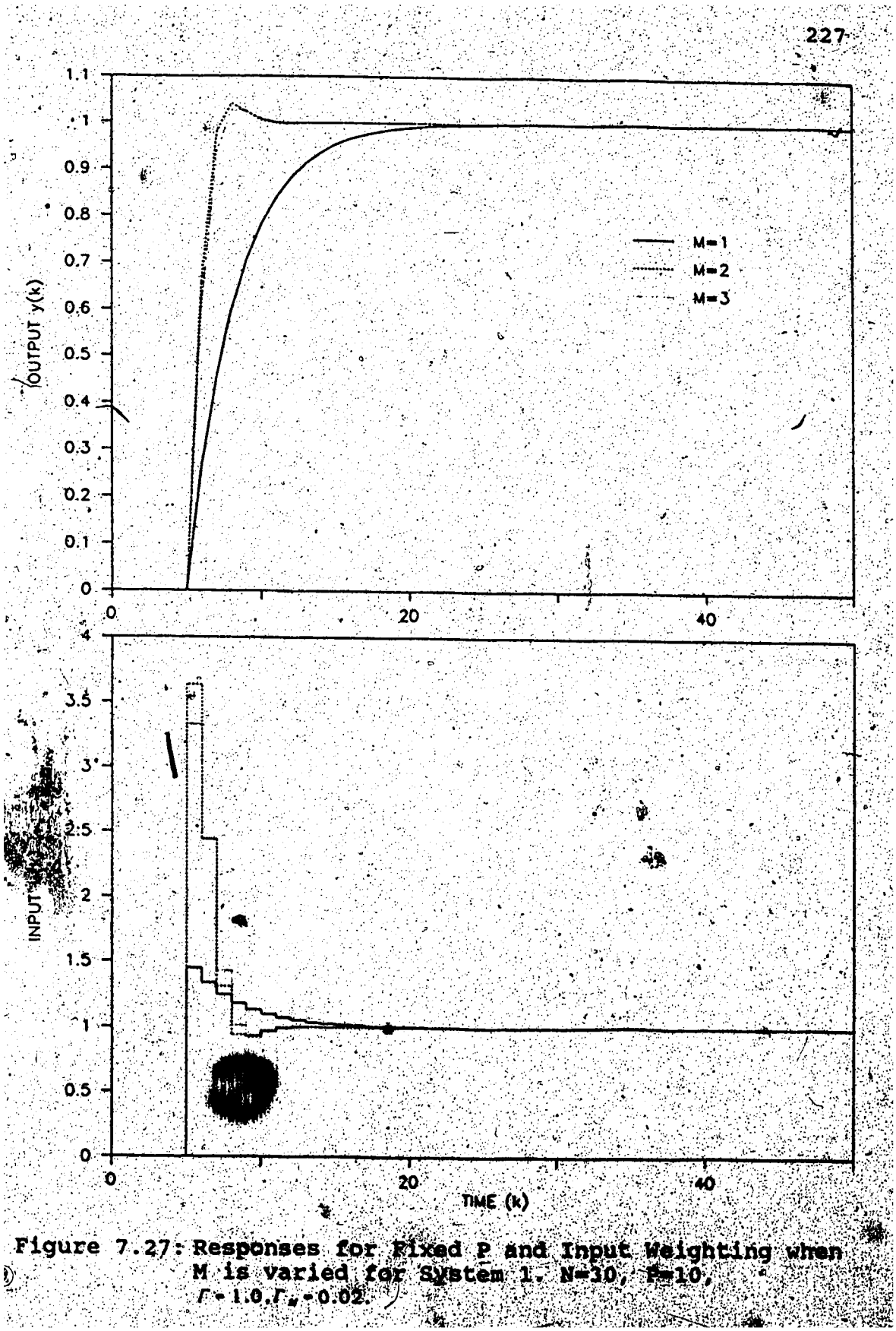

Figure 7.27: Responses for Fixed P and Input Weighting when M is varied for System 1. N=30, P=10, $\Gamma = 1.0, \Gamma_u = 0.02$.

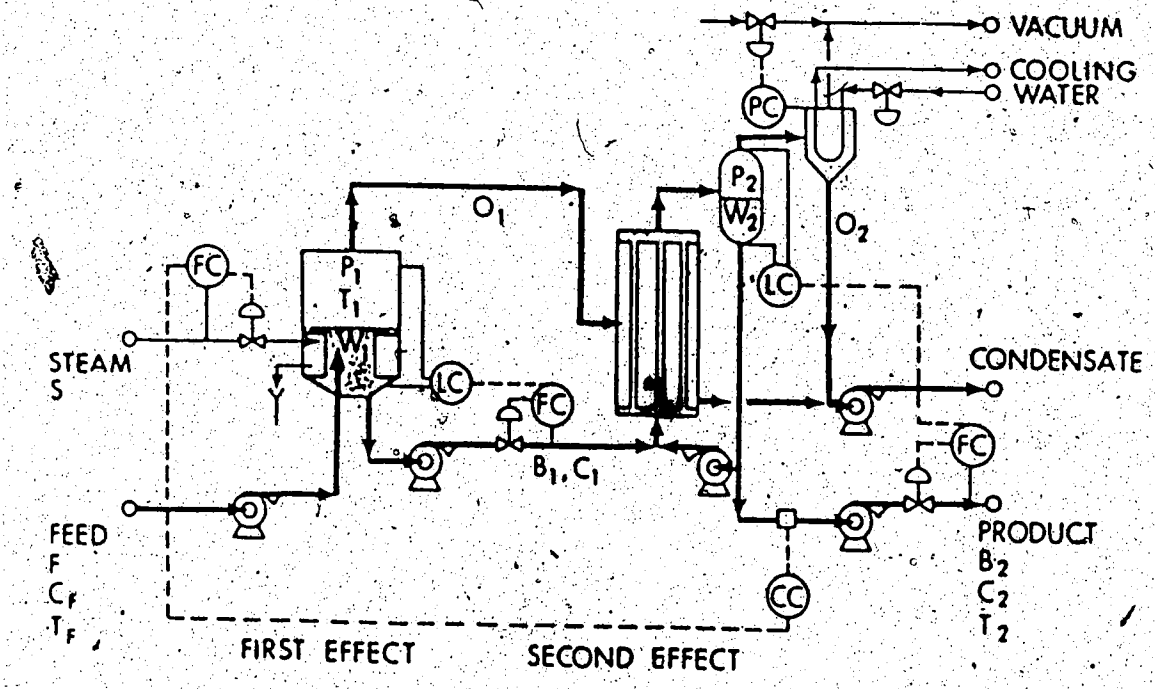

Figure 7.28: Schematic Diagram of Double Effect Evaporator.

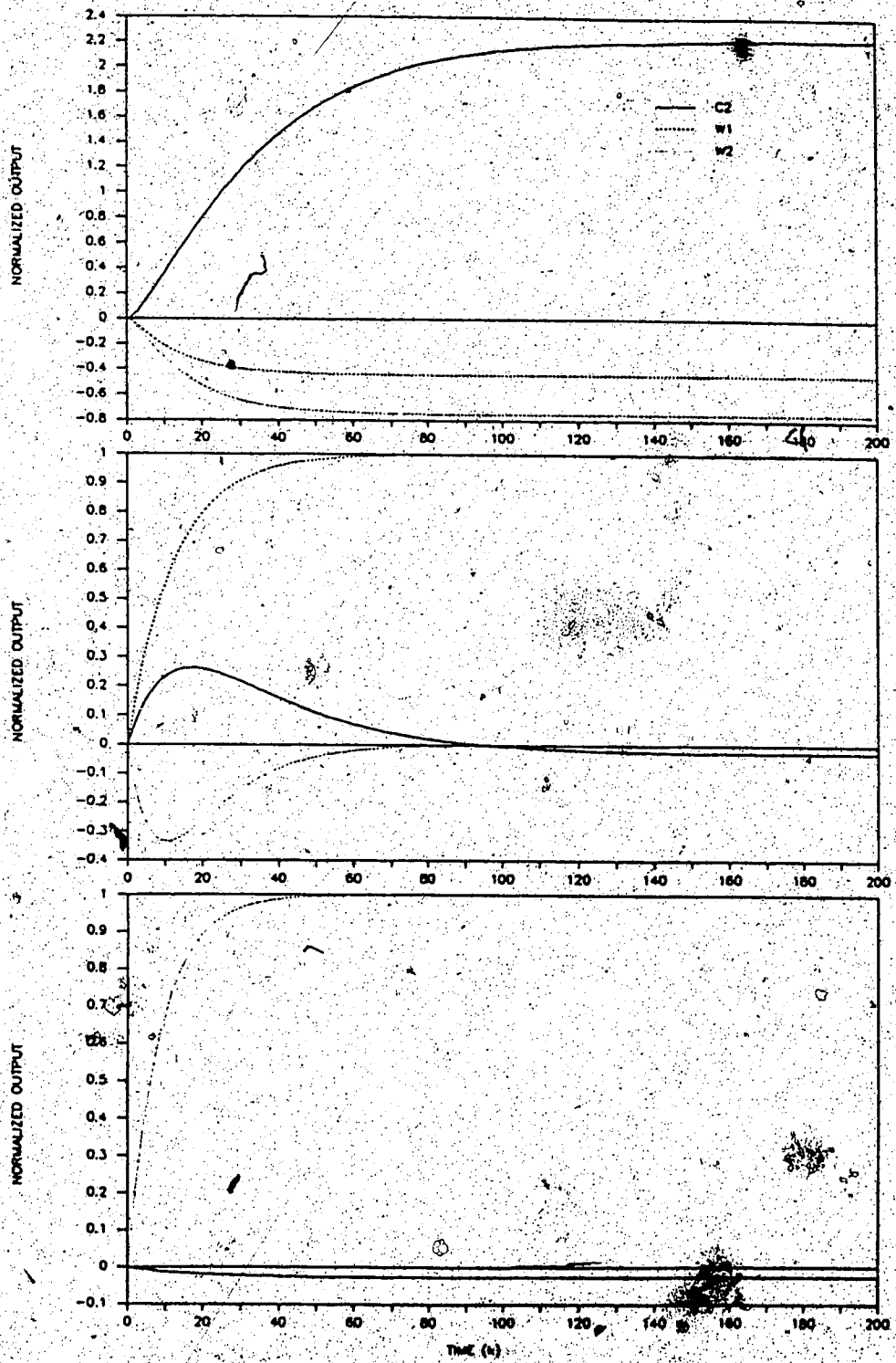

Figure 7.29: Evaporator Step Responses Obtained From 5th Order Non-linear Model. Step in Steam (Top), Step in First Effect Level Setpoint (Middle) and Step in Second Effect Level Setpoint (Bottom).

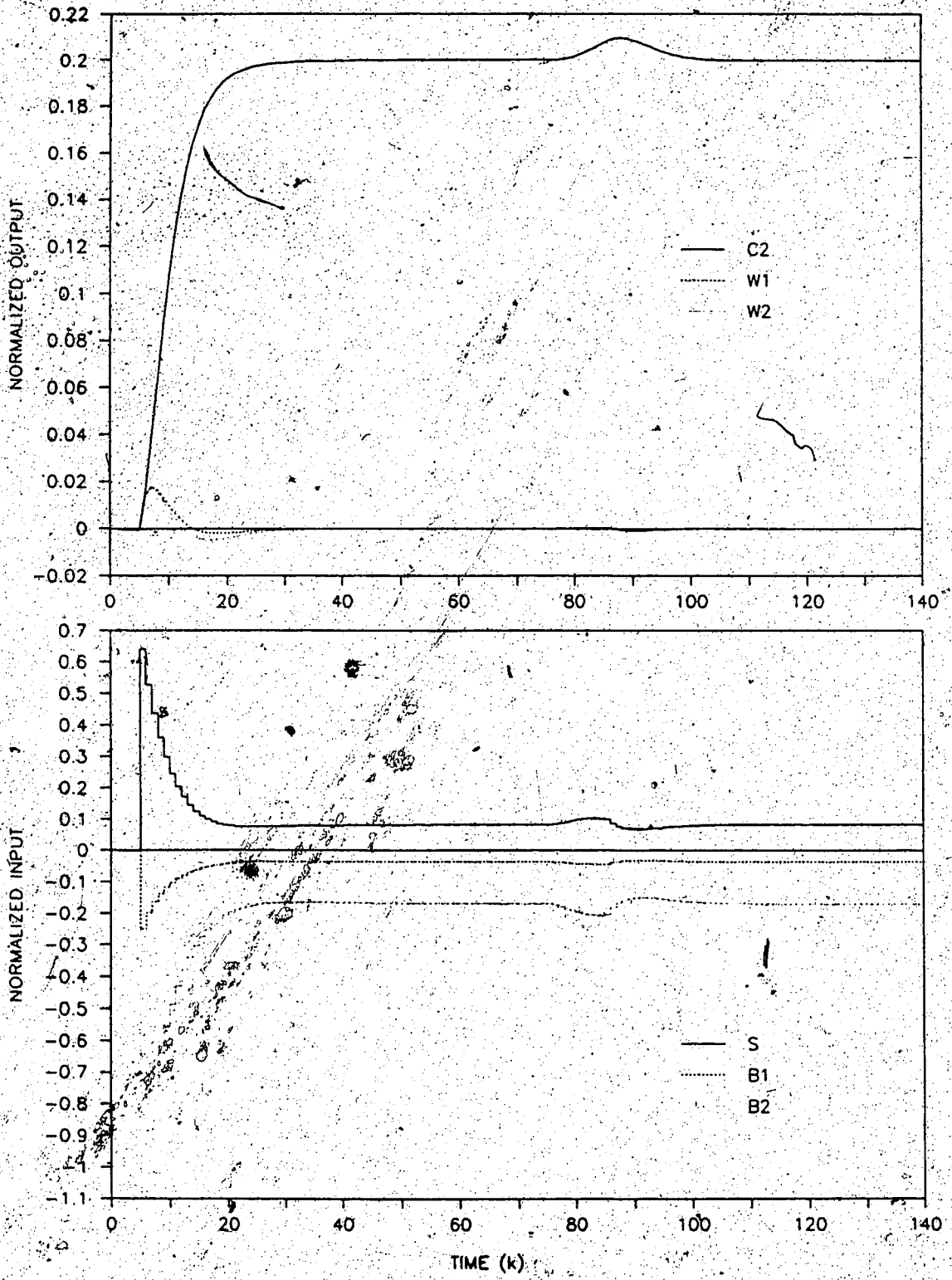

Figure 7.30: Effect of Truncation Error Due to N Using Non-linear Model. N=80, P=10, M=1, Γ=1.0, Γ_m=0.0. 20% Step Change in C2 Setpoint.

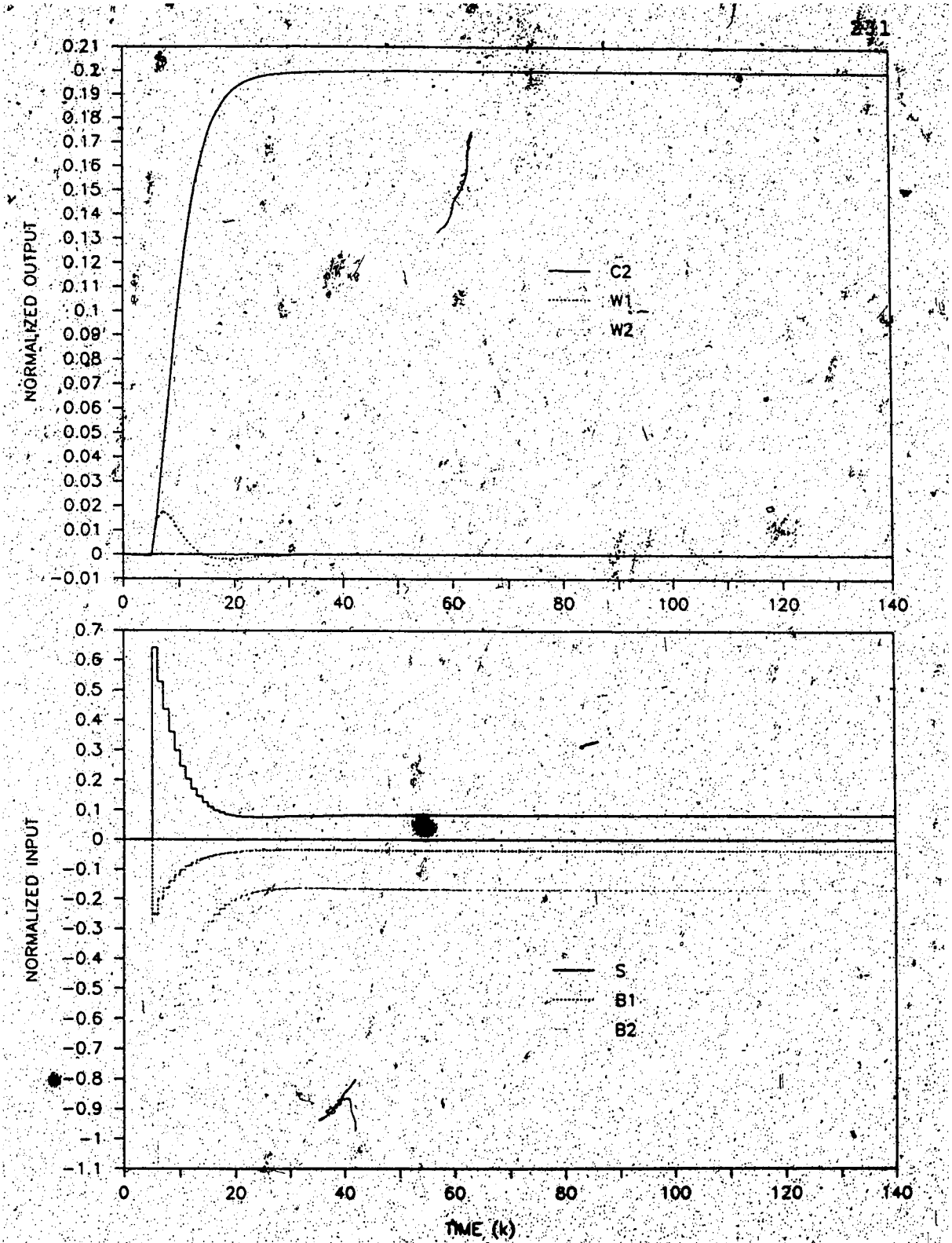

Figure 7.31: Effect of Truncation Error Due to N Using Non-linear Model. All N=100 except $N_{11}=N_{12}=150$, P=10, M=1, $\gamma=1.0, \gamma_u=0.0$. 20% Step Change in C2 Setpoint.

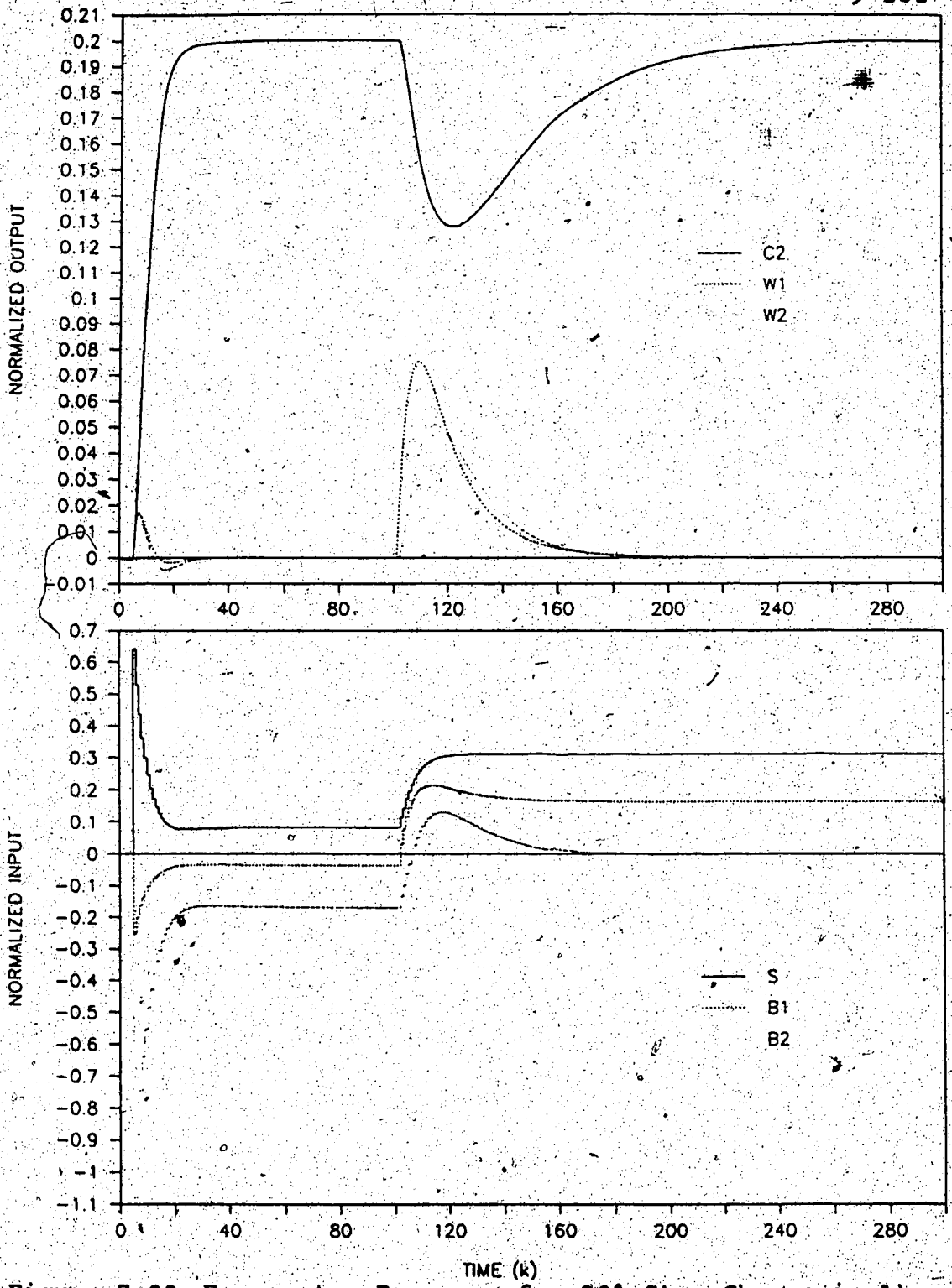

Figure 7.32: Evaporator Response for 20% Step Change in 1) C2 Setpoint and 2) Feed Flowrate Using the Non-linear Model. All N=100 except $N_{11}=N_{12}=150$, P=10, M=1, $\Gamma=(1.0,1.0,1.0)$, $\Gamma_m=(0.0,0.0,0.0)$.

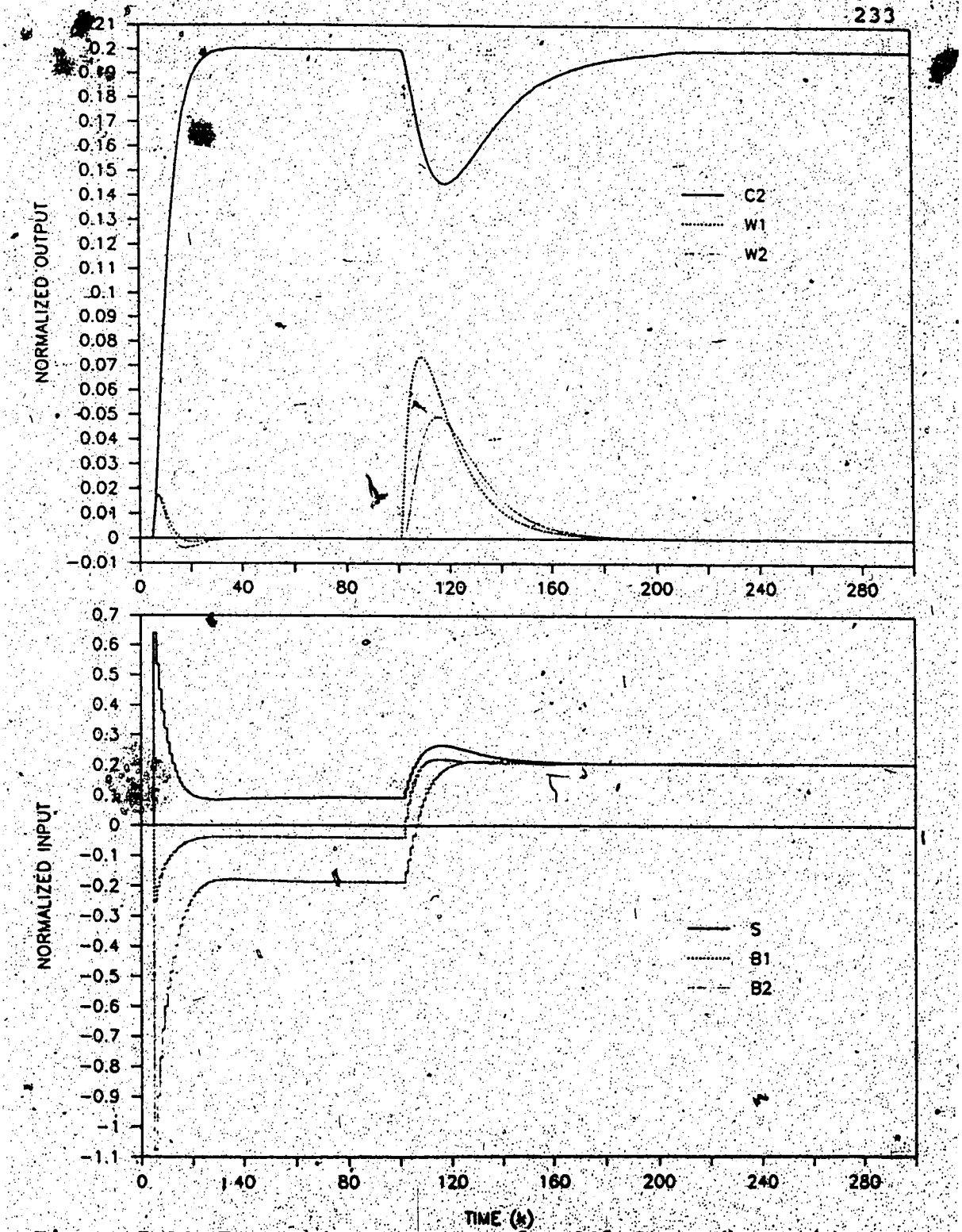

Figure 7.33: Evaporator Response for 20% Step Change in 1) C2 Setpoint and 2) Feed Flowrate Using the Simulator. All N=100 except $N_{11}=N_{12}=150$, P=10, M=1, $\Gamma=1.0, \Gamma_u=0.0$.

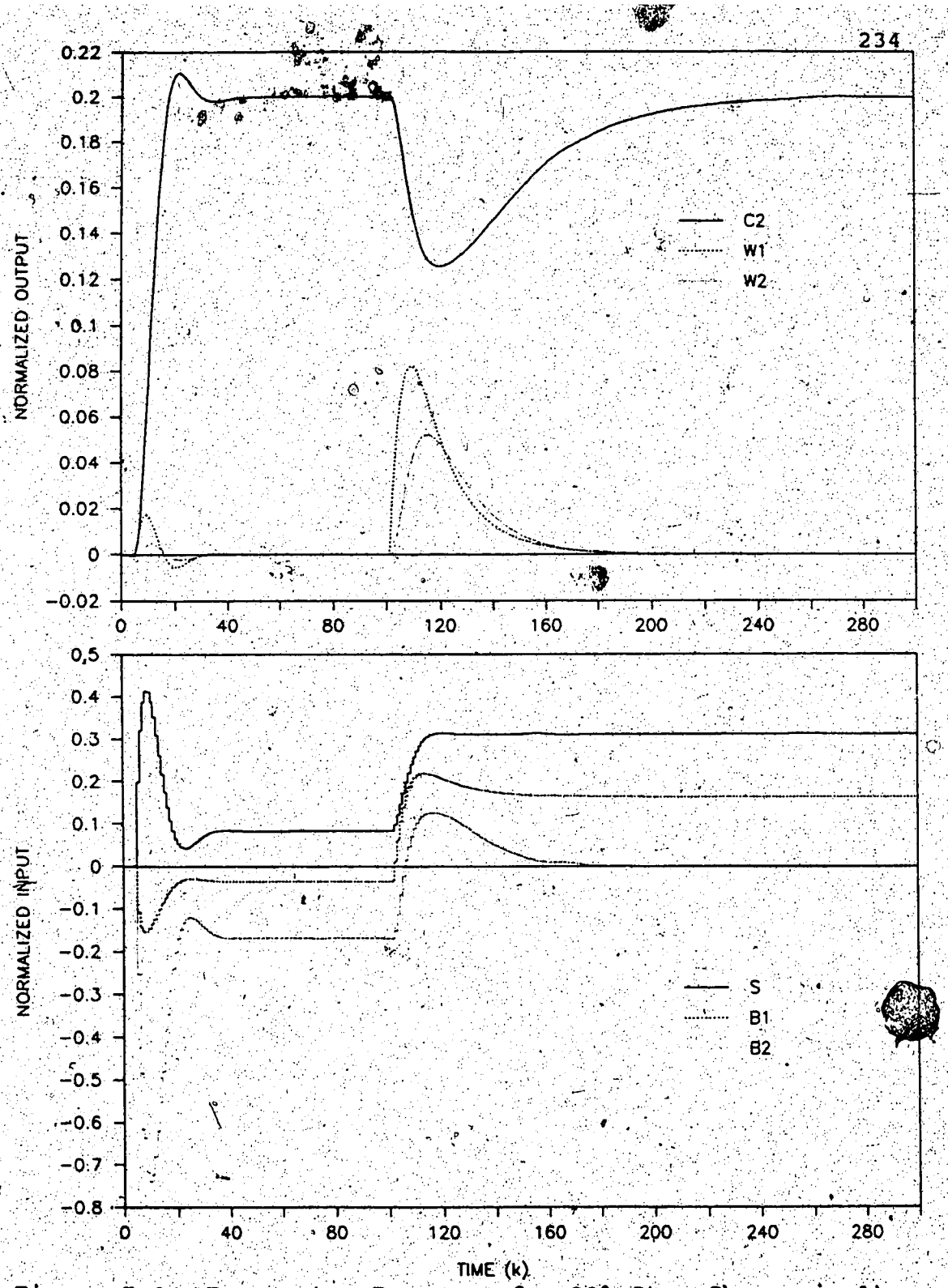

Figure 7.34: Evaporator Response for 20% Step Change in 1) C2 Setpoint and 2) Feed Flowrate Using the Non-linear Model. All N=100 except $N_{11}=N_{12}=150$, P=10, M=1, $\Gamma_1=(1.0,1.0,1.0)$, $\Gamma_2=(1.0,1.0,1.0)$.

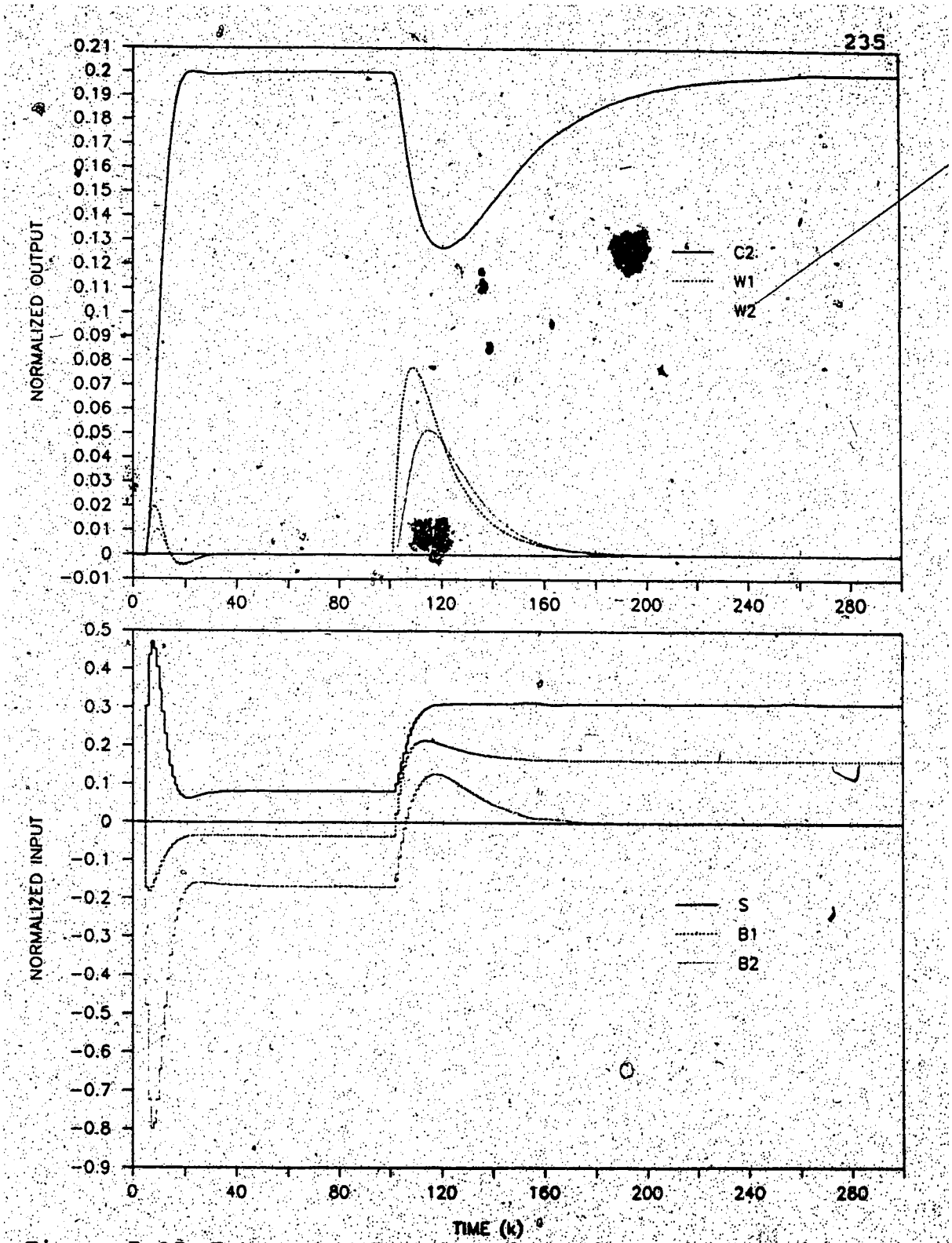

Figure 7.35: Evaporator Response for 20% Step Change in 1) C2 Setpoint and 2) Feed Flowrate Using the Non-linear Model. All N=100 except $N_{11}=N_{12}=150$, P=10, M=1, $\Gamma=(1.0,1.0,1.0)$, $\Gamma_u=(0.5,0.5,0.5)$.

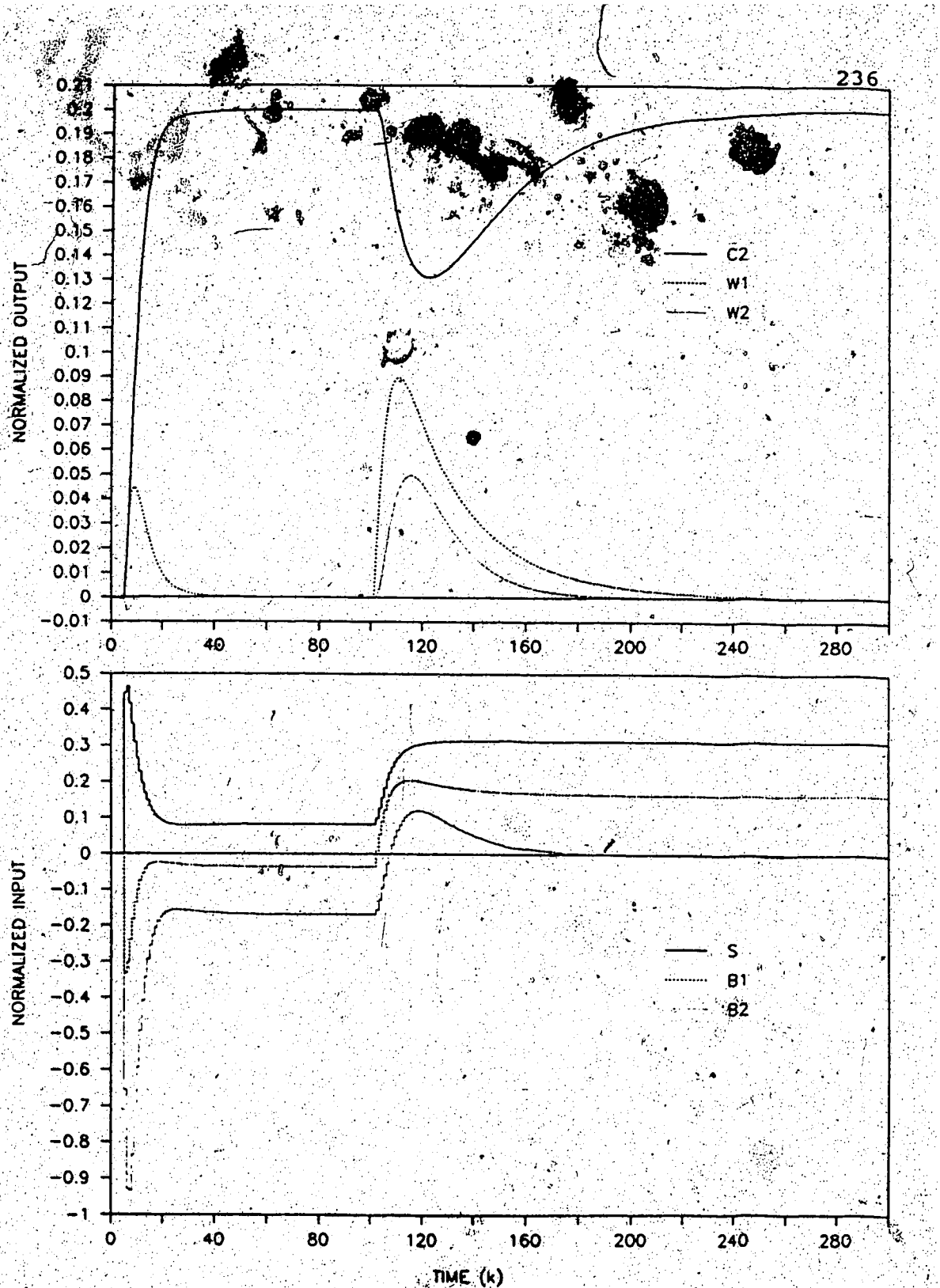

Figure 7.36: Evaporator Response for 20% Step Change in 1) C2 Setpoint and 2 Feed Flowrate Using the Non-linear Model. All N=100 except $N_{11}=N_{12}=150$, P=10, M=1, $\Gamma=(10.0,1.0,1.0)$, $\Gamma_u=(0.5,0.1,0.5)$.

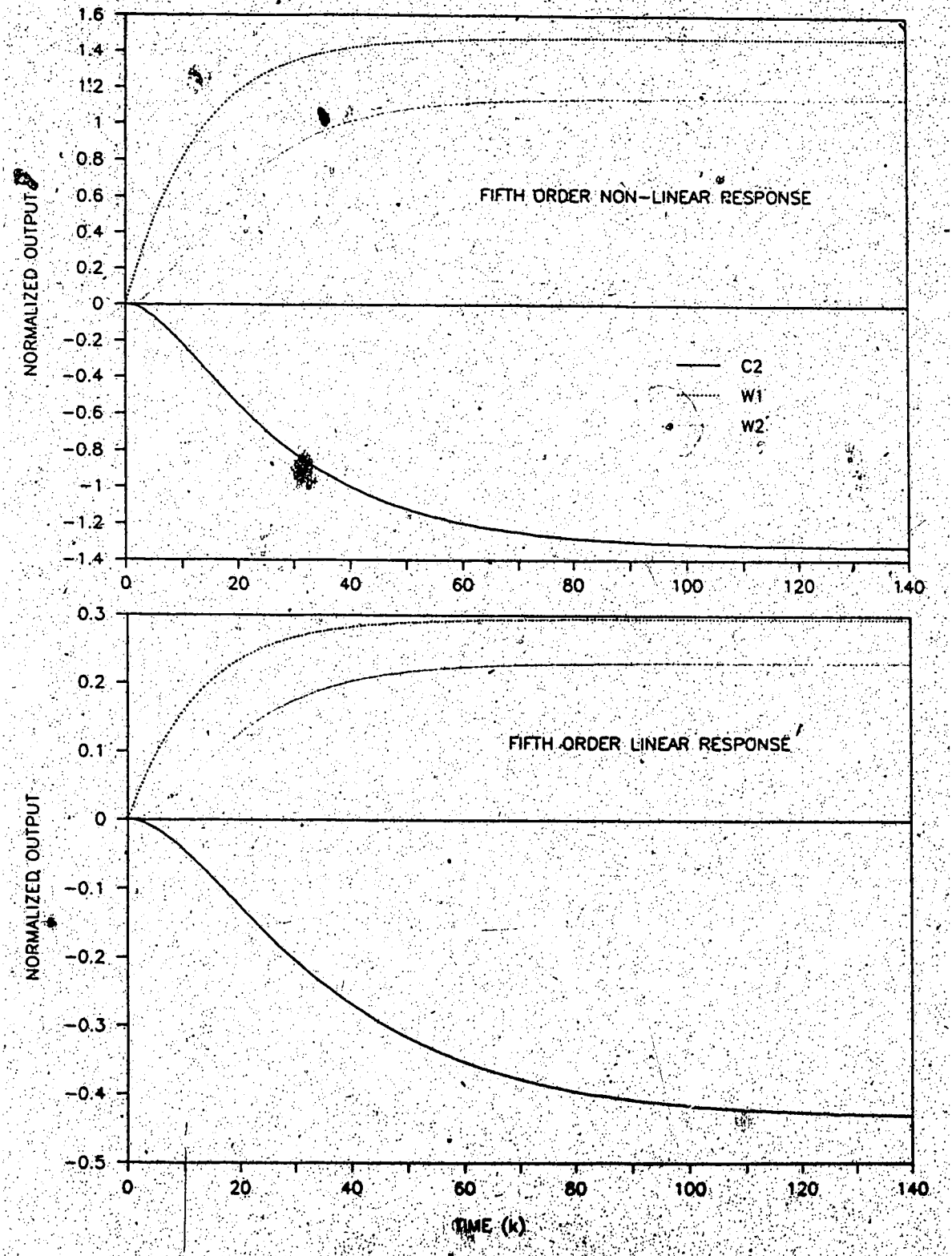

Figure 7.37: 5th Order Non-linear and Linear Evaporator Response for Step in Feed Flow Rate.

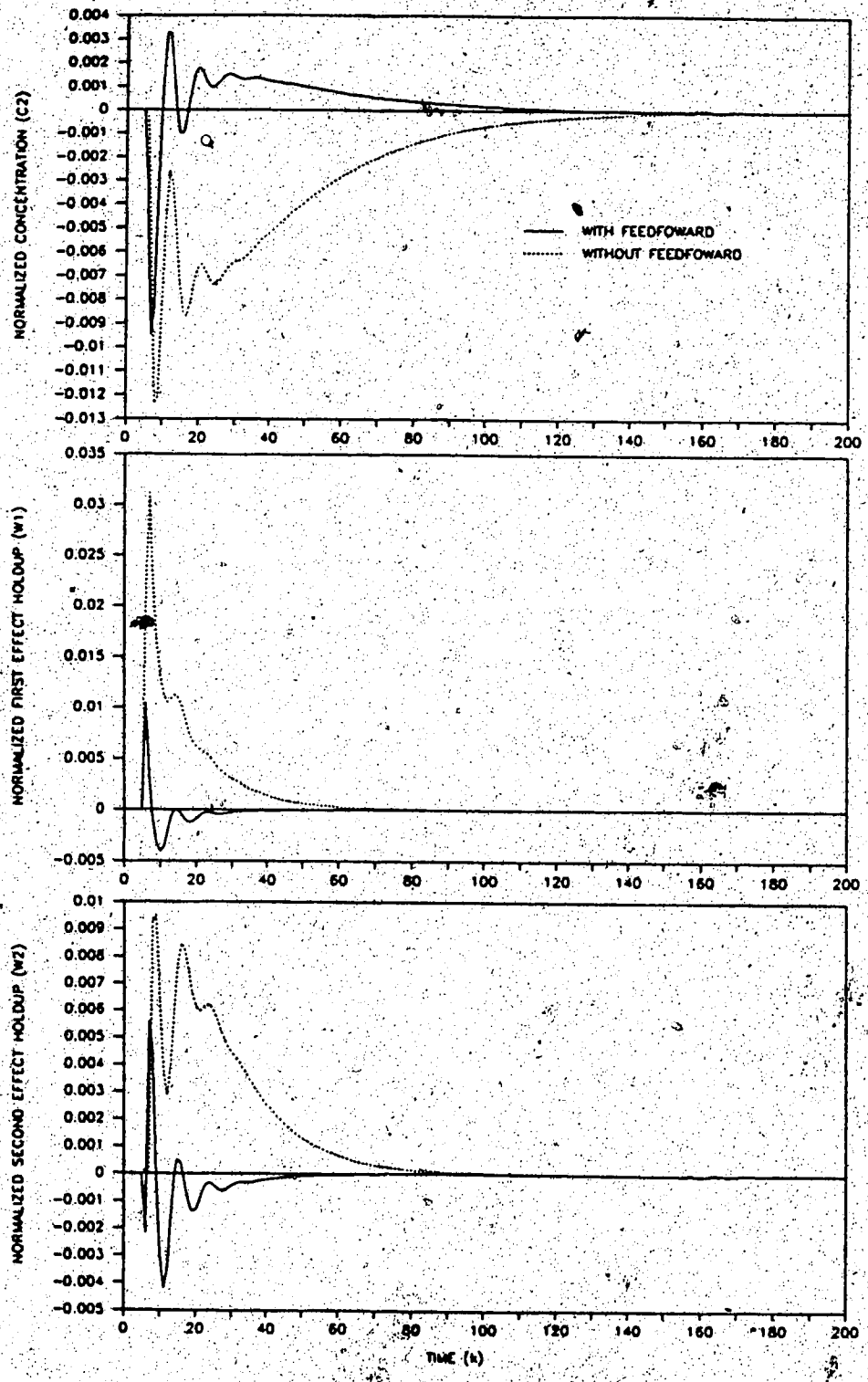

Figure 7.38: Response of Linear Evaporator to 20% Step Change in Feed Flowrate With and Without Feedfoward Control, All N=100 except $N_{11}=N_{12}=150$, P=1, M=1, $\Gamma=(1.0,1.0,1.0)$, $\Gamma_u=(0.0,0.0,0.0)$.

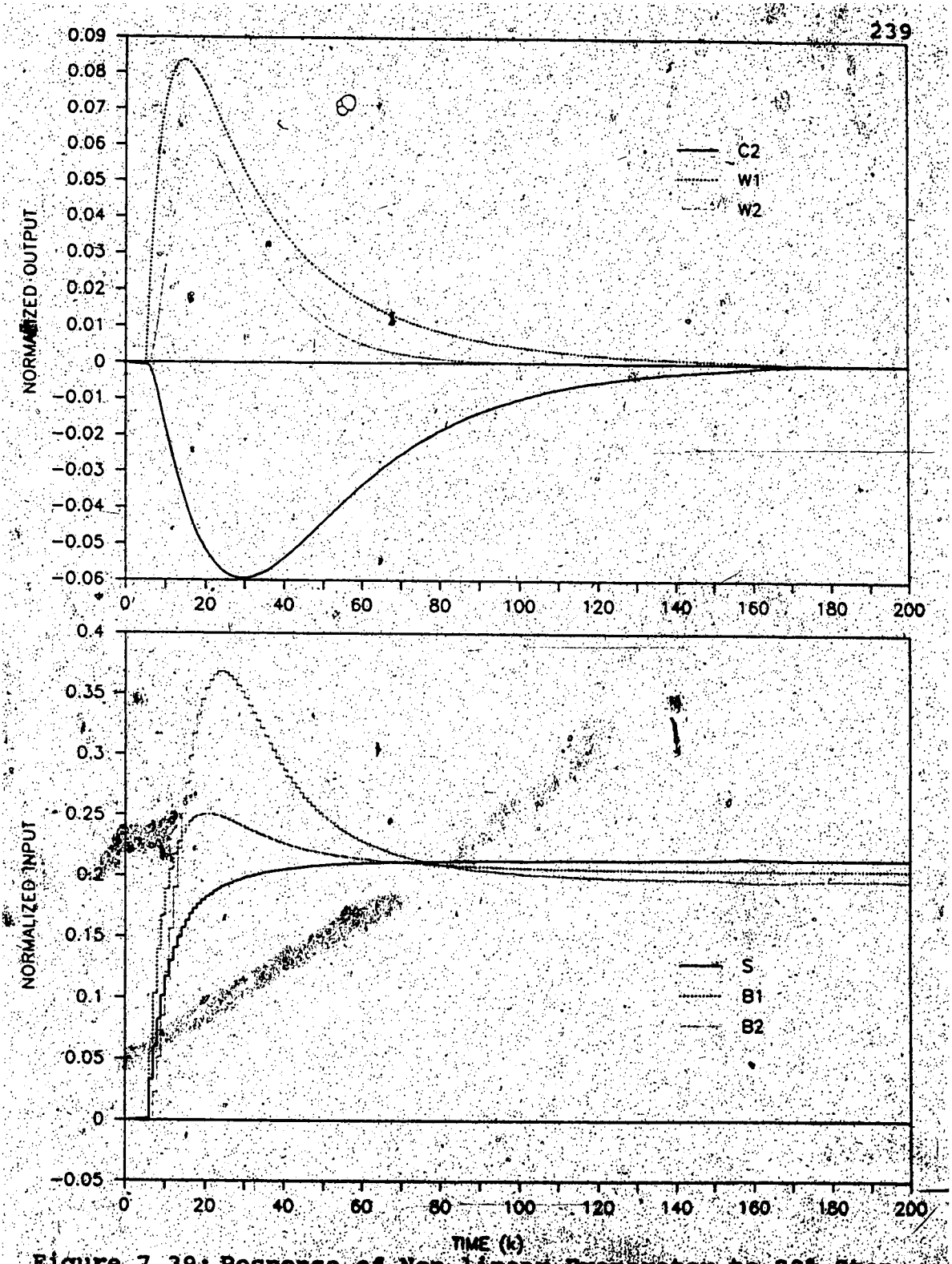

Figure 7.39: Response of Non-linear Evaporator to 20% Step Change in Feed Flowrate Without Feedfoward Control. All N=100 except N_{11}=N_{12}=150, P=10, M=1, Γ=(1.0,1.0,1.0), Γ_f=(0.0,0.0,0.0).

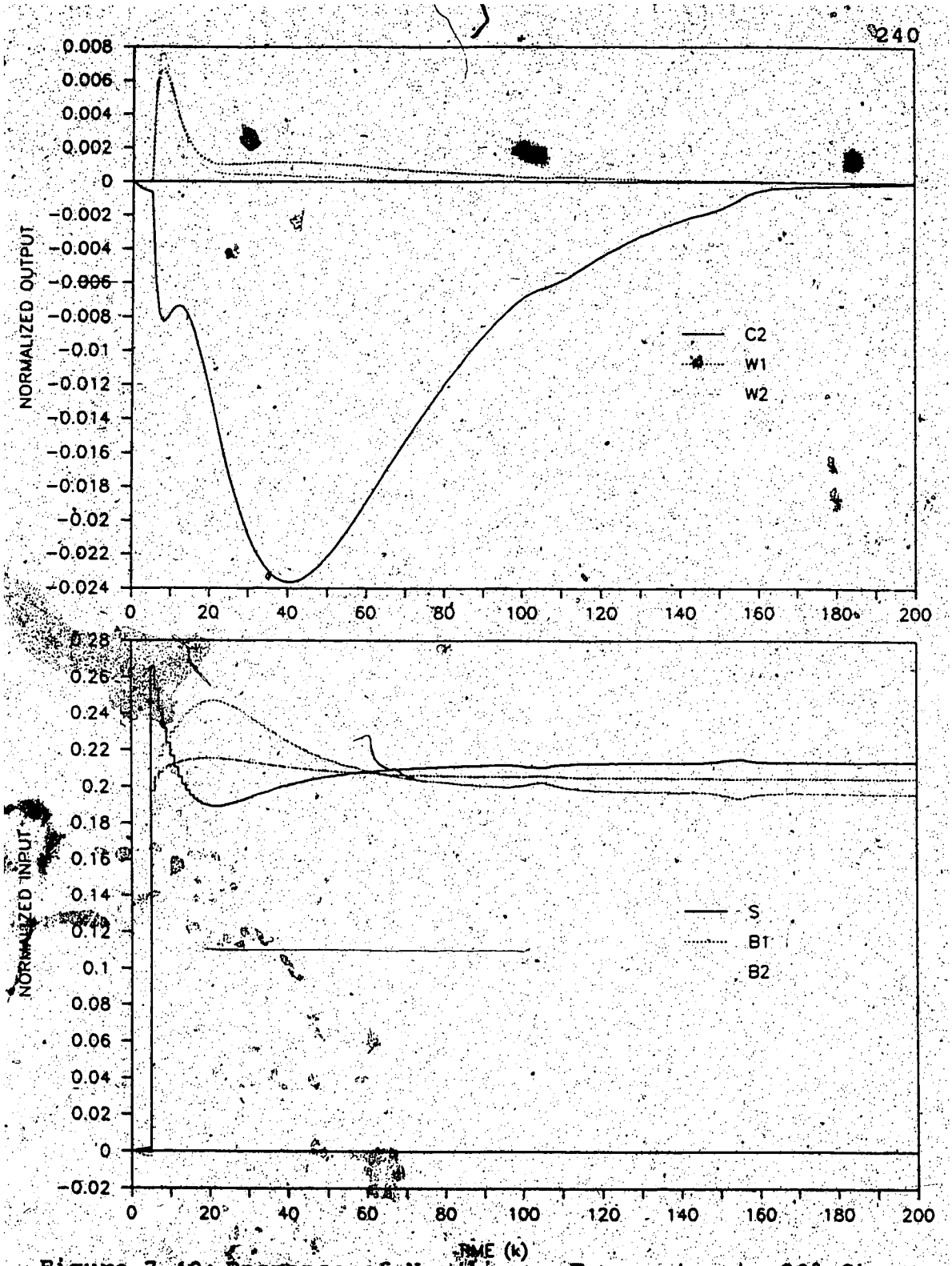

Figure 7.40: Response of Non-linear Evaporator to 20% Step Change in Feed Flowrate With Feedfoward Control. All N=100 except N_{11}=N_{12}=150, P=10, M=1; Γ=(1.0,1.0,1.0), Γ_d=(0.0,0.0,0.0).

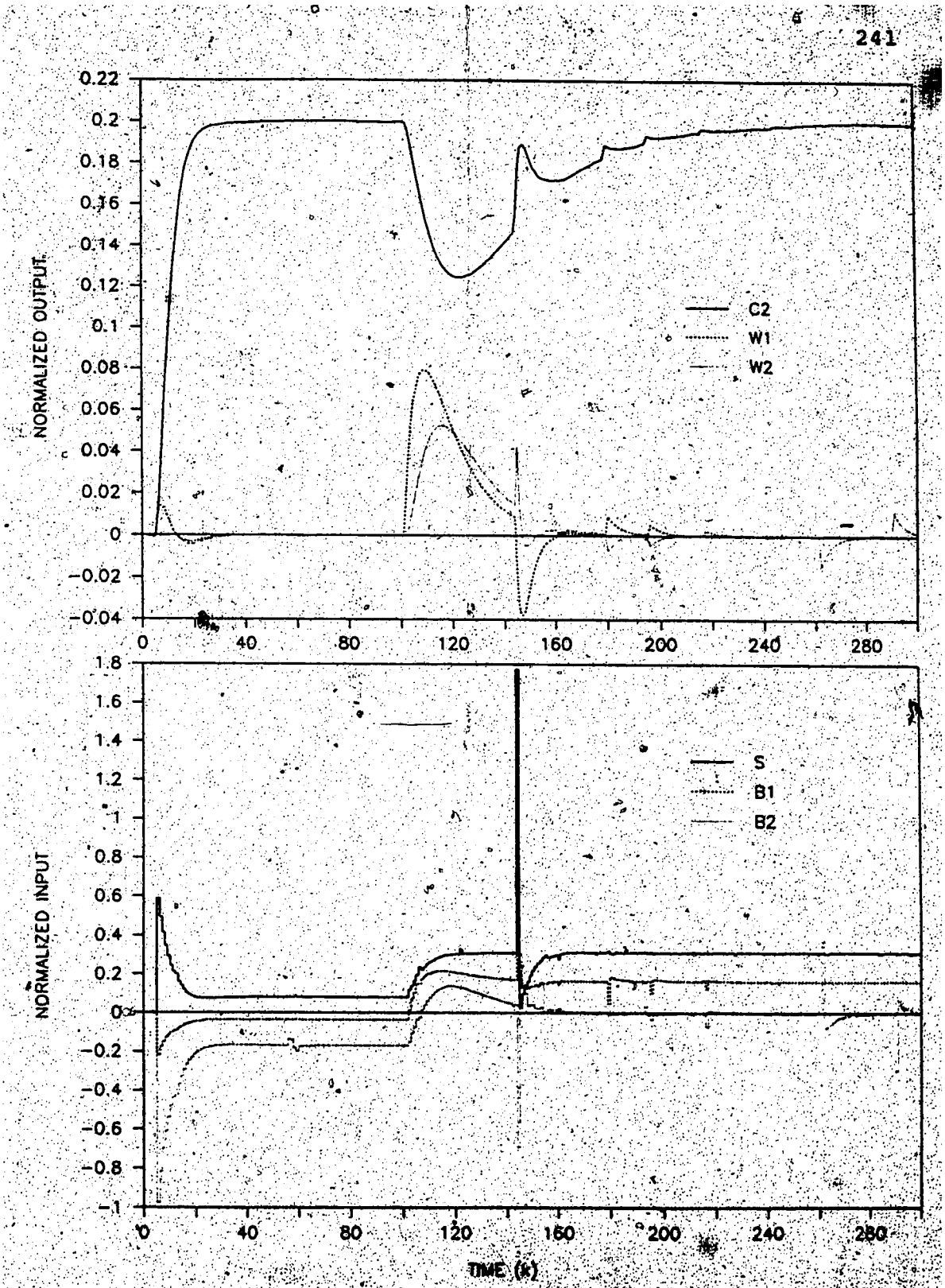

Figure 7.41: LP IAE Control of Non-Linear Evaporator Without Constraints. All N=100 except $N_{11}=N_{12}=150$, P=10, M=1, Γ=(1.0,1.0,1.0).

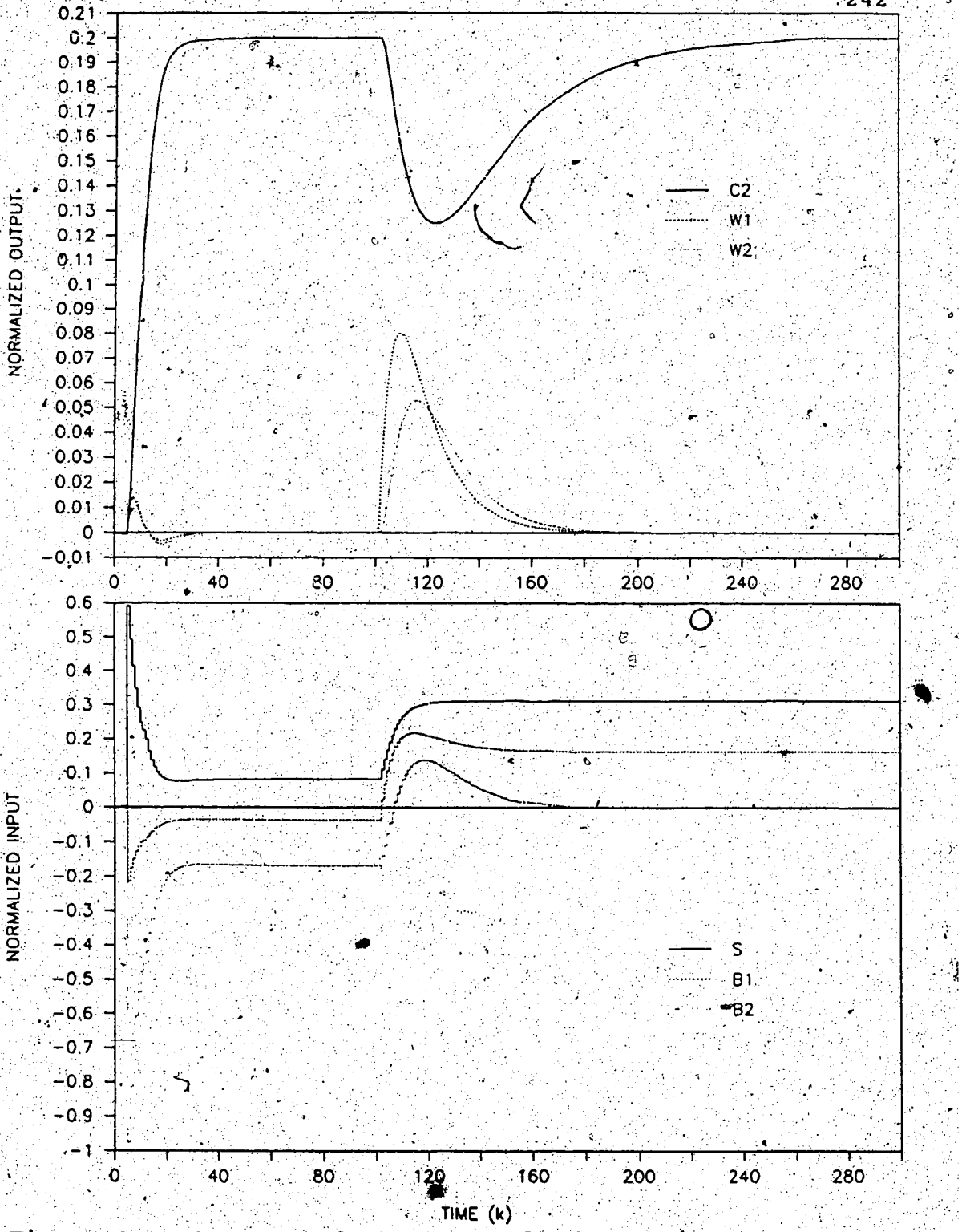

Figure 7.42: LP IAE and IA ΔU Control of Non-Linear Evaporator Without Constraints. All N=100 except $N_{11}=N_{12}=150$, P=10, M=1, $\Gamma = (1.0, 1.0, 1.0)$, $\Gamma_{\Delta} = (0.1, 0.1, 0.1)$.

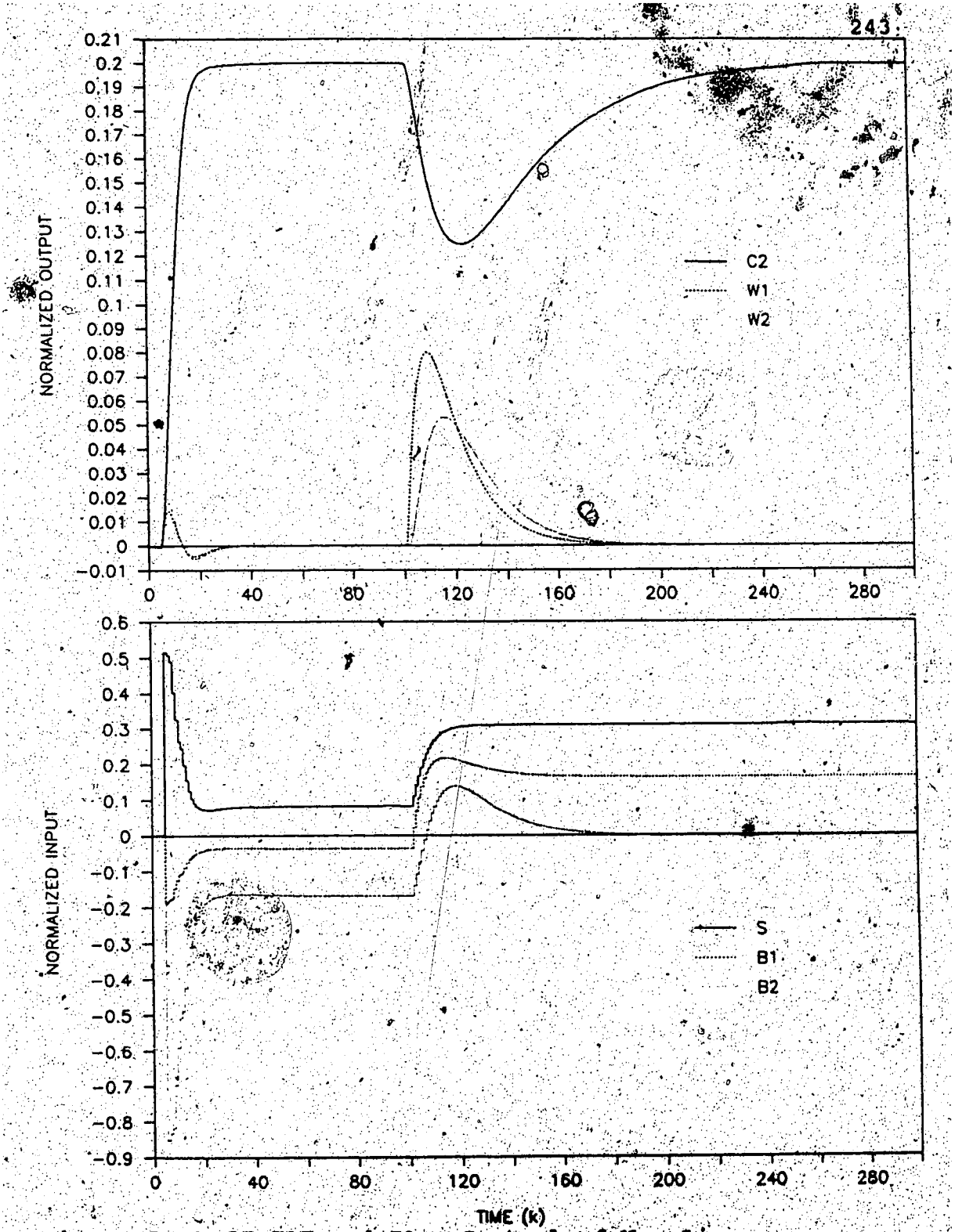

Figure 7.43: LP IAE and IAΔU Control of Non-Linear Evaporator Without Constraints. All N=100 except $N_{11}=N_{12}=150$, P=10, M=1, $\Gamma=(1.0,1.0,1.0)$, $\Gamma_{\Delta}=(0.5,0.1,0.5)$.

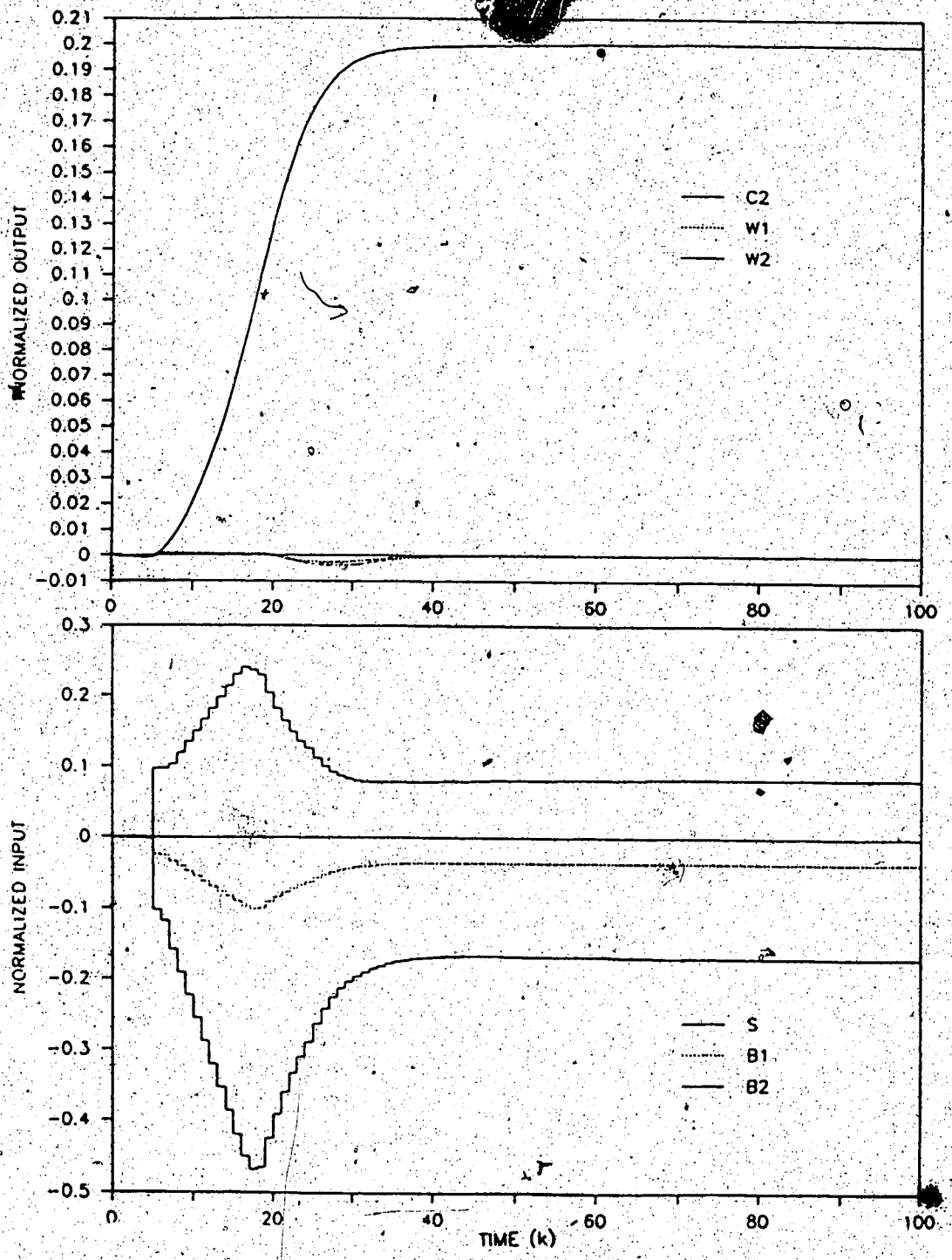

Figure 7.44: LP IAE and IAΔU Control of Non-Linear Evaporator With Output Constraints to Minimize Interaction. Upper Bound Output Deviation of (0.01, 0.001, 0.001). All N=100 except $N_{11}=N_{12}=150$, P=10, M=1, $\Gamma=(1.0,1.0,1.0)$, $\Gamma_m=(0.1,0.1,0.1)$.

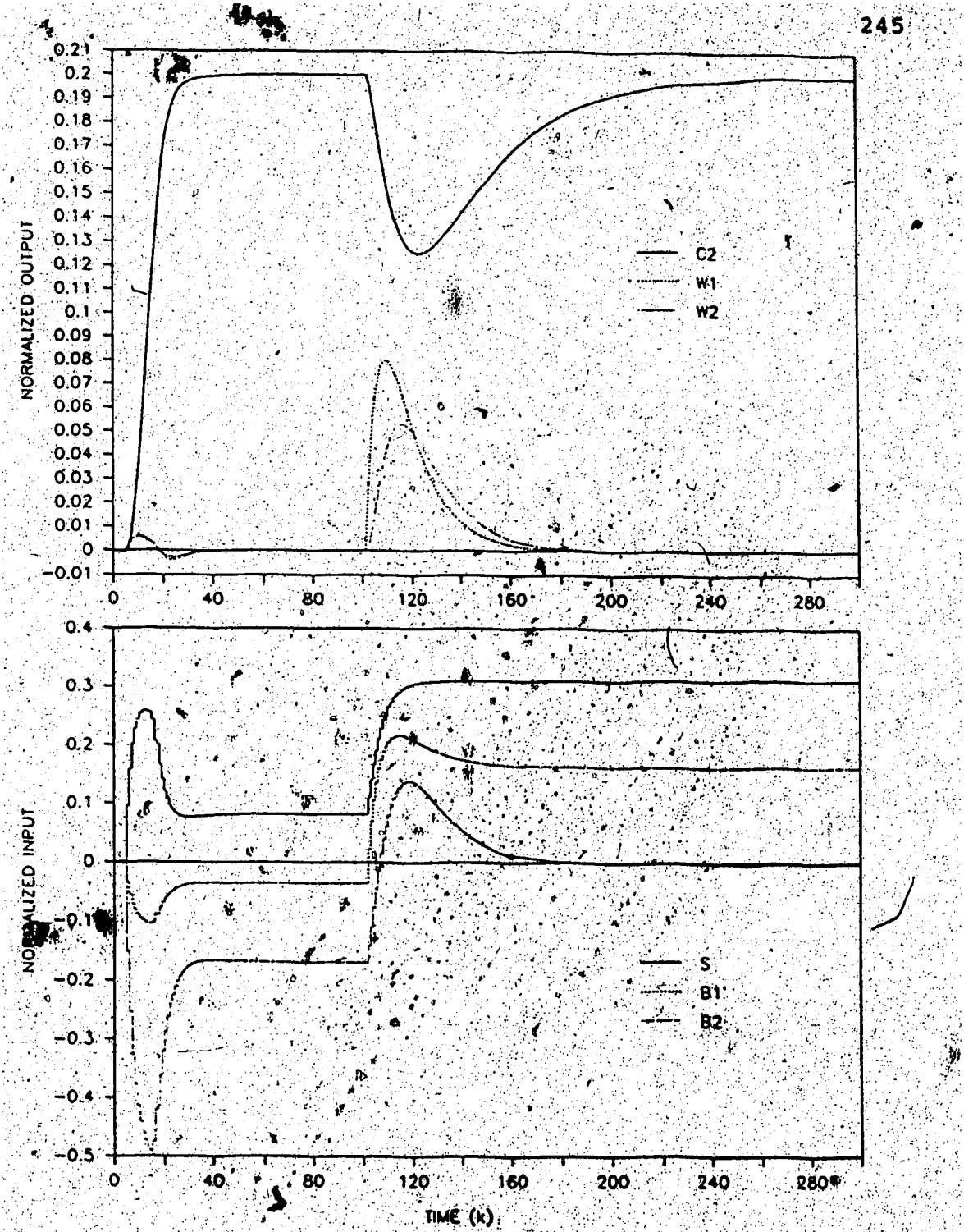

Figure 7.45: LP IAE and IAΔU Control of Non-Linear Evaporator With Input Constraints; Upper Bound of (0.5,0.5,0.182) and Lower Bound of (-1.0,-0.5,-0.182). All N=100 except $N_{11}=N_{12}=150$, P=10, M=1, $\Gamma=(1.0,1.0,1.0)$, $\Gamma_{\Delta}=(0.1,0.1,0.1)$.

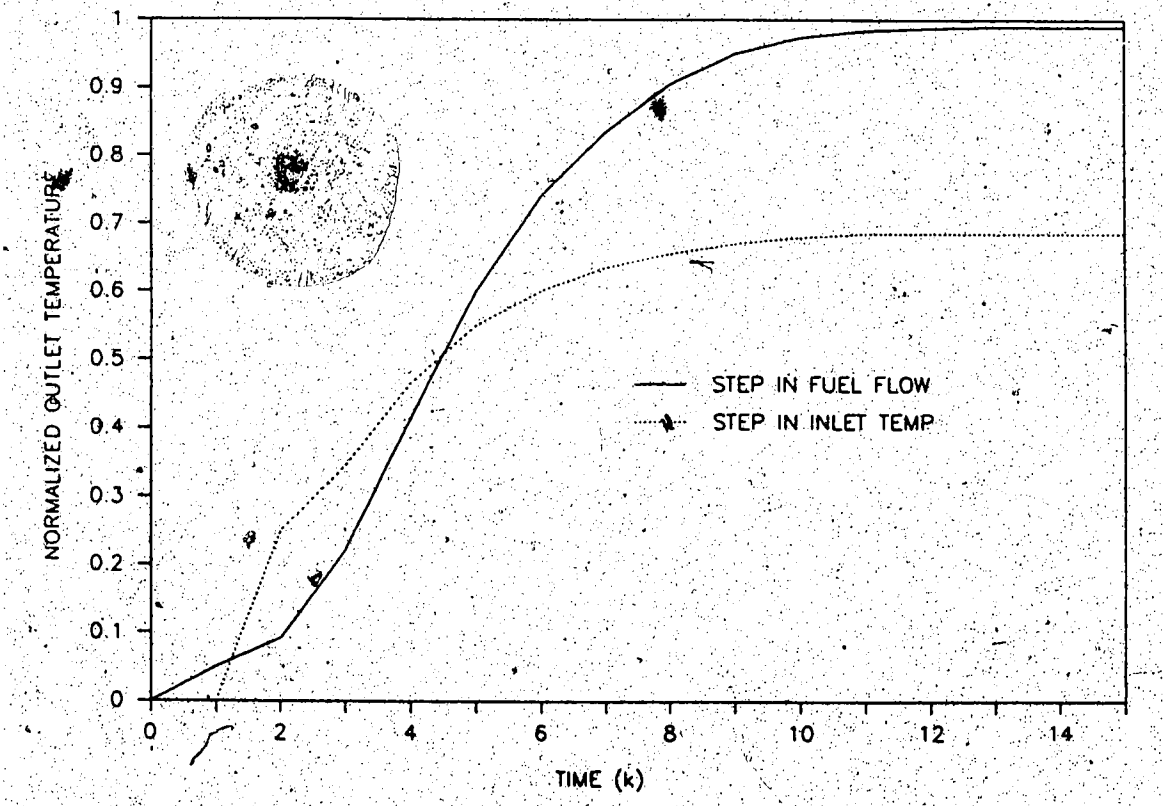

Figure 7.46: Outlet Temperature Responses of a Furnace for Unit Step Change in Fuel Flowrate and Inlet Temperature.

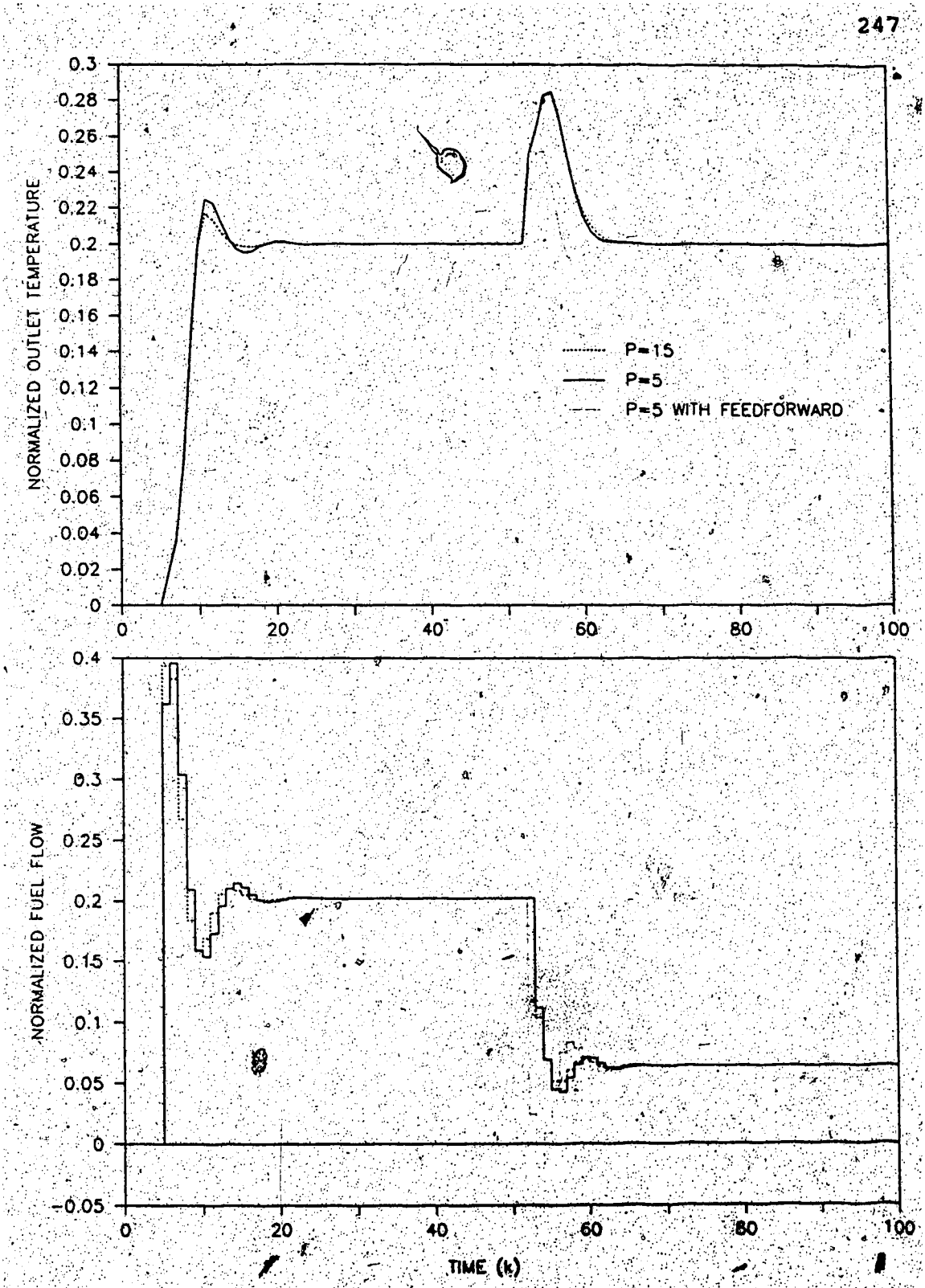

Figure 7.47: Simulated Responses of Furnace for 20% Step Change Outlet Temperature and Inlet Temperature. N=15, M=3, $\Gamma = 1.0, \Gamma_u = 0.1$.

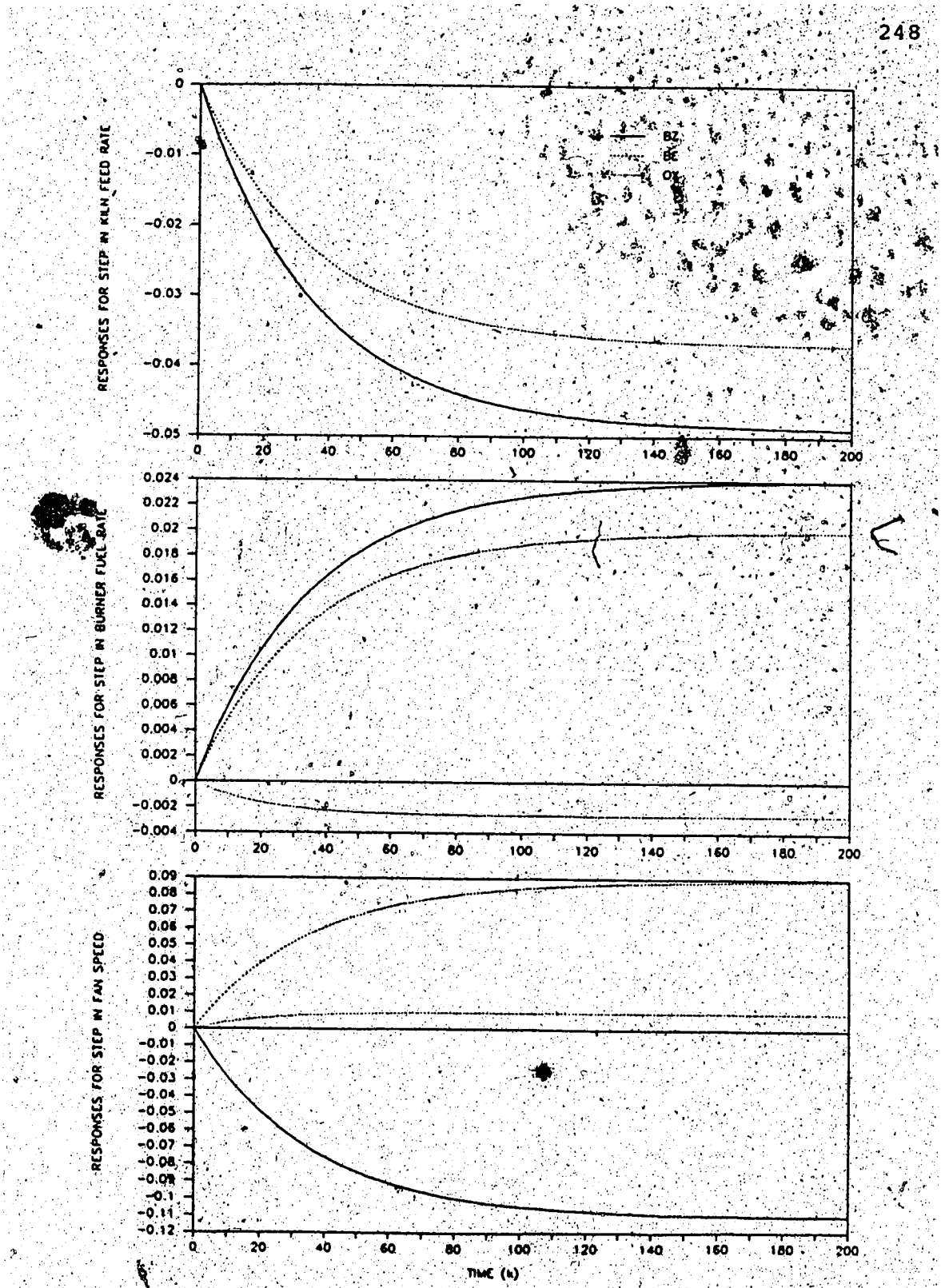

Figure 7-48: Cement Kiln Step Responses Obtained From Linear Model (Equation 7-17). Sampling Time of 2 Minutes.

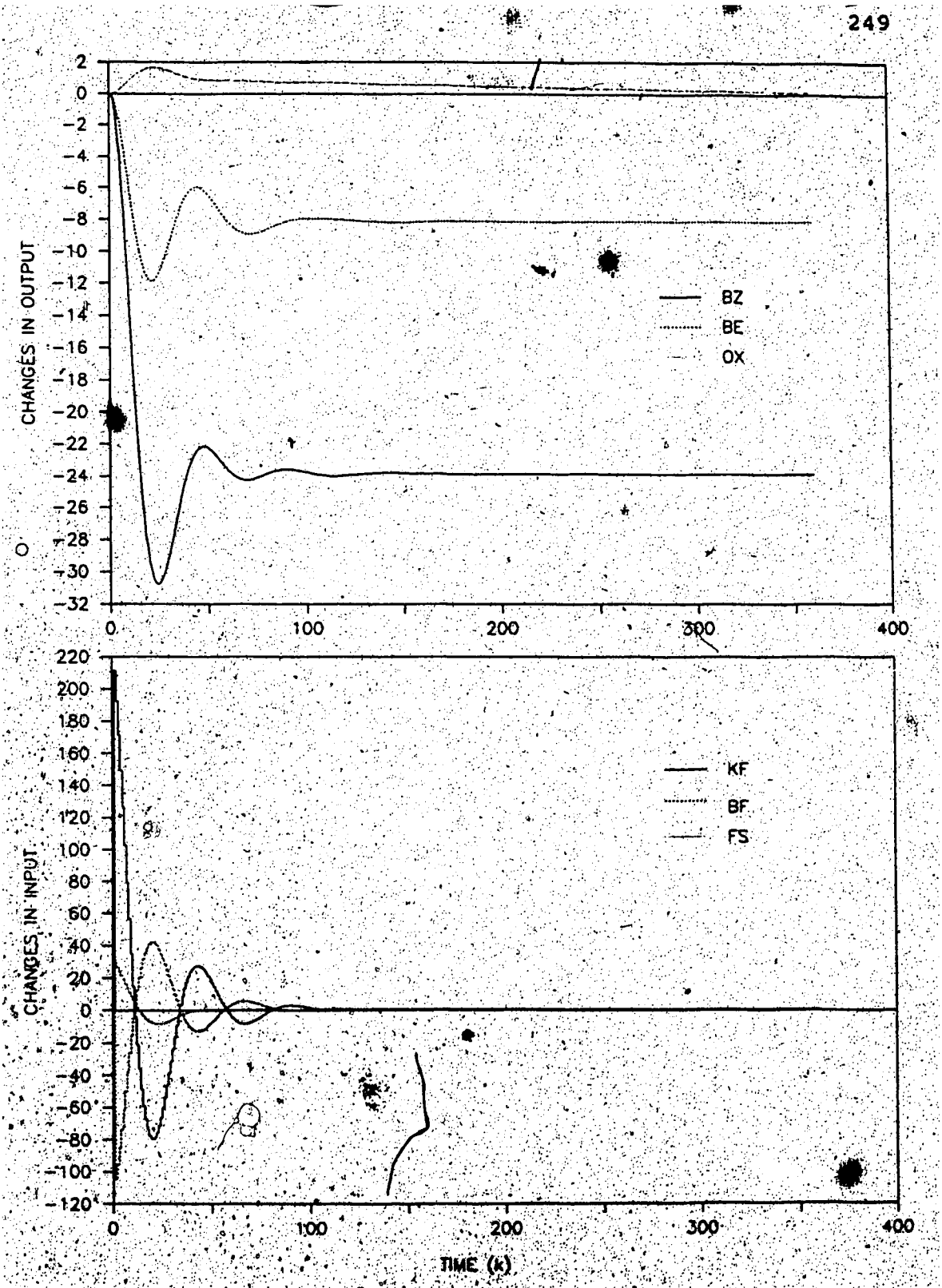

Figure 7.49: Responses of Cement Kiln Using Weighted Least Squares. Ts=2.0, N=200, P=5, M=3
$\Gamma = (1.0, 1.0, 1.0)$, $\Gamma_u = (0.0025, 0.0025, 0.025)$.

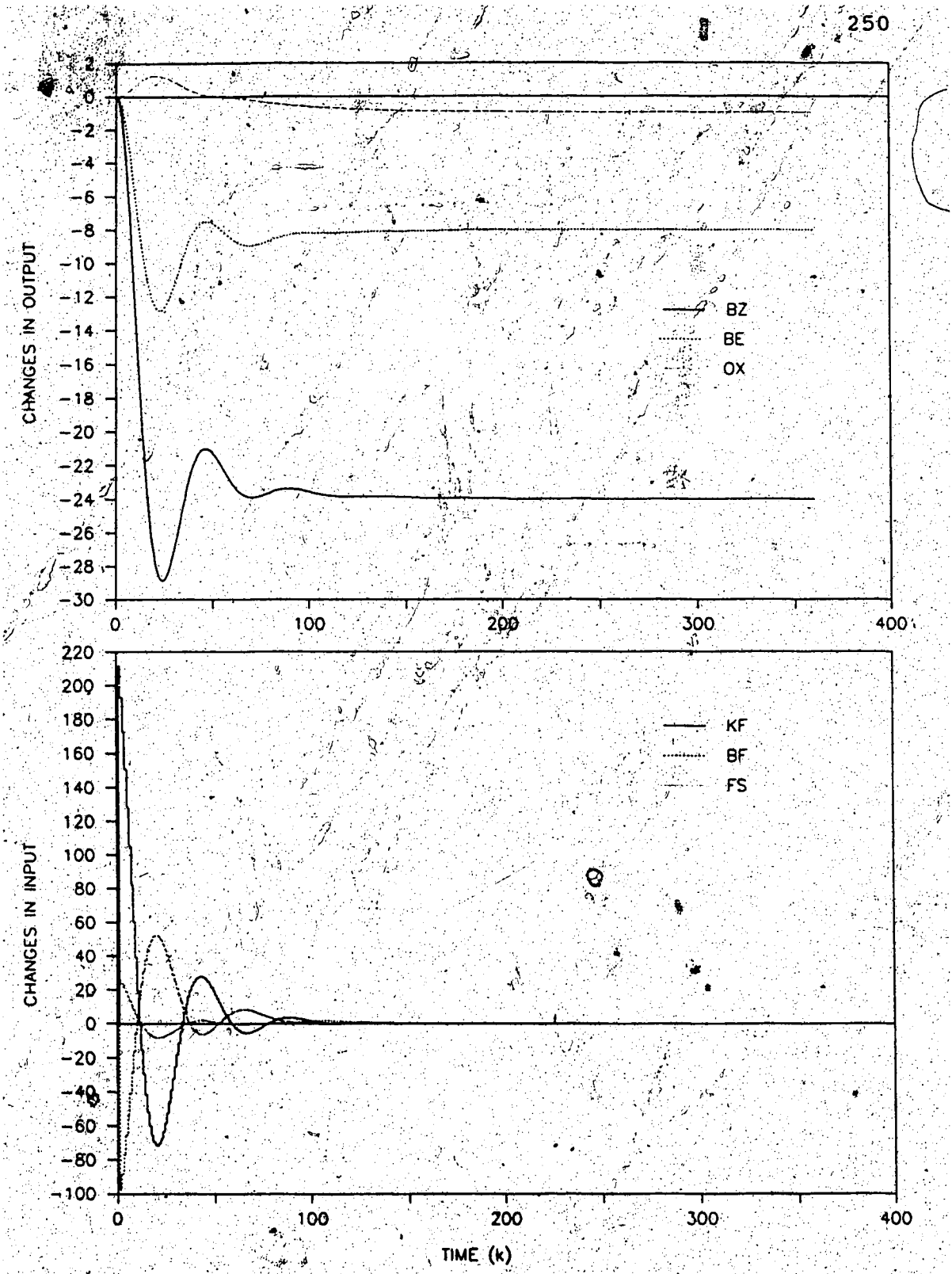

Figure 7.50: Responses of Cement Kiln Using Weighted Least Squares. Ts=2.0, N=200, P=5, M=3
$\Gamma = (1.0, 1.0, 10.0)$, $\Gamma_m = (0.0025, 0.0025, 0.025)$.

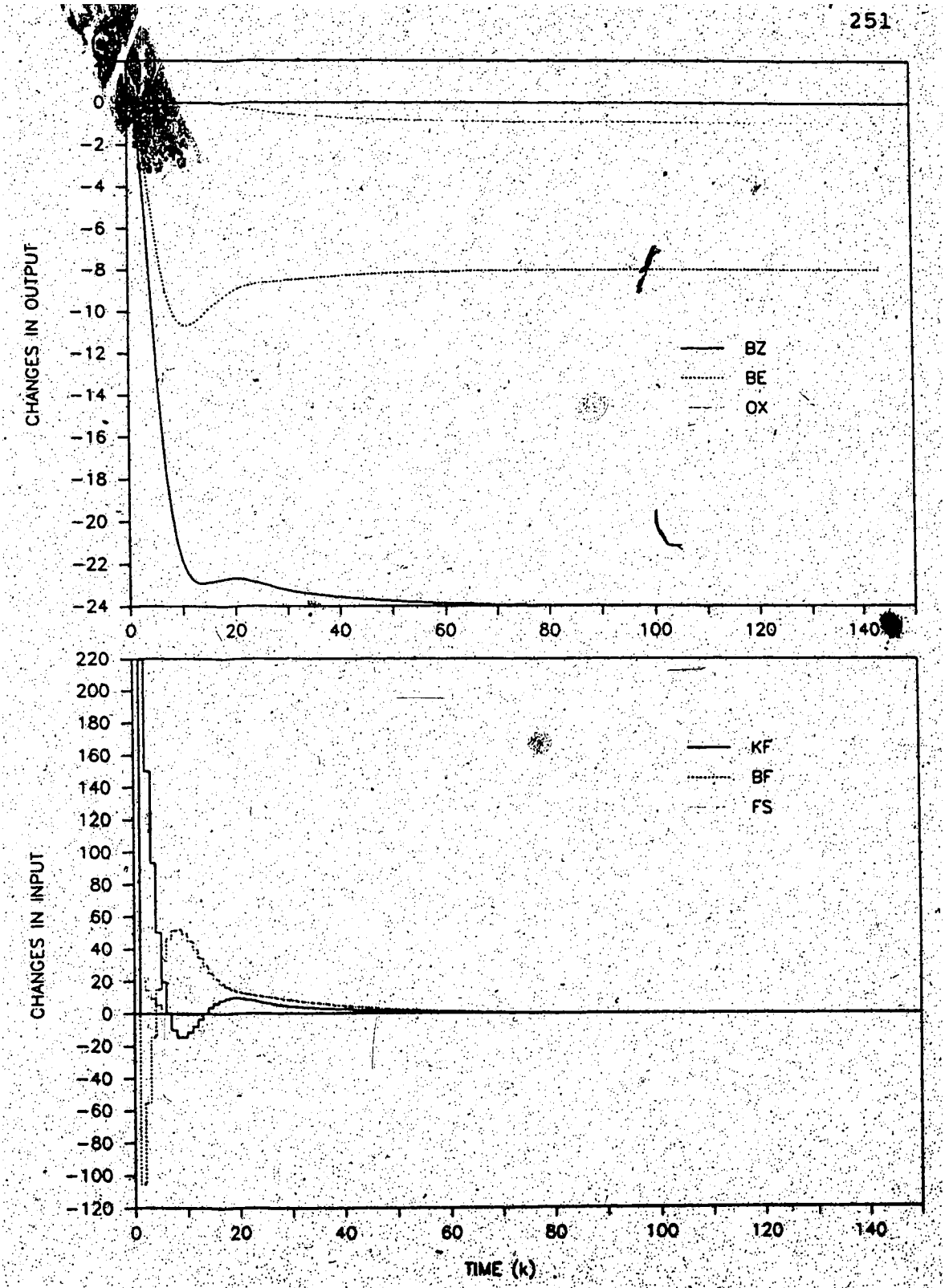

Figure 7.51: Responses of Cement Kiln Using Weighted Least Squares. Ts=10.0, N=80, P=5, M=3
Γ=(1.0,1.0,10.0), Γ_u=(0.006,0.005,0.075).

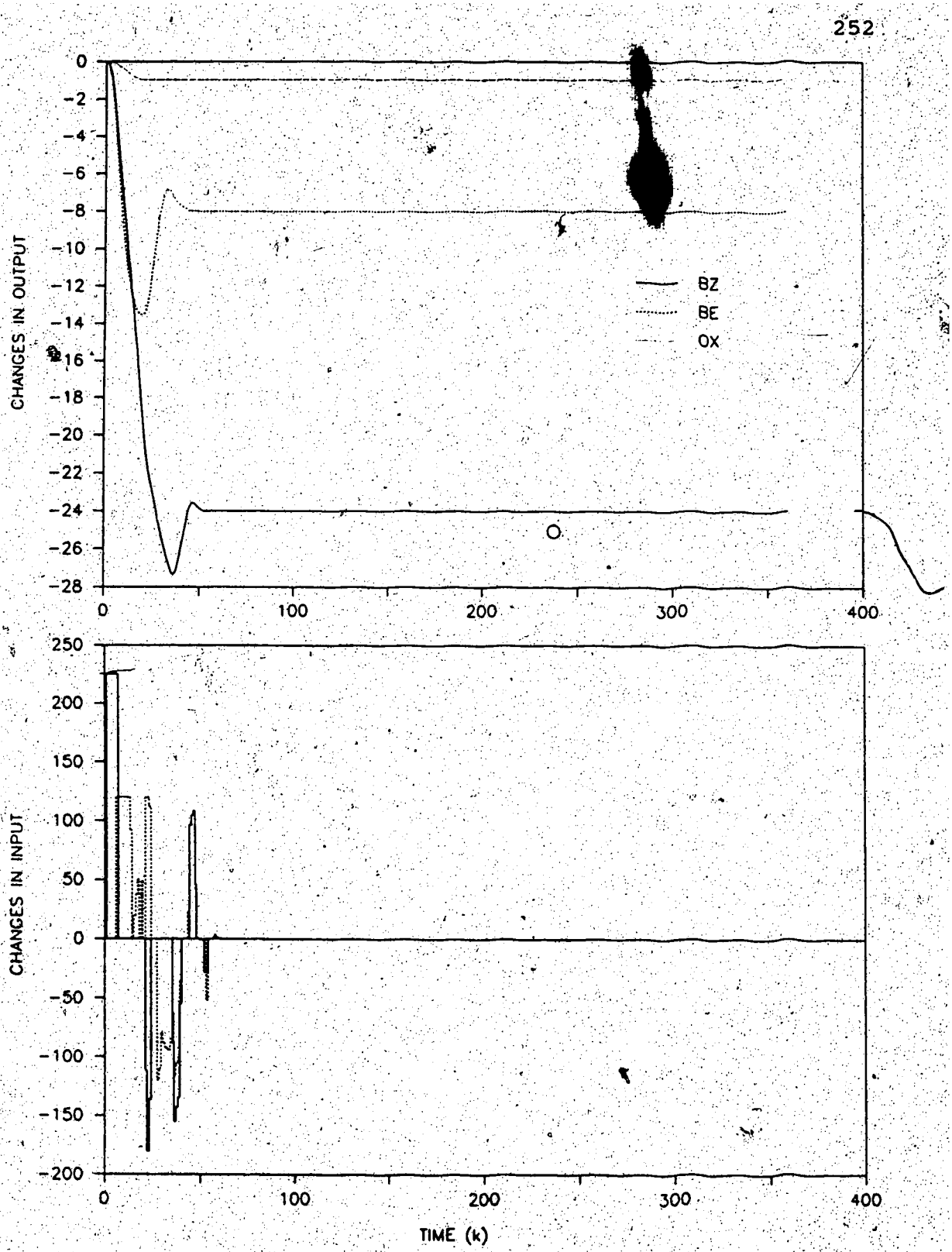

Figure 7.52: Responses of Cement Kiln Using LP IAE and IA ΔU With Change in Input Constraints. Ts=2.0, N=200, P=5, M=3 Γ = (1.0,1.0,10.0), h_u = (0.01,0.01,0.01).

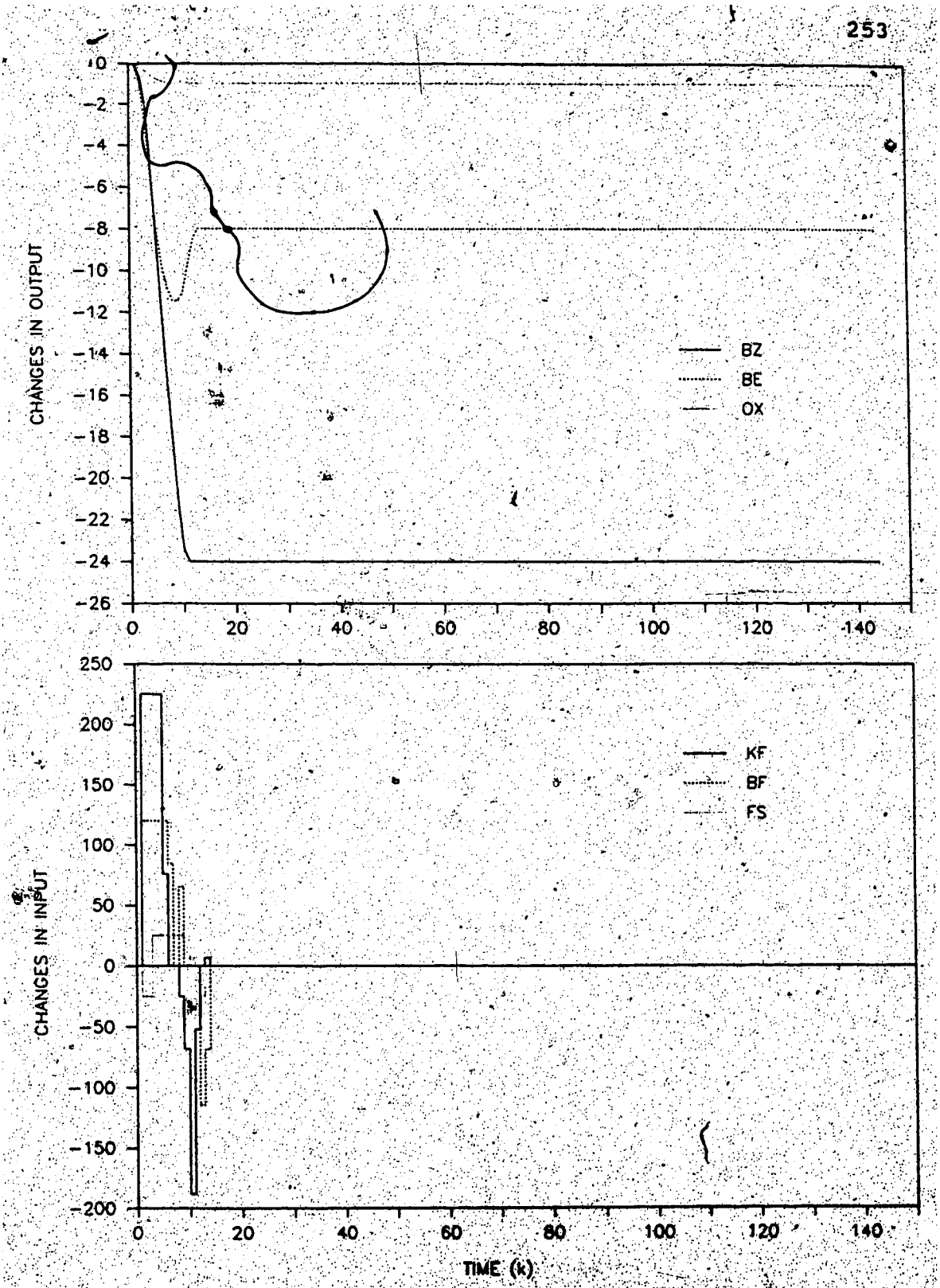

Figure 7.53: Responses of Cement Kiln Using LP IAE and IAΔU With Change in Input Constraints. Ts=10.0, N=80, P=5, M=3 Γ=(1.0,1.0,10.0), Γ_u=(0.01,0.01,0.01).

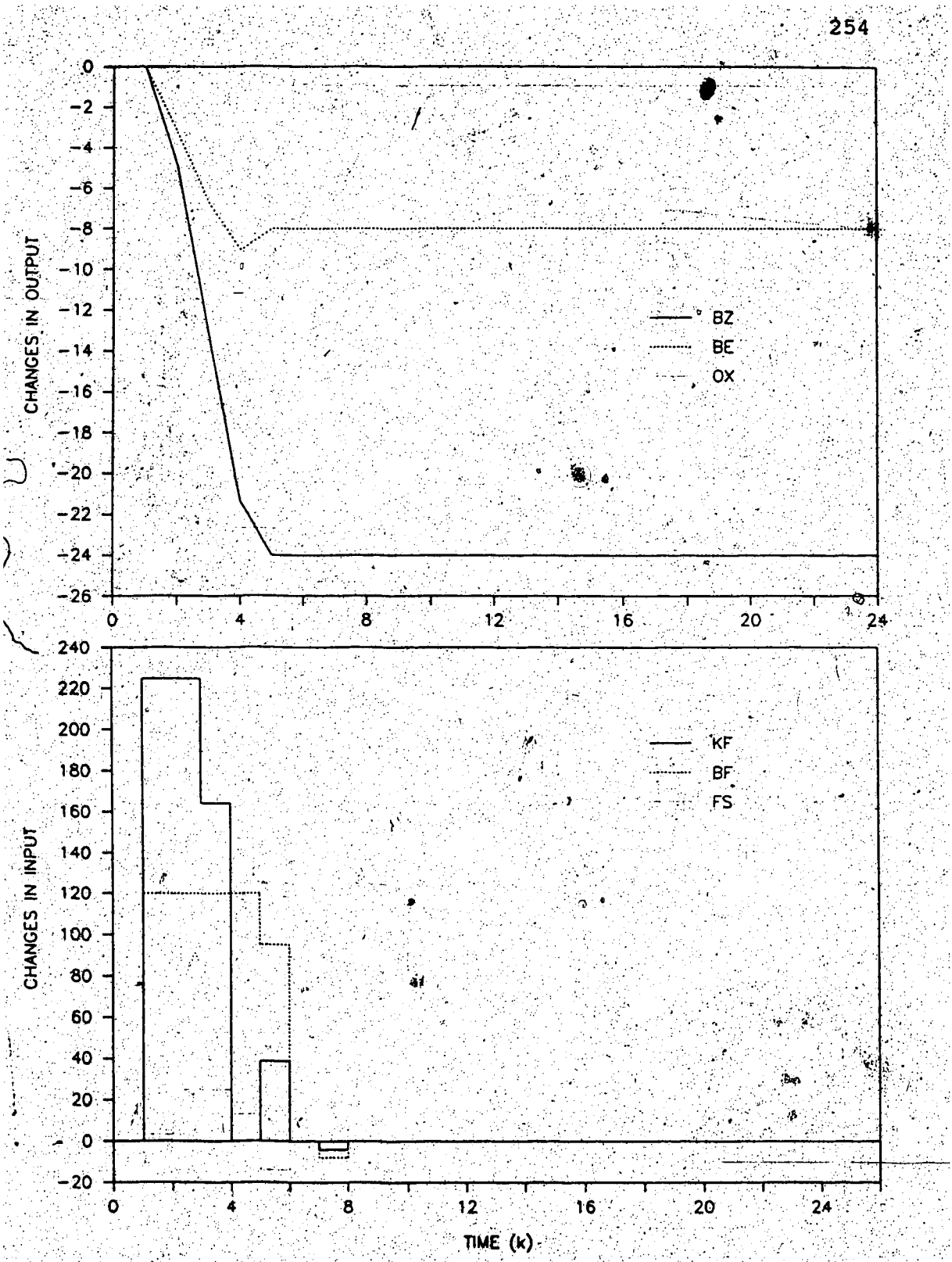

Figure 7.54: Responses of Cement Kiln Using LP IAE and IA ΔU With Change in Input Constraints. Ts=60.0, N=20, P=5, M=3 Γ=(1.0,1.0,10.0), Γ_m=(0.01,0.01,0.01).

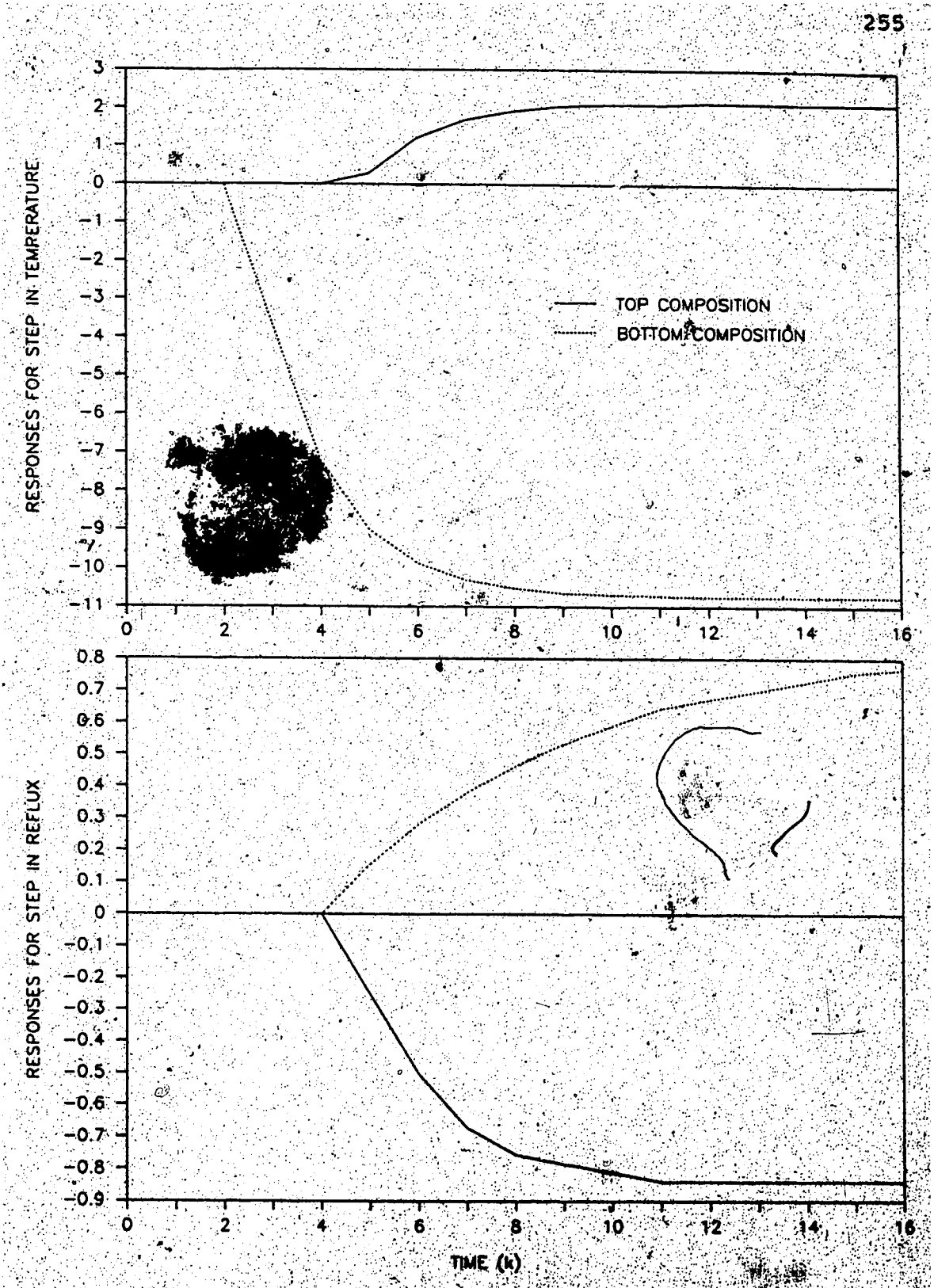

Figure 7.55: Responses of Distillation Column For Unit Step Change in Reflux and Temperature.

Cutler, C. R. and Ramaker, B. L., (1979) "Dynamic Matrix Control - A Computer Control Algorithm", AIChE 86th National Meeting, Paper No. 51B (Also published in Proceedings of Joint Automatic Control Conference, San Francisco, USA, 1980, Paper WP5-B).

Cutler, C. R., (1982) "Dynamic Matrix Control of Imbalance Systems", ISA Transactions, 21, No. 1, 1-6.

Cutler, C. R., (1983) "Dynamic Matrix Control: An Optimal Multivariable Control Algorithm with Constraints", Ph.D. Thesis, University of Houston, Texas, USA.

Cutler, C. R. and Johnston, C. R., (1985) "Comparison of the Quality Criterion for PID and Predictive Controllers", Proceedings American Control Conference, Boston, USA, 214-219.

Cutler, C. R. and Hawkins, R. B., (1987) "Constrained Multivariable Control of a Hydrocracker Reactor", Proceedings American Control Conference, Minneapolis, USA, 1014-1028.

Daclin, E., (1980) "Design Studies for the Automatic Piloting of the Tripartite Mine Hunter Boat", Proceedings of Joint Automatic Control Conference, San Francisco, USA, Vol. 2, paper FA9-E.

Economou, C. G., Morari, M. and Paisson, B. O., (1986) "Internal Model Control: 5. Extension to Non-linear Systems", Industrial and Engineering Chemistry: Process Design and Development, 25, No. 1, 266-274.

Economou, C. G. and Morari, M., (1986) "Internal Model Control: 6. Multi-loop Design", Industrial and Engineering Chemistry: Process Design and Development, 25, No. 1, 275-283.

Engrand, J. C., (1980) "Application of Multivariable Control in a Refinery and Implementation on a Dedicated Minicomputer", Proceedings of Joint Automatic Control Conference, San Francisco, USA, Vol. 2, paper FA9-D.

Favier, G., (1987) "Self-Tuning Long-Range Predictive Controllers", IFAC 10th World Congress on Automatic Control Pre-prints, (R. Isermann, Ed.), Munich, Federal Republic of Germany, Vol. 10, 80-87.

Fisher, D. G. and Seborg, D. E., (1976) Multivariable Computer Control: A Case Study, North-Holland/American Elsevier, New York, USA

Foigel, J. K. and Richalet, J., (1979) "Self-Adapting IDCOM", 5th IFAC Symposium on Identification and System Parameter Estimation, Vol. 2, 1095-1102

Freedman, R. W. and Bhatia, A., (1985) "Adaptive Dynamic Matrix Control: On-line Evaluation of the DMC Model Coefficients", Proceedings American Control Conference, Boston, USA, 220-225.

Froisy, J. B. and Richalet, J., (1986) "Industrial Applications of IDCOM", Proceedings of the Third International Conference on Chemical Process Control (CPC III), (Morari, M. and McAvoy, T., Eds.), CACHE and Elsevier, Amsterdam.

Garcia, C. E. and Morari, M., (1982) "Internal Model Control: 1. A Unifying Review and Some New Results", Industrial and Engineering Chemistry: Process Design and Development, 21, No. 2, 308-323.

Garcia, C. E. and Morari, M., (1985a) "Internal Model Control: 2. Design Procedure For Multivariable Systems", Industrial and Engineering Chemistry: Process Design and Development, 24, No. 2, 472-484.

Garcia, C. E. and Morari, M., (1985b) "Internal Model Control: 3. Multivariable Control Law Computation and Tuning Guidelines", Industrial and Engineering Chemistry: Process Design and Development, 24, No. 2, 484-494.

Garcia, C. E. and Morshedi, A. M., (1986) "Quadratic Programming Solution of Dynamic Matrix Control (QDMC)", Chemical Engineering Communications, 46, No. 1-3, 73-87.

Garcia, C. E. and Prett, D. M., (1986) "Advances in Industrial Model Predictive Control", Proceedings of the Third International Conference on Chemical Process Control (CPC III), (Morari, M. and McAvoy, T., Eds.), CACHE and Elsevier, Amsterdam.

Garcia, C. E., (1987) "Real Benefits of Model Predictive Control", IFAC 10th World Congress on Automatic Control Pre-prints, (R. Isermann, Ed.), Munich, Federal Republic of Germany, Vol. 11, 8.

Gerald, C. F. and Wheatley, P. O., (1984) Applied Numerical Analysis 3rd Edition, Addison-Wesley Publishing Co., USA.

Gill, P. E., Murray, W. and Wright, M. H., (1981) Practical Optimization, Academic Press, New York, USA.

Goodwin, G. C. and Sin, K. S., (1984) Adaptive Filtering Prediction and Control, Prentice-Hall, N. J., USA.

Graham, B. P., Lee, P. L. and Newell, R. B., (1985) "Simulation of Multivariable Control of a Cement Kiln", Chemical Engineering Research and Design, 63, November 1985.

Hmood, S. A. and Prasad, R. M., (1987) "Generalized Approach for Model Algorithmic Control", International Journal of Control, 18, No. 8, 1395-1410

Huang, H., Chao, Y. and Liu, P., (1985) "Predictive Adaptive Control System for Unmeasured Disturbances", Industrial and Engineering Chemistry Process Design and Development, 24, No. 3, 666-673.

Kiparissides, C., Sidiropoulou, E., Voutetakis, S. and Sahinidis, N. B., (1987) "A Comparative Study of LQC, DMC and Extended STR Control Strategies", IFAC 10th World Congress on Automatic Control Pre-prints. (R. Isermann, Ed.), Munich, Federal Republic of Germany, Vol. 3, 123-128.

Kozub, D. J. and MacGregor, J. F., (1987) "Optimal IMC Design Via Spectral Factorization", Proceedings American Control Conference, Minneapolis, USA, 652-658.

Kuo, B. C., (1982) Automatic Control Systems, 4th Edition, Prentice-Hall, N. J., USA.

Land, A. H. and Powell, S., (1973) Fortran Codes for Mathematical Programming: Linear, Quadratic and Discrete, John Wiley and Sons, London, UK.

Laughlin, D. L. and Morari, M., (1987) "Smith Predictor Design for Robust Performance", Proceedings American Control Conference, Minneapolis, USA, 637-642.

Lawson, C. L. and Hanson, R. J., (1974) Solving Least Squares Problems, Prentice-Hall, Englewood Cliffs, N. J., USA.

Lebourgeois, F., (1980) "IDCOM Applications and Experiences on a PVC Production Plant", Proceedings of Joint Automatic Control Conference, San Francisco, USA, Vol. 2, paper FA9-C.

Lee, P. L. and Newell, R. B., (1985) "Multivariable Control of a Grinding Circuit", IFAC Symposium on Automation for Mineral Resource Development, 283-287

Levien, K. L. and Morari, M., (1987) "Internal Model Control of Coupled Distillation Columns", AIChE Journal, 33, No. 1, 83-98

Li, S., Fisher, D. G. and Lim, K. Y., (1986) "A State Space Formulation for Model Predictive Control", Pre-prints of the 36th National CSChE Conference, Sarnia, Ontario, Canada.

Little, D. L. and Edgar, T. F., (1986) "Predictive Control Using Constrained Optimal Control", Proceedings American Control Conference, Seattle, USA, 1365-1371.

Mahmood, S. and Larimore, W. E., (1986) "Multivariable Adaptive Control of a Missile Using MAC and CVA", Proceedings American Control Conference, Seattle, USA, 1482-1489.

Man, D. S., (1984) "Process Control Using Single Series Forecasting", MSc. Thesis, Department of Chemical Engineering, University of Alberta, Edmonton, Alberta, Canada.

Marchetti, J. L., Mellichamp, D. A. and Seborg, D. E., (1983a) "Predictive Control Based on Discrete Convolution Models", Industrial and Engineering Chemistry: Process Design and Development, 22, No. 3, 488-495.

Marchetti, J. L., Mellichamp, D. A. and Seborg, D. E., (1983b) "Design of Predictive Control Systems Based on Pole Assignment", Proceedings American Control Conference, San Francisco, USA, 1132-1138.

Martin, G. D., (1981) "Long Range Predictive Control", AIChE Journal, 27, No. 5, 748-753.

Martin, G. D., Van Horn, L. D. and Cassaday, K. M., (1984) "Experience With Hydrotreater Computer Control", Hydrocarbon Processing, 63, No. 3, 66-70.

Matsko, T. N., (1985) "Internal Model Control For Chemical Recovery", Chemical Engineering Progress, 81, No. 12, 46-51.

Maurath, P. R., Mellichamp, D. A. and Seborg, D. E., (1985a), "Predictive Controller Design for SISO Systems", Proceedings American Control Conference, Boston, USA, 1546-1552.

Maurath, P. R., Seborg, D. E. and Mellichamp, D. A., (1985b), "Predictive Controller Design by Principal Component Analysis", Proceedings American Control Conference, Boston, USA, 1059-1065.

Maurath, P. R., Seborg, D. E. and Mellichamp, D. A., (1986), "Achieving Decoupling with Predictive Controllers", Proceedings American Control Conference, Seattle, USA, 1372-1377.

McDonald, K. A., McAvoy, T. J. and Tits, A., (1986) "Optimal Averaging Level Control", AIChE Journal, 32, No. 1, 75-86

McDonald, K. A. and McAvoy, T. J., (1987) "Application of Dynamic Matrix Control to Moderate and High Purity Distillation Towers", Industrial and Engineering Chemistry: Process Design and Development, 25, No. 5, 1011-1018.

Mehra, R. K., Rouhani, R. Rault, A. and Reid, J. G., (1979) "Model Algorithmic Control: Theoretical Results on Robustness", Proceedings of Joint Automatic Control Conference, 387-392.

Mehra, R. K. and Rouhani, R., (1980) "Theoretical Considerations on Model Algorithmic Control for Non-minimum Phase Systems", Proceedings of Joint Automatic Control Conference, San Francisco, USA, Paper TA8-B.

Mehra, R. K., Rouhani, R. and Praly, L. (1980) "New Theoretical Development in Multivariable Predictive Algorithmic Control", Proceedings of Joint Automatic Control Conference, San Francisco, USA, Paper FA9-B.

Mehra, R. K., Rouhani, R., Eterno, J., Richalet, J. and Rault, A., (1981) "Model Algorithmic Control (MAC); Review and Recent Developments", Proceedings of the Second Engineering Foundation Conference on Chemical Process Control (CPC II), (Edgar, T. F. and Seborg, D. E., Eds.), United Engineering Trustees, Inc., New York, USA.

Mereau, P., Guillaume, D. and Mehra, R. K., (1978) "Flight Control Application of Model Algorithmic Control With IDCOM (IDENTIFICATION AND COMMAND)", Proceedings of the 1978 IEEE Conference on Decision and Control, San Diego, California, USA, 977-982.

Mijares, G. and Holland C. D., (1987) "Comments On: IMC 2. Design Procedure for Multivariable Systems", Industrial and Engineering Chemistry Research, 26, No: 3, 633-634.

Morari, M., Rivera, D. F. and Skogestad, S., (1984) "Implications of Internal Model Control for PID Controllers", Proceedings American Control Conference, San Diego, USA, 661-666.

Morari, M., (1987) "Response to Comments on: IMC 2. Design Procedure for Multivariable Systems", Industrial and Engineering Chemistry Research, 26, No. 3, 634.

Morningred, J. D., Mellichamp, D. A. and Seborg, D. E., (1987) "On-Line Adaptation of Predictive Controllers", Proceedings American Control Conference, Minneapolis, USA, 1731-1736.

Morshedi, A. M., Cutler, C. R. and Skrovanek, T. A., (1985) "Optimal Solution of Dynamic Matrix Control with Linear Programming (LDMC)", Proceedings American Control Conference, Boston, USA, 199-208.

Morshedi, A. M., (1986) "Universal Dynamic Matrix Control", Proceedings of the Third International Conference on Chemical Process Control (CPC III), (Morari, M. and McAvoy, T., Eds.), CACHE and Elsevier, Amsterdam.

Navratil, J., (1988) "Disturbance Prediction in MOCCA", M.Sc. Thesis in progress, Department of Chemical Engineering, University of Alberta, Edmonton, Alberta, Canada.

Newell, R. B., (1971) "Multivariable Computer Control of an Evaporator", Ph.D. Thesis, Department of Chemical Engineering, University of Alberta, Edmonton, Alberta, Canada.

Ogunnaike, B. A., (1983) "A Statistical Appreciation of Dynamic Matrix Control", Proceedings American Control Conference, San Francisco, USA, 1126-1131.

Ogunnaike, B. A. and Adewale, K. E. P., (1986) "Dynamic Matrix Control for Process with Time Varying Parameters", Chemical Engineering Communication, 47, 295-314.

Palazoglu, A., (1987) "Robust Stability In IMC Framework Using the Numerical Range Approach", Proceedings American Control Conference, Minneapolis, USA, 643-648.

Papadoulis, A. V. and Svoronos, S. A., (1987) "Adaptive MDC Control", Proceedings American Control Conference, Minneapolis, USA, 2038-2044.

Phillips, C. L. and Nagle Jr., H. T., (1984) Digital Control System Analysis and Design, Prentice-Hall, Englewood Cliffs, N. J., USA.

Prett, D. M. and Gillette, R. D., (1980) "Optimization and Constrained Multivariable Control of a Catalytic Cracking Unit", Proceedings of Joint Automatic Control Conference, San Francisco, USA, Paper WP5-C.

Prett, D. M. and Garcia, C. E., (1987) "Design of Robust Process Controllers", IFAC 10th World Congress on Automatic Control Pre-prints, (R. Isermann, Ed.), Munich, Federal Republic of Germany, Vol. 2, 291-296.

Quigley, V. P. and Brosilow, C., (1986) "Adaptive Horizon Adjustment for Model Predictive Control", Proceedings American Control Conference, Seattle, USA, 344-348.

Richalet, J., Rault, A., Testud, J. L. and Papon, J., (1978) "Model Predictive Heuristic Control: Application to Industrial Processes", Automatica, 14, 413-428. (Original version published in 4th IFAC Symposium on Identification and System Parameter Estimation, Tbilisi, Georgian Republic, USSR, 1976)

Richalet, J., (1980) "General Principles of Scenario Predictive Control Techniques", Proceedings of Joint Automatic Control Conference, San Francisco, USA, Paper FA9-A.

Richalet, J., (1987) "Why Predictive Control ?", IFAC 10th World Congress on Automatic Control Pre-prints, (R. Isermann, Ed.), Munich, Federal Republic of Germany, Vol. 11, 7.

Rickel, M. L., (1985) "Use of Quadratic Programming for Constrained Internal Model Control", Industrial and Engineering Chemistry: Process Design and Development, 24, No. 4, 925-936.

[illegible], M. L., Sim, T. and Cheng, C. M., (1986) "Predictive Control of a Multi-effect Evaporation System", Proceedings American Control Conference, Seattle, USA, 355-359.

Rivera, D. E., Morari, M. and Skogestad, S., (1986) "Internal Model Control: 4. PID Controller Design", Industrial and Engineering Chemistry: Process Design and Development, 25, No. 1, 252-265.

Rotta, M. A. and Marchetti, J. L., (1987) "Internal Model Control Using the Linear Quadratic Regulator Theory", Industrial and Engineering Chemistry Research, 26, No. 3, 577-581.

Rouhani, R. and Mehra, R. K., (1982) "Model Algorithmic Control (MAC): Basic Theoretical Properties", Automatica, 18, 401-414.

Smylie, B. A., (1984) "An Evaluation of Internal Model Control", M.Sc. Thesis, Department of Chemical Engineering, University of Alberta, Edmonton, Alberta, Canada.

Sripada, N. R. and Fisher, D. Grant, (1985) "Multivariable Optimal Constrained Control Algorithm (MOCCA): Part 1. Formulation and Application", Proceedings International Conference on Industrial Process Modeling and Control, Vol. 1, Hangzhou, China, June 1985.

Sripada, N. R. and Fisher, D. Grant, (1987) "Improved Least Squares Identification", submitted for publication to International Journal of Control.

Stephanopoulos, G., (1984) Chemical Process Control: An Introduction to to Theory and Practice, Prentice-Hall, Englewood Cliffs, N. J., USA.

Stephanopoulos, G. and Huang, H., (1987) "Two-Port Control System: A Generalized Predictive Controller", International Journal of Control, 45, No. 2, 617-639.

Svoronos, S. A., (1986) "Disturbance Rejection Through Model Dependent Control", Proceedings American Control Conference, Seattle, USA, 664-668.

Tabak, D. and Kuo, B. C. (1971) Optimal Control by Mathematical Programming, Prentice-Hall, Englewood Cliffs, N. J., USA.

Tung L. S., (1983). "Sequential Predictive Control of Industrial Processes", Proceedings American Control Conference, San Francisco, USA, 349-355.

Usoro, P. B. and Mehra, R. K., (1984) "Model Algorithmic Control of a Non-linear Three Phase Electric Arc Furnace", Proceedings American Control Conference, San Diego, 679-685.

Walgama, K. S., (1986) "Multivariable Adaptive Predictive Control for Stochastic Systems with Time Delays", M.Sc. Thesis, Department of Chemical Engineering, University of Alberta, Edmonton, Alberta, Canada.

Wassick, J. M. and Tummala, R. L., (1987) "Multivariable Internal Model Control Designed for a Full Scale Industrial Distillation Column", Proceedings American Control Conference, Minneapolis, USA, 1778-1784.

Wellons, M. C. and Edgar, T. F., (1987) "The Generalized Analytical Predictor", Industrial and Engineering Chemistry Research, 26, No. 8, 1523-1536.

Wolfgang, M. G., Lee, P. L. and Callaghan, P. J., (1986) "Practical Robust Predictive Control of a Heat Exchange Network", Internal Report, Department of Chemical Engineering, University of Queensland, St. Lucia, Australia.

Wong, P. M., Taylor, P. A. and Wright, J. D., (1986) "An Experimental Evaluation of Saturation Algorithms for Advanced Digital Controllers", Report #1007, Department of Chemical Engineering, McMaster University, Hamilton, Ontario, Canada.

Yuan, P. and Seborg, D. E., (1986) "Predictive Control Using an Observer for Load Estimation", Proceedings American Control Conference, Seattle, USA, 669-675.

Zafiriou, E. and Morari, M., (1987a) "Setpoint Tracking vs. Disturbance Rejection for Stable and Unstable Processes", Proceedings American Control Conference, Minneapolis, USA, 649-651.

Zafiriou, E. and Morari, M., (1987b) "Robust H_2-Type IMC Controller Design Via the Structured Singular Value", IFAC 10th World Congress on Automatic Control Pre-prints, (R. Isermann, Ed.), Munich, Federal Republic of Germany, Vol. 2, 275-280.

10 Appendix A

Model Formulation For Unstable (Integrating) Processes By Differentiation

For a stable process, the following equation (equation (3.7)) can be used to predict the future trajectory using step response data:

$$y_m(k+i|k+i) = \sum_{j=1}^{N} a_j \Delta u(k+i-j) + a_{ss} u(k+i-N-1) \qquad \text{(A-1)}$$

where i=1,2,...,P; $P \leq N$. If the process were unstable then N and a_{ss} would no longer be finite. As a result the summation term in equation (A-1) could no longer be truncated, and an "infinite" order model would be required, e.g.

$$y_m(k+i|k+i-1) = \sum_{j=1}^{\infty} a_j \Delta u(k+i-j) \qquad \text{(A-2)}$$

A model for unstable (integrating) processes can, however, be formulated by differentiating equation (A-2) with respect to time. The simplest approximation for a first derivative is the backward difference formula:

$$\frac{dy}{dt} = \frac{y(i)-y(i-1)}{\Delta t} \qquad \text{(A-3)}$$

Now differentiating equation (A-2) on both side, using equation (A-3) whenever appropriate gives:

$$\frac{y_m(k+i|k+i)-y_m(k+i-1|k+i)}{\Delta t} = \sum_{j=1}^{\infty}\frac{(a_j-a_{j-1})}{\Delta t}\Delta u(k+i-j) \qquad \text{(A-4)}$$

Now redefine (a_j-a_{j-1}) as $\bar{a}_j$. Note that $a_1 = \bar{a}_1$ since $a_0 = 0.0$ (due to normalization). Equation (A-4) can then be written as:

$$y_m(k+i|k+i)-y_m(k+i-1|k+i) = \sum_{j=1}^{\infty}\bar{a}_j\Delta u(k+i-j) \qquad \text{(A-5)}$$

At this point, it is noted that the output response of a unstable integrating process does not come to a steady state value but to a steady state ramp change. The coefficients $\bar{a}_j$ can then be truncated to $\bar{a}_n$ after N data points, i.e. $\bar{a}_j$ = constant for j>N. Therefore equation (A-5) may be rewritten as:

$$y_m(k+i|k+i)-y_m(k+i-1|k+i) = \sum_{j=1}^{N}\bar{a}_j\Delta u(k+i-j)+\bar{a}_n\sum_{j=N+1}^{\infty}\Delta u(k+i-j) \qquad \text{(A-6)}$$

Since the output at any time interval k is, by assumption, a function only of changes in input over the past N intervals and of the input (N+1) intervals ago, equation (A-6) may be rewritten by truncating Δu at N and u at N+1:

$$y_m(k+i|k+i) - y_m(k+i-1|k+i) = \sum_{j=1}^{N} \tilde{a}_j \Delta u(k+i-j) + \tilde{a}_{ss} u(k+i-N-1) \qquad \text{(A-7)}$$

which is similar to equation (3-7) except that the left hand side represents the change in the output. The MOCCA algorithm for integrating processes can then be implemented as follows:

1. $\tilde{a}_i$ is the difference step response data (impulse response data)
2. the output trajectory is now expressed in terms of changes in output, $\Delta y_m(\cdot)$
3. setpoint changes must be implemented as changes in desired trajectory, $\Delta y_d(\cdot)$
4. the residuals in the feedback paths must also be expressed as increments
5. the control signal is still in terms of changes in input, however the controller gains are calculated based on the step response obtained in step one.

This formulation of MOCCA can be applied directly to integrating processes but may suffer from increased sensitivity to measurement noise due to the differentiation. The alternative is to stabilize the process before applying the standard MOCCA formulation.

11 Appendix B

Reduced Order State Space Model

Assume that the process can be represented by Figure B.1,

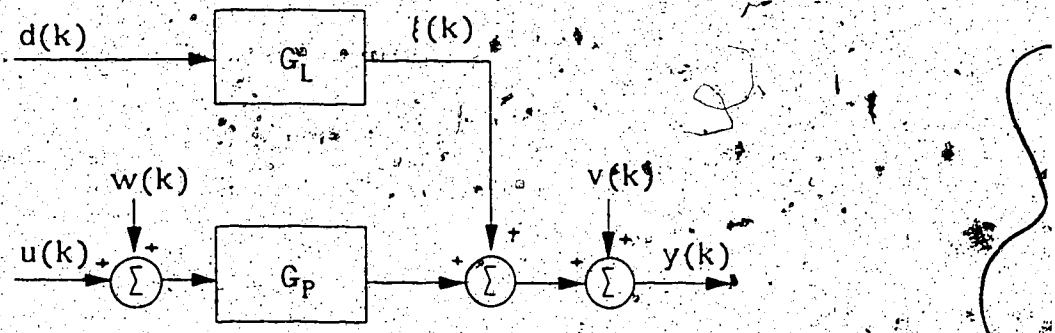

Figure B.1: Schematic Diagram of Process With Load

where w(k) is process noise, v(k) is measurement noise and the process input/output relationship, G_p, is defined by the known step response data $(a_1, a_2, \ldots, a_N, a_{ss})$ or the equivalent impulse response data $(h_1, h_2, \ldots, h_N, h_{ss})$ where $h_1 = a_1$ and $h_i = a_i - a_{i-1}$. Similarly, let the relationship between the external disturbance, d(k-1) and the output be defined by the known step response data $(b_1, b_2, \ldots, b_N, b_{ss})$ or the equivalent impulse response data $(g_1, g_2, \ldots, g_N, g_{ss})$

From the definition of the step response data (plus the assumption of superposition and linear scaling) it is possible to develop the following equations that estimate future values of the process output. The notation $y_m(k|k)$ means the estimated value of the model output at time k is calculated using input values up to and including time k-1.

Now assume that at time k an estimate of the future output trajectory $(y_m(k+i|k-1),\ i=0,1,\ldots,P)$ is known. Also assume that at time k, the input variables $\Delta u(k-1)$ and $\Delta d(k-1)$ are known. Then by definition, the prediction of the output trajectory at time k is

$$\begin{aligned} y(k|k) &= y(k+1|k-1) + a_1\Delta u(k-1) + b_1\Delta d(k-1) \\ y(k+1|k) &= y(k+2|k-1) + a_2\Delta u(k-1) + b_2\Delta d(k-1) \\ &\ldots \qquad \ldots \qquad \ldots \qquad \ldots \\ y(k+P|k) &= y(k+P+1|k-1) + a_{P+1}\Delta u(k-1) + b_{P+1}\Delta d(k-1) \end{aligned} \tag{B-1}$$

Using the past contributions of equation (3-9), the following may be written for $y_m(k+P+1|k-1)$ and $y_m(k+P|k-1)$:

$$\begin{aligned} y_m(k+P+1|k-1) = {} & a_{P+2}\Delta u(k-2) + a_{P+3}\Delta u(k-3) + \ldots \\ & + a_N\Delta u(k-N+P) + a_{ss}u(k-N+P-1) \\ & + b_{P+2}\Delta d(k-2) + b_{P+3}\Delta d(k-3) + \ldots \\ & + b_N\Delta d(k-N+P) + b_{ss}d(k-N+P-1) \end{aligned} \tag{B-2}$$

$$y_m(k+P\,|\,k-1) = a_{P-1}\Delta u(k-2) + a_{P-2}\Delta u(k-3) + \ldots$$
$$+ a_N \Delta u(k-N+P-1) + a_{ss}u(k-N+P-2)$$
$$+ b_{P-1}\Delta d(k-2) + b_{P-2}\Delta d(k-3) + \ldots \qquad \text{(B-3)}$$
$$+ b_N \Delta d(k-N+P-1) + b_{ss}d(k-N+P-2)$$

Subtracting equation (B-3) from (B-2), and collecting all the similar terms yields:

$$y_m(k+P+1\,|\,k-1) = y_m(k+P\,|\,k-1)$$
$$+ (a_{P-2} - a_{P-1})\Delta u(k-2)$$
$$+ (a_{P-3} - a_{P-2})\Delta u(k-3) + \ldots \qquad \text{(B-4)}$$
$$+ (a_N - a_{N-1})\Delta u(k-N+P-1)$$
$$- a_N \Delta u(k-N+P-1) + a_{ss}\Delta u(k-N+P-1)$$
$$+ (b_{P-2} - b_{P-1})\Delta d(k-2)$$
$$+ (b_{P-3} - b_{P-2})\Delta d(k-3) + \ldots$$
$$+ (b_N - b_{N-1})\Delta d(k-N+P)$$
$$- b_N \Delta d(k-N+P-1) + b_{ss}\Delta d(k-N+P-1)$$

Now, the difference between step response coefficients can be replaced by impulse response coefficients and the last two terms can also be combined, to give (since $h_{ss} = a_{ss} - a_N$:

$$
\begin{aligned}
y_m(k+P+1\,|\,k-1) = {} & y_m(k+P\,|\,k-1) + h_{P-2}\Delta u(k-2) \\
& + h_{P-3}\Delta u(k-3) + \ldots + h_N \Delta u(k-N+P) \\
& + h_{ss}\Delta u(k-N+P-1) + g_{P-2}\Delta d(k-2) \\
& + g_{P-3}\Delta d(k-3) + \ldots + g_N \Delta d(k-N+P) \\
& + g_{ss}\Delta d(k-N+P-1)
\end{aligned}
\qquad \text{(B-5)}
$$

Now substituting Eqn (B-5) for $y_m(k+P+1|k-1)$ in the last equation of equation (B-1) and rewriting equation (B-1) in vector/ matrix form

$$
\begin{bmatrix} y_m(k\,|\,k) \\ y_m(k+1\,|\,k) \\ \cdots \\ y_m(k+P\,|\,k) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdots & \cdots & \cdots & 0 & 1 \\ 0 & \cdots & \cdots & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} y_m(k\,|\,k-1) \\ y_m(k+1\,|\,k-1) \\ \cdots \\ \cdots \\ y_m(k+P\,|\,k-1) \end{bmatrix} \qquad \text{(B-6)}
$$

$$
+ \begin{bmatrix} 0 & \cdots & \cdots & \cdots & a_1 \\ 0 & \cdots & \cdots & \cdots & a_2 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdots & 0 & a_P \\ h_{ss} & h_N & \cdots & h_{P-2} & a_{P-1} \end{bmatrix} \begin{bmatrix} \Delta u(k-N+P-1) \\ \Delta u(k-N+P) \\ \cdots \\ \Delta u(k-2) \\ \Delta u(k-1) \end{bmatrix}
$$

$$
+ \begin{bmatrix} 0 & \cdots & \cdots & \cdots & b_1 \\ 0 & \cdots & \cdots & \cdots & b_2 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdots & 0 & b_P \\ g_{ss} & g_N & \cdots & g_{P-2} & b_{P-1} \end{bmatrix} \begin{bmatrix} \Delta d(k-N+P-1) \\ \Delta d(k-N+P) \\ \cdots \\ \Delta d(k-2) \\ \Delta d(k-1) \end{bmatrix}
$$

Note: d(k) is any measured or known process input signal that affects $y_m(.)$ in the manner defined by the step response data (b_1, b_2, ..., b_N, b_{ss}). Typically d(.) is a measured input (disturbance). If d is non-zero then the

controller compensates for this input by using feedfoward control based on the coefficient matrix in equation (B-6). Equation (B-6) is equivalent to equation (3-40) if P=N since $h_{P,2} \ldots h_{n} \rightarrow 0$ and $a_{P,1} \rightarrow a_{n}$ and the matrix becomes a column vector (also applies to the last term).

12 Appendix C

Closed Loop Observer Derivation

Consider a system represented by equations (3-40) and (3-41):

$$X(k) = \Phi X(k-1) + \Theta \Delta U(k-1) \qquad \text{(C-1)}$$

$$Y(k) = H X(k) \qquad \text{(C-2)}$$

We would like to design a state observer to estimate the states X(k). In equations (C-1) and (C-2) the matrices Φ, Θ and vector H are known, the signal $Y(k) = y_m(k)$ can be measured and the $\Delta U(k-1)$ sequence is known since we generate it. A transfer function approach in designing the observer (Phillips and Nagle Jr., 1984) will be used. Taking the z transform of equation (C-1) yields:

$$zX(z) = \Phi X(z) + \Theta \Delta U(z)$$

and solving for X(z)

$$X(z) = [zI - \Phi]^{-1} \Theta \Delta U(z) \qquad \text{(C-3)}$$

Since the observer has two inputs, Y(k) and $\Delta U(k)$, we can write the observer equation as:

$$\hat{X}(k) = F\hat{X}(k-1) + KY(k) + H' \Delta U(k-1) \qquad \text{(C-4)}$$

where F, K and H' are the parameters to be designed. Solving equation (C-4) for $\hat{X}(z)$ gives:

$$z\hat{X}(z) = F\hat{X}(z) + KzY(z) + H' \Delta U(z)$$

$$\hat{X}(z) = [zI - F]^{-1}\left[KzY(z) + H\,\Delta U(z)\right] \qquad \text{(C-5)}$$

Taking the z transform of equation (C-2) yields:

$$Y(z) = HX(z) \qquad \text{(C-6)}$$

Substituting equation (C-6) into equation (C-5) yields:

$$\hat{X}(z) = [zI - F]^{-1}\left[KHzX(z) + H\,\Delta U(z)\right] \qquad \text{(C-7)}$$

Substituting equation (C-3) into equation (C-7) yields:

$$\hat{X}(z) = [zI - F]^{-1}\left[KHz[zI - \Phi]^{-1}\Theta + H\right]\Delta U(z) \qquad \text{(C-8)}$$

The criterion in the transfer function matrix design approach is that the transfer function matrix from $\Delta U(z)$ to $\hat{X}(z)$ be the same as $\Delta U(z)$ to X(z). Therefore, comparing equations (C-3) and (C-8):

$$(zI - \Phi)^{-1}\Theta = \left[(zI - F)^{-1}\right]\left[KHz(zI - \Phi)^{-1}\Theta + H\right]$$

or

$$\left[I - (zI - F)^{-1}KHz\right](zI - \Phi)^{-1}\Theta = (zI - F)^{-1}H \qquad \text{(C-9)}$$

Equation (C-9) can be rewritten as:

$$(zI - F)^{-1}[zI - (F + KHz)](zI - \Phi)^{-1}\Theta = (zI - F)^{-1}H \qquad \text{(C-10)}$$

$$(zI - \Phi)^{-1}\Theta = [zI - (F + KHz)]^{-1}H$$

Therefore choosing $H = \Theta$ and $\Phi = F + KHz$, we could rewrite the observer (equation (C-4)) as:

$$\hat{X}(k) = (\Phi - KHz)\hat{X}(k-1) + KY(k) + \Theta\Delta U(k-1) \qquad \text{(C-11)}$$

which satisfies the transfer function matrix design criterion. Note that here only K, the observer gain is unspecified (the gain in this particular case is a column vector).

Now, consider the errors in the state estimation process. Defining the error vector e(k) as:

$$\hat{e}(k) = X(k) - \hat{X}(k) \tag{C-12}$$

Then from equations (C-1), (C-2) and (C-11), equation (C-12) can be rewritten, after some algebraic manipulation, as:

$$\begin{aligned}\hat{e}(k) &= \Phi X(k-1) - KHzX(k-1) - (\Phi - KHz)\hat{X}(k-1) \\ &= (\Phi - KHz)(X(k-1) - \hat{X}(k-1)) \\ &= (\Phi - KHz)\hat{e}(k-1)\end{aligned} \tag{C-13}$$

Equation (C-13) can be rewritten as:

$$[zI - (\Phi - KHz)]\hat{e}(k) = 0.0 \tag{C-14}$$

or in a more familiar form

$$[zI - [I + KH)]^{-1}\Phi]\hat{e}(k) = 0.0 \tag{C-15}$$

Hence the error dynamic has a characteristic equation given by the following determinant:

$$|zI - [I - KH]^{-1}\Phi| = 0.0 \tag{C-16}$$

Consider now the observer state equation (equation (C-11). All the matrices in the equation are determined by the plant equations except K. However the observer design criterion is satisfied independent of the choice of K. In

choosing the values of K, the fact that K determines the error dynamics, as shown in equation (C-16) is used. Using the characteristic equation, the gain vector, K, can be chosen such that the observer has the desired dynamic response and is asymptotically stable, i.e. the roots of equation (C-16) lie within the unit circle. Note that the solution of equation (C-16) involves solving for the roots of a polynomial with order equal to number of states, e.g. order N+1 for the full order state space model. Hence the use of a reduced order state space model would reduce the complexity of the closed loop observer. Thus an observer can be designed once an appropriate characteristic equation for the error dynamic is specified by the user.

Sources of error in state estimation are 1) model process mismatch, 2) choice of initial conditions and 3) plant and measurement noise. The first source of error can only be eliminated by choice of a better model. The second source of error, initial conditions, requires that the observer be stable and drive the error to zero as time increases. The third source, noise, will prevent the error from going to zero due to the excitation (i.e. noise). A stochastic observer such as Kalman filter can be designed to provide optimal estimates for the stochastic case.

An alternative interpretation of the choice of K can be found by rewriting equation (C-11) as:

$$\hat{X}(k) = \Phi\hat{X}(k-1) + \Theta\Delta U(k-1) + K[Y(k) - H_z\hat{X}(k-1)] \qquad \text{(C-17)}$$
$$= \Phi\hat{X}(k-1) + \Theta\Delta U(k-1) + K[Y(k) - H\hat{X}(k)]$$

or in terms of MOCCA's variables, the process output, $y_p(k)$ and it's corresponding estimate, $y_m(k|k)$:

$$\hat{X}(k) = \Phi\hat{X}(k-1) + \Theta\Delta U(k-1) + K\left[y_p(k) - y_m^*(k|k)\right] \qquad \text{(C-18)}$$

The block diagram of equation (C-18) is shown in Figure C.1

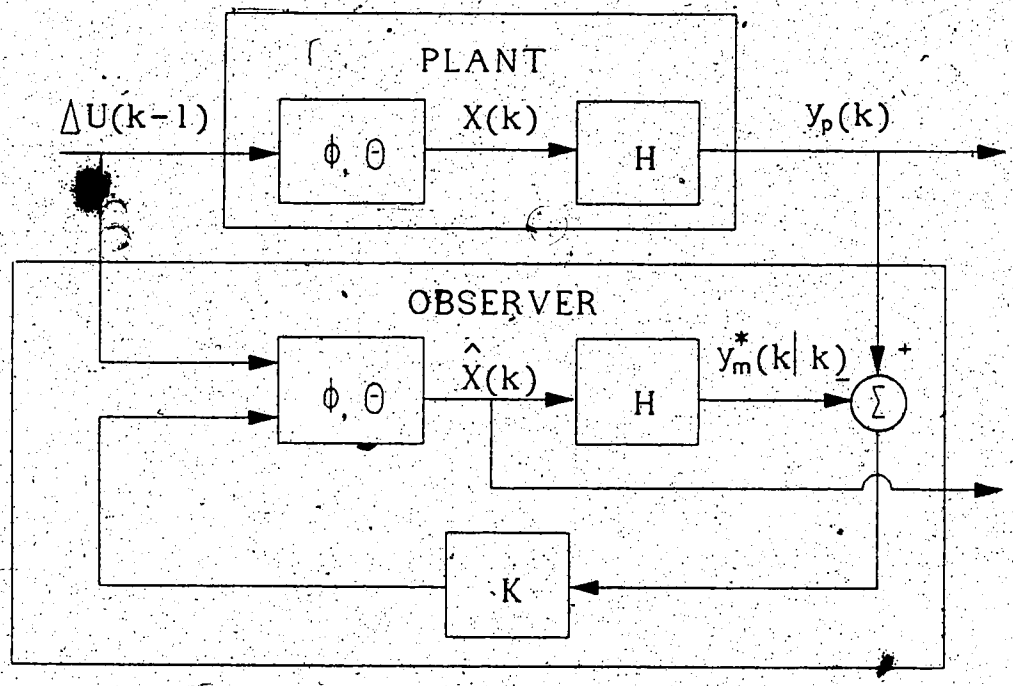

Figure C.1: Alternative View of the Observer

From Figure C.1, if the output of the observer and of the plant are approximately equal (i.e. the model is good), there is little effect from the feedback through K. Therefore $\hat{X}(k)$ is determined principally by $\Delta u(k)$. However if $\hat{X}(k)$ differs significantly from X(k), then the effect of K

is much more important and the measurement y(k) is more important than $\Delta u(k)$ in determining $\hat{x}(k)$. Thus we can view the choice of K as related to the relative importance that we attach to the effects of $\Delta u(k)$ and to the effects of the measured output on $\hat{x}(k)$. In summary, the choice of K can be viewed as a feedback correction to the state estimate (Figure C.1) to account for modeling errors and disturbances. If the errors are significant, K should be relatively large. However, if the measurement, y(k) is not reliable then K should be relatively small. The value of K can be checked via simulation of the system in Figure C.1. The use of an observer accomplishes the model based prediction function of MOCCA.

13 Appendix D

Alternate Feedback Path Analysis

Richalet et. al. (1978) in their MPHC algorithm use a different feedback path than MOCCA. They set $y_d(k)$ equal to the current measured output $y_p(k)$ at every control interval and generate a new setpoint trajectory using a first order filter so that $y_d(k+i)$ moves exponentially from $y_p(k)$ to the new setpoint. An analysis of this feedback path will be performed here. For a single input single output system, the feedback path may be written as:

$$y_d(k+i) = \beta y_d(k+i-1) + (1.0-\beta) y_{sp}(k) \qquad \text{(D-1)}$$

$$y_d(k) = y(k)$$

Equation (D-1) may be rewritten for i=1,2,...

$$i=1 \quad y_d(k+1) = \beta y_d(k) + (1.0-\beta) y_{sp}(k) \qquad \text{(D-2)}$$

$$= \beta y(k) + (1.0-\beta) y_{sp}(k)$$

$$i=2 \quad y_d(k+2) = \beta y_d(k+1) + (1.0-\beta) y_{sp}(k) \qquad \text{(D-3)}$$

Substituting equation (D-2) into equation (D-3) yields:

$$y_d(k+2) = \beta^2 y(k) + (\beta+1.0)(1.0-\beta) y_{sp}(k) \qquad \text{(D-4)}$$

Similarly for i=3,

$$y_d(k+3) = \beta^3 y(k) + (\beta^2+\beta+1.0)(1.0-\beta) y_{sp}(k) \qquad \text{(D-5)}$$

or for i=1,2,...

$$y_d(k+i) = \beta^i y(k) + \left[\sum_{j=1}^{i-1} \beta^j\right](1.0-\beta)y_{sp}(k) \qquad \text{(D-6)}$$

Now consider the case where there is no feedback of the measured output to the setpoint filter where $\bar{y}_d(k) = \bar{y}_d(k+1|k-1)$, i.e. the trajectory starts at where it expects (i.e. perfect control) the process to be as opposed to where the process is. The notation $\bar{y}$ is to distinguish between the two cases. As in the above, successive substitution yields for i=1,2,...

$$\bar{y}_d(k+i) = \beta^i \bar{y}_d(k+1|k-1) + \left[\sum_{j=1}^{i-1} \beta^j\right](1.0-\beta)y_{sp}(k) \qquad \text{(D-7)}$$

The block diagram for equation (D-6) is shown in Figure D.1.

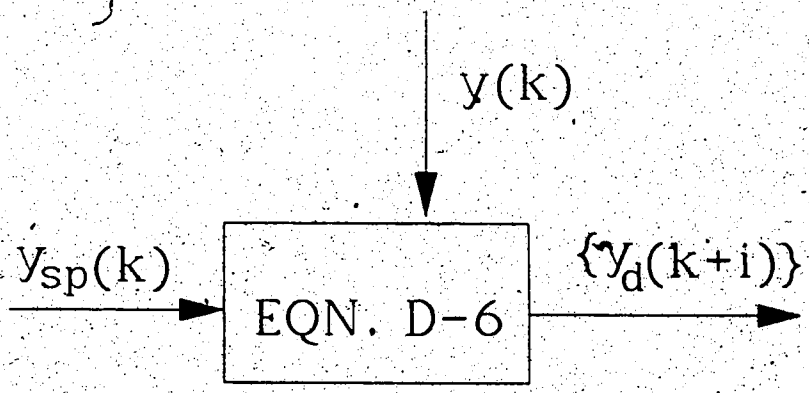

Figure D.1: Setpoint Filter in MPHC

Figure D.2 is the block diagram of equation (D-7). The block with the question mark (?) indicates a proposed block that will make Figure D.1 equivalent to Figure D.2.

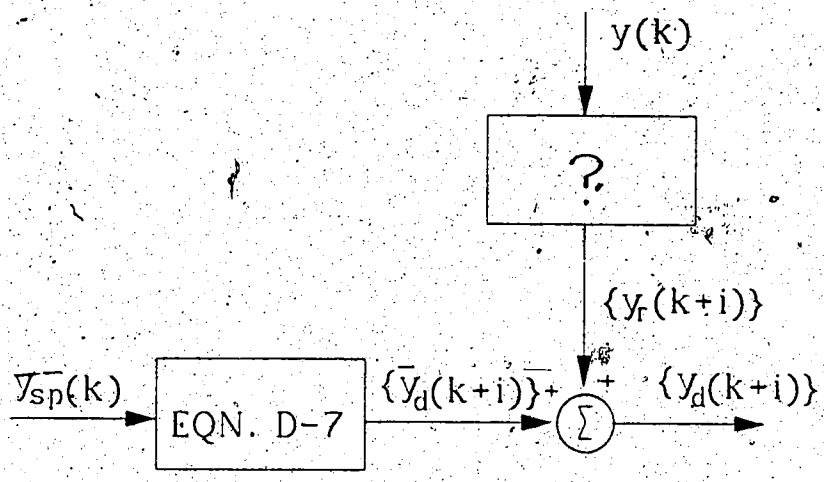

Figure D.2: Proposed Equivalent of Figure D.1

For the two block diagrams to be equivalent, the outputs in Figures D.1 and D.2 must be equal, i.e.

$$\{y_r(k+i)\} + \{\bar{y}_d(k+i)\} = \{y_d(k+i)\} \qquad \text{(D-8)}$$

Substituting equations (D-6) and (D-7) into equation (D-8):

$$\{y_r(k+i)\} = \{y_d(k+i)\} - \{\bar{y}_d(k+i)\} \qquad \text{(D-9)}$$

$$= \beta' y(k) + (1.0-\beta)\left[\sum_{j=0}^{i-1}\beta^j\right] y_{sp}(k)$$

$$-[\beta' \bar{y}_d(k+1\,|\,k-1) + (1.0-\beta)\left[\sum_{j=0}^{i-1}\beta^j\right] y_{sp}(k)$$

$$= \beta'[y(k) - \bar{y}_d(k+1\,|\,k-1)]$$

Figure D.3 shows the block diagram corresponding to equation (D-9).

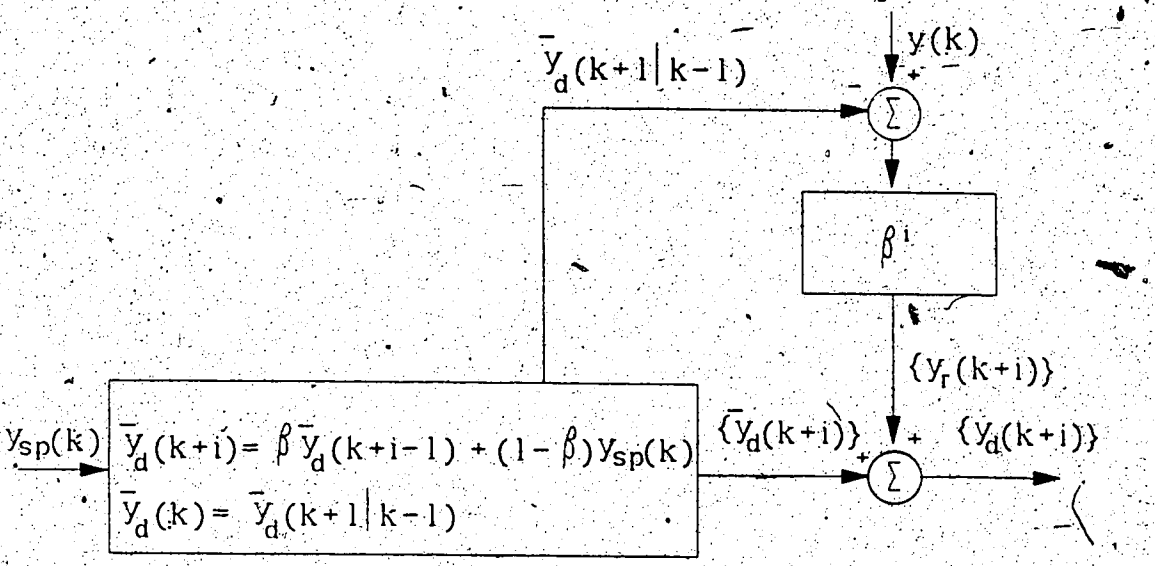

Figure D.3: Alternative View of Figure D.1

Hence, the "filter" is $\beta^i, i = 1,\ldots,P$ acting on the difference between the actual (current) and previous desired output. An alternate way of drawing Figure D.3 can be obtained by rewriting $\{y_r(k+i)\}$. Defining $y_r(k) = y(k) - \bar{y}_d(k+1|k-1)$ (i.e. the error between where the process is, and where the process should be), then equation (D-9) can be rewritten as:

$$y_r(k+i) = \beta^i y_r(k) \qquad \text{(D-10)}$$

Writing equation (D-10) for i=1,2,...

$$i=1 \quad y_r(k+1) = \beta y_r(k) \qquad \text{(D-11)}$$

$$i=2 \quad y_r(k+2)=\beta^2 y_r(k)=\beta\left(\beta y_r(k)\right)=\beta y_r(k+1) \qquad \text{(D-12)}$$

Successive substitution yields the following equation for i=1,2,...

$$y_r(k+i)=\beta y_r(k+i-1) \qquad \text{(D-13)}$$

$$y_r(k)=y(k)-\bar{y}_d(k+1\,|\,k-1)$$

Figure D.4 shows an alternative view of Figure D.3:

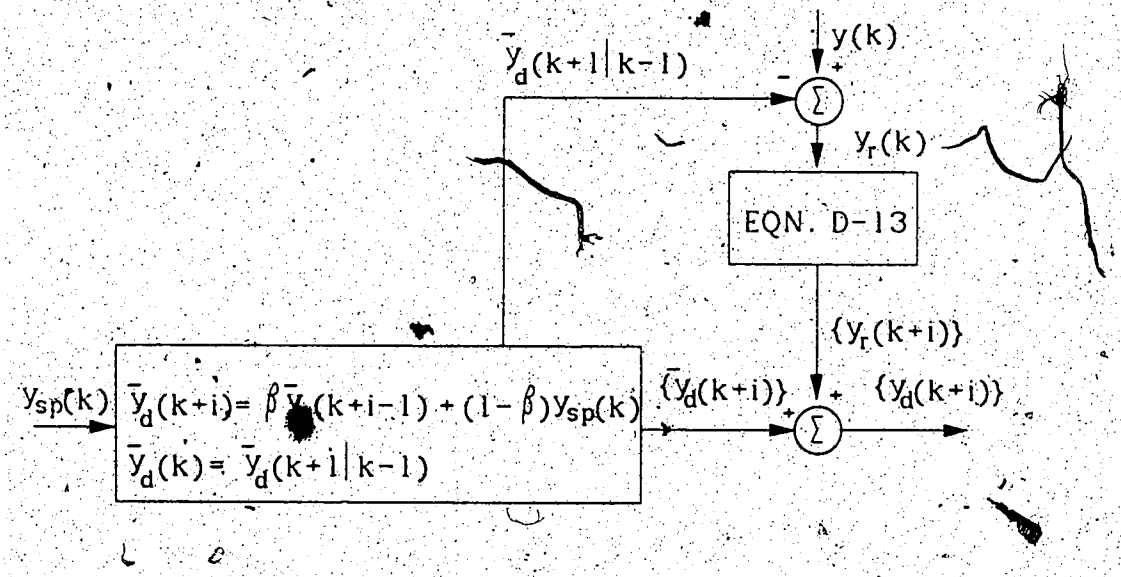

Figure D.4: Alternative View of Figure D.3

Equation (D-13) represents a first order filter with zero desired final output. Hence Richalet's et. al. MPHC feedback path with filter is equivalent to an exponential discounting filter acting on the residual.

A feedback path similar to that of MPHC can be constructed in the MOCCA block diagram (compare Figure D.5 versus Figure 2.1), by replacing the filter and disturbance

predictor blocks by a discounting filter block acting on the residual, $y_r(k) = y_p(k) - y_m(k)$. The residual term includes model process mismatch which was not included in the residual term for MPHC.

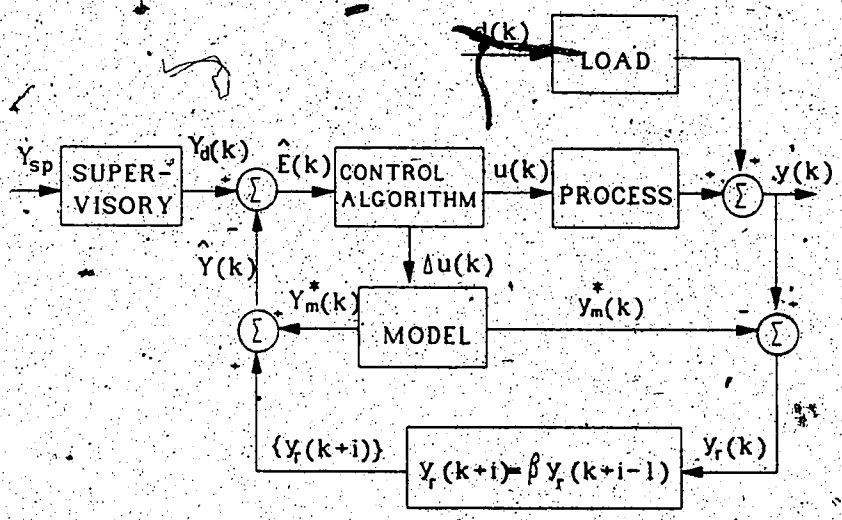

Figure D.5: MOCCA With Exponentially Discounting Filter

14 Appendix E

Feedforward Control Using Model Predictive Control

In order to facilitate the implementation of constraints, the feedforward and predictive controller calculations are combined into one calculation (i.e. one constrained optimization problem). Figure E.1 shows a schematic of the simplified MOCCA system where G_P and G_L represent the process and load respectively, G_{PM} and G_{LM} represent the model (step response) of the process and load respectively and G_C is the predictive controller transfer function.

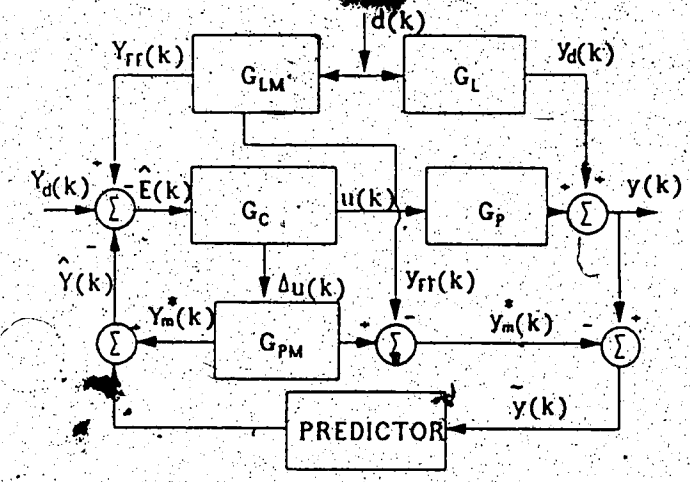

Figure E.1: Schematic Diagram of MOCCA With Feedfoward

The transfer function relationships between y and d can be written as follows:

$$y = G_P \Delta u(k) + G_L d(k) \qquad \text{(E-1)}$$

$$\Delta u(k) = G_c \hat{E}(k) = G_c\left[Y_d(k) - Y_{ff}(k) - \hat{Y}(k)\right] \qquad \text{(E-2)}$$

$$Y_{ff}(k) = G_{LM} d(k) \qquad \text{(E-3)}$$

In equation (E-2) only Y_{ff} is a function of d. Dropping the other two terms for convenience, equation (E-1) can be rewritten as:

$$y(k) = \left(G_L - G_P G_c G_{LM}\right) d(k) \qquad \text{(E-4)}$$

Since the MOCCA predictive controller approximates a model inverse controller in the ideal case, that is $G_c = G_P^{-1}$, equation (E-4) reduces down to:

$$y(k) = \left(G_L - G_{LM}\right) d(k) \qquad \text{(E-5)}$$

Now assuming perfect modeling, the desired effect of the feedforward controller, i.e. y=0.0 is achieved. The presence of model process mismatch will deteriorate the performance of the feedforward control but will not affect the implementation. Mismatch between the process and process model as well as mismatch between load and load model will appear in the feedback residual $\hat{y}(k)$.

In the absence of measurable disturbances, an "inferential feedforward" controller can still be implemented in MOCCA. The effect of the disturbance on the current output can be estimated (residual) in the feedback path. An estimate of the future effect of the disturbance on the output can be generated using a disturbance predictor (cf. Figure E.1) and used in the error trajectory calculation.

which is equivalent to the feedforward contribution when the disturbance is measurable. In this case, the disturbance predictor is essentially a model of the load disturbance with the residual being the disturbance term.

15 Appendix F

One Step Ahead Predictive Control

If the prediction horizon, P and the control horizon, M are both set equal to one, then the resulting control law becomes a one step ahead predictive controller, i.e. the predicted value is forced to match the desired output of the system after one sampling period. With P=M=1, equations (3-8) and (3-9) can be rewritten as:

$$y_m(k+1|k+1) = y_m^*(k+1|k) + a_1 \Delta u(k) \tag{F-1}$$

$$y_m^*(k+1|k) = \sum_{j=2}^{N} a_j \Delta u(k-j+1) + a_{ss} u(k-N) \tag{F-2}$$

To include the effect of disturbances and model mismatch, a corrected prediction based on the assumption that the residual at time k+1 is equal to the residual at time k can be defined as (cf. Figure 2.1):

$$\hat{y}(k+1) = y_m^*(k+1|k) + [y(k) - y_m^*(k|k)] \tag{F-3}$$

where

$$y_m^*(k|k) = \sum_{j=1}^{N} a_j \Delta u(k-j) + a_{ss} u(k-N-1) \tag{F-4}$$

Substituting equations (F-2) and (F-4) into (F-3) and collecting terms yields:

$$\hat{y}(k+1) = y(k) + (a_2 - a_1)\Delta u(k-1) \qquad \text{(F-5)}$$
$$+ (a_3 - a_2)\Delta u(k-2) + \ldots + (a_{n} - a_N)\Delta u(k-N)$$

Defining $h_j = (a_j - a_{j-1})$, equation (F-5) can be rewritten as:

$$\hat{y}(k+1) = y(k) + \sum_{j=2}^{N+1} h_j \Delta u(k-j+1) \qquad \text{(F-6)}$$

Solving equation (F-1) by setting $y_d(k+1|k) = y_m(k+1|k)$, as the desired output and replacing the term $y_m(k+1|k)$ by the corrected prediction (equation (F-6)) yields:

$$\Delta u(k) = \frac{1}{a_1}\left[y_d(k+1) - y(k) - \sum_{j=2}^{N+1} h_j \Delta u(k-j+1) \right] \qquad \text{(F-7)}$$

Equation (F-7) is the one step ahead predictive control when there is no delay. For the case where $\lambda \neq 0$ where λ is the time delay of the process, equation (F-6) can be rewritten as (using equations (3-82) and (3-83)):

$$\hat{y}(k+1) = y(k) + \sum_{j=2}^{N+1} a_j \Delta u(k-j+1) - \sum_{j=1}^{N+1} a_j \Delta u(k-j-\lambda) \qquad \text{(F-8)}$$

where $a_{N+1} = a_{n}$. The control law, equation (F-7), then becomes:

$$\Delta u(k) = \frac{1}{a_1}[y_d(k+1+\lambda) - y(k) - \sum_{j=2}^{N+1} a_j \Delta u(k-j+1) \qquad \text{(F-9)}$$
$$+ \sum_{j=1}^{N+1} a_j \Delta u(k-j-\lambda)]$$

Multiplying equation (F-9) by a_1 and combining the term $a_1\Delta u(k)$ into the summation, equation (F-9) can be rearranged into:

$$y_d(k+1+\lambda)-y(k)=\sum_{j=1}^{N-1}a_j\Delta u(k-j+1)+\sum_{j=1}^{N-1}a_j\Delta u(k-j-\lambda) \qquad \text{(F-10)}$$

The setpoint filter for SISO system with P=M=1 and delay is given by:

$$y_d(k+1+\lambda)=\beta y_d(k+\lambda)+(1.0-\beta)y_{sp}(k) \qquad \text{(F-11)}$$

Taking the z-transform of equations (F-10) and (F-11) and combining them to gives the following closed loop transfer function (after some manipulation):

$$\frac{y(z)}{y_{sp}(z)}=\frac{(1-\beta)}{(1-\beta z^{-1})}\left(1+\frac{1-z^{-\lambda-1}G_M(z)}{\quad}\right) \qquad \text{(F-12)}$$

where $G_M(z)$ is the delay free process model given by:

$$G_M(z)=\sum_{j=1}^{N-1}a_j z^{-j+1} \qquad \text{(F-13)}$$

Assuming no model process mismatch, the delay in the process along with the discretization delay can be extracted,

$$G_P(z)=z^{-\lambda-1}G_m(z) \qquad \text{(F-14)}$$

Equation (F-14) can be substituted into (F-12) and with some algebraic manipulation, the closed loop transfer function can be written as:

$$\frac{y(z)}{y_{sp}(z)} = \frac{(1-\beta)}{(1-\beta z^{-1})} z^{-\lambda+1} \qquad (F-15)$$

When $\beta = 0.0$ (i.e. with no setpoint filtering), the result is a $\lambda+1$ step ahead predictive control law where the process output is forced to match the desired output of the system after $\lambda+1$ sampling interval (Stephanopoulos, 1984). For $\beta \neq 0.0$ then the result is a perfect model following controller since, the first term in equation (F-15) represents a first order filter (i.e. equation (F-11)) and the second term represents the known delay of the process. In the case where $P = M \neq 1$ (i.e. multi-step predictive control), the control law defined by equations (F-7) and (F-9) is still true provided that only $\Delta u(k)$ is implemented at time k. This is due to the lower triangular form of the dynamic matrix A_2 when P=M, since the inverse of a lower triangular matrix is still a lower triangular matrix (Bronson, 1970). That is $\Delta u(k)$ is calculated from the first row of the inverted matrix (which has a non-zero in the first column and zeroes in the rest) multiplied by the error vector.

A similar analysis can be performed on the closed loop version of MPHC (Mehra et al., 1981) using a MPHC setpoint filter (with feedback, cf. equation (3-74)). The resulting closed loop transfer function is given by

$$\frac{y(z)}{y_{sp}(z)} = \frac{(1-\beta)z^{-\lambda-1}}{\left(1-\beta z^{-\lambda-1}\right)} \qquad \text{(F-16)}$$

When $\beta = 0.0$ the closed loop transfer function is the same as in MOCCA. For $\beta \neq 0.0$ perfect model following of the MPHC filter is achieved.

16 Appendix G

Conditioning of Dynamic Matrix

The following discussion on ill-conditioning of matrices were taken from Applied Numerical Analysis by Gerald and Wheatley, 1984 (pages 114-124) and Practical Optimization by Gill et al., 1981 (pages 27-30). We will start by reviewing some of the mathematics involved in the discussion of ill-conditioning and the practical implications of ill-conditioning in the solution of system of linear equations. Parallel conclusions with respect to MOCCA can then be drawn from these discussions.

A measure of the magnitude of a vector or matrix can be obtained by using the norm. There are three basic definition of norms for vectors:

$$\|X\|_p = \left[\sum_{i=1}^{n} |X_i|^p\right]^{1/p} \qquad (G\text{-}1)$$

where p=1 is the sum of magnitudes

p=2 is the Euclidean norm

$p = \infty$ is the maximum magnitude norm.

Equivalent definitions of norms can be defined for matrices. The norm-2 of a matrix (also known as the spectral norm) is related to the eigenvalues of the matrix and provides the "tightest" measure of the magnitude of a matrix. It is also the most difficult to calculate. The spectral norm of a

matrix, A is given by the square root of the largest eigenvalues of A^TA. The Frobenius norm provides a easier way of calculating the magnitude of a matrix. The norm is given by:

$$||A||_F = \left[\sum_{i=1}^{M}\sum_{j=1}^{N} a_{ij}^2\right]^{1/2} \qquad \text{(G-2)}$$

The condition of a matrix can be defined using the condition number (using perturbation theory):

$$\text{cond}(A) = ||A||\,||A^{-1}|| \qquad \text{(G-3)}$$

If the cond(A) is large, then the exact solution may change substantially by even a small change in the data. The matrix A is said to be ill-conditioned if cond(A) is large and well conditioned if cond(A) is small. The above statements are made with respect to the solution of a linear system Ax=b. The following observations were made by Gerald and Wheatley, 1984 regarding the solution of a linear system:

1. The accuracy of a solution can be dependent on the algorithm used to invert the matrix. For example, Gaussian elimination with pivoting gives better accuracy than without pivoting.
2. Sometimes the problem itself may be very sensitive to the effects of small errors. This is usually indicated by small diagonal values of the triangularized system. This

indicates that the determinant of the original coefficient matrix tends to be small and since a singular matrix occurs when the determinant is zero, we can say that the small determinant indicates "near-singularity".

3. In actual application, the coefficient matrix of A may be obtained experimentally, hence the coefficients are only as precise as the measurements. An analysis can be performed in this case. The problem is a linear system, Ax=b. Assume that the error in the measurement, E, is such that the following is solved instead of the true Ax=b.

$$(A+E)\bar{x}=b \tag{G-4}$$

where $\bar{x}$ represent the solution of the perturbed system and A represents the true coefficients. Let $\bar{A}=A+E$, then we want to know how large $x-\bar{x}$ is. Since $Ax=b$ and $\bar{A}\bar{x}=b$ then

$$\begin{aligned} x=A^{-1}b &= A^{-1}(\bar{A}\bar{x})=A^{-1}(A+\bar{A}-A)\bar{x} \\ &=\left[I+A^{-1}(\bar{A}-A)\right]\bar{x} \\ &=\bar{x}+A^{-1}(\bar{A}-A)\bar{x} \end{aligned} \tag{G-5}$$

Since $\bar{A}-A=E$, then

$$x-\bar{x}=A^{-1}E\bar{x} \tag{G-6}$$

$$\|x-\bar{x}\|\le\|A^{-1}\|\|E\|\|\bar{x}\|=\|A^{-1}\|\|A\|\|E\|\frac{1}{\|A\|}\|\bar{x}\| \tag{G-7}$$

$$\frac{||x-\bar{x}||}{||\bar{x}||} \leq cond(A)||E|\frac{|}{||A||} \qquad \text{(G-8)}$$

The error of the solution relative to the norm of the computed solution can be as large as the relative error in the coefficients of A multiplied by the condition number. Therefore if the coefficients of A are known to only four digit precision and the condition number is 1000, then the computed vector x may have only one digit accuracy. In fact if

$$\frac{||x-\bar{x}||}{||\bar{x}||} \text{is} 10^{-p} \qquad \text{(G-9)}$$

then $\bar{x}$ is probably correct to only p-[illegible]ts.

Now we can draw parallel conclusions for MOCCA from the above observations.

1. If the dynamic matrix is ill-conditioned (i.e. the condition number is large) then the solution, in this case $\Delta u(k)$ will be sensitive to small changes in the error vector.
2. Since the coefficients of the dynamic matrix need to be measured in most cases, then the argument in observation 3 holds. Therefore noisy and incorrect measurements will likely cause trouble due to the ill-conditioned dynamic matrix.

3. Observation 2 is also relevant to MOCCA, where the inverse will contain small values, if not zero, i.e. since the pseudo inverse is taken, A_2 can have row of zeros and still be invertible and naturally the inverse will have zeros.

Ill-conditioning can be avoided by proper treatment of the data by scaling or normalization. Future work could include the use of scaling matrices that minimizes the condition number (Sripada and Fisher, 1987) to solve the weighted least square problem in MOCCA. It has also been suggested that the calculation of the pseudo inverse (cf. equation (4-15)) using normal algorithms (e.g. Gaussian elimination) can caused stability problems with regard to propagation of data errors and uncertainty. To avoid this problem, algorithms for finding the pseudo inverse using orthogonal or elimination matrices (e.g. QR factorization or singular value decomposition) should be used since it leaves the Euclidean length of the vector invariant (e.g. the Euclidean length used in least square problem is preserved by orthogonal matrices Lawson & Hanson, 1974, Gill et al., 1981).

17 Appendix H

SISO Weighted Least Square Example Translation and Simulation Session

The following session log provides a simple weighted least square controller design using the MOCCA translator/simulator software package. Simulation of the resulting controller was performed using the simulator option. The purpose of including this example is to show that the implementation of MOCCA is straightforward and is defined strictly in terms of "control terminology" e.g. the user does not need to know how to formulate the weighted least square, linear programming or quadratic programming problems.

```
Slog Output: Kian Lim, KLIM, Job#=2688, Host=UALTAMTS, 12:13:02 Fri Dec 18/87
#   $.03,    $ 737
#R M_MOCCA RUN
#12:13:08
 ********************************************
 **                                        **
 **            WELCOME TO MOCCA            **
 **       TRANSLATOR AND SIMULATOR         **
 **              PROGRAM                   **
 **          VERSION: 87/07/20             **
 **                                        **
 ********************************************
THE FOLLOWING OPTIONS ARE AVAILABLE:
1  TRANSLATOR   TRANSLATE MOCCA PROBLEM
                INTO CONTROL ALGORITHM
2  SIMULATOR    SIMULATE CONTROL ALGORITHM
                OBTAINED FROM TRANSLATOR
PLEASE SPECIFY OPTION NUMBER
1
YOU HAVE SPECIFIED OPTION NUMBER 1
WOULD YOU LIKE TO CHANGE THIS ?
ANSWER YES OR NO
NO
 ********************************************
 **                                        **
 ** MOCCA TRANSLATOR PROGRAM OPTIONS       **
 **                                        **
 ** 1. WEIGHTED LEAST SQUARE               **
 **                                        **
 ** 2. INTERVAL PROGRAMMING (NOT READY)    **
 **                                        **
 ** 3. LINEAR PROGRAMMING                  **
 **                                        **
 ** 4. QUADRATIC PROGRAMMING               **
 **                                        **
 ********************************************
PLEASE SPECIFY OPTION NUMBER
1
YOU HAVE SPECIFIED OPTION NUMBER 1
WOULD YOU LIKE TO CHANGE THIS ?
ANSWER YES OR NO
NO
PLEASE SPECIFY STEP RESPONSE DATA FILE NAME
FILE NAME MUST START WITH AN ALPHABET AND CAN HAVE UP TO 6 CHARACTERS
STEPA1
PLEASE SPECIFY CONTROLLER PARAMETER FILE NAME
FILE NAME MUST START WITH AN ALPHABET AND CAN HAVE UP TO 6 CHARACTERS
-A1
PLEASE SPECIFY DESIGN INFORMATION FILE NAME
FILE NAME MUST START WITH AN ALPHABET AND CAN HAVE UP TO 6 CHARACTERS
-A2
THE FOLLOWING DATA ARE REQUIRED FOR THE
TRANSLATOR/SIMULATOR INFORMATION FILE.
YOU MAY LEAVE IT BLANK BY PRESSING THE
RETURN KEY
NAME OF ENGINEER
K. LIM
DATE:
DECEMBER 18, 87
YOU ARE ALLOWED 3 LINES OF 80 CHARACTERS EACH
TO DESCRIBE THE SYSTEM YOU ARE DESIGNING
LINE 1
WEIGHTED LEAST SQUARE DESIGN EXAMPLE
LINE 2
SISO SYSTEM: GAIN=1.0, DELAY=0, TIME CONSTANT=10.0
LINE 3

 ********************************************
 *  WEIGHTED LEAST SQUARE CONTROLLER DESIGN  *
 ********************************************
SYSTEM HAS 3 INPUTS AND 3 OUTPUTS
DO YOU WANT TO CHANGE THE SYSTEM SIZE ?
ANSWER YES OR NO
YES
SPECIFY THE NUMBER OF INPUT VARIABLES
1
SPECIFY THE NUMBER OF OUTPUT VARIABLES
1
SYSTEM HAS 1 INPUTS AND 1 OUTPUTS
DO YOU WANT TO CHANGE THE SYSTEM SIZE ?
ANSWER YES OR NO
NO
OUTPUT 1 PREDICTION HORIZON = 10
DO YOU WANT TO CHANGE THE PREDICTION HORIZON ?
ANSWER YES OR NO
NO
INPUT 1 CONTROL HORIZON =  3
DO YOU WANT TO CHANGE THE CONTROL HORIZON ?
ANSWER YES OR NO
NO
OUTPUT 1 KNOWN TIME DELAY =  0
DO YOU WANT TO CHANGE THE OUTPUT TIME DELAY ?
ANSWER YES OR NO
NO
OUTPUT 1 WEIGHTING SEQUENCE IS
 1.000   1.000   1.000   1.000   1.000   1.000   1.000   1.000   1.000   1.000

DO YOU WANT TO CHANGE THE WEIGHTING SEQUENCE ?
ANSWER YES OR NO
NO
INPUT 1 WEIGHTING SEQUENCE IS
 0.0     0.0     0.0
DO YOU WANT TO CHANGE THE WEIGHTING SEQUENCE ?
ANSWER YES OR NO
YES
SPECIFY THE WEIGHTING SEQUENCE FOR INPUT 1
0.25 0.25 0.25
```

```
INPUT 1 WEIGHTING SEQUENCE IS
 0.250  0.250  0.250
DO YOU WANT TO CHANGE THE WEIGHTING SEQUENCE ?
ANSWER YES OR NO
NO
VARIABLE GAIN OPTION, DEFAULT VALUE ARE UNITY
 KP( 1, 1) =  0.100000E+01
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
STEADY STATE STEP RESPONSE DEFAULTED TO FINAL VALUE OF THE STEP RESPONSE
 A11:  STEADY STATE VALUE =  0.381656E+00
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
FEEDBACK PATH:  1 = BETA,  2 = FILTER/PREDICTOR,  CURRENT PATH = 2
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
EXPONENTIAL AND VELOCITY (UNITON) FILTER
ARE AVAILABLE.  THE DEFAULT FILTER IS
EXPONENTIAL
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
OUTPUT 1 EXPONENTIAL FILTER CONSTANTS = 0.0
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
DISTURBANCE PREDICTION OPTIONS
1 = FUTURE PREDICTION EQUAL CURRENT ESTIMATE
2 = USER SUPPLIED SUBROUTINE (NOT IMPLEMENTED)
DEFAULT DISTURBANCE PREDICTION IS
FUTURE PREDICTION EQUAL CURRENT ESTIMATE
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
OUTPUT 1 BETA         FILTER CONSTANTS = 0.0
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
IS THERE ANY MEASURABLE DISTURBANCE ?
ANSWER YES OR NO
NO
DO YOU WANT TO TRANSLATE ANOTHER PROBLEM ?
ANSWER YES OR NO
NO
YOU HAVE FINISHED TRANSLATING A PROBLEM
WOULD YOU LIKE TO USE THE SIMULATOR ?
ANSWER YES OR NO
YES
**********************************************
**                                          **
** MOCCA SIMULATOR PROGRAM OPTIONS          **
**                                          **
** 1. WEIGHTED LEAST SQUARE                 **
**                                          **
** 2. INTERVAL PROGRAMMING (NOT READY)      **
**                                          **
** 3. LINEAR PROGRAMMING                    **
**                                          **
** 4. QUADRATIC PROGRAMMING (NOT READY)     **
**                                          **
**********************************************
PLEASE SPECIFY CONTROLLER TYPE
USE THE NUMBER GIVEN ABOVE
1
YOU HAVE SPECIFIED OPTION NUMBER 1
WOULD YOU LIKE TO CHANGE THIS ?
ANSWER YES OR NO
NO
PLEASE SPECIFY STEP RESPONSE DATA FILE NAME
FILE NAME MUST START WITH AN ALPHABET AND CAN HAVE UP TO 6 CHARACTERS
STEPA1
PLEASE SPECIFY CONTROLLER PARAMETER FILE NAME
FILE NAME MUST START WITH AN ALPHABET AND CAN HAVE UP TO 6 CHARACTERS
-A1
PLEASE SPECIFY SIMULATION INFORMATION FILE NAME
FILE NAME MUST START WITH AN ALPHABET AND CAN HAVE UP TO 6 CHARACTERS
-S1
THE FOLLOWING DATA ARE REQUIRED FOR THE
TRANSLATOR/SIMULATOR INFORMATION FILE.
YOU MAY LEAVE IT BLANK BY PRESSING THE
RETURN KEY
NAME OF ENGINEER:
K. LIM
DATE:
DECEMBER 18, 87
YOU ARE ALLOWED 3 LINES OF 60 CHARACTERS EACH
TO DESCRIBE THE SYSTEM YOU ARE DESIGNING
LINE 1
WEIGTED LEAST SQUARE SIMULATION EXAMPLE
LINE 2
STEP RESPONSE DATA FOR SIMULATING PROCESS IS THE SAME
LINE 3
AS THE ONE USED FOR DESIGNING THE CONTROLLER
SPECIFY SIMULATION END TIME IN INTEGER
60
ASSUME PROCESS IS AT STEADY STATE
HOW MANY SET OF SETPOINT CHANGES IS REQUIRED ?
MINIMUM IS ZERO (0) AND MAXIMUM IS TEN (10)
1
SETPOINT CHANGE SET NUMBER  1
SPECIFY TIME AT WHICH SETPOINT SET NUMBER  1 IS TO BE IMPLEMENTED
5
PLEASE SPECIFY SETPOINT FOR OUTPUT 1
1.0
DO YOU WANT TO SIMULATE DISTURBANCES ?
```

```
(MEASURABLE OR UNMEASURABLE CAN BE SIMULATED)
ANSWER YES OR NO
NO
SIMULATION BEGINS
SIMULATION COMPLETED
CHECK SIMULATION INFORMATION FILE FOR
INFORMATION REGARDING SIMUCATION RESULTS
DO YOU WANT TO SIMULATE ANOTHER PROBLEM ?
ANSWER YES OR NO
NO
WOULD YOU LIKE TO USE THE TRANSLATOR/SIMULATOR AGAIN ?
ANSWER YES OR NO
NO
* END OF MOCCA TRANSLATOR/SIMULATOR PROGRAM *
#12 18 07  T=0 877  RC=0
#    9 56,    $1.59T
#LI -A1  (COMMENT: CONTROLLER FILE USED IN WEIGHTED LEAST SQUARE ALGORITHM)

      1      MOCCA WLSQ CONTROLLER PARAMETERS
      2      1.1,
      3      10,
      4      3,
      5      0,
      6      40,
      7      0.95182398E-01,0.18128688E+00,0.25918112E+00,0.32867902E+00,0.38346833E+00,0.4511
             871E+00,
      8      0.50341328+00,0.55068894E+00,0.59342868E+00,0.6321187E+00,0.6671270E+00,0.6988
             038E+00,
      9      0.72746828+00,0.75340108E+00,0.77688778E+00,0.7981012E+00,0.8173141E+00,0.8348
             968E+00,
     10      0.85042908+00,0.86468238E+00,0.87754112E+00,0.88919438E+00,0.8997388E+00,0.9092
             798E+00,
     11      0.91791248E+00,0.92572338E+00,0.93278198E+00,0.93918738E+00,0.9449741E+00,0.9502
             103E+00,
     12      0.95498488E+00,0.95923518E+00,0.9631141E+00,0.96662408E+00,0.9697988E+00,0.9726
             737E+00,
     13      0.97527388E+00,0.97782688E+00,0.9801548E+00,0.98218618E+00,
     14      0.21044108E+00,0.27008258E+00,0.32847578E+00,0.38081938E+00,0.4287468E+00,0.1259
             158E+00,
     15      0.95018008E-01,0.72777188E-01,0.48937078E-01,0.2827077E-01,
     16      1,
     17      .9818818,
     18      2,
     19      1,
     20      0,
     21      1,
     22      0,
     23      0,
#   $ 03,    $1.72T
#LI -A2  (COMMENT: DESIGN INFORMATION GENERATED BY SOFTWARE PACKAGE)

      1      NAME OF ENGINEER: K. LIM
      2      DATE: DECEMBER 18, 87
      3      DESCRIPTION OF SYSTEM:
      4      WEIGHTED LEAST SQUARE DESIGN EXAMPLE
      5      SISO SYSTEM: GAIN=1.0, DELAY=0, TIME CONSTANT=10.0
      6
      7
      8      ****************************************
      9      *  WEIGHTED LEAST SQUARE CONTROLLER DESIGN  *
     10      ****************************************
     11      SYSTEM HAS 1 INPUTS AND 1 OUTPUTS
     12      OUTPUT 1 PREDICTION HORIZON = 10
     13      INPUT 1 CONTROL HORIZON =  3
     14      OUTPUT 1 KNOWN TIME DELAY =  0
     15      OUTPUT 1 WEIGHTING SEQUENCE IS
     16       1.000   1.000   1.000   1.000   1.000   1.000   1.000   1.000   1.000   1.000
     17
     18      INPUT 1 WEIGHTING SEQUENCE IS
     19       0.250   0.250   0.250
     20       A11: NUMBER OF STEP RESPONE DATA =  40 ,
     21       DYNAMIC MATRIX
     22        0.951628-01   0.0             0.0
     23        0.181278+00   0.951628-01     0.0
     24        0.258158+00   0.181278+00     0.951628-01
     25        0.329688+00   0.258158+00     0.181278+00
     26        0.393478+00   0.329688+00     0.258158+00
     27        0.451108+00   0.393478+00     0.329688+00
     28        0.503418+00   0.451108+00     0.393478+00
     29        0.550678+00   0.503418+00     0.451108+00
     30        0.593438+00   0.550678+00     0.503418+00
     31        0.632128+00   0.593438+00     0.550678+00
     32       DYNAMIC MATRIX FROBENIUS NORM = 4.507356,
     33       DYNAMIC MATRIX INVERSE:
     34        0.210448+00   0.270088+00   0.228488+00   0.190828+00   0.158758+00   0.12592E
             +00   0.980198E-01   0.727778-01   0.489378-01   0.252718-01
     35       -0.130788+00   0.887638-02   0.311018-01   0.502188-01   0.675128-01   0.831818
             -01   0.972218-01   0.110148+00   0.121738+00   0.132228+00
     36       -0.955828-01  -0.268318+00  -0.198158+00  -0.113818+00  -0.373818-01   0.308728
             -01   0.829308-01   0.144818+00   0.188078+00   0.244838+00
     37       DYNAMIC MATRIX INVERSE FROBENIUS NORM =  .8288155
     38       ESTIMATED CONDITION NUMBER = NORM(A)*NORM(A INVERSE) = 1.879888,
     39      VARIABLE GAIN OPTION, DEFAULT VALUE ARE UNITY
     40       KP(I,1) =  0.10000E+01
     41       A11: STEADY STATE VALUE =  0.981888E+00
     42      FEEDBACK PATH: 1 = BETA, 2 = FILTER/PREDICTOR, CURRENT PATH = 2
     43      FEEDBACK FILTER CHOSEN IS EXPONENTIAL
     44      OUTPUT 1 EXPONENTIAL FILTER CONSTANTS = 0.0
     45      TYPE OF DISTURBANCE PREDICTION CHOSEN IS
     46      1 = FUTURE PREDICTION EQUAL CURRENT ESTIMATE
     47      OUTPUT 1 BETA        FILTER CONSTANTS = 0.0
#   $ 04,    $1.78T
#LI -S1 (COMMENT: SIMULATION INFORMATION FILE GENERATED BY SOFTWARE PACKAGE)

      1      NAME OF ENGINEER: K. LIM
      2      DATE: DECEMBER 18, 87
      3      DESCRIPTION OF SYSTEM:
      4      WEIGTED LEAST SQUARE SIMULATION EXAMPLE
```

```
      5       STEP RESPONSE DATA FOR SIMULATING PROCESS IS THE SAME
      6       AS THE ONE USED FOR DESIGNING THE CONTROLLER
      7
      8       WEIGHTED LEAST SQUARE SIMULATION
      9       STEP RESPONSE DATA FILE USED: STEPA1
     10       CONTROLLER PARAMETER FILE USED: -A1
     11
     12                    SETPOINT CHANGES FOR THIS SIMULATION
     13       SET #    IMPLEMENTATION TIME          SETPOINTS OUTPUT #
     14
     15       ------------------------------------------------------------------------
     16          1                5              1.000
     17       ------------------------------------------------------------------------
     18
     19       SIMULATION RESULTS ARE IN THE FOLLOWING FILES
     20            TYPE OF RESULTS              FILE NAME
     21       ------------------------------  ------------
     22          PROCESS OUTPUT                YOUT
     23          MODEL OUTPUT                  YMOUT
     24          MANIPULATED INPUT             UOUT
     25          CHANGE IN INPUT               DUOUT
     26          SETPOINT IMPLEMENTED          SPTOUT
     27          MEASURABLE DISTURBANCE        DISOUT
  $.  $.03.    $1.787
#LI YOUT (COMMENT: LISTING OF SIMULATION PROCESS OUTPUT FILE)
    0   0.0
    1   0.0
    2   0.0
    3   0.0
    4   0.0
    5   0.0
    6   0.1383E+00
    7   0.3222E+00
    8   0.5078E+00
    9   0.6875E+00
   10   0.7928E+00
   11   0.8637E+00
   12   0.9451E+00
   13   0.9832E+00
   14   0.1005E+01
   15   0.1015E+01
   16   0.1018E+01
   17   0.1017E+01
   18   0.1014E+01
   19   0.1011E+01
   20   0.1005E+01
   21   0.1005E+01
   22   0.1003E+01
   23   0.1002E+01
   24   0.1001E+01
   25   0.1000E+01
   26   0.9998E+00
   27   0.9997E+00
   28   0.9997E+00
   29   0.9997E+00
   30   0.9998E+00
   31   0.9998E+00
   32   0.9999E+00
   33   0.9999E+00
   34   0.1000E+01
   35   0.1000E+01
   36   0.1000E+01
   37   0.1000E+01
   38   0.1000E+01
   39   0.1000E+01
   40   0.1000E+01
   41   0.1001E+01
   42   0.1001E+01
   43   0.1002E+01
   44   0.1003E+01
   45   0.1004E+01
   46   0.1003E+01
   47   0.1001E+01
   48   0.9999E+00
   49   0.9969E+00
   50   0.9964E+00
#LI UOUT (COMMENT: LISTING OF SIMULATION PROCESS MANIPULATED INPUT FILE)
    0   0.0
    1   0.0
    1   0.0
    2   0.0
    2   0.0
    3   0.0
    3   0.0
    4   0.0
    4   0.0
    5   0.0
    5   0.1432E+01
    6   0.1432E+01
    6   0.2090E+01
    7   0.2090E+01
    7   0.2270E+01
    8   0.2270E+01
    8   0.2165E+01
    8   0.2158E+01
    9   0.1984E+01
   10   0.1984E+01
   10   0.1748E+01
   11   0.1748E+01
   11   0.1528E+01
   12   0.1528E+01
   12   0.1348E+01
   13   0.1348E+01
   13   0.1208E+01
   14   0.1208E+01
   14   0.1119E+01
   15   0.1119E+01
   15   0.1047E+01
```

15	0.1047E+01
16	0.1008E+01
17	0.1008E+01
17	0.9860E+00
18	0.9860E+00
18	0.9785E+00
19	0.9785E+00
19	0.9780E+00
20	0.9780E+00
20	0.9803E+00
21	0.9803E+00
21	0.9841E+00
22	0.9841E+00
22	0.9862E+00
23	0.9862E+00
23	0.9918E+00
24	0.9918E+00
24	0.9948E+00
25	0.9948E+00
25	0.9970E+00
26	0.9970E+00
26	0.9985E+00
27	0.9985E+00
27	0.9994E+00
28	0.9994E+00
28	0.1000E+01
28	0.1000E+01
29	0.1000E+01
30	0.1000E+01
30	0.1000E+01
31	0.1000E+01
31	0.1000E+01
32	0.1000E+01
32	0.1000E+01
33	0.1000E+01
33	0.1000E+01
34	0.1000E+01
34	0.1000E+01
35	0.1000E+01
35	0.1000E+01
36	0.1000E+01
36	0.1000E+01
37	0.1000E+01
37	0.1000E+01
38	0.1000E+01
38	0.1001E+01
39	0.1001E+01
39	0.1002E+01
40	0.1002E+01
40	0.1003E+01
41	0.1003E+01
41	0.1005E+01
42	0.1005E+01
42	0.1008E+01
43	0.1008E+01
43	0.1011E+01
44	0.1011E+01
44	0.1018E+01
45	0.1018E+01
45	0.1020E+01
46	0.1020E+01
46	0.1024E+01
47	0.1024E+01
47	0.1028E+01
48	0.1028E+01
48	0.1030E+01
49	0.1030E+01
49	0.1030E+01
50	0.1030E+01
50	0.1030E+01

18 Appendix I

MIMO Linear Programming Example Translation Session

The following design session was performed using step response data from the cement kiln example. The design example uses the linear programming option of the MOCCA translation package and the simulation results are presented in section 7.4.2.

```
Slog Output  KIAN YAP LIM, KLIM, Job#=4882, Host=UALTAMTS, 02.37.18 Sat Apr 02/88
#    $.03,    $.787
#R M_MOCCA.RUN
#02.37.21
 ********************************************
 **                                        **
 **              WELCOME TO MOCCA          **
 **         TRANSLATOR AND SIMULATOR       **
 **                 PROGRAM                **
 **          VERSION DATE: 86/02/23        **
 **                                        **
 ********************************************
THE FOLLOWING OPTIONS ARE AVAILABLE:
1  TRANSLATOR:  TRANSLATE MOCCA PROBLEM
                INTO CONTROL ALGORITHM
2  SIMULATOR:   SIMULATE CONTROL ALGORITHM
                OBTAINED FROM TRANSLATOR
PLEASE SPECIFY OPTION NUMBER
1
YOU HAVE SPECIFIED OPTION NUMBER 1
WOULD YOU LIKE TO CHANGE THIS ?
ANSWER YES OR NO
NO
 ********************************************
 **                                        **
 ** MOCCA TRANSLATOR PROGRAM OPTIONS:      **
 **                                        **
 ** 1  WEIGHTED LEAST SQUARE               **
 **                                        **
 ** 2  LINEAR PROGRAMMING                  **
 **                                        **
 ** 3  QUADRATIC PROGRAMMING               **
 **                                        **
 ********************************************
PLEASE SPECIFY OPTION NUMBER
2
YOU HAVE SPECIFIED OPTION NUMBER 2
WOULD YOU LIKE TO CHANGE THIS ?
ANSWER YES OR NO
NO
PLEASE SPECIFY STEP RESPONSE DATA FILE NAME
FILE NAME MUST START WITH AN ALPHABET AND CAN HAVE UP TO 6 CHARACTERS
KILN60
PLEASE SPECIFY CONTROLLER PARAMETER FILE NAME
FILE NAME MUST START WITH AN ALPHABET AND CAN HAVE UP TO 6 CHARACTERS
-A1
PLEASE SPECIFY DESIGN INFORMATION FILE NAME
FILE NAME MUST START WITH AN ALPHABET AND CAN HAVE UP TO 6 CHARACTERS
-A2
PLEASE SPECIFY LP/QP DATA FILE NAME
FILE NAME MUST START WITH AN ALPHABET AND CAN HAVE UP TO 6 CHARACTERS
-A3
PLEASE SPECIFY CONSTRAINTS DATA FILE NAME
FILE NAME MUST START WITH AN ALPHABET AND CAN HAVE UP TO 6 CHARACTERS
-A4
THE FOLLOWING DATA ARE REQUIRED FOR THE
TRANSLATOR/SIMULATOR INFORMATION FILE.
YOU MAY LEAVE IT BLANK BY PRESSING THE
RETURN KEY
NAME OF ENGINEER:
K. LIM
DATE:
APRIL 1, 1988
YOU ARE ALLOWED 3 LINES OF 80 CHARACTERS EACH
TO DESCRIBE THE SYSTEM YOU ARE DESIGNING
LINE 1
MOCCA LINEAR PROGRAMMING CONTROLLER FOR CEMENT KILN SIMULATION
LINE 2
SAMPLING TIME=60 MINUTES
LINE 3

 ********************************************
 *  LINEAR PROGRAMMING CONTROLLER DESIGN    *
 ********************************************
 ********************************************
 ** CURRENT AVAILABLE INDICES ARE:         **
 ** NOTE: ALL INDICES HAS WEIGHTING FACTOR **
 ** TIME VARYING INDEX CAN BE IMPLEMENTED  **
 ** VIA THE POCC (PMC) BLOCKS              **
 ** IAE  - INTEGRAL OF ABSOLUTE ERROR (LP) **
 ** IAEU - INTEGRAL OF ABSOLUTE ERROR AND  **
 **        DELTA U (LP)                    **
 ** WLSQ - WEIGHTED LEAST SQUARE (QP)      **
 ** ISE  - INTEGRAL OF ERROR SQUARE (QP)   **
 ********************************************
PLEASE SPECIFY PERFORMANCE INDEX (USE ACRONYM)
IAEU
 PERFORMANCE INDEX CHOSEN IS IAEU
DO YOU WANT TO CHANGE THIS (YES/NO) ?
NO
SYSTEM HAS 3 INPUTS AND 3 OUTPUTS
DO YOU WANT TO CHANGE THE SYSTEM SIZE ?
ANSWER YES OR NO
NO
OUTPUT 1 PREDICTION HORIZON = 10
OUTPUT 2 PREDICTION HORIZON = 10
OUTPUT 3 PREDICTION HORIZON = 10
DO YOU WANT TO CHANGE THE PREDICTION HORIZON ?
ANSWER YES OR NO
YES
SPECIFY THE PREDICTION HORIZON FOR OUTPUT 1
5
SPECIFY THE PREDICTION HORIZON FOR OUTPUT 2
5
SPECIFY THE PREDICTION HORIZON FOR OUTPUT 3
5
OUTPUT 1 PREDICTION HORIZON =  5
```

```
OUTPUT 2 PREDICTION HORIZON =  5
OUTPUT 3 PREDICTION HORIZON =  5
DO YOU WANT TO CHANGE THE PREDICTION HORIZON ?
ANSWER YES OR NO
NO
INPUT 1 CONTROL HORIZON =  3
INPUT 2 CONTROL HORIZON =  3
INPUT 3 CONTROL HORIZON =  3
DO YOU WANT TO CHANGE THE CONTROL HORIZON ?
ANSWER YES OR NO
NO
OUTPUT 1 KNOWN TIME DELAY =  0
OUTPUT 2 KNOWN TIME DELAY =  0
OUTPUT 3 KNOWN TIME DELAY =  0
DO YOU WANT TO CHANGE THE OUTPUT TIME DELAY ?
ANSWER YES OR NO
NO
VARIABLE GAIN OPTION, DEFAULT VALUE ARE UNITY
 KP(1,1) =  0.10000E+01
 KP(1,2) =  0.10000E+01
 KP(1,3) =  0.10000E+01
 KP(2,1) =  0.10000E+01
 KP(2,2) =  0.10000E+01
 KP(2,3) =  0.10000E+01
 KP(3,1) =  0.10000E+01
 KP(3,2) =  0.10000E+01
 KP(3,3) =  0.10000E+01
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
STEADY STATE STEP RESPONSE DEFAULTED TO FINAL VALUE OF THE STEP RESPONSE
 A11: STEADY STATE VALUE =  0.49000E-01
 A12: STEADY STATE VALUE =  0.24000E-01
 A13: STEADY STATE VALUE =  0.11100E+00
 A21: STEADY STATE VALUE =  0.37000E-01
 A22: STEADY STATE VALUE =  0.20000E-01
 A23: STEADY STATE VALUE =  0.88999E-01
 A31: STEADY STATE VALUE =  0.0
 A32: STEADY STATE VALUE =  0.28000E-02
 A33: STEADY STATE VALUE =  0.90000E-01
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
FEEDBACK PATH: 1 = BETA, 2 = FILTER/PREDICTOR;CURRENT PATH =  2
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
EXPONENTIAL AND VELOCITY (UNION) FILTER
ARE AVAILABLE.  THE DEFAULT FILTER IS
EXPONENTIAL
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
OUTPUT 1 EXPONENTIAL FILTER CONSTANTS =  0.0
OUTPUT 2 EXPONENTIAL FILTER CONSTANTS =  0.0
OUTPUT 3 EXPONENTIAL FILTER CONSTANTS =  0.0
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
DISTURBANCE PREDICTION OPTIONS:
1 = FUTURE PREDICTION EQUAL CURRENT ESTIMATE
2 = USER SUPPLIED SUBROUTINE (USER MUST MODIFY
    ON-LINE FEEDBACK PATH ROUTINE: FPATH
DEFAULT DISTURBANCE PREDICTION IS
FUTURE PREDICTION EQUAL CURRENT ESTIMATE
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
OUTPUT 1 BETA          FILTER CONSTANTS =  0.0
OUTPUT 2 BETA          FILTER CONSTANTS =  0.0
OUTPUT 3 BETA          FILTER CONSTANTS =  0.0
DO YOU WANT TO CHANGE THIS ?
ANSWER YES OR NO
NO
IS THERE ANY MEASURABLE DISTURBANCE ?
ANSWER YES OR NO
NO
** LP WEIGHTING AND CONSTRAINT DESIGN **
OUTPUT 1 WEIGHTING SEQUENCE IS
 1.000   1.000   1.000   1.000   1.000
OUTPUT 2 WEIGHTING SEQUENCE IS
 1.000   1.000   1.000   1.000   1.000
OUTPUT 3 WEIGHTING SEQUENCE IS
 1.000   1.000   1.000   1.000   1.000
DO YOU WANT TO CHANGE THE WEIGHTING SEQUENCE ?
ANSWER YES OR NO
YES
SPECIFY THE WEIGHTING SEQUENCE FOR OUTPUT 1
1,1,1,1,1
SPECIFY THE WEIGHTING SEQUENCE FOR OUTPUT 2
1,1,1,1,1
SPECIFY THE WEIGHTING SEQUENCE FOR OUTPUT 3
10,10,10,10,10
OUTPUT 1 WEIGHTING SEQUENCE IS
 1.000   1.000   1.000   1.000   1.000
OUTPUT 2 WEIGHTING SEQUENCE IS
 1.000   1.000   1.000   1.000   1.000
OUTPUT 3 WEIGHTING SEQUENCE IS
10.000  10.000  10.000  10.000  10.000
DO YOU WANT TO CHANGE THE WEIGHTING SEQUENCE ?
ANSWER YES OR NO
NO
INPUT 1 WEIGHTING SEQUENCE IS
 0.0     0.0     0.0
INPUT 2 WEIGHTING SEQUENCE IS
 0.0     0.0     0.0
```

```
 INPUT 3 WEIGHTING SEQUENCE IS
  0.0     0.0     0.0
DO YOU WANT TO CHANGE THE WEIGHTING SEQUENCE ?
ANSWER YES OR NO
YES
SPECIFY THE WEIGHTING SEQUENCE FOR INPUT 1
0.01,0.01,0.01
SPECIFY THE WEIGHTING SEQUENCE FOR INPUT 2
0.01,0.01,0.01
SPECIFY THE WEIGHTING SEQUENCE FOR INPUT 3
0.01,0.01,0.01
INPUT 1 WEIGHTING SEQUENCE IS
 0.010   0.010   0.010
INPUT 2 WEIGHTING SEQUENCE IS
 0.010   0.010   0.010
INPUT 3 WEIGHTING SEQUENCE IS
 0.010   0.010   0.010
DO YOU WANT TO CHANGE THE WEIGHTING SEQUENCE ?
ANSWER YES OR NO
NO
DO YOU WANT CONSTRAINTS ON CHANGE IN INPUTS (YES/NO) ?
YES
SPECIFY CHANGE IN INPUT 1
 UPPER CONSTRAINT TRAJECTORY
225,225,225
 LOWER CONSTRAINT TRAJECTORY
-225,-225,-225
SPECIFY CHANGE IN INPUT 2
 UPPER CONSTRAINT TRAJECTORY
120,120,120
 LOWER CONSTRAINT TRAJECTORY
-120,-120,-120
SPECIFY CHANGE IN INPUT 3
 UPPER CONSTRAINT TRAJECTORY
25,25,25
 LOWER CONSTRAINT TRAJECTORY
-25,-25,-25
CHANGE IN INPUT 1
 UPPER CONSTRAINT TRAJECTORY IS
225.0 225.0 225.0
 LOWER CONSTRAINT TRAJECTORY IS
-225.0 -225.0 -225.0
CHANGE IN INPUT 2
 UPPER CONSTRAINT TRAJECTORY IS
120.0 120.0 120.0
 LOWER CONSTRAINT TRAJECTORY IS
-120.0 -120.0 -120.0
CHANGE IN INPUT 3
 UPPER CONSTRAINT TRAJECTORY IS
25.00 25.00 25.00
 LOWER CONSTRAINT TRAJECTORY IS
-25.0 -25.0 -25.0
DO YOU WANT TO CHANGE THE CONSTRAINT (YES/NO) ?
NO
DO YOU WANT CONSTRAINTS ON INPUTS (YES/NO) ?
NO
DO YOU WANT CONSTRAINTS ON OUTPUTS (YES/NO) ?
NO
DO YOU WANT CONSTRAINT ON UNCONTROLLED OUTPUTS (YES/NO) ?
NO
DO YOU WANT TO TRANSLATE ANOTHER PROBLEM ?
ANSWER YES OR NO
NO
YOU HAVE FINISHED TRANSLATING A PROBLEM
WOULD YOU LIKE TO USE THE SIMULATOR ?
ANSWER YES OR NO
NO
* END OF MOCCA TRANSLATOR/SIMULATOR PROGRAM *
#02.40.37   TP=0.575   RC=0
#   S1.71.     S2.497
#LL -A1 (COMMENT  CONTROLLER FILE USED IN MOCCA LP ALGORITHM)
      1       MOCCA LP CONTROLLER PARAMETERS
      2       3,3,
      3       5,5,5,
      4       3,3,3,
      5       0,0,0,
      6       20,
      7       -.28205708E-01,-.4017564E-01,-.45255038E-01,-.47410174E-01,-.48325648E-01,-.4871
              3778E-01,
      8       -.48878528E-01,-.48948443E-01,-.48978108E-01,-.48990658E-01,-.48995803E-01,-.4899
              8308E-01,
      9       -.48998278E-01,-.48999878E-01,-.48999965E-01,-.48999992E-01,-.48999998E-01,-.4899
              9988E-01,
     10       -.48999998E-01,-.48999998E-01,
     11       20,
     12       0.13815038E-01,0.19677978E-01,0.22165758E-01,0.23221588E-01,0.23669668E-01,0.2385
              9868E-01,
     13       0.23940468E-01,0.23974738E-01,0.23989268E-01,0.23995438E-01,0.23998058E-01,0.2399
              8168E-01,
     14       0.23999318E-01,0.23999438E-01,0.23999918E-01,0.23999958E-01,0.23999978E-01,0.2399
              9968E-01,
     15       0.23999968E-01,0.23999988E-01,
     16       20,
     17       -.62884618E-01,-.81009508E-01,-.10251848E+00,-.10733958E+00,-.10947188E+00,-.1103
              8138E+00,
     18       -.11072448E+00,-.11088268E+00,-.11095018E+00,-.11097888E+00,-.11098078E+00,-.1109
              8868E+00,
     19       -.11098908E+00,-.11098988E+00,-.11098998E+00,-.11098998E+00,-.11098998E+00,-.1109
              8998E+00,
     20       -.11098998E+00,-.11098998E+00,
     21       20,
     22       -.21286168E-01,-.30338958E-01,-.34172208E-01,-.35789948E-01,-.36490718E-01,-.3675
              3488E-01,
     23       -.36908288E-01,-.36981088E-01,-.37063468E-01,-.37092978E-01,-.37097018E-01,-.3699
              8728E-01,
     24       -.36998468E-01,-.36998768E-01,-.36998868E-01,-.36998938E-01,-.36998988E-01,-.3699
              8998E-01,
```

```
25      -.38888888E-01,-.38888888E-01,
26       20,
27      0.11612828E-01,0.16286138E-01,0.18471458E-01,0.19351318E-01,0.19724708E-01,0.1988
        3188E-01,
28      0.19950408E-01,0.19978948E-01,0.19991058E-01,0.19996198E-01,0.19998278E-01,0.1999
        928E-01,
29      0.19999868E-01,0.19999888E-01,0.19999928E-01,0.19999958E-01,0.19999978E-01,0.1999
        998E-01,
30      0.19999988E-01,0.19999998E-01,
31       20,
32      0.51230768E-01,0.72971588E-01,0.82187768E-01,0.86113108E-01,0.87774638E-01,0.8847
        978E-01,
33      0.88778878E-01,0.88908998E-01,0.88958938E-01,0.88982828E-01,0.88992548E-01,0.8999
        668E-01,
34      0.88998328E-01,0.88999038E-01,0.88999338E-01,0.88999338E-01,0.88999338E-01,0.8999
        838E-01,
35      0.88999338E-01,0.88999338E-01,
36       20,
37      0.0          ,0.0          ,0.0          ,0.0          ,0.0          ,0.0
38      0.0          ,0.0          ,0.0          ,0.0          ,0.0          ,0.0
39      0.0          ,0.0          ,0.0          ,0.0          ,0.0          ,0.0
40      0.0          ,0.0
41       20,
42      -.20198588E-02,-.24708518E-02,-.25711158E-02,-.25935958E-02,-.25985618E-02,-.2599
        6758E-02,
43      -.25998268E-02,-.25999638E-02,-.25999958E-02,-.25999998E-02,-.25999998E-02,-.2599
        9958E-02,
44      -.25999998E-02,-.25999998E-02,-.25999998E-02,-.25999998E-02,-.25999998E-02,-.2599
        9958E-02,
45      -.25999998E-02,-.25999998E-02,
46       20,
47      0.77568728E-02,0.95021138E-02,0.98488898E-02,0.99752038E-02,0.99944568E-02,0.9998
        754E-02,
48      0.99997118E-02,0.99999318E-02,0.99999798E-02,0.99999798E-02,0.99999798E-02,0.9999
        978E-02,
49      0.99999798E-02,0.99999798E-02,0.99999798E-02,0.99999798E-02,0.99999798E-02,0.9999
        978E-02,
50      0.99999798E-02,0.99999798E-02,
51       1.,1.,1.,
52       1.,1.,1.,
53       1.,1.,1.,
54      -.46998868E-01,.23999988E-01,-.11098883,
55      -.38888888E-01,.18888888E-01,.68888132E-01,
56      0.,-.25878888E-02,.99999798E-02,
57       2,
58       1,
59       0,
60       0,
61       0,
62       1,
63       0,
64       0,
65       0,
66       0,
# 5.04    52.52T
FLI  A2 (COMMENT: DESIGN INFORMATION FILE GENERATED BY SOFTWARE PACKAGE)

1       NAME OF ENGINEER: K. LIM
2       DATE: APRIL 1, 1988
3       DESCRIPTION OF SYSTEM:
4       MOCCA LINEAR PROGRAMMING CONTROLLER FOR CEMENT KILN SIMULATION
5       SAMPLING TIME=60 MINUTES
6
7
8        ********************************************
9        *   LINEAR PROGRAMMING CONTROLLER DESIGN   *
10       ********************************************
11       PERFORMANCE INDEX CHOSEN IS IAEU
12      SYSTEM HAS 3 INPUTS AND 3 OUTPUTS
13      OUTPUT 1 PREDICTION HORIZON =  5
14      OUTPUT 2 PREDICTION HORIZON =  5
15      OUTPUT 3 PREDICTION HORIZON =  5
16      INPUT 1 CONTROL HORIZON =  3
17      INPUT 2 CONTROL HORIZON =  3
18      INPUT 3 CONTROL HORIZON =  3
19      OUTPUT 1 KNOWN TIME DELAY =  0
20      OUTPUT 2 KNOWN TIME DELAY =  0
21      OUTPUT 3 KNOWN TIME DELAY =  0
22       A11: NUMBER OF STEP RESPONSE DATA =  20
23       A12: NUMBER OF STEP RESPONSE DATA =  20
24       A13: NUMBER OF STEP RESPONSE DATA =  20
25       A21: NUMBER OF STEP RESPONSE DATA =  20
26       A22: NUMBER OF STEP RESPONSE DATA =  20
27       A23: NUMBER OF STEP RESPONSE DATA =  20
28       A31: NUMBER OF STEP RESPONSE DATA =  20
29       A32: NUMBER OF STEP RESPONSE DATA =  20
30       A33: NUMBER OF STEP RESPONSE DATA =  20
31       DYNAMIC MATRIX:
32       -0.282068E-01  0.0           0.0           0.138158E-01  0.0           0.0
          -0.626868E-01  0.0           0.0
33       -0.401788E-01 -0.282068E-01  0.0           0.198788E-01  0.138158E-01  0.0
          -0.818088E-01 -0.626868E-01  0.0
34       -0.482668E-01 -0.401788E-01 -0.282068E-01  0.221868E-01  0.198788E-01  0.138158
         -01 -0.102528E+00 -0.818088E-01 -0.626868E-01
35       -0.474118E-01 -0.482668E-01 -0.401788E-01  0.233228E-01  0.221868E-01  0.198788
         -01 -0.107408E+00 -0.102528E+00 -0.818088E-01
36       -0.482328E-01 -0.474118E-01 -0.482668E-01  0.238708E-01  0.233228E-01  0.221868
         -01 -0.108478E+00 -0.107408E+00 -0.102528E+00
37       -0.212988E-01  0.0           0.0           0.118138E-01  0.0           0.0
           0.512318E-01  0.0           0.0
38       -0.302378E-01 -0.212988E-01  0.0           0.168868E-01  0.118138E-01  0.0
          -0.729728E-01  0.512318E-01  0.0
39       -0.341728E-01 -0.302378E-01 -0.212988E-01  0.192718E-01  0.168868E-01  0.118138
         -01  0.821988E-01  0.729728E-01  0.512318E-01
```

```
 40      -0.35800E-01 -0.34172E-01 -0.30337E-01  0.19351E-01  0.18471E-01  0.18398E
      -01  0.86113E-01  0.82198E-01  0.72872E-01
 41      -0.38461E-01 -0.35800E-01 -0.34172E-01  0.19728E-01  0.19351E-01  0.18471E
      -01  0.87776E-01  0.86113E-01  0.82198E-01
 42        0.0          0.0          0.0         -0.20199E-02  0.0          0.0
           0.77687E-02  0.0          0.0
 43        0.0          0.0          0.0         -0.24706E-02 -0.20199E-02  0.0
           0.95021E-02  0.77687E-02  0.0
 44        0.0          0.0          0.0         -0.25711E-02 -0.24706E-02 -0.20199E
      -02  0.98889E-02  0.95021E-02  0.77687E-02
 45        0.0          0.0          0.0         -0.25935E-02 -0.25711E-02 -0.24706E
      -02  0.99752E-02  0.98889E-02  0.95021E-02
 46        0.0          0.0          0.0         -0.25985E-02 -0.25935E-02 -0.25711E

      -02  0.99945E-02  0.99752E-02  0.98889E-02
 47    VARIABLE GAIN OPTION, DEFAULT VALUE ARE UNITY
 48     KP(1,1) =  0.10000E+01
 49     KP(1,2) =  0.10000E+01
 50     KP(1,3) =  0.10000E+01
 51     KP(2,1) =  0.10000E+01
 52     KP(2,2) =  0.10000E+01
 53     KP(2,3) =  0.10000E+01
 54     KP(3,1) =  0.10000E+01
 55     KP(3,2) =  0.10000E+01
 56     KP(3,3) =  0.10000E+01
 57     A11  STEADY STATE VALUE = -0.48000E-01
 58     A12  STEADY STATE VALUE =  0.24000E-01
 59     A13  STEADY STATE VALUE = -0.11100E+00
 60     A21  STEADY STATE VALUE = -0.37000E-01
 61     A22  STEADY STATE VALUE =  0.20000E-01
 62     A23  STEADY STATE VALUE =  0.88889E-01
 63     A31  STEADY STATE VALUE =  0.0
 64     A32  STEADY STATE VALUE = -0.28000E-02
 65     A33  STEADY STATE VALUE =  0.10000E-01
 66    FEEDBACK PATH  1 = BETA, 2 = FILTER/PREDICTOR;CURRENT PATH = 2
 67    FEEDBACK FILTER CHOSEN IS EXPONENTIAL
 68    OUTPUT 1 EXPONENTIAL FILTER CONSTANTS = 0.0
 69    OUTPUT 2 EXPONENTIAL FILTER CONSTANTS = 0.0
 70    OUTPUT 3 EXPONENTIAL FILTER CONSTANTS = 0.0
 71    TYPE OF DISTURBANCE PREDICTION CHOSEN IS
 72    1 = FUTURE PREDICTION EQUAL CURRENT ESTIMATE
 73    OUTPUT 1 BETA          FILTER CONSTANTS = 0.0
 74    OUTPUT 2 BETA          FILTER CONSTANTS = 0.0
 75    OUTPUT 3 BETA          FILTER CONSTANTS = 0.0
 76    ** LP WEIGHTING AND CONSTRAINT DESIGN **
 77    OUTPUT 1 WEIGHTING SEQUENCE IS
 78     1.000  1.000  1.000  1.000  1.000
 79    OUTPUT 2 WEIGHTING SEQUENCE IS
 80     1.000  1.000  1.000  1.000  1.000
 81    OUTPUT 3 WEIGHTING SEQUENCE IS
 82    10.000 10.000 10.000 10.000 10.000
 83    INPUT 1 WEIGHTING SEQUENCE IS
 84     0.010  0.010  0.010
 85    INPUT 2 WEIGHTING SEQUENCE IS
 86     0.010  0.010  0.010
 87    INPUT 3 WEIGHTING SEQUENCE IS
 88     0.010  0.010  0.010
 89    CHANGE IN INPUT 1
 90     UPPER CONSTRAINT TRAJECTORY IS
 91    ***** ***** *****
 92     LOWER CONSTRAINT TRAJECTORY IS
 93    ***** ***** *****
 94    CHANGE IN INPUT 2
 95     UPPER CONSTRAINT TRAJECTORY IS
 96    ***** ***** *****
 97     LOWER CONSTRAINT TRAJECTORY IS
 98    ***** ***** *****
 99    CHANGE IN INPUT 3
100     UPPER CONSTRAINT TRAJECTORY IS
101    25.00 25.00 25.00
102     LOWER CONSTRAINT TRAJECTORY IS
103    ***** ***** *****
#  8.05.    82.867
#LI -A3 (COMMENT: LP DATA FILE USED IN MOCCA LP ALGORITHM)

  1          33         48          0          0          0          0
  2      1 -0.1000E+01  2 -0.1000E+01  3 -0.1000E+01  4 -0.1000E+01  5 -0.1000E+01
         6 -0.1000E+01  7 -0.1000E+01  8 -0.1000E+01
  3      9 -0.1000E+01 10 -0.1000E+01 11 -0.1000E+02 12 -0.1000E+02 13 -0.1000E+02
        14 -0.1000E+02 15 -0.1000E+02 16 -0.1000E+01
  4     17 -0.1000E+01 18 -0.1000E+01 19 -0.1000E+01 20 -0.1000E+01 21 -0.1000E+01
        22 -0.1000E+01 23 -0.1000E+01 24 -0.1000E+01
  5     25 -0.1000E+01 26 -0.1000E+02 27 -0.1000E+02 28 -0.1000E+02 29 -0.1000E+02
        30 -0.1000E+02 31 -0.1000E-01 32 -0.1000E-01
  6     33 -0.1000E-01 34 -0.1000E-01 35 -0.1000E-01 36 -0.1000E-01 37 -0.1000E-01
        38 -0.1000E-01 39 -0.1000E-01 40 -0.1000E-01
  7     41 -0.1000E-01 42 -0.1000E-01 43 -0.1000E-01 44 -0.1000E-01 45 -0.1000E-01
        46 -0.1000E-01 47 -0.1000E-01 48 -0.1000E-01
  8     999999999
  9      11 0.0          21 0.0          31 0.0          41 0.0          51 0.0
         61 0.0          71 0.0          81 0.0
 10      91 0.0         101 0.0         111 0.0         121 0.0         131 0.0
        141 0.0         151 0.0         161 0.0
 11     171 0.0         181 0.0         190 0.0         200 0.0         210 0.0
        220 0.0         230 0.0         240 0.0
 12     250 0.0         260 0.0         270 0.0         280 0.0         290 0.0
        300 0.0         310 0.0         320 0.0
 13     330 0.0
 14     999999999
 15          0.1000E+01 31  0.1000E+01 40 -0.1000E+01
 16          0.2000E+01 32  0.1000E+01 41 -0.1000E+01
 17          0.3000E+01 33  0.1000E+01 42 -0.1000E+01
 18          0.4000E+01 34  0.1000E+01 43 -0.1000E+01
 19          0.5000E+01 35  0.1000E+01 44 -0.1000E+01
 20          0.6000E+01 36  0.1000E+01 45 -0.1000E+01
 21          0.7000E+01 37  0.1000E+01 46 -0.1000E+01
 22          0.8000E+01 38  0.1000E+01 47 -0.1000E+01
 23          0.9000E+01 39  0.1000E+01 48 -0.1000E+01
```

```
 24          0.1000E+02 31 -0.1000E+01 40  0.1000E+01
 25          0.1100E+02 32 -0.1000E+01 41  0.1000E+01
 26          0.1200E+02 33 -0.1000E+01 42  0.1000E+01
 27          0.1300E+02 34 -0.1000E+01 43  0.1000E+01
 28          0.1400E+02 35 -0.1000E+01 44  0.1000E+01
 29          0.1500E+02 36 -0.1000E+01 45  0.1000E+01
 30          0.1600E+02 37 -0.1000E+01 46  0.1000E+01
 31          0.1700E+02 38 -0.1000E+01 47  0.1000E+01
 32          0.1800E+02 39 -0.1000E+01 48  0.1000E+01
 33          0.1900E+02  1  0.1000E+01 16 -0.1000E+01 31 -0.2821E-01 34  0.1382E-01
      37 -0.8388E-01 40  0.2821E-01 43 -0.1382E-01
 34          0.1900E+02 46  0.8388E-01
 35          0.2000E+02  2  0.1000E+01 17 -0.1000E+01 31 -0.4018E-01 32 -0.2821E-01
      34  0.1958E-01 35  0.1382E-01 37 -0.9101E-01
 36          0.2000E+02 38 -0.8388E-01 40  0.4018E-01 41 -0.2821E-01 43 -0.1958E-01
      44 -0.1382E-01 46  0.9101E-01 47  0.8388E-01
 37          0.2100E+02  3  0.1000E+01 18 -0.1000E+0  31 -0.4526E-01 32 -0.4018E-01
      33 -0.2821E-01 34  0.2217E-01 35  0.1958E-01
 38          0.2100E+02 36  0.1382E-01 37 -0.1025E+00 38 -0.9101E-01 39 -0.8388E-01
      40  0.4526E-01 41  0.4018E-01 42  0.2821E-01
 39          0.2100E+02 43 -0.2217E-01 44 -0.1958E-01 45 -0.1382E-01 48  0.1025E+00
      47  0.9101E-01 48  0.8388E-01
 40          0.2200E+02  4  0.1000E+01 19 -0.1000E+01 31 -0.4741E-01 32 -0.4526E-01
      33 -0.4018E-01 34  0.2322E-01 35  0.2217E-01
 41          0.2200E+02 36  0.1958E-01 37 -0.1074E+00 38 -0.1025E+00 39 -0.9101E-01
      40  0.4741E-01 41  0.4526E-01 42  0.4018E-01
 42          0.2200E+02 43 -0.2322E-01 44 -0.2217E-01 45 -0.1958E-01 46  0.1074E+00

      47  0.1025E+00 48  0.9101E-01
 43          0.2300E+02  5  0.1000E+01 20 -0.1000E+01 31 -0.4833E-01 32 -0.4741E-01
      33 -0.4526E-01 34  0.2387E-01 35  0.2322E-01
 44          0.2300E+02 36  0.2217E-01 37 -0.1088E+00 38 -0.1074E+00 39 -0.1025E+00
      40  0.4833E-01 41  0.4741E-01 42  0.4526E-01
 45          0.2300E+02 43 -0.2387E-01 44 -0.2322E-01 45 -0.2217E-01 46  0.1088E+00
      47  0.1074E+00 48  0.1025E+00
 46          0.2400E+02  6  0.1000E+01 21 -0.1000E+01 31 -0.2130E-01 34  0.1181E-01
      37  0.5123E-01 40  0.2130E-01 43 -0.1181E-01
 47          0.2400E+02 46 -0.5123E-01
 48          0.2500E+02  7  0.1000E+01 22 -0.1000E+01 31 -0.3034E-01 32 -0.2130E-01
      34  0.1840E-01 35  0.1181E-01 37  0.7297E-01
 49          0.2500E+02 38  0.5123E-01 40  0.3034E-01 41  0.2130E-01 43 -0.1840E-01
      44 -0.1181E-01 46 -0.7297E-01 47 -0.5123E-01
 50          0.2500E+02  8  0.1000E+01 23 -0.1000E+01 31 -0.3417E-01 32 -0.3034E-01
      33 -0.2130E-01 34  0.1847E-01 35  0.1840E-01
 51          0.2600E+02 36  0.1181E-01 37  0.8220E-01 38  0.7297E-01 39  0.5123E-01
      40  0.3417E-01 41  0.3034E-01 42  0.2130E-01
 52          0.2600E+02 43 -0.1847E-01 44 -0.1840E-01 45 -0.1181E-01 46 -0.8220E-01
      47 -0.7297E-01 48 -0.5123E-01
 53          0.2700E+02  9  0.1000E+01 24 -0.1000E+01 31 -0.3580E-01 32 -0.3417E-01
      33 -0.3034E-01 34  0.1938E-01 35  0.1847E-01
 54          0.2700E+02 36  0.1840E-01 37  0.8811E-01 38  0.8220E-01 39  0.7297E-01
      40  0.3580E-01 41  0.3417E-01 42  0.3034E-01
 55          0.2700E+02 43 -0.1938E-01 44 -0.1847E-01 45 -0.1840E-01 46 -0.8811E-01
      47 -0.8220E-01 48 -0.7297E-01
 56          0.2800E+02 10  0.1000E+01 25 -0.1000E+01 31 -0.3648E-01 32 -0.3580E-01
      33 -0.3417E-01 34  0.1972E-01 35  0.1938E-01
 57          0.2800E+02 36  0.1847E-01 37  0.8777E-01 38  0.8811E-01 39  0.8220E-01
      40  0.3648E-01 41  0.3580E-01 42  0.3417E-01
 58          0.2800E+02 43 -0.1972E-01 44 -0.1938E-01 45 -0.1847E-01 46 -0.8777E-01
      47 -0.8811E-01 48 -0.8220E-01
 59          0.2900E+02 11  0.1000E+01 26 -0.1000E+01 34 -0.2020E-02 37  0.7768E-02
      43  0.2020E-02 46 -0.7768E-02
 60          0.3000E+02 12  0.1000E+01 27 -0.1000E+01 34 -0.2471E-02 35 -0.2020E-02
      37  0.8802E-02 38  0.7768E-02 43  0.2471E-02
 61          0.3000E+02 44  0.2020E-02 46 -0.8802E-02 47 -0.7768E-02
 62          0.3100E+02 13  0.1000E+01 28 -0.1000E+01 34 -0.2571E-02 35 -0.2471E-02
      36 -0.2020E-02 37  0.8888E-02 38  0.8802E-02
 63          0.3100E+02 39  0.7768E-02 43  0.2571E-02 44  0.2471E-02 45  0.2020E-02
      46 -0.8888E-02 47 -0.8802E-02 48 -0.7768E-02
 64          0.3200E+02 14  0.1000E+01 29 -0.1000E+01 34 -0.2594E-02 35 -0.2571E-02
      36 -0.2471E-02 37  0.8975E-02 38  0.8888E-02
 65          0.3200E+02 39  0.8802E-02 43  0.2594E-02 44  0.2571E-02 45  0.2471E-02
      46 -0.8975E-02 47 -0.8888E-02 48 -0.8802E-02
 66          0.3300E+02 15  0.1000E+01 30 -0.1000E+01 34 -0.2598E-02 35  0.2594E-02
      36 -0.2571E-02 37  0.8994E-02 38  0.8975E-02
 67          0.3300E+02 39  0.8888E-02 43  0.2598E-02 44  0.2594E-02 45  0.2571E-02
      45 -0.8994E-02 47 -0.8975E-02 48 -0.8888E-02
 68       9999999999
 69                  0
#   $.05    $2.837
#LIST-MC (COMMENT (CONSTRAINT DATA FILE USED IN MOCCA LP ALGORITHM)

      225.,225.,225.,120.,120.,120.,25.,25.,25.,
      -225.,-225.,-225.,-120.,-120.,-120.,-25.,-25.,-25.
 5
 6
 7
#   $.01    $2.8
#LOG OFF
```

19 Appendix J

Steady State Operating Data For The Evaporator Model

The following steady state operating data for the double effect evaporator were taken from Wilson's thesis (1974).

Variable	Description	Steady state
T_s	Steam temperature	177.8 °C
T_{w1}	First effect tube wall temperature	108.3 °C
W_1	First effect holdup	20.64 kg
C_1	First effect concentration	4.59% glycol
H_1	First effect enthalpy	440.1 kj/kg
T_{w2}	Second effect tube wall temperature	82.8 °C
W_2	Second effect holdup	18.82 kg
C_2	Second effect concentration	10.11% glycol
H_2	Second effect enthalpy	311.9 kj/kg
T_{w3}	Condenser tube wall temperature	42.2 °C
S	Steam flow rate	15.1 kg/s
B_1	First effect bottom flowrate	26.3 kg/s
B_2	Second effect bottom flowrate	12.0 kg/s
F	Feed flowrate	37.8 kg/s
C_F	Second effect concentration	3.2% glycol
H_F	Feed enthalpy	364.9 kj/kg