INTEGRATED SYSTEM OPTIMISATION AND PARAMETER ESTIMATION METHODS FOR ON-LINE CONTROL OF INDUSTRIAL PROCESSES

BY

SHENG CHEN

A THESIS SUBMITTED FOR THE

DEGREE OF

DOCTOR OF PHILOSOPHY

CONTROL ENGINEERING CENTRE

THE CITY UNIVERSITY

LONDON

MAY, 1986

CONTENTS

TABLES

FIGURES

ACKNOWLEDGEMENTS

I would particularly like to thank Professor P.D. Roberts, my thesis supervisor, not only for his advice, guidance and encouragements but also for his sincere and understanding character throughout my research work.

I also wish to thank [redacted] of the Technical University of Warsaw, who was a senior visiting research fellow in the Control Engineering Centre, for his many advices during my first year of the research. Some of the results in the research were obtained based on his suggestions.

Thanks are also due to [redacted] for his assistance in the computer laboratory work, and to [redacted] for his assistance with using the Prime 550 minicomputer. In addition, I would like to express my gratitude to Mr. [redacted], of The City University Computer Unit, for designing and supplying 3 channel 8-bit D to A convertors using BBC microcomputer's user port.

I am also indebted to [redacted], secretary of the Control Enginnering Centre, and other members of the Centre who assisted me while working on this project.

DECLARATION

The author grants powers of discretion to the University Librarian to allow this thesis to be copied in whole or in part without further reference to him. This permission covers only single copies made for study purposes, subject to normal conditions of acknowledgement.

PUBLICATION

The following two papers, based on the work described in this thesis, will be published soon:

An extension to the modified two-step algorithm for steady state system optimisation and parameter estimation [5];

An integrated system optimisation and parameter estimation technique for hierarchical control of steady state systems [10].

The following paper, using the work reported in the thesis, has been presented at the 1st European Workshop on Parallel Processing Techniques for Simulation:

Integrated system optimisation and parameter estimation technique using a distributed hierarchical computer system [12].

The following paper has been submitted to a Chinese Journal, Control Theory and Applications:

Comparison of several on-line integrated system optimisation and parameter estimation methods for steady state systems [11].

The following paper has been submitted to the 25th IEEE Conference on Decision And Control, Athens, Greece, Dec.10-12, 1986:

Hierarchical integrated system optimisation and parameter estimation using a microcomputer based system [14].

ABSTRACT

In this research on-line steady state optimisation techniques capable of being applied to the situation where process structure is highly uncertain have been systematically investigated. The main results obtained in the research reveal that by adopting an iterative procedure of updating model parameter values and appropriately modifying model optimisation problems the model uncertainty can be overcome and the process optimal operating condition can be attained. Algorithms for solving integrated system optimisation and parameter estimation problems have been developed by utilizing normal Lagrange analysis as a basic mathematical tool. It has been shown that a very simple model can be employed to determine the process optimum operating condition using these algorithms. If any disturbances occur at any stage during the operation (for instance, changes in raw material composition) the algorithms will adapt to proceed towards a new optimum.

This research also involves the improvement of a distributed computer network where new integrated system optimisation and parameter estimation techniques developed in the research can be studied under a real time environment. Problems associated with the practical implementation of the techniques have been discussed and methods for dealing with these problems have been suggested and tested using a microcomputer based system.

The author hopes that the work reported in this thesis will provide a useful foundation for evaluating and extending the techniques of on-line integrated system optimisation and parameter estimation both in methodology and application aspects.

SYMBOLS

symbol:	description:	Chapter:
arg	argument	2,3,4,6,7
Arg	argument	5
A	feasible set of model parameter	3,5,6
b(.)	real-valued upper semicontinuous function	5
B(.)	positive real-valued function	5
c	control (set point) vector,	2 to 7
c_i	ith element of control vector,	3 to 6
	ith subsystem's control vector	7
c_{ij}	jth element of ith subsystem's control vector	7,9
c_{opt}	real optimal control	3,4
$\hat{c}(.)$	control solution mapping of optimisation	3 to 7
C	feasible set of control	3,5,6
CA, $\overline{CA}$	feasible set of control and model parameter	3
CY	feasible set of control and output	2,3
CUY	feasible set of control, input and output	7
CUY_i	ith subsystem's feasible set of control, input and output	7,9
C_α	model feasible set of control with given α	3,5
D	constant matrix	5
e(.)	error function	2
e_i	unit vector with unity ith element	4
$E_z\{.\}$	mathematical expectation with respect to z	2
F(.)	model input-output mapping	2 to 7
$F_i(.)$	ith subsystem's model input-output mapping	7

$F^*(.)$	reality input-output mapping	2 to 7
$F_i^*(.)$	ith subsystem's reality input-output mapping	7
$F_s^*(.)$	reality input-output mapping with given state s	2
$g(.),G(.)$	constraint mapping	2 to 7
$g^*(.),G^*(.)$	reality constraint mapping	2,5,6
$g_i(.),G_i(.)$	ith element of constraint mapping,	5
	ith subsystem's constraint mapping	7
H,H_i,H_{ij}	interconnection matrices	7
$HK^*(v)$	reality interaction input vector	7
inf	infimum of	5
I	identity matrix	3,4,5,7,9
K_p	gain matrix for iterating price	7,9
K_v	gain matrix for iterating control	4,5,9
K_{v_i}	gain matrix for iterating ith subsystem's control	7
K_ξ	gain matrix for iterating Lagrange multiplier	4
K_{ξ_i}	gain matrix for iterating ith subsystem's Lagrange multiplier	7
$K^*(.)$	reality control-output mapping	7
$K_i^*(.)$	ith subsystem's reality control-output mapping	7
lim	limit of	5
L(.)	Lagrangian	3,4,7
$L_i(.)$	ith subsystem's Lagrangian	7
$L^*(.)$	(reality) Lagrangian	3
m	number of outputs	4
n	number of control inputs	4,7
N	number of subsystems	7
p	price vector	7
p_i	ith subsystem's price vector	7

p_{ij}	jth element of ith subsystem's price vector	9
$P_n(.)$	projection operator from $R^n x R^m$ to R^n	3
$q(.),Q(.)$	performance index	2 to 7
$q_i(.),Q_i(.)$	ith subsystem's performance index	7,9
Q_{in}	input flow	2
Q_{out}	output flow	2
Q_{opt}	real optimal value of performance index	3,4
$q^*(.)$	reality performance index	5,6
$\bar{q}(.)$	supremal performance index	6
R	one-dimensional real space	3,6,7
R^a	a-dimensional real space, e.g., a=m,n,p or t	2,3,5,6,9
r_j	filter parameter	4
$r_{min}(.)$	minimal eigenvalue of	5
s	state vector	2
sup	supremum of	5
T(.)	infimal algorithmic mapping	6
u	interaction input vector	7
u_i	ith subsystem's interaction input vector	7
u_{ij}	jth element of ith subsystem's interaction input vector	7,9
$\hat{u}(.)$	interaction input solution mapping of optimisation	7
$\Delta \hat{u}$	model interaction imbalance	7
v	control vector (applied to the process)	4 to 7
v_i	ith subsystem's control vector	7
$\hat{v}(.)$	infimal solution mapping	6
w	control action	2
WY	model constraint set	2

WY^*	reality constraint set	2
$x, \bar{x}$	measurement feedback	2
$\bar{x}_i$	measurement feedback to layer i	2
y	(interaction) output vector	2 to 7
y_i	ith element of output vector,	4
	ith subsystem's output vector	7
y_{ij}	jth element of ith subsystem's output vector	7,9
y^*	reality output vector	4,7
y_{ij}^*	jth element of ith subsystem's reality output vector	7,9
$\bar{y}^*$	filtered output measurement	4
z	disturbance input vector	2
α	model parameter vector	2 to 7
α_i	ith element of model parameter vector,	4
	ith subsystem's model parameter vector	7
α_{ij}	jth element of ith subsystem's model parameter vector	7,9
α^c	controller's coefficients	2
$\hat{\alpha}(.)$	solution mapping of parameter estimation	3 to 6
β_p	desired tolerance for price	7,9
β_v	desired tolerance for control	4,7,9
β_ξ	desired tolerance for Lagrange multiplier	4,7
$\hat{\delta}(.)$	component mapping of modified two-step algorithm	5
ε	small positive scalar	5
$\varepsilon_m, \varepsilon_M$	minimum and maximum perturbation on control	4
η	Lagrange multiplier	4 to 7
θ or θ_v	gain parameter for iterating control	4,5
θ_ξ	gain parameter for iterating Lagrange multiplier	4

λ	modifier vector	3 to 7,9
λ_i	ith element of modifier vector,	4
	ith subsystem's modifier vector	7
λ_{ij}	jth element of ith subsystem's modifier vector	9
λ_{opt}	optimal supremal solution	6
$\bar{\lambda}$	filtered modifier vector	4
$\tilde{\nu}(.)$	component mapping of modified two-step algorithm	5
ξ or $\bar{\xi}$	Lagrange multiplier	3 to 7
ξ_i	ith subsystem's Lagrange multiplier	7
$\hat{\xi}(.)$	Lagrange multiplier associated with optimisation solution	4 to 7
ξ_{opt}	Lagrange multiplier associated with real optimum	3,4
ρ , ρ_1, ρ_2	Lipschitz constants	5,6
σ	positive scalar	5
σ_i	perturbation on ith element of control vector	4
τ	positive scalar	5
$\Phi(.)$	algorithmic mapping of modified two-step algorithm	5
$\Psi_p(.)$	iterative strategy for price	7
$\Psi_v(.)$	iterative strategy for control	4
$\Psi_{v_i}(.)$	iterative strategy for ith subsystem's control	7
$\Psi_\xi(.)$	iterative strategy for Lagrange multiplier	4
$\Psi_{\xi_i}(.)$	iterative strategy for ith subsystem's Lagrange multiplier	7
$\hat{\omega}(.)$	component mapping of modified two-step algorithm	5
Ω	algorithm solution set	5
Ω^*	solution set of real optimisation problem	3,5

$\mathcal{C}$	finite-dimensional real space of control	7
$\mathcal{C}_i$	finite-dimensional real space of ith subsystem's control	7
$\mathcal{U}$	finite-dimensional real space of interaction input	7
$\mathcal{U}_i$	finite-dimensional real space of ith subsystem's interaction input	7
$\mathcal{Y}$	finite-dimensional real space of output	7
$\mathcal{Y}_i$	finite-dimensional real space of ith subsystem's output	7
$\mathcal{A}$	supremal constraint set	6
$\bar{\mathcal{A}}$	finite-dimensional real space of model parameter	7
$\bar{\mathcal{A}}_i$	finite-dimensional real space of ith subsystem's model parameter	7
$\in$	is member of	2 to 7,9
Σ	sum of	4,7
$(.)^T$	transpose of	3,4,5,7,9
$\frac{\partial}{\partial}(.)$	derivative of	3,4,5,7
$\nabla\,(.)$	gradient of	3,4,5,7
$(.)^{-1}$	inverse of	4,5,7
$\|.\|$	absolute value of	4,6,7,9
$\|\|.\|\|$	Euclidean norm	5,6
$\forall$	all of	3

CHAPTER 1. INTRODUCTION

1.1 Systems control in industrial processes

Control systems play an ever-increasing role in improving the quantity and quality of manufactured products. As our manufacturing activities get more and more automated, the importance of using increasingly more refined and advanced control technology becomes evident. The most important technological advance that we are currently in the process of seeing is a revolution that is taking place in the processing of information and it already has a major impact on industrial control. More and more control functions in three basic categories of industrial control-direct control, supervisory control and sequencing control-are being taken by microprocessors because of the advantages of lower cost, fewer components, increased reliability and versatility offered by microprocessing systems.

Direct (regulatory) control is concerned with ways to keep the values of controlled process variables as near to their set points as possible. The set points are either fixed or changing according to some prescribed programme which should be able to take account of the changing conditions in the process. As far as the direct controllers are concerned, the set points are some given values. Sequencing control, on the other hand, is concerned with carrying out a series of process steps in a prescribed order, such as starting and stopping an industrial process. Supervisory control refers to a methodical quantitative approach to determine the best set points for the direct controllers, either as functions of time or, more generally, as

functions of the inputs to the system; such inputs might be raw material properties, product specifications, external disturbances, and economic factors, etc.

As microcomputers are becoming cheaper and more powerful, and as the continuous advance is being made in computer network theory, a distributed control system is not only a concept but also a reality in industry. Many sophisticated hierarchical decentralized techniques appear. The new emerging notions are subsystem, interconnection, decentralization, coordination, synchronization, distributed computer networks, etc. Despite numerous researches, refining these techniques and applying them to industry remains to be a widely open area. Many researches give emphasis to distributed hierarchical optimisation techniques. This is no doubt due to increasing pressure upon industry to improve efficiency and productivity and to reduce pollution, and due to the capability of computation provided by new microcomputer technology.

1.2 Steady-state optimisation

A typical application of supervisory control is aimed at optimizing a chosen performance index under changing external or process conditions. It is assumed that the optimal operating condition changes slowly as external factors change. The supervisory control system then finds and tracks the optimum condition. When a performance index has been chosen the key step in such supervisory control is to construct a model which relates the set points to the external and process conditions. Various identification techniques may be used to

determine model structure and model parameter values. If a model is obtained the problem becomes a standard linear or nonlinear programming problem and can readily be solved by mathematical programming methods, including hierarchical decomposition techniques, on computers. However, in general, the solution obtained is a model optimum which may be far from the process optimum and may even violate the real system constraints because the model is only an approximation to the real process due to uncertain knowledge of the process structure and deliberate simplification for the sake of computation.

Some attempts have been made to develop the techniques of direct on-line optimisation using measurements. Only very limited cautious reports of success have been heard. In practice, direct on-line optimisation using measurements involves severe difficulties. Firstly, such algorithms are very inefficient because of process dynamic characteristics which are often slow. Secondly, the presence of noise can produce erratic behaviour of the algorithms. Thirdly, during the optimisation procedure the algorithms may demand totally unacceptable operating conditions from the process.

Efforts have been focused on developing optimisation techniques which are based on a mathematical model and use measurement feedback to improve the model based solution. For example, in the large-scale system case Findeisen and co-workers (1980) gave a variety of methods which employ a (fixed) model and measurement feedback to form an iterative procedure. While the solution obtained using these methods is often better than the purely model based solution it is, in general, not the real optimum solution.

Other techniques attempt to overcome the model uncertainty by employing an iterative procedure between model parameter estimation and model optimisation (for example, Durbeck,1965; Foord,1974; Youle and Duncanson, 1970). However, simply combining two separate optimisation and estimation steps does not guarantee that the real optimum operating condition will be achieved as will be shown in the following chapters.

1.3 Scope of the research

A fundamental question which has to be answered by any on-line process optimisation technique is that under what conditions the solution given by the optimisation technique is also the solution of the real process optimum. Durbeck (1965) and Foord (1974) formulated sufficient conditions for a model based optimisation solution to give the correct process optimum. Part of the conditions demands that the derivatives of model outputs with respect to the controller set points match exactly the corresponding derivatives in the real process at the solution of model optimisation. This demand, in general, requires that the model structure (kind and nonlinear degree of equations, etc.) is the same as that of the real process. For many industrial processes their structures are highly uncertain and, hence, such a requirement is obviously unrealistic.

Roberts (1978) introduced the modified two-step method which involves an iterative procedure of updating model parameters and modifying the model optimisation problem. The main characteristic of the method is that the optimisation objective index is modified by an extra term which contains a comparison between model and process output

derivatives with respect to controller set points. Therefore, by measuring the differences of these two sets of derivatives, the real optimal steady-state operating condition can be achieved in spite of model inaccuracies, or at least the solution obtained is very close to the real optimum if the derivatives are poorly approximated.

The modified two-step method has been successfully applied to many example problems, for example, using computer simulation to study its application in the steady-state optimisation control of a chemical reactor (Roberts, 1979; Roberts and Williams, 1981; Ellis and Roberts, 1981). However, four main aspects concerned with this method remained to be exploited and further studies were needed. Firstly, it can be shown that the optimality of the original method given in (Roberts, 1978, 1979; etc.) is restricted to a class of problems where process inequality constraints do not depend on process outputs. Examples can be found to demonstrate that if inequality constraints depend on process outputs the original method may fail to have a feasible solution (Brdys, Chen and Roberts, 1984). Secondly, the convergence conditions needed to be derived, which was a difficult task due to the complex adaptive nature of the method. Thirdly, the method is very sensitive to noise measurements because it uses the real output derivatives, and the differentiation amplifies the errors in measurements. The question of how to attenuate the influence of measurement noise naturally arises. Finally, could the method be extended to solve the interconnected large-scale system problems? If the answer is yes a new useful tool to tackling complex large-scale system problems will be provided.

This thesis is aimed at a study of these aspects. An extension to the original method, proposed by Brdys', Chen and Roberts (1984), removes the restriction on inequality constraints. Global convergence conditions, at least for a class of problems where the inequality constraints do not involve the process outputs, have been derived (Brdys' and Roberts, 1984b). Brdys' suggested, in a manuscript, the original idea of a two-level method which does not require the knowledge of the real output derivatives and satisfies the optimal conditions in a similar manner to that of the modified two-step method. This idea was later developed into a two-level algorithm and was compared with the modified two-step algorithm (Chen, Roberts and Brdys', 1984). Michalska, Ellis and Roberts (1985) presented an algorithm for interconnected subsystems by combining the modified two-step algorithm with the price correction mechanism. They had difficulties in dealing with inequality constraints and made some assumptions which actually require that the inequality constraints do not depend on the process outputs. Brdys' and Roberts (1984a) removed these unnecessary assumptions and developed a group of hierarchical adaptive optimal algorithms. One of these algorithms was incorporated with a double iterative technique based on previous work of Shao and Roberts (1983) to achieve the benefits of reducing the time for determining the optimal operating condition (Chen, Brdys' and Roberts, 1985).

This research not only concentrates on the formulation and study of several on-line integrated system optimisation and parameter estimation methods but is also involved with the development of a distributed computer network where the methods can be tested under a real-time environment (Chen, Wadhwani and Roberts, 1985). A new

integrated system optimisation and parameter estimation technique developed in the research has been implemented using part of the network and encouraging results have been obtained (Chen, Roberts and Wadhwani, 1985; Chen, Wadhwani and Roberts, 1986). The purpose of this experiment is to study problems associated with real-time application of the technique and to develop counter measures. The experience gained will benefit the application of the technique to industrial processes.

1.4 Outline of the thesis

In Chapter 2 the multilayer approach is employed to decompose an industrial control system into a multilayer hierarchy. Two important aspects--the model-reality differences and the complexity of the control system are emphasized. The analysis in this chapter provides a broad background where the research reported in this thesis is based on.

Interdependence of system optimisation and parameter estimation revealed in the analysis of Chapter 2 gives rise to the necessity of integrated system optimisation and parameter estimation (ISOPE). In Chapter 3 a class of ISOPE problems are defined, and basic definitions and theories related with ISOPE problems are presented.

A systematic formulation of the modified two-step method is presented in Chapter 4. The extension to the original formulation (for example, Roberts, 1978, 1979) is emphasized, implementation aspects of this method are discussed and computer off-line simulation results are

presented.

Optimality and global convergence conditions of the modified two-step algorithm are investigated in Chapter 5. The convergence theories in this chapter are based on the paper (Brdys and Roberts, 1984b). It is shown that there are advantages of using simpler (linear) models when solving the nonlinear steady-state optimal control problems by the modified two-step algorithm.

In Chapter 6 a two-level ISOPE method is investigated. Due to its structure, the convergence rate of this algorithm is slower than that of the previous modified two-step algorithm. However, this algorithm does not require real process output derivatives, and consequently, it may be preferred when the process measurements are contaminated with noise. Both the modified two-step algorithm and the two-level type algorithm can be viewed as two modified versions of the basic two-step method given in Chapter 3.

The techniques presented in Chapter 3 (two-step algorithm), Chapter 4 (modified two-step algorithm) and Chapter 6 (two-level type algorithm) can be combined with a price correction mechanism to yield a group of adaptive algorithms for hierarchical control of steady-state systems. To demonstrate this, following from the ideas of Michalska, Ellis and Roberts (1985) and Brdys and Roberts (1984a), in Chapter 7 a hierarchical structure is formulated by combining price coordination and the modified two-step techniques. It is important to note that this is by no means the only structure that can be derived by combining price coordination and the modified two-step methods (see, Michalska,

Ellis and Roberts, 1985; Brdys and Roberts, 1984a). The structure given in this thesis has shown a fast convergence rate in computer simulation studies. One of the important results revealed in this research is that by using a double iterative scheme (Shao and Roberts, 1983) the time for determining the optimum steady-state condition of a large-scale process can be significantly reduced.

In Chapter 8, a distributed computer network and its applications to systems control are briefly described. Serial data communication between a DEC LSI11/23 minicomputer and an Acorn BBC microcomputer is discussed.

A real-time simulation study of the algorithms formulated in Chapter 7 is presented in Chapter 9. The experiment was conducted on a small distributed computer system where an analogue computer simulated two interconnected subsystems, a LSI11/23 minicomputer served as the coordinator and two BBC microcomputers played the roles of local control units.

Chapter 10 contains some final conclusions of this research and suggestions for further research in the area of integrated system optimisation and parameter estimation.

CHAPTER 2. CONTROL OF INDUSTRIAL PROCESSES

2.1 Introduction

During the last two decades considerable advances have been made in the field of process control. There is an increasing interest in maintaining an industrial process at its optimal operating condition efficiently. This may be due to the increasing limitations on natural resources and rising costs of energy and raw materials. Furthermore, the improvement in microcomputer technology has a great impact on this subject.

The traditional concept of control, when applied to industrial systems, concerning the problem of how to manipulate inputs to the process so that (a) output variables follow the predetermined time trajectories or (b) the system state is transferred (optimally) from some initial value to a specified final value, has been broadened to include all aspects of decision-making applied to industrial systems operating in real time.

In this chapter we study some basic elements of the industrial process and utilize the multilayer concept (Lefkowitz, 1966, 1977) to split the task of determining control into a multilayer hierarchy. The interdependence between the twin problems of process optimisation and parameter estimation arises naturally within the hierarchical structure, which indicates the need for joint treatment of these two problems.

2.2 Some elements of the industrial control system

Control is a means of influencing an object to behave in a desired way. The common characteristic of the control function, hence, is the basing of responses, decisions, actions, etc. on information describing the state of the system (and its environment) interpreted through the medium of appropriate models relevant to system performance. The term controller, therefore, is used to denote the agent by which the control functions are carried out.

A deeper understanding of the basic elements of a control system--their functions and their relationship to each other, will certainly help to design a better control system. Referring to Fig. 2.1., some basic elements of a control system are defined based on previous work of Lefkowitz (1977).

2.2.1 Plant

The term plant is usually referred to the controlled system or means of production. It may be a processing unit like a topping unit in the crude distillation process, a whole chemical factory, etc. The fundamental assumptions are that the plant can be described (in principle) by a set of input-output relations and some of its inputs are free to be selected by a decision-maker or controller so as to influence the plant's behaviour in a desired way. The variables associated with these relations (or with the plant) then can be classified as follows:

(1) Disturbance inputs (disturbances) z. These are inputs which

represent the effects of interactions of the plant with its environment, and which are beyond our influence. Disturbances should essentially be considered random variables. In general, a disturbance input causes the plant outputs to deviate from desired behaviour and hence motivates control action.

(2) Contol inputs w. These are the results of the decision-making by the controller which are intended to compensate for the undesired influence of disturbances z or to change the outputs to follow the order r from some higher level organisation.

(3) Outputs y. These are the plant's response to the inputs. Apart from functionally dependent on the inputs, we would prefer the plant outputs to be available explicitly either through direct measurement or through inference (via a model) from the measurement of other variables, and to be relevant with respect to the evaluation of plant performance so that they can be used by the controller in determining its decisions or control actions and in observing the effectiveness of a control policy or action.

(4) State variables s. These are variables associated with the memory characteristics of the plant such as the status of energy or material storages in the plant. In the dynamic case, the state vector compacts the past history of the plant, such that, knowing the inputs to the plant over a given time interval, the outputs are determined uniquely over that same period. In a steady-state condition, the state vector is one of the main factors which define the structure of the input-output relations of the plant, i.e., at different static states the input-output relations may have different forms.

Through the above discussion we may write the general form of the

plant input-output relations

$$y = F_s^*(w, z), \tag{2.1}$$

where y, w and z denote output, control and disturbance vectors, respectively. The dependence on the state vector is indicated by the index s. If the state vector changes, the functional relation F_s^* may also change. Because of technical limitations and because of the limitations imposed by the environment, the values of w and y cannot be arbitrary, rather, they are likely to be limited in some set of values. In most cases these limitations can be described by a set of inequality constraints

$$(w, y) \in WY^* \triangleq \{(w, y): G^*(w, y) \leq 0\}. \tag{2.2}$$

Thus, we adopt, a plant is represented by the relations (2.1) and (2.2). On the other hand, our understanding of the plant is referred to as the plant model. Consequently, we may write the plant model

$$y = F(w, z, \alpha), \tag{2.3}$$

$$(w, y) \in WY \triangleq \{(w, y): G(w, y) \leq 0\}, \tag{2.4}$$

where α is a parameter vector which has been determined by some technique. Usually, the constraints imposed on the plant are known explicitly, therefore, it may be reasonable to assume that (2.4) is exactly the same as (2.2). However, our knowledge of the plant input-output relations cannot be perfect due to the nature of time-varying unknown disturbances and inaccurate knowledge of the state vector. As the result, differences between (2.1) and (2.3) inevitably exist. We refer to this as model-reality differences. Throughout our discussion we will continuously emphasize the existence of such differences.

2.2.2 Environment

The plant can be viewed as a subsystem of a larger system which we term the environment, i.e., all aspects of the external world that interact with the plant. The interactions, in general, are mutual. While the environment affects the plant through disturbances z and order r (which affects the objectives or the constraints to be applied by the controller acting on the plant) the plant's responses (outputs) also have influence on the environment. As far as the controller acting on the plant is concerned, only the performance of the plant is of importance. How the outputs affect the environment (the higher decision-making units may be concerned about this) is beyond its interests.

For example, a controller in charge of keeping constant the level of fluid within a tank (Fig. 2.2.) may try to change Q_{out} as fast as possible responding to a sudden change in Q_{in}. The decision-maker in charge of the whole process, however, may worry about the influence of the quick changing rate in Q_{out} to the down-stream units and hence imposes some constraints on the changing rate. In the view of the whole process, Q_{out} is considered as an interaction input to the down-stream units.

2.2.3 Performance index

Since control decisions aim at achieving a certain goal it is necessary to define some criterion for measuring the degree that the goal is achieved. The practical criterion will be based generally on

the approximation and simplification of technological requirements and economic demands. Such an approximation and simplification is motivated by the limited capability of computational tools, the form and accuracy of the model used, the nature of the information required and the reliability of the data available.

It is assumed that the aim of optimizing the performance of the control system has been simplified as that of minimizing (or maximizing) a scalar function which is known as the performance index

$$Q = Q(w, y, z), \tag{2.5}$$

where w, y and z are defined as in 2.2.1. The problem of determining the control w is referred to as deterministic if we assume that z has a known value or a known function of time. If the probabilistic properties of z are known and we make an appropriate use of them, the control problem is stochastic, and we may set the problem target to be the minimization (or maximization) of the expected value of Q:

$$\min_{w} E_{z}\{Q\}.$$

If we use the deterministic approach, it is important to investigate how the control system behaves when the real disturbance differs from the value used in the model. We must require that the performance of the control system is relatively insensitive to this model-reality difference.

2.2.4 Controller

The general purpose of the controller is to generate the control inputs w applied to the plant which result in an optimum value of the

performance Q while satisfying the variety of constraints imposed on the system. Also the controller should be able to give non-normal responses to ensure the safety of the system in contingency situations.

In order to achieve the control goal, the controller needs: (1) to determine the structure of the plant model and to set the optimal values of the parameters in the model with respect to some objective index according to the information obtained through the measurements; (2) to make control decisions, i.e., to determine the best controller set points based on the model; (3) finally, to implement the control decisions. A good controller should be able to compensate for the model-reality differences so that the controller is robust when available information is imperfect. We can see that the controller requires complex multi-function capabilities.

2.2.5 Information processor

As noted above, the decisions are based on the information gathered. The quality of decisions depends on the quality of information. Some major functions of the information processor include:

(1) Gathering data and distributing them to points of usage (including sensors, data input devices, communication links, etc.).

(2) Processing raw data into the form required by the controller, e.g., data smoothing, noise filtering, making inference of the values of variables which cannot directly be measured.

(3) Storing the information required for operating the system.

The design of a control system actually involves the design of a

controller and an information processor.

2.3 Multilayer control hierarchy

As discussed in 2.2.4, the task of determining and realizing the control w is a complex and, maybe, formidable problem. The multilayer control hierarchy (Lefkowitz, 1966, 1977) provides a rational and systematic procedure for resolving it. In the multilayer control hierarchy (see Fig. 2.3.) the overall problem is split into a set of subproblems which are identified by the four layers of the hierarchy, namely, implementation, optimisation, adaptation and organization layers. In this section, the functions of each layer are briefly discussed.

2.3.1 Implementation layer (regulation layer)

This first layer plays the role of implementing the decisions given by the second layer (optimisation layer) and copes with the dynamics of the plant. The control w is generated in this layer according to the rule

$$w = w(\bar{x}_1, c_d, \alpha^c) \qquad (2.6)$$

where $\bar{x}_1$ denotes the feedback information from the plant, c_d denotes the controller set points given by the second layer and α^c denotes the parameters associated with this layer. Conventional PID regulation or follow-up control can be considered as one kind of implementation of this layer in which control is generated to cause the (appropriately chosen) regulated variables to follow their desired values c_d. In this case, α^c is the vector of controller's PID coefficients.

One important consequence must be noted. The dynamics of the problem and fast disturbance inputs may be effectively "absorbed" at the first layer so that the system virtually operates in the steady state (or more precisely, in a sequence of steady states). Such a situation is very common in process control. For example, in a continuous chemical process, the process variables such as temperatures, pressures, concentrations, etc. may have to be varied from time to time to adjust to some slow disturbances, say, changes in raw material composition. The new desired values c_d for these process variables may be determined by a steady state approach. On the other hand, the process control inputs w, such as valve positions, have to be adjusted much more frequently in response to fast disturbances, such as heating gas pressure. This adjustment can be achieved by the regulation layer which tries to keep the process temperatures, pressures, etc. to their desired and constant values c_d as prescribed by the optimisation layer.

The above discussion is the foundation for choosing a steady-state approach in the higher layers. For convenience, we will term the plant together with its implementation control layer as real process or simply reality.

2.3.2 Optimisation layer

The optimisation layer is the agency where control decisions are determined, which then will be implemented by the implementation layer. The basic information available to this layer is a model (including the model structure, predetermined parameter values and estimated

disturbance values) and a constraint set. The model is valid only in the neighbourhood of a given "state" or set of circumstances. As these change with time, it is necessary to update the model. The updating is carried out by the third and/or fourth layers. This means that we can eliminate from the optimisation problem formulation those factors or disturbance inputs which tend to change infrequently relative to the period of control action in the optimisation layer, since they may be compensated through the higher layers. Based on this justification we will assume that the disturbances are constant in the single run of determining desired controller set points and, hence, will be dropped to simplify the notation.

The task of the optimisation layer may be described as the model based optimisation problem of finding the desired controller set points c_d such that

$$c_d = \arg\min_{c} \Omega(c, y) \tag{2.7}$$

$$\text{(MOP)} \quad \text{s.t. } y = F(c, \alpha) \tag{2.8}$$

$$(c, y) \in CY \overset{\Delta}{=} \{(c, y) \in R^n \times R^m : G(c, y) \leq 0\} \tag{2.9}$$

where Ω is the performance function, F are the model equations, G are the constraint functions and α are given model parameters.

It is worth pointing out that the feedback information from the real process is necessary to improve the quality of the control decisions because of the model-reality differences. How the feedback information is utilized may distinguish various on-line optimisation techniques.

2.3.3 Adaptation layer

The fundamental task of the adaptation layer is estimating parameter and disturbance values in a given model structure which then will be used in making control decisions in the optimisation layer. Since according to our assumption the disturbances are used as constants in the optimisation layer they may be viewed as some kind of parameters.

Consider the following model parameter estimation technique where the model parameter values are determined by minimizing a given comparison index e(.) of the difference between the model and reality outputs.

$$\min_{\alpha} e(F^*(c) - F(c, \alpha)) \tag{2.10}$$

(MEP) s.t. $y = F(c, \alpha)$ (2.11)

$$G(c, y) \leq 0, \tag{2.12}$$

where $F^*(c)$ is the real output vector, $F(c, \alpha)$ is the model output vector. A weighted least squares criterion may be employed as the comparison index.

The procedure of the parameter estimation involves: Choose a suitable set of controller set points c_e, apply c_e to the real process, measure the corresponding outputs $F^*(c_e)$, then adjust the parameter to optimize the comparison index. However, guidelines are required on how to choose a suitable c_e. Ideally, the adaptation layer should be able to give "real optimal" parameter values such that the lower optimisation layer can use these parameters in the decision-making process to

achieve the best control decision which is the optimal operating condition in the real process. It is clear that if a random set of c_e is used the parameter values calculated accordingly may not achieve such a desired goal.

2.3.4 Organization layer

When the conditions in which a model is valid have changed significantly, updating the model parameters may not be enough to ensure the model is a good approximation of the reality. In this situation, the organization layer acts to adjust the model structure. Also the organization layer should be able to judge the appearance of contingency events and to shift the system from normal operation to emergency operation. It is recognized that the organization layer acts much less frequently compared with the lower layers.

Human intervention in the control system is mainly through the organization layer, e.g., managing, planning, operating, etc. The functions performed by man including those requiring judgements that cannot be standardized, or decision processes that have not been adequately established, or coordinations that involve the integration of a great many factors whose subtleties or non-quantifiable attributes defy computer implementation.

2.4 Interdependence of optimisation layer and adaptation layer

The coupling relationship between the optimisation layer and adaptation layer is inherent in the nature of searching for an "optimal

solution". Consider the task of obtaining the real optimum values of the controller set points when a model structure with some unknown parameters is given. Assume that the following two steps are performed: First solve the model parameter estimation problem (MEP) to estimate the parameter values and then solve the model optimisation problem (MOP) with these parameter values to determine the controller set points. Obviously, such a procedure cannot generate the real optimal set points unless the model structure is exactly the same as that of the real process with only the real values of some parameters unknown. Moreover, the controller set points generated may be unacceptable to the real process when the model-reality structure differences are significant.

It is apparent that interaction between the optimisation layer and the adaptation layer naturally exists. In general, the two layers interact in that the solution of the optimisation problem is dependent on the values of the model parameters and the parameter estimates will change according to the controller settings. If the parameters are estimated under non-optimal conditions, accordingly, the solution generated from such models is non-optimal.

2.5 Summary

In this chapter we have examined the components of the control system and the functions of the controller. A multilayer approach is used to analyze a modern control system which is really a multi-function complex. Two facts are particularly worthy of emphasising: The model which is our understanding of the reality is

unlikely a perfect representation of the reality; the interdependence of the optimisation layer and the adaptation layer inevitably exists. The coupling relationship of these two layers is strongly strengthened because of the existence of the model-reality differences.

The discussion in this chapter gives a wider environment where the research of this thesis is within. The interaction between the optimisation and adaptation layers is a great challenge to on-line optimisation techniques and motivates the idea of integrated system optimisation and parameter estimation. One of the main tasks in this research is to develop some on-line methods which can tackle the problems of optimisation and parameter estimation revealed in this chapter.

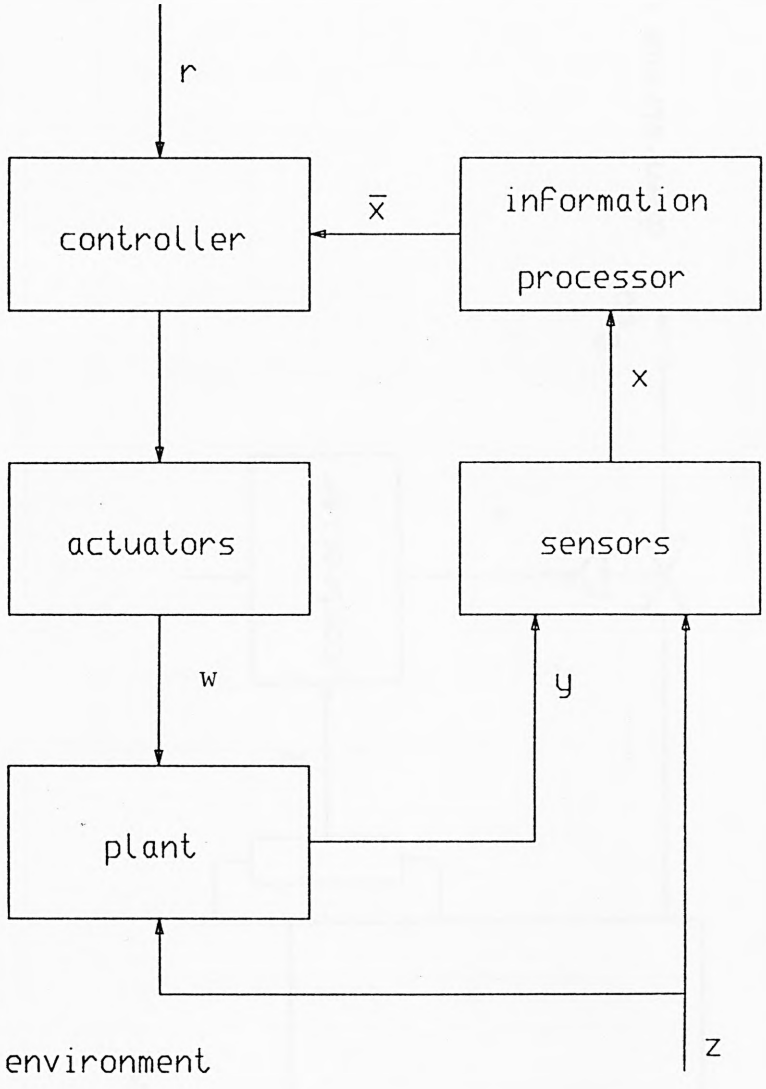


Fig. 2.1. Basic elements of control system

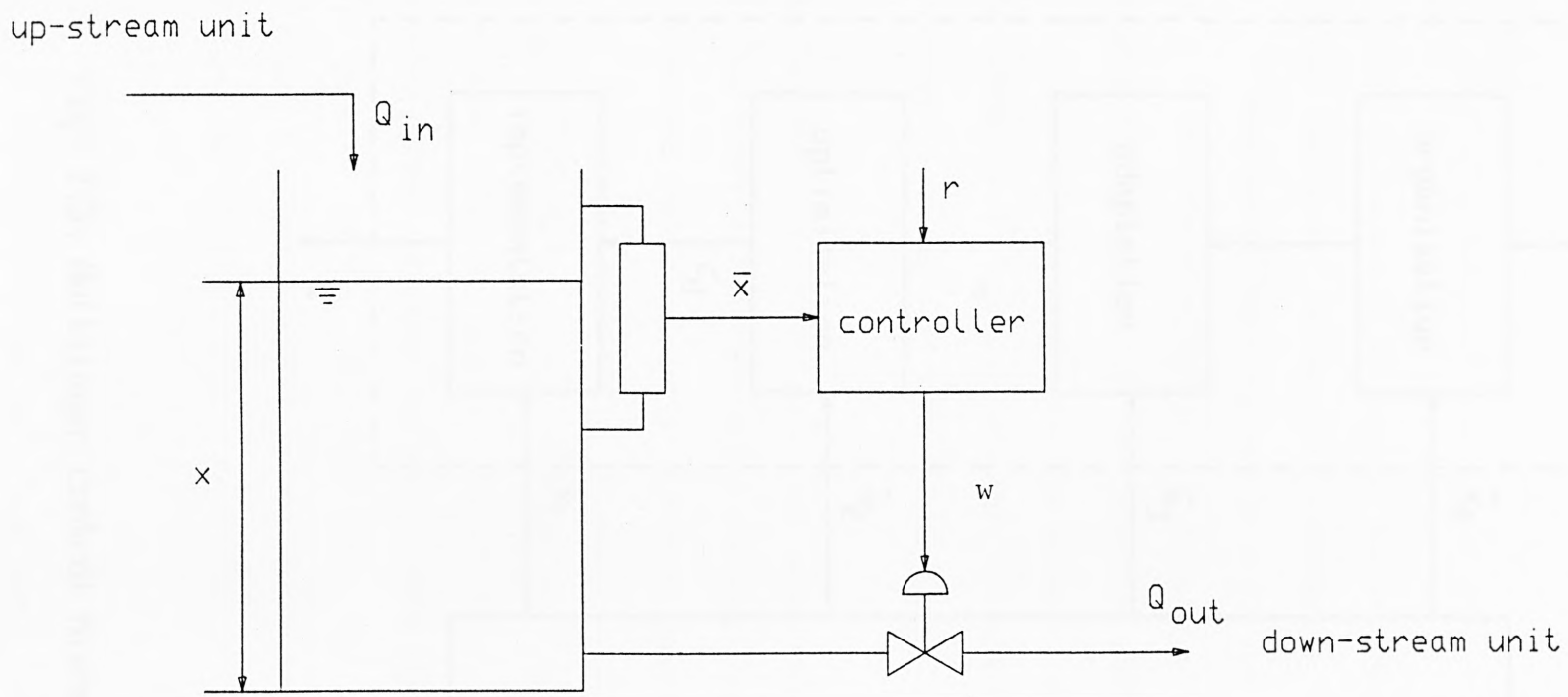


Fig. 2.2. Level control of a tank

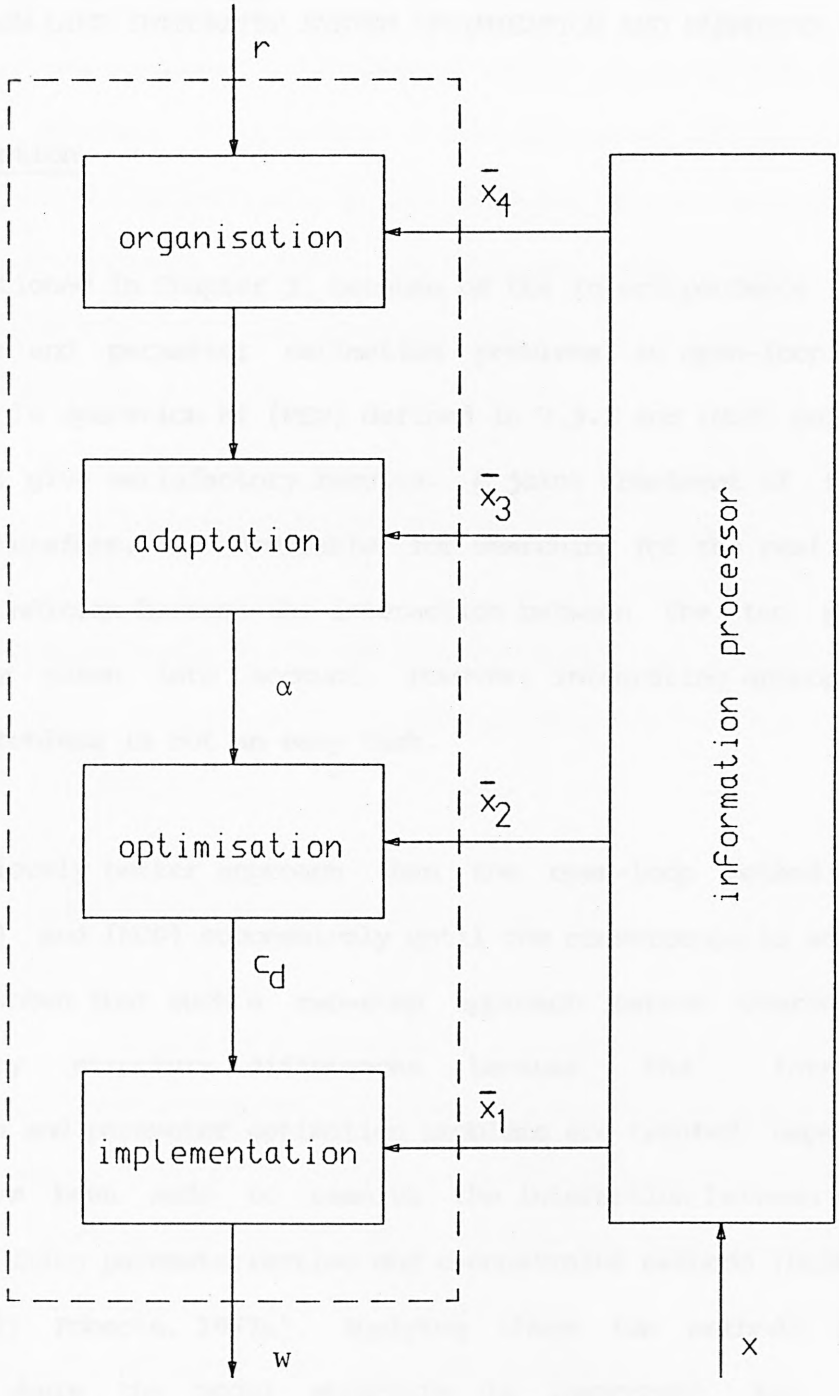


Fig. 2.3. Multilayer control hierarchy

CHAPTER 3. ON-LINE INTEGRATED SYSTEM OPTIMISATION AND PARAMETER ESTIMATION

3.1 Introduction

As mentioned in Chapter 2, because of the interdependence of the optimisation and parameter estimation problems, an open-loop method, i.e., a single operation of (MEP) defined in 2.3.3 and (MOP) defined in 2.3.2 cannot give satisfactory results. A joint treatment of the two problems, therefore, is desirable for searching for the real optimum operating condition because the interaction between the two problems can then be taken into account. However, integrating appropriately these two problems is not an easy task.

An obviously better approach than the open-loop method is to repeat (MEP) and (MOP) successively until the convergence is achieved. It is well-known that such a two-step approach cannot overcome the model-reality structure differences because the interacting optimisation and parameter estimation problems are treated separately. Attempts have been made to resolve the interaction between the two problems by using parameterization and ε-constraint methods (Haimes and Wismer, 1972; Roberts, 1977a). Applying these two methods in the situation where the model structure is inaccurate has proved unsuccessful (Roberts, 1977b).

A control scheme (Fig. 3.1.) which has found on-line practical application in industry (Youle and Duncanson, 1970) involves determining the optimum operating condition (controller set points) of the real process using a model and implementing the condition on the

process through regulatory controllers. The model contains parameters whose values are estimated by comparing output responses from both the model and the process. This defines the parameter estimation problem. In turn, the model is employed to determine the optimum controller set point values to optimize a given performance index. This defines system optimisation. In the next section the integrated formulation for two such problems is given.

3.2 Description of a class of ISOPE problems

The task of determining the optimum controller set points c for a real process can be defined as the following real optimum problem:

$$\min_{c} Q(c,y) \tag{3.1}$$

$$\text{(ROP)} \qquad \text{s.t. } y = F^{*}(c) \tag{3.2}$$

$$(c,y) \in CY \triangleq \{(c,y) \in R^{n} \times R^{m} : G(c,y) \leqslant 0 \} \tag{3.3}$$

where $Q: R^{n} \times R^{m} \rightarrow R$ is a given performance function, c and y are the control and output vectors, respectively, $F^{*}: R^{n} \rightarrow R^{m}$ is the real process input-output mapping, and $G: R^{n} \times R^{m} \rightarrow R^{p}$ are the process inequality constraints. Let us denote a set $C \triangleq P_{n}(CY)$, where P_{n} is the projection operator from $R^{n} \times R^{m}$ to R^{n}, and the solution set of (ROP) as Ω^{*}. C is known as the feasible control set. The following two definitions are introduced.

Definition 3.1 (feasible control)

A feasible control c_{f} is such that $c_{f} \in C$.

Definition 3.2 (real optimum control)

A real optimum control c_{opt} is such that $c_{opt} \in \Omega^{*}$.

Solving (ROP) directly on-line to find a c_{opt} is inefficient because of the presence of process noise and dynamic characteristics which are often slow. However, we can introduce a process model and formulate another optimisation problem:

$$\min_{c,\ \alpha} Q(c, F(c, \alpha)) \tag{3.4}$$

(OP1) $$\text{s.t. } F(c, \alpha) = F^*(c) \tag{3.5}$$

$$(c, \alpha) \in CA \triangleq \{(c, \alpha) \in R^n \times R^t : G(c, F(c, \alpha)) \leq 0\} \tag{3.6}$$

where $F: R^n \times R^t \to R^m$ is the model input-output mapping which is only an approximation to F^* because of uncertain knowledge of the real process structure and deliberate simplification, and α is the model parameter vector.

From the point of view of the equality (3.5) a very simple property required for a chosen model is such that given a point in the feasible set C, we can always match exactly the model and real outputs by an appropriate choice of parameter values. Brdys(1983) named such a property as point-parametric.

Definition 3.3 (point-parametric)

The process model is point-parametric at $\bar{c} \in C$ if there exists $\bar{\alpha} \in R^t$ such that $F(\bar{c}, \bar{\alpha}) = F^*(\bar{c})$. The model is point-parametric on C if it is point-parametric at every point $c \in C$.

The importance of introducing (OP1) becomes clear in Lemma 3.1.

lemma 3.1

If the process model is point-parametric on C then (ROP) and (OP1) are

equivalent.

Proof Let $(\bar{c}, \bar{\alpha})$ solves (OP1). Then $\bar{c}$ solves (ROP). Otherwise, there exists $\tilde{c} \in C$ such that $Q(\tilde{c},F^*(\tilde{c})) < Q(\bar{c},F^*(\bar{c}))$. But for $\tilde{c}$ there exists $\tilde{\alpha}$ satisfying $F^*(\tilde{c}) = F(\tilde{c}, \tilde{\alpha})$ and $(\tilde{c}, \tilde{\alpha}) \in CA$. Therefore, $Q(\tilde{c}, F(\tilde{c}, \tilde{\alpha})) < Q(\bar{c}, F(\bar{c}, \bar{\alpha}))$ which contradicts the fact that $Q(\bar{c}, F(\bar{c}, \bar{\alpha}))$ is the minimum.
Similarly let c^* solves (ROP). Then there exists α^* satisfying $F^*(c^*) = F(c^*, \alpha^*)$ such that (c^*, α^*) solves (OP1). Q.E.D.

The significance of Lemma 3.1 is that in order to find a c_{opt} one can solve (OP1) instead of (ROP). Based on (OP1) it is comparatively easy to construct some on-line algorithm to find, at least, a good approximation of c_{opt}. A point-parametric model on C can always be designed easily, e.g., let t = m and choose $F(c, \alpha) = f(c) + I\alpha$, where I is the mxm identity matrix.

3.3 Two-step method

A basic method for solving (OP1) is to employ a two-step approach (Durbeck, 1965; Foord, 1974). The on-line implementation (Fig. 3.2.) of the method applied to (OP1) is given as the following algorithm:
Start from some initial control $c^0 \in C$.
Step1 Apply c^k to the real process. After the transient response of the real process is finished the steady-state measurements $F^*(c^k)$ are taken. Then estimate parameter values α^k by solving the equation

$$F(c^k, \alpha) = F^*(c^k). \qquad (3.7)$$

Step2 With fixed α^k solve the model optimisation problem to find a new vector of set points c^{k+1}

$$\min_{c} \Phi(c, F(c, \alpha^k)) \tag{3.8}$$

$$\text{s.t. } G(c, F(c, \alpha^k)) \lesssim 0. \tag{3.9}$$

The iteration is terminated when $c^{k+1} = c^k$.

Obviously if the algorithm converges, the real constraints will be preserved. This is because the control $\bar{c}$ obtained satisfies

$$G(\bar{c}, F(\bar{c}, \bar{\alpha})) \lesssim 0 \text{ and } F(\bar{c}, \bar{\alpha}) = F^*(\bar{c}).$$

Lemma 3.2 (feasibility)

The control obtained by the two-step method satisfies the real process constraints.

Sufficient conditions for the applicability of this basic method are concerned with the existence of a solution. Assume that the model is point-parametric on C and the parameter estimation problem (3.7) gives a unique solution (a necessary condition is $t = m$). Let us denote

$$A \triangleq \{ \alpha \in R^m: F(c, \alpha) = F^*(c), G(c, F(c, \alpha)) \lesssim 0 \text{ and } c \in C \} \tag{3.10}$$

$$\overline{CA} \triangleq \{(c, \alpha) \in R^n \times R^m: G(c, F(c, \alpha)) \lesssim 0 \text{ and } F(c, \alpha) = F^*(c)\} \tag{3.11}$$

$$C_\alpha \triangleq \{ c \in R^n: G(c, F(c, \alpha)) \lesssim 0 \}. \tag{3.12}$$

The parameter estimation problem then can be represented by the mapping

$$\hat{\alpha}: P_n(\overline{CA}) \to A \tag{3.13}$$

The model optimisation problem with fixed α can be rewritten as

$$(MOP_\alpha) \qquad \min_{c} \Phi(c, F(c, \alpha)) \quad \text{s.t. } c \in C_\alpha. \tag{3.14}$$

The solution of (MOP_α) is denoted by $\hat{c}(\alpha)$.

Theorem 3.1 (existence of solution)

Assume 1. The model is point-parametric on C. 2. $\hat{\alpha}$ is continuous on $P_n(\overline{CA})$. 3. For any $\alpha \in A$, $\hat{c}(\alpha)$ is unique. 4. $\hat{c}$ is continuous on A. 5. $\overline{CA}$ is compact and convex. Then there exists at least one solution $\bar{c}$ of the two-step algorithm.

Proof Due to assumptions 1., 2., 3. and 4. the composition mapping $\hat{c}\hat{\alpha}$ is a well-defined point-to-point mapping continuous on $P_n(\overline{CA})$. By the definitions of $\hat{c}$ and $\hat{\alpha}$, and assumption 1., $\forall c \in P_n(\overline{CA})$, $\hat{c}\hat{\alpha}(c) \in P_n(\overline{CA})$,

i.e. $\hat{c}\,\hat{\alpha}: P_n(\overline{CA}) \to P_n(\overline{CA})$.

Finally $P_n(\overline{CA})$ is compact and convex as a linear transformation of the compact and convex set $\overline{CA}$ (assumption 5.). Schauder's theorem (Appendix A), therefore, ensures that there exists at least one point $\bar{c} \in P_n(\overline{CA})$ such that $\hat{c}\hat{\alpha}(\bar{c}) = \bar{c}$, i.e., $\hat{c}(\bar{\alpha}) = \bar{c}$ with $F(\bar{c}, \bar{\alpha}) = F^*(\bar{c})$. Q.E.D.

It is necessary to point out that in general the two-step algorithm cannot really solve (OP1) (i.e. obtain an optimal solution of (OP1)). Rather it can only obtain a suboptimal solution of (OP1). When the model-reality differences are significant the solution given by the two-step method may be quite far from the real optimum. To demonstrate this, consider the following simple example.

Simple example 1

The model and reality equations are:

$$y = F(c, \alpha) = -1.5c + \alpha ,$$

$$y = F^*(c) = c + 1.$$

Constraint and performance index are:

$$G(c, y) = c + y + 1 \leqslant 0,$$

$$Q(c, y) = c^2 + y^2.$$

The following results are readily verifed (Appendix C).

(a) Real optimum: $c_{opt}=-1$, $Q_{opt} = 1$ with associated Lagrange multiplier $\xi_{opt}=1$. Notice that the real constraint set is $c \leqslant -1$ and the constraint is active.

(b) Two-step method: $\bar{c} = -3$, $\bar{Q} = 13$ with $\bar{\xi} = 0$.

3.4 Sufficient conditions for the model based solution to give the correct process optimum

To determine the conditions when the basic two-step method gives the correct optimal solution of (OP1) (i.e. (ROP)) let us examine sufficient conditions for the model solution to be the correct process optimum. Such conditions have been discussed by Durbeck (1965), Foord (1974), and Ellis and Roberts (1982). We recall (ROP) and (MOP_α), and assume α could be any desired value in R^t and its value has been given.

Theorem 3.2

Assume (a) $Q(c,y)$ and $G(c,y)$ are continuously differentiable on $R^n \times R^m$, F^* and $F(.,\alpha)$ are continuously differentiable on R^n. (b) All relative minimum points are regular points. (c) C and C_α are convex. (d) $Q(c,F^*(c))$ is convex on C and $Q(c,F(c,\alpha))$ is convex on C_α. Then the sufficient conditions for a solution $\bar{c}$ of (MOP_α) to be a solution of (ROP) are

(i) $F(\bar{c},\alpha) = F^*(\bar{c})$,

(ii) $$\frac{\partial F}{\partial c}\bigg|_{c=\bar{c}} = \frac{\partial F^*}{\partial c}\bigg|_{c=\bar{c}}.$$

Proof Write the Lagrangians of (ROP) and (MOP_α)

$$L^*(c, \xi) = Q(c, F^*(c)) + \xi^T G(c, F^*(c)),$$

$$L(c, \xi) = Q(c, F(c, \alpha)) + \xi^T G(c, F(c, \alpha)).$$

According to our assumptions the necessary and sufficient conditions (see, for example, Luenberger, 1973) for $\bar{c}$ and $\bar{\xi}$ to solve (MOP_α) are

$$\nabla_c L(\bar{c}, \bar{\xi}) = \left[\frac{\partial^T Q}{\partial c} + \frac{\partial^T F}{\partial c}\frac{\partial^T Q}{\partial y} + \left[\frac{\partial^T G}{\partial c} + \frac{\partial^T F}{\partial c}\frac{\partial^T G}{\partial y}\right]\bar{\xi}\right]_{c=\bar{c}} = 0$$

$$G(\bar{c}, F(\bar{c}, \alpha)) \leq 0 , \quad \bar{\xi} \geq 0$$

$$\bar{\xi}^T G(\bar{c}, F(\bar{c}, \alpha)) = 0$$

but $\nabla_c L^*(c, \xi) = \frac{\partial^T Q}{\partial c} + \frac{\partial^T F^*}{\partial c}\frac{\partial^T Q}{\partial y} + \left[\frac{\partial^T G}{\partial c} + \frac{\partial^T F^*}{\partial c}\frac{\partial^T G}{\partial y}\right]\xi$, due to (i)

and (ii)

$$\nabla_c L^*(\bar{c}, \bar{\xi}) = \nabla_c L(\bar{c}, \bar{\xi}) = 0$$

$$G(\bar{c}, F^*(\bar{c})) = G(\bar{c}, F(\bar{c}, \alpha)) \leq 0$$

$$\bar{\xi}^T G(\bar{c}, F^*(\bar{c})) = \bar{\xi}^T G(\bar{c}, F(\bar{c}, \alpha)) = 0 ,$$

i.e. $\bar{c}$ and $\bar{\xi}$ solve (ROP). Q.E.D.

Conditions (i) is, of course, very mild, e.g., point-parametric models can always achieve it. On the other hand, condition (ii) is rarely achieved in practice. It requires that the model and the real process have the same structure up to first-order derivatives. It becomes clear why the two-step algorithm given in Section 3.3 cannot find the real optimum solution when the model structure is inaccurate. The situation, however, is not hopeless. There are two possible ways for improving the standard two-step method so that the correct process

optimum may be achieved.

Foord (1974) proposed a method to satisfy condition (ii) directly by extending the parameter estimation problem (3.7) so that not only the outputs but also the first-order derivatives of the model and the real process are matched, i.e., solving α^k from

$$F(c^k, \alpha) = F^*(c^k)$$

$$\frac{\partial F(c^k, \alpha)}{\partial c} = \frac{\partial F^*(c^k)}{\partial c} .$$

This approach, in addition to other practical difficulties, requires a much more complex model and suffers computational drawbacks. A point-parametric model is not enough. The chosen model must also be capable of matching the real process output derivatives. Suppose that the parameter estimation problem is unique, the parameter number would be $t = n \times m + m$ instead of $t = m$. Solving such a large number of equations provides a heavy computational burden.

The second method attempts to modify (MOP_α) so that the sufficient conditions to give the correct process optimum can be weakened. This approach is discussed in the next section.

3.5 Sufficient conditions for the modified model based solution to give the correct process optimum

As discussed above, any attempt to satisfy condition (ii) of Theorem 3.2 will meet great practical difficulties. Therefore, we will allow that model-reality derivative differences exist and try to

compensate such differences by modifying the model based optimisation problem. Let us consider the following modified model optimisation problem, By "modified" we mean that a modification term $-\lambda^T c$ is added to the performance function. The meaning of modifier λ will become clear later on.

$$(MMOP_\alpha) \qquad \begin{aligned} &\min_c \{ Q(c, F(c, \alpha)) - \lambda^T c \} \\ &\text{s.t. } c \in C_\alpha . \end{aligned} \tag{3.15}$$

Theorem 3.3

Assume that assumptions (a) to (d) in Theorem 3.2 hold. Then the sufficient conditions for a solution $\bar{c}$ of $(MMOP_\alpha)$ to be a solution of (ROP) are

(i) $F(\bar{c}, \alpha) = F^*(\bar{c})$,

(ii) $$\lambda = \left[\frac{\partial^T F}{\partial c} - \frac{\partial^T F^*}{\partial c}\right]\left[\frac{\partial^T Q}{\partial y} + \frac{\partial^T G}{\partial y}\bar{\xi}\right]\Bigg|_{c=\bar{c}}$$

where $\bar{\xi}$ is chosen such that the associated Lagrange multiplier of $(MMOP_\alpha)$ at $\bar{c}$ is $\bar{\xi}$.

Proof Obviously $Q(c, F(c, \alpha)) - \lambda^T c$ is convex on C_α due to the convexity of $Q(c, F(c, \alpha))$. Therefore, the necessary and sufficient conditions for $\bar{c}$ and $\bar{\xi}$ to solve $(MMOP_\alpha)$ are

$$\nabla_c L(\bar{c}, \bar{\xi}) = \frac{\partial^T Q}{\partial c} + \frac{\partial^T F}{\partial c}\frac{\partial^T Q}{\partial y} + \left[\frac{\partial^T G}{\partial c} + \frac{\partial^T F}{\partial c}\frac{\partial^T G}{\partial y}\right]\bar{\xi}\Bigg|_{c=\bar{c}} - \lambda = 0$$

$$G(\bar{c}, F(\bar{c}, \alpha)) \leq 0 , \quad \bar{\xi} \geq 0$$

$$\bar{\xi}^T G(\bar{c}, F(\bar{c}, \alpha)) = 0 .$$

Because of (i) and (ii)

$$\nabla_c L(\bar{c}, \bar{\xi}) = \left[\frac{\partial^T Q}{\partial c} + \frac{\partial^T F^*}{\partial c} \frac{\partial^T Q}{\partial y} + \left[\frac{\partial^T G}{\partial c} + \frac{\partial^T F^*}{\partial c} \frac{\partial^T G}{\partial y} \right] \bar{\xi} \right] \Bigg|_{c = \bar{c}}$$

$$= \nabla_c L^*(\bar{c}, \bar{\xi}) = 0$$

$$G(\bar{c}, F^*(\bar{c})) = G(\bar{c}, F(\bar{c}, \alpha)) \leq 0$$

$$\bar{\xi}^T G(\bar{c}, F^*(\bar{c})) = \bar{\xi}^T G(\bar{c}, F(\bar{c}, \alpha)) = 0 ,$$

i.e. $\bar{c}$ and $\bar{\xi}$ solve (ROP). Q.E.D.

Condition (ii) of Theorem 3.3 is much weaker than that of Theorem 3.2. It is no longer required to match the output derivatives. Instead, the model-reality derivative differences are allowed to exist and compensated by the modifier λ. How to implement condition (ii) of Theorem 3.3 is discussed in the next chapter.

3.6 Summary

In this chapter a class of ISOPE problems is discussed which has found on-line practical application in industry. The mathematical formulation of such problems is defined as (OP1). Lemma 3.1 provides a basis for determining the real optimum operating condition by solving (OP1).

Developing a suitable algorithm to solve ISOPE problems proves to be nontrivial. Using the two-step method to solve (OP1) has been discussed. Theorem 3.2 reveals the suboptimal nature of the standard

two-step method when applied to (OP1). Any attempt to fulfil condition (ii) of Theorem 3.2 will meet serious practical difficulties.

The sufficient conditions in Theorem 3.3 have much practical sense. It has been shown that by modifying appropriately the model based optimisation problem a model solution can be the real optimum and yet the structure differences between the model and the process are allowed to exist. This result has significant importance because it shows that an algorithm can be constructed by modifying the iterative two-step algorithm given in this chapter. The resulted algorithm is capable of overcoming the model uncertainties in on-line operations. In the next chapter technical details of this algorithm are presented.

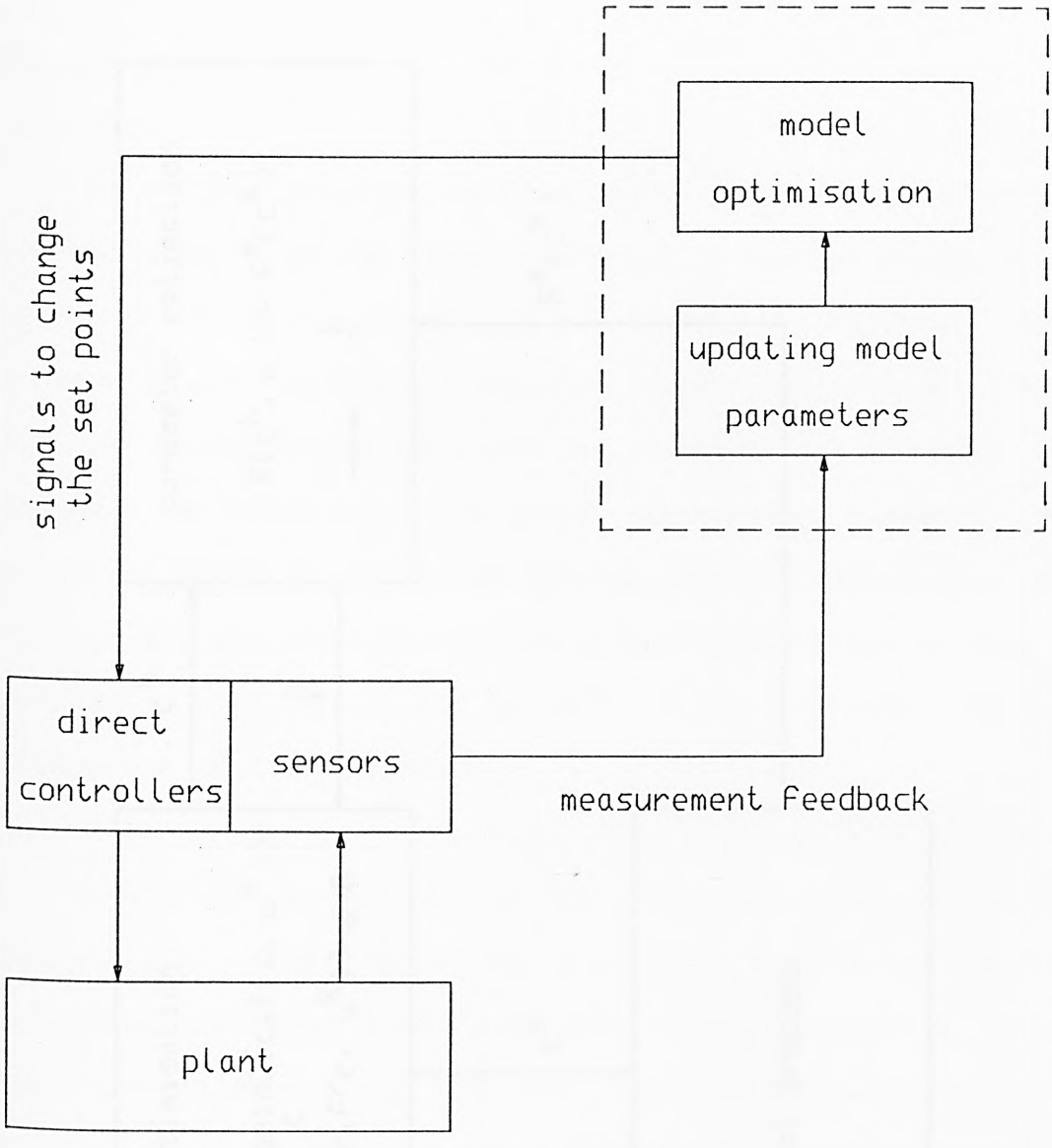


Fig. 3.1. An on-line control scheme

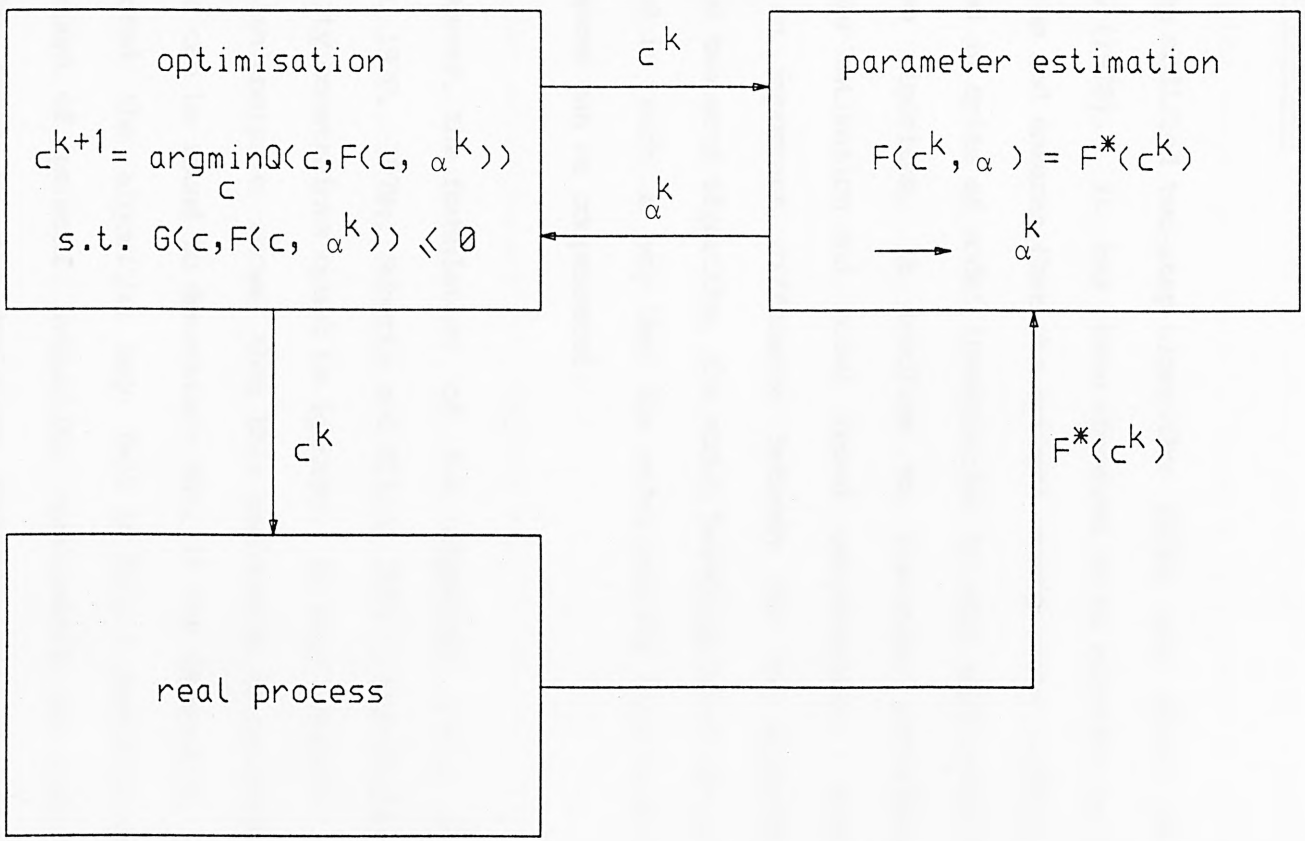


Fig. 3.2. Two-step algorithm

CHAPTER 4. MODIFIED TWO-STEP METHOD

4.1 Introduction

The modified two-step algorithm (MTS) was first introduced by Roberts (1978). It has been observed to be superior to the two-step algorithm and ensures that the optimal steady-state condition can be achieved in spite of model inaccuracies in many applications. Like the two-step algorithm, it involves an iterative procedure of model parameter estimation and model based optimisation. However, there exists an important difference between the two algorithms. In the modified two-step algorithm, the model based optimisation problems are modified in such a way that the model-reality first-order derivative differences can be compensated.

However, the formulation of the algorithm (see, for example, Roberts, 1978, 1979; Roberts and Ellis, 1981) assumed that the system inequality constraints could be ignored. In this chapter a thorough Lagrangian analysis shows that this assumption is not entirely valid. Examples can be found to demonstate that if the inequality constraints are ignored the algorithm may fail to have a feasible solution. To take account of general inequality constraints an additional term should be added to the modifier formula. This appears to give the benefit of a better rate of convergence and guarantees that the optimal condition is achieved even when the inequality constraints involve the system outputs. The previous algorithm may be viewed as a special case, i.e., when the system inequality constraints do not involve the system outputs or when these constraints are not active during the

whole iteration procedure.

In general, the real process outputs are often contaminated with measurement noise and hence, accurate measurements of real outputs cannot be assumed. The errors in measurements are further amplified by the use of finite difference approximations to determine the process output derivatives with respect to the controller set points. This forms a major source of inaccuracy. Because the derivatives are required in order to calculate the modifiers, an suitable simple method for reducing the influence of noise is to filter the modifier vector using low-pass digital filters. Employing simple averaging techniques will also reduce the influence of zero mean value noise, which is a common noise in practice.

In the following sections the new version of the modified two-step algorithm is formulated, some implementation aspects of the algorithm are given and simulation results are presented and discussed.

4.2 Formulation of the new version of the modified two-step algorithm

It is assumed that the external disturbances which affect the system are varying slowly and can be considered as constant in the considered time interval of determining the steady state optimal control and, hence, the task of integrated system optimisation and parameter estimation may be described by (OP1) defined in Chapter 3. Rewrite (OP1) as

$$\min_{c,\ \alpha} q(c,\ \alpha), \tag{4.1}$$

$$\text{(OP1)'} \qquad \text{s.t. } F(c, \alpha) = F^{*}(c), \tag{4.2}$$

$$g(c, \alpha) \lesssim 0, \tag{4.3}$$

where $q(c, \alpha) = Q(c, F(c, \alpha))$ and $g(c, \alpha) = G(c, F(c, \alpha))$.

In order to separate system optimisation and parameter estimation an additional equality constraint is introduced as $v = c$, and (OP1)' becomes

$$\min_{c,v,\alpha} q(c, \alpha), \tag{4.4}$$

$$\text{(OP2)} \qquad \text{s.t. } F(v, \alpha) = F^{*}(v), \tag{4.5}$$

$$g(c, \alpha) \lesssim 0, \tag{4.6}$$

$$v = c. \tag{4.7}$$

The Lagrangian function associated with (OP2) is

$$L(c,v, \alpha, \lambda, \eta, \xi) = q(c, \alpha) + \lambda^{T}(v - c) + \eta^{T}[F(v,\alpha) - F^{*}(v)] + \xi^{T}g(c, \alpha), \tag{4.8}$$

where λ, η and ξ are Lagrange multiplier vectors. Assume that all required derivatives exist and regularity conditions are satisfied. Then the Kuhn-Tucker necessary optimal conditions (for example, Luenberger, 1973) take the form:

$$\nabla_{c}L = \frac{\partial^{T}q(c, \alpha)}{\partial c} - \lambda + \frac{\partial^{T}g(c, \alpha)}{\partial c}\xi = 0, \tag{4.9}$$

$$\nabla_{v}L = \lambda + \left[\frac{\partial^{T}F(v, \alpha)}{\partial v} - \frac{\partial^{T}F^{*}(v)}{\partial v}\right]\eta = 0, \tag{4.10}$$

$$\nabla_{\alpha}L = \frac{\partial^{T}q(c, \alpha)}{\partial \alpha} + \frac{\partial^{T}F(v, \alpha)}{\partial \alpha}\eta + \frac{\partial^{T}g(c, \alpha)}{\partial \alpha}\xi = 0, \tag{4.11}$$

$$\nabla_{\lambda} L = v - c = 0, \tag{4.12}$$

$$\nabla_{\eta} L = F(v, \alpha) - F^{*}(v) = 0, \tag{4.13}$$

$$\nabla_{\xi} L = g(c, \alpha) \leqslant 0, \tag{4.14}$$

$$\xi^{T} \nabla_{\xi} L = \xi^{T} g(c, \alpha) = 0, \quad \xi \geqslant 0. \tag{4.15}$$

Let us consider how to obtain a solution of (4.9) to (4.15).

Assuming that $\left[\frac{\partial^{T} F(v, \alpha)}{\partial \alpha}\right]^{-1}$ exists, then η is solved from (4.11) and substituted into (4.10) to obtain the formula of λ

$$\lambda = \lambda(c, v, \alpha, \xi) = \left[\frac{\partial^{T} F(v, \alpha)}{\partial v} - \frac{\partial^{T} F^{*}(v)}{\partial v}\right]\left[\frac{\partial^{T} F(v, \alpha)}{\partial \alpha}\right]^{-1} \cdot \left[\frac{\partial^{T} q(c, \alpha)}{\partial \alpha} + \frac{\partial^{T} g(c, \alpha)}{\partial \alpha} \xi\right]. \tag{4.16}$$

Compared with the previous work (Roberts, 1978,1979; Roberts and Ellis, 1981) a new term $\frac{\partial^{T} g(c, \alpha)}{\partial \alpha} \xi$ is introduced which takes into account the influence of system inequality constraints. Because of equation (4.12) $v = c$, (4.16) can be written as

$$\lambda = \lambda(v, \alpha, \xi) = \left[\frac{\partial^{T} F(v, \alpha)}{\partial v} - \frac{\partial^{T} F^{*}(v)}{\partial v}\right]\left[\frac{\partial^{T} F(v, \alpha)}{\partial \alpha}\right]^{-1}$$

$$\cdot\left[\frac{\partial^T q(v,\alpha)}{\partial\alpha} + \frac{\partial^T g(v,\alpha)}{\partial\alpha}\xi\right]. \qquad (4.17)$$

Equations (4.9), (4.14) and (4.15) are satisfied by solving the modified model optimisation problem with given α and λ

$$\min_c \{q(c,\alpha) - \lambda^T c\}, \qquad (4.18)$$

(MMOP)

$$\text{s.t. } g(c,\alpha) \leqslant 0. \qquad (4.19)$$

For given v and ξ let us denote the solution of (4.13) by $\hat{\alpha}(v)$, evaluate λ according to formula (4.17) and denote the solution of (MMOP) and associated Lagrange multipliers by $\hat{c}(v,\alpha,\xi)$ and $\hat{\xi}(v,\alpha,\xi)$, respectively. Then any solution of the operator equations

$$\hat{c}(v,\hat{\alpha}(v),\xi) = v, \qquad (4.20)$$

$$\hat{\xi}(v,\hat{\alpha}(v),\xi) = \xi, \qquad (4.21)$$

is a solution of (4.9) to (4.15). The resulting scheme is the same as in the previous work except for the difference in the formula for calculating modifiers λ and for the need to solve a new operator equation (4.21). The information structure of the algorithm is shown in Fig. 4.1. Interestingly, if the system inequality constraints do not depend on the system outputs, or if those involving the system outputs are not active during the whole iterative procedure then

$\frac{\partial^T g(c,\alpha)}{\partial\alpha}\xi = 0$, (4.20) and (4.21) reduce to

$$\hat{c}(v,\hat{\alpha}(v)) = v,$$

and (4.16) to

$$\lambda = \lambda(c,v,\alpha) = \left[\frac{\partial^T F(v,\alpha)}{\partial v} - \frac{\partial^T F^*(v)}{\partial v}\right]\left[\frac{\partial^T F(v,\alpha)}{\partial \alpha}\right]^{-1}\frac{\partial^T q(c,\alpha)}{\partial \alpha}$$

which, as expected, agrees with the formula given in the previous work.

Simple example 1 given in Section 3.3 is used to demonstrate the importance of the new term in the modifier computation. The following results show that if the new term in (4.16) is ignored the procedure may fail to reach a feasible solution.

(1) Using the previous formula for computing λ gives:

$$\hat{c} = -0.5 + 0.125\,\hat{\xi}.$$

Since $\hat{\xi} \geq 0$ the result violates the real system constraint $\hat{c} \leq -1$, hence, the solution is not feasible.

(2) Using the new formula (4.16) (or (4.17)) for computing λ gives:

$$\hat{c} = -0.5 - 0.5\,\hat{\xi}.$$

It follows $\hat{\xi} = \xi_{opt} = 1$, $\hat{c} = c_{opt} = -1$ and $\hat{Q} = Q_{opt} = 1$ (see Appendix C).

4.3 Implementation aspects of the algorithm

4.3.1 Description of the algorithm

Suppose we have properly chosen some iterative strategies for finding the solutions of equations (4.20) and (4.21)

$$v^{k+1} = \Psi_v(c^k, v^k), \tag{4.22}$$

$$\xi^{k+1} = \Psi_\xi(\hat{\xi}^k, \xi^k). \tag{4.23}$$

Given some initial v^0 inside the feasible set and some initial guess ξ^0 the on-line implementation involves repeatedly the following two

step procedure

Step 1: Apply v^k to the real process and obtain the corresponding steady-state measurements $F^*(v^k)$. Determine α^k by solving

$$F(v^k, \alpha) = F^*(v^k).$$

Perform additional perturbations about v^k and measure the corresponding process outputs to compute finite difference approximations of the derivatives $\dfrac{\partial^T F^*(v^k)}{\partial v}$. Finally, use equation (4.16) to calculate

$$\lambda^k = \lambda(c^{k-1}, v^k, \alpha^k, \xi^k).$$

λ^k may be calculated according to equation (4.17). In this case

$$\lambda^k = \lambda(v^k, \alpha^k, \xi^k).$$

Step 2: With fixed α^k and λ^k, solve the model-based optimisation problem (4.18) and (4.19) to obtain c^k and $\hat{\xi}^k$.

The iteration is terminated when $c^k = v^k$ and $\hat{\xi}^k = \xi^k$.

In practice, one may terminate the procedure when all elements of c and $\hat{\xi}$ satisfy

$$| c_i^k - v_i^k | < \beta_v, \quad i=1,\ldots,n, \tag{4.24}$$

$$| \hat{\xi}_j^k - \xi_j^k | < \beta_\xi, \quad j=1,\ldots,p, \tag{4.25}$$

where β_v and β_ξ are some desired tolerances. It is suggested that the iterative strategies (4.22)and (4.23) have the form

$$v^{k+1} = v^k + K_v(c^k - v^k), \tag{4.26}$$

$$\xi^{k+1} = \xi^k + K_\xi(\hat{\xi}^k - \xi^k), \tag{4.27}$$

where K_v and K_ξ are diagonal gain matrices usually with positive diagonal elements less than or equal to unity.

It is obvious that if the algorithm converges the solution is feasible and condition (i) and (ii) of Theorem 3.3 are fulfilled. Therefore, if the Kuhn-Tucker necessary optimal conditions are also sufficient (e.g., in convex cases) a real process optimum solution is found.

The justification of using either (4.16) or (4.17) to calculate λ^k is that the solutions obtained are identical. However, the iteration procedure may not necessarily be exactly the same. The author's simulation experience suggests that equation (4.16) is preferred in that better behaviour (such as faster convergence) of the iteration process can usually be achieved. On the other hand, using (4.16) the convergence conditions of the algorithm become extremely difficult to formulate. The reason for this will be explained in the next chapter.

4.3.2 Approximating real output derivatives

Calculating the modifiers using (4.16) (or (4.17)) needs the real output derivatives with respect to controller set points v. These derivatives can only be approximated by perturbing v, measuring the corresponding changes in outputs y^* and applying finite difference formulae because the real output equation $F^*(v)$ is unknown. A simple finite difference formula is

$$\frac{\partial F_j^*(v)}{\partial v_i} \approx \frac{F_j^*(v + e_i \sigma_i) - F_j^*(v)}{\sigma_i}, \quad j=1,\ldots,m, \ i=1,\ldots,n, \qquad (4.28)$$

where e_i is a unit vector with unity ith element and others all zero, σ_i is a small perturbation on the ith element of v, m is the number of

outputs and n is the number of controller set points. An advantage of (4.28) is ease of implementation. Let T_m be the measurement time required to wait for the process to reach a new steady state after the set points have been changed, and let T_o be the computing time required for Step 2. Changing v^{k-1} to v^k requires time T_m, another nT_m is needed for calculating the derivatives using (4.28). Therefore, the time required for one iteration will approximately be $(n + 1)T_m + T_o$.

Roberts and Ellis (1981) showed that this time can be reduced by combining process derivative perturbations with controller set point changes produced using equation (4.26). Consider using another simple finite difference formula

$$\frac{\partial F_j^*(\bar{v})}{\partial v_s} \approx \frac{F_j^*(v + \Delta v(s)) - F_j^*(v + \Delta v(s-1))}{\sigma_s}, \quad j=1,\ldots,m, \quad s=1,\ldots,n, \tag{4.29}$$

where $\sigma_s = v_s^k - v_s^{k-1}$, $\Delta v(s) = \sum_{i=1}^{s} e_i \sigma_i$, $\Delta v(0) = 0$, $v = v^{k-1}$ and $\bar{v} = v + \Delta v(n) = v^k$. Using (4.29) to approximate derivatives first and at the end calculate α^k, the time required for one iteration will be reduced to $nT_m + T_o$. However, to achieve an acceptable accuracy it is important to limit both the minimum and maximum possible change in controller set points

$$\varepsilon_M > | v_i^k - v_i^{k-1} | > \varepsilon_m. \tag{4.30}$$

For processes with slow dynamics using equation (4.29) is worthwhile because saving T_m for each iteration is quite significant. However, if the desired tolerance β_v in (4.24) is smaller than ε_m equation (4.29) should not be used when the iteration procedure reaches its convergent stage.

To reduce the measurement time further the points generated by the algorithm may be used to approximate the process output derivatives. Such a scheme requires additional perturbations only at the starting point. For very slow dynamic systems this method clearly has advantage. On the other hand, poor quality of the derivatives may adversely affect the behaviour of the algorithm. It is recommended that when the iteration procedure is near its convergent stage better finite difference formulae such as (4.28) should be used.

4.3.3 Simple techniques for attenuating the influence of noise

It is expected that the performance of the modified two-step algorithm will deteriorate considerably under noisy measurement conditions because the differentiation amplifies measurement errors. However, employing simple filter techniques can significantly reduce the influence of noise. In this short section, it is not intended to construct general filter structures. Rather, only some very simple techniques are suggested. Their effectiveness can be observed in the simulation studies given in the next section.

Since the derivatives are used in calculating the modifier vector, an efficient method for reducing the influence of noise is to filter the modifier vector using low-pass digital filters. A first-order low-pass filter with unity steady-state gain is represented by the difference equation

$$\bar{\lambda}_j^k = r_j \bar{\lambda}_j^{k-1} + (1 - r_j)\lambda_j^k \qquad (4.31)$$

where $\bar{\lambda}_j^k$ and λ_j^k are filtered and unfiltered values of the jth element of the modifier vector, respectively; r_j is a constant,

$0 \leq r_j < 1$, typical values being $0.9 \sim 0.98$.

Employing simple averaging techniques will also help to reduce the influence of zero mean value noise which is the common noise in practice. For example, take T measurements and apply the averaging formula

$$\bar{y}_j^* = \frac{1}{T} \sum_{i=1}^{T} y_j^*(i). \tag{4.32}$$

4.4 Simulation study

4.4.1 Simulation examples

Three numerical examples have been used in computer simulation studies.

Example 1. The model and reality equations are:

$$y_1 = c_1 - c_2 + 2y_2 + \alpha_1$$
$$y_2 = c_3 - c_4 + y_1 - 3y_4 + \alpha_2$$
$$y_3 = 2c_4 - c_5 - y_1 + y_4 + \alpha_3$$
$$y_4 = c_6 + 2.5c_7 - 4y_3 + \alpha_4,$$

$$y_1^* = 1.3c_1 - c_2 + 2y_2^* + 0.15y_2^* c_1$$
$$y_2^* = c_3 - c_4 + 1.2y_1^* - 3y_4^* + 0.1c_4^2$$
$$y_3^* = 2c_4 - 1.25c_5 - y_1^* + y_4^* + 0.25c_4c_5 + 0.1$$
$$y_4^* = 0.8c_6 + 2.5c_7 - 4.2y_3^*.$$

The system constraints and the performace index are:

$$CY \triangleq \{(c,y) \in R^{11}: c_1^2 + c_2^2 \leq 1,\ 0 \leq y_2 \leq 0.5,\ 0.5c_3 + c_4 + 2c_5 \leq 1,$$
$$4c_3^2 + 2c_3y_1 + 0.4y_1 + c_3c_5 + 0.5c_5^2 + y_1^2 \leq 4,\ 0 \leq c_7 \leq 1 \text{ and}$$

$$c_6 + y_3 + 0.5 \geq 0 \},$$

$$Q(c,y) = 5(c_1 + c_2 - 2)^2 + 2(c_3 - 2)^2 + c_4^2 + 3c_5^2 + (c_6 + 1)^2 + 2.5c_7^2 + 4y_1^2 + (y_2 - 1)^4 + (y_3 - 1)^2 + y_4^2.$$

Example 2. The model and reality equations are:

$$y_1 = 1.4375c_1 - 0.1875c_2 + 1.5y_2 + \alpha_1$$

$$y_2 = 0.5c_3 - 1.5c_4 + y_1 - 2y_4 + \alpha_2$$

$$y_3 = 2.5c_4 - 0.5c_5 - y_1 + 1.5y_4 + \alpha_3$$

$$y_4 = 1.25c_6 - 3y_3 + \alpha_4,$$

$$y_1^* = c_1 - c_2 + 2y_2^* - 0.5c_1^2 + 0.5(c_1 + c_2 - 2)y_2^*$$

$$y_2^* = c_3 - c_4 + y_1^* - 3y_4^*$$

$$y_3^* = 2c_4 - c_5 - y_1^* + y_4^*$$

$$y_4^* = c_6 - 4y_3^* + 0.5c_6 y_3^*.$$

The system constraints and the performance index are:

$$CY \triangleq \{(c,y) \in R^{10}: 1.006 - c_1 - y_2 \geq 0,\ 0.375 + 2.25c_6 - 2.75y_3 - y_4 \geq 0,$$

$$|c_i| \leq 0.5,\ i=1,4,5,6,\ 0 \leq c_2 \leq 2.5 \text{ and } 0 \leq c_3 \leq 2\},$$

$$Q(c,y) = c_1^2 + (c_2 - 2)^2 + 2(c_3 - 2)^2 + c_4^2 + 3c_5^2 + (c_6 + 1)^2 + 4y_1^2 + (y_2 - 1)^2 + (y_3 - 1)^2 + y_4^2.$$

Example 3. The model and reality equations are:

$$y_1 = c_1 - c_2 + 2y_2 + \alpha_1$$

$$y_2 = c_3 - c_4 + y_1 + \alpha_2$$

$$y_3 = 2c_4 - c_5 - y_1 + \alpha_3,$$

$$y_1^* = 1.4c_1 - 0.6c_2 + 1.8y_2^*$$

$$y_2^* = 1.3c_3 - 1.1c_4 + 1.1y_1^*$$

$$y_3^* = 2.3c_4 - 0.7c_5 - 1.1y_1^*.$$

The system constraints and the performance index are:

$$CY \overset{\Delta}{=} \{(c,y) \in R^8 : |c_i| \leq 1,\ i=1,\ldots,5,\ y_j \geq 0,\ j=1,2,3,$$

$$0.8 - c_2 - 0.6y_2 \geq 0 \text{ and } 2.04 + 1.05y_1 - c_3^2 - c_4^2 - c_5^2 \geq 0\},$$

$$Q(c,y) = \sum_{i=1}^{5} c_i^2 + (y_1 - 1)^2 + 2(y_2 - 2)^2 + (y_3 - 3)^2.$$

In each example, the aim is to use the iterative modified two-step algorithm to determine the optimum (minimum) value of the real system performance $Q(c, y^*)$ inspite of the fact that the model used to solve the optimisation problem is not a faithful representation of reality. The model-reality differences are taken into account by updating model parameters and through modification of the optimisation problem.

4.4.2 Simulation results

The simulation work was performed on a Prime 550 computer. The optimisation problems were solved using the NAG library subroutine E04UAF which uses direct search and a sequential augmented Lagrangian method. Initial conditions were

$$v^0 = 0 \text{ and } \xi^0 = 0.$$

Desired tolerances in (4.24) and (4.25) were chosen as

$$\beta_v = 5\text{x}10^{-5} \text{ and } \beta_\xi = 10^{-3}.$$

Gain matrices K_v and K_ξ in (4.26) and (4.27) were chosen as

$$K_v = \theta_v I \text{ and } K_\xi = \theta_\xi I,$$

where I is the identity matrix. Suitable values of θ_v and θ_ξ were found by experiment. The modifier vector was updated using equation (4.16) and the finite difference formula (4.28) was used to approximate

the real output derivatives.

TABLE 4.1 compares the performance of the new version of the algorithm with its previous version under noise-free conditions. TABLE 4.2 illustrates detailed results of Example 3.

To investigate the behaviour of the MTS algorithm in the presence of noise, random noise sources with normal statistical properties have been added to all elements of the real output vector measurements. The ith sequence of the noise vector has a zero-mean value and $0.01|y_i^*|$ standard deviation, where y_i^* is the real optimum value of the ith element of the output vector (in these examples all y_i^* are none zero). Every element in the sequences of the noise vector are generated using NAG library subroutines G05CCF and G05DDF so that they are all different to each other and have different values in separate runs of the program.

Fig. 4.2. to Fig. 4.6. illustrate the effect of noise on real system performance during the course of the iterations and the significant improvement achieved by the use of the filter techniques (4.31) and (4.32), where the number T in (4.32) is chosen as 10.

4.4.3 Discussion of results

In all three examples the system inequality constraints depend on the system outputs and at least one of them is active. TABLE 4.1 shows that in all these examples the new version of the modified two-step algorithm attained the real system optimum condition with a

significantly improved rate of covergence in two of the examples. The previous version failed to reach the real optimum condition, illustrating the danger that it may fail to achieve a feasible solution in some situations, due to ignoring the term $\frac{\partial^T g(c, \alpha)}{\partial \alpha} \xi$ as shown in Section 4.2. It has been observed that after convergence the associated Lagrange multipliers $\hat{\xi}$ in the new version matched the corresponding ξ_{opt} associated with the real system optimisation problem. This did not occur in the previous version of the algorithm.

The simulation results show that, as expected, the measurement noise has serious influence on the performance of the modified two-step algorithm. However, the simulation results also illustrate that the simple filter techniques (4.31) and (4.32) are effective. In the examples tested it has been found that the filter technique (4.31) is powerful enough to attenuate the influence of the given noise and contributes the most improvement.

4.5 Summary

In this chapter a method for solving ISOPE problems has been presented. Using a modification of the two-step algorithm to take account of differences between real process and model output first-order derivatives with respect to the controller set points enables the iterative technique to achieve the correct optimal operating point on the real process in spite of errors in the structure of the mathematical model.

The need to evaluate the real output derivatives represents a significant drawback of the modified two-step algorithm. However, two major advantages are gained which may partly compensate the above drawback. Firstly, the self-adaptive nature of the method enables simpler models to be used and results in saving in modelling. If any disturbances occur at any stage during the operation the algorithm will adapt to proceed towards a new optimum. Secondly, using simpler models (e.g. linear models) simplifies the optimisation computation.

It is considered that the modified two-step algorithm is very suitable for applying in the situation where highly uncertain mathematical models exist. Further work is required to extend the algorithm to dynamic optimisation.

TABLE 4.1 Comparison of performances (noise-free)

example	MTS algorithm	θ_v	θ_ξ	iterations	real performance	real optimum
one	previous version	0.9	-	15	6.3394	6.3266
	new version	0.9	0.9	10	6.3266	
two	previous version	0.4	-	31	2.1409	2.1405
	new version	0.4	0.8	32	2.1405	
three	previous version	0.3	-	35	5.9273	5.9261
	new version	0.3	0.8	29	5.9261	

TABLE 4.2 Comparison of results of Example three (noise-free)

set points	previous MTS	-0.73870	0.10857	0.90761	1.00000	-0.83271
	new MTS	-0.71740	0.11836	0.89968	1.00000	-0.82994
	real optimum	-0.71740	0.11836	0.89966	1.00000	-0.82990

outputs	previous MTS	0.97502	1.15242	1.81037
	new MTS	0.96953	1.13606	1.81448
	real optimum	0.96955	1.13607	1.81443

real constraint values (≥ 0)	previous MTS	-0.000025	0.546620
	new MTS	0.000002	0.559789
	real optimum	0.000000	0.559895

Lagrange multipliers	previous MTS	0.308415	0.000000
	new MTS	0.378316	0.000000
	real optimum	0.378188	0.000000

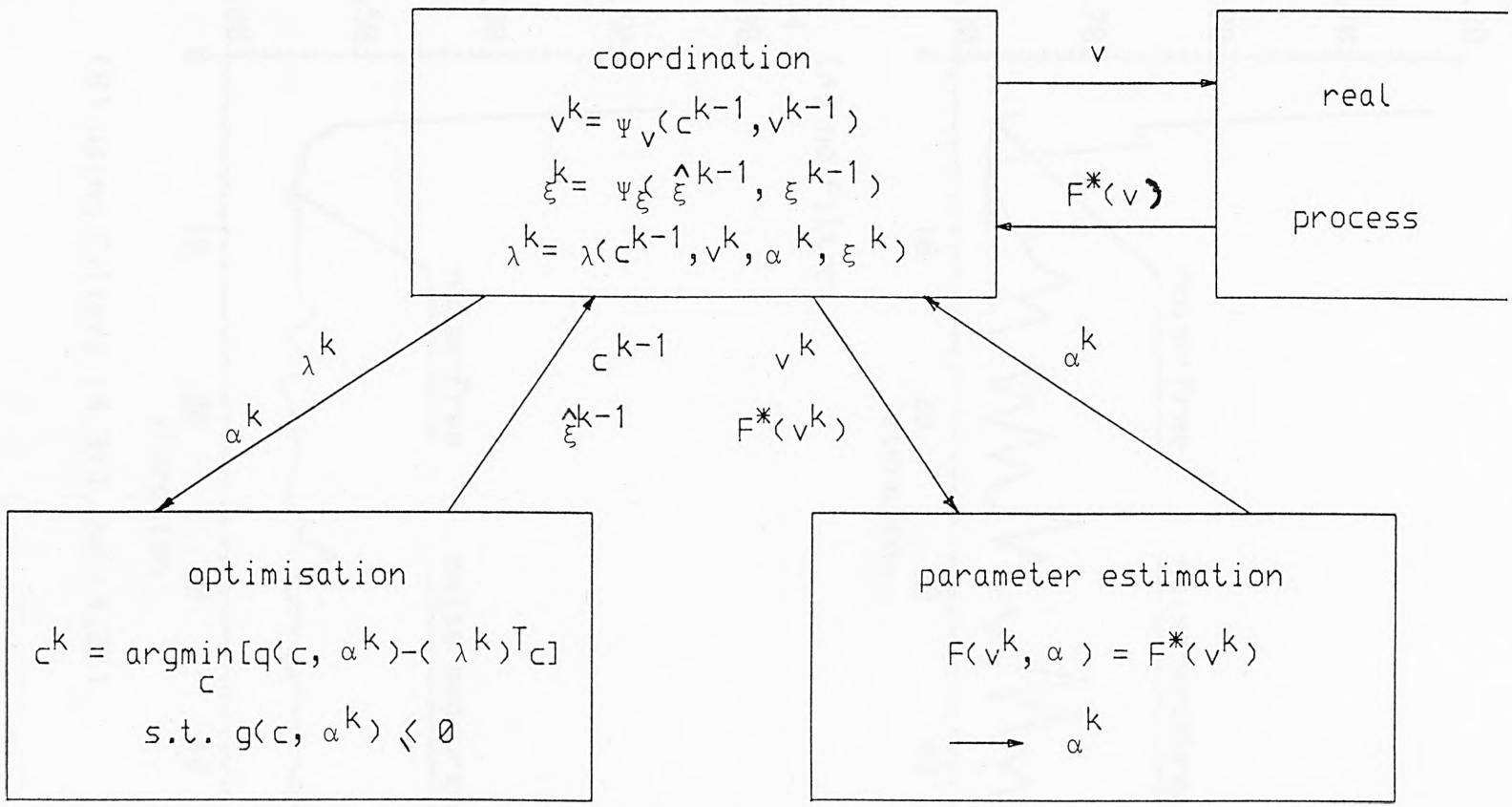
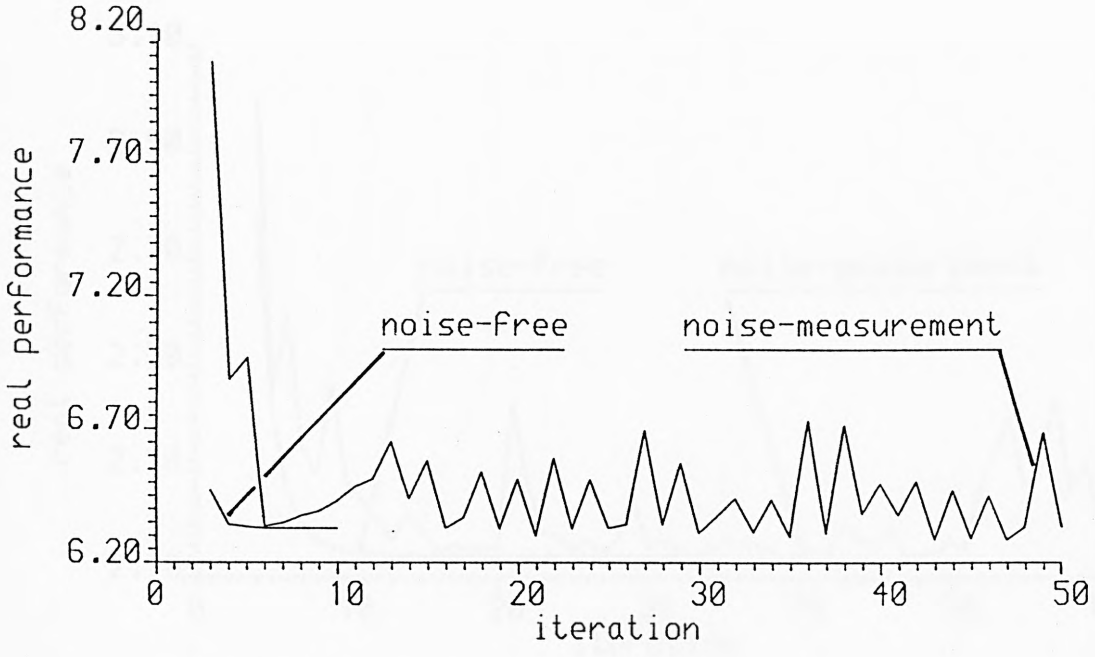
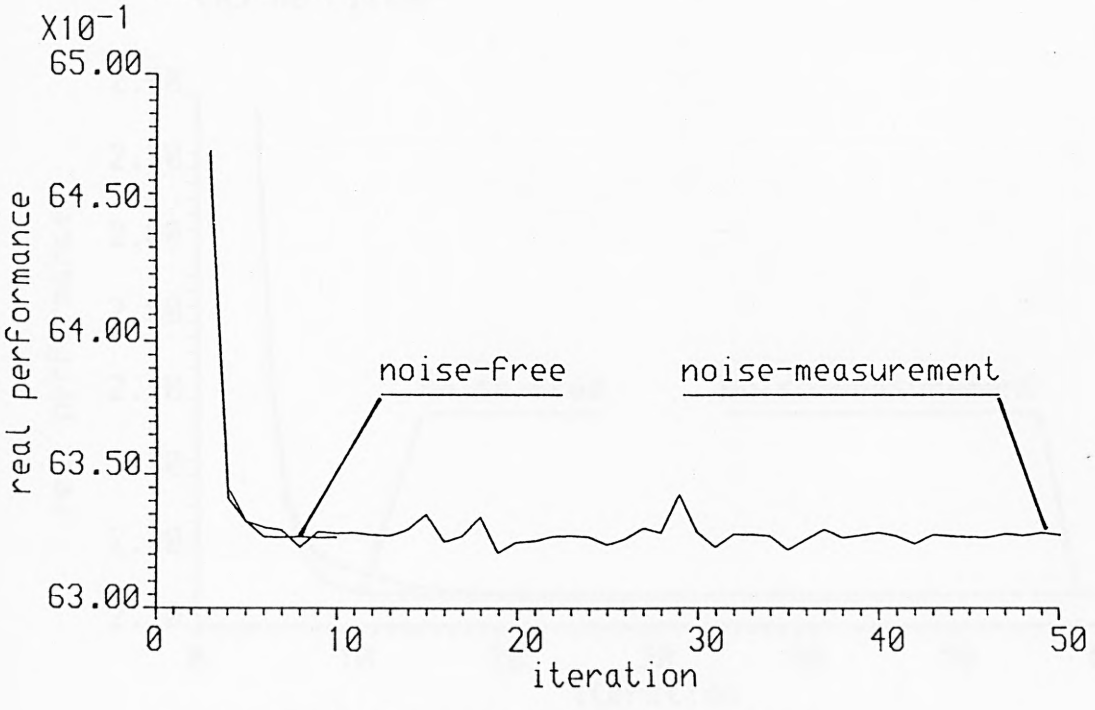


Fig. 4.1. Modified two-step algorithm

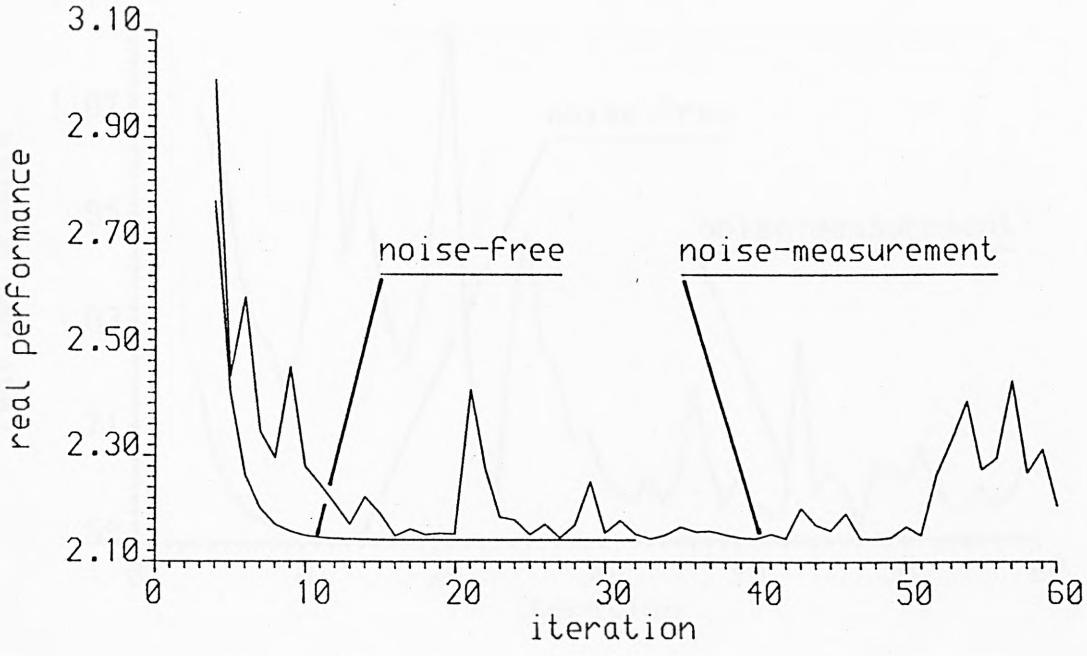


(A) no filter

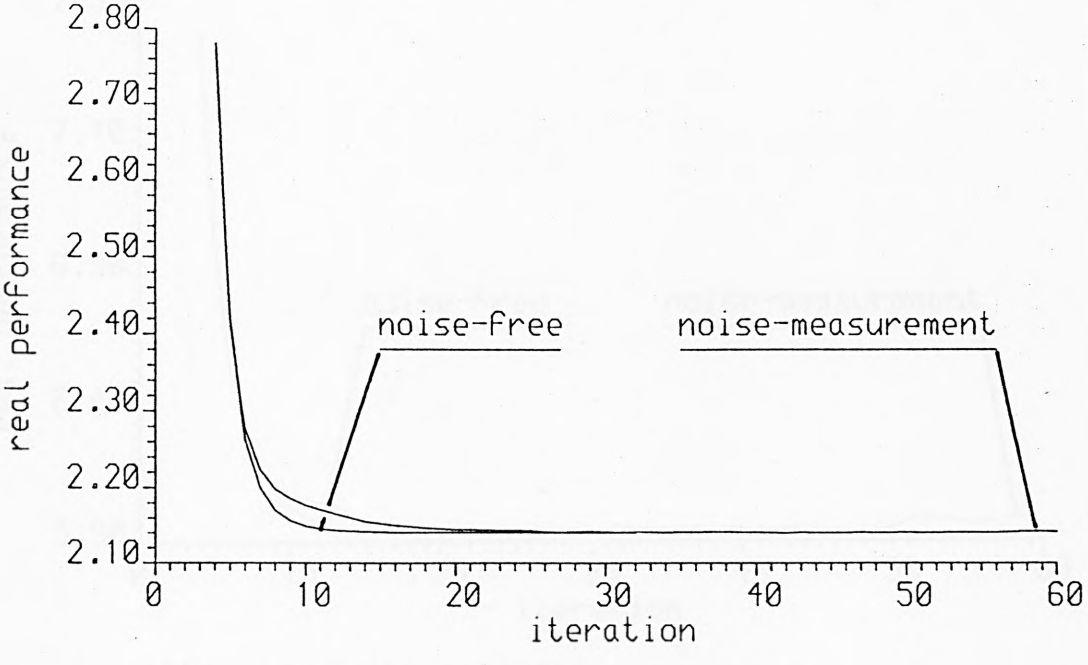


(B) using filters (4.31) and (4.32)

Fig. 4.2. Performance of Example one in the presence of noise

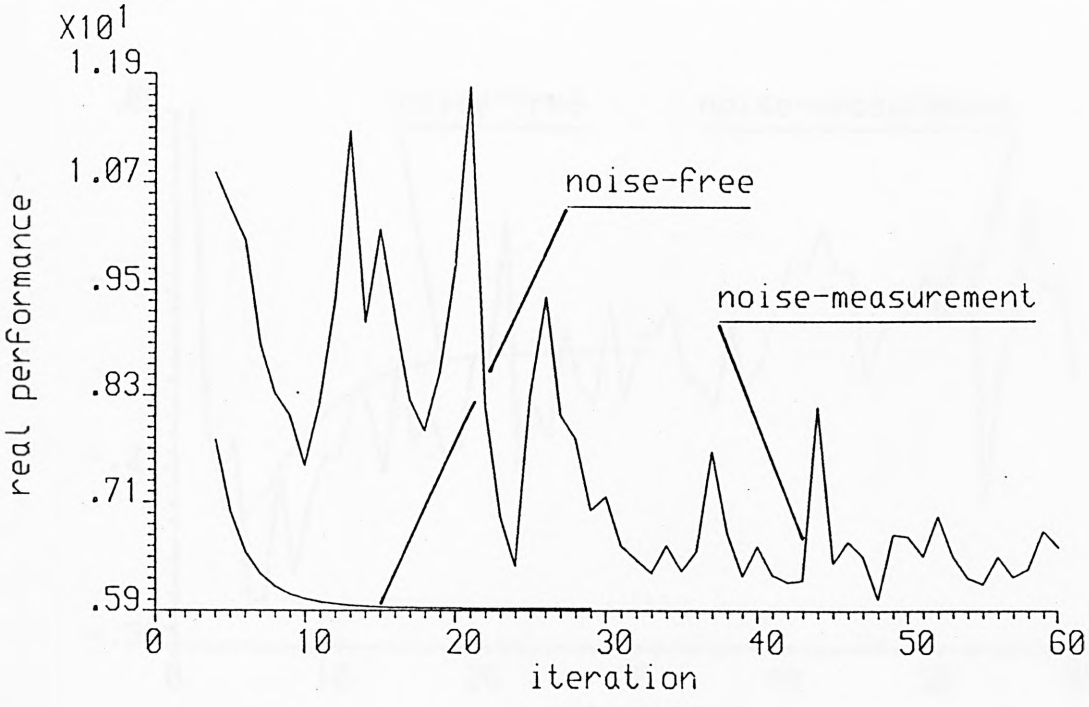


(A) no filter

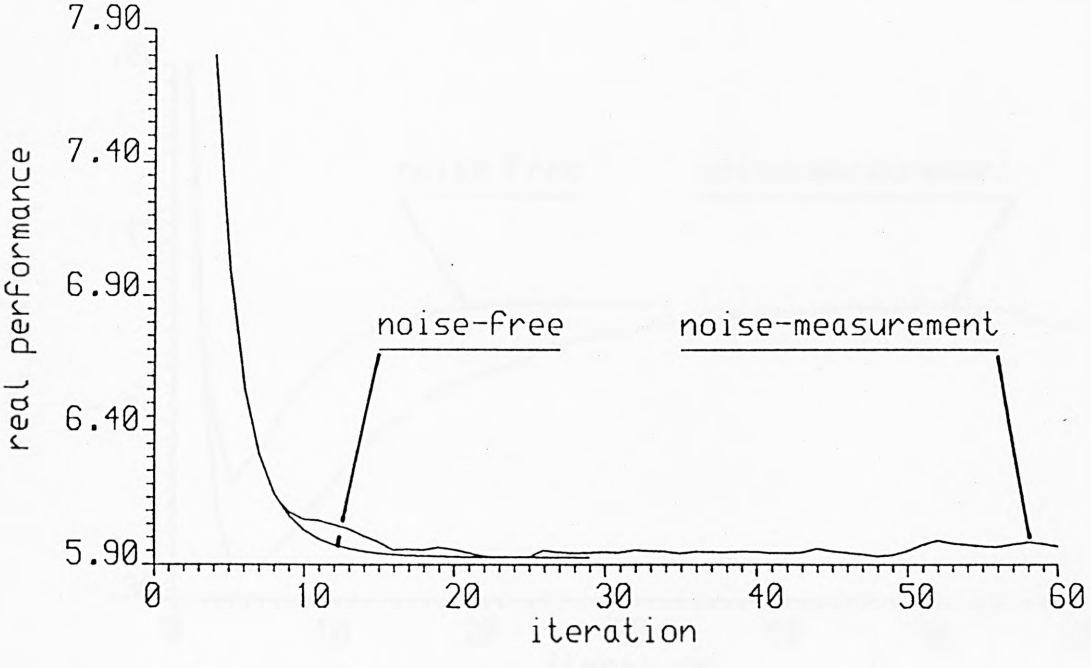


(B) using filters (4.31) and (4.32)

Fig. 4.3. Performance of Example two in the presence of noise

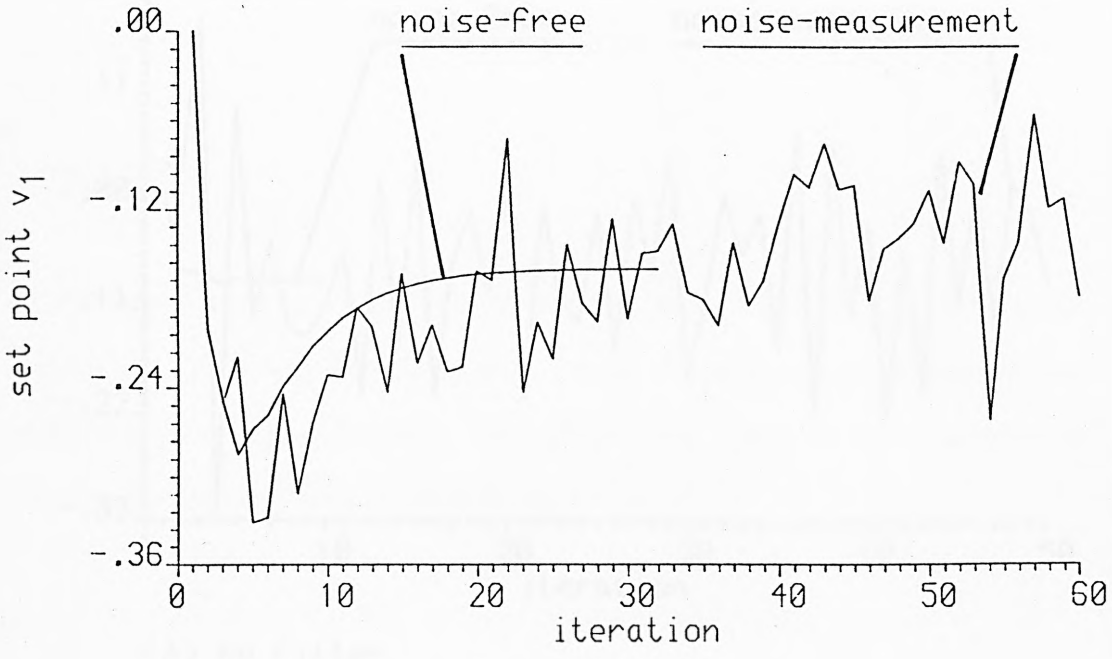


(A) no filter

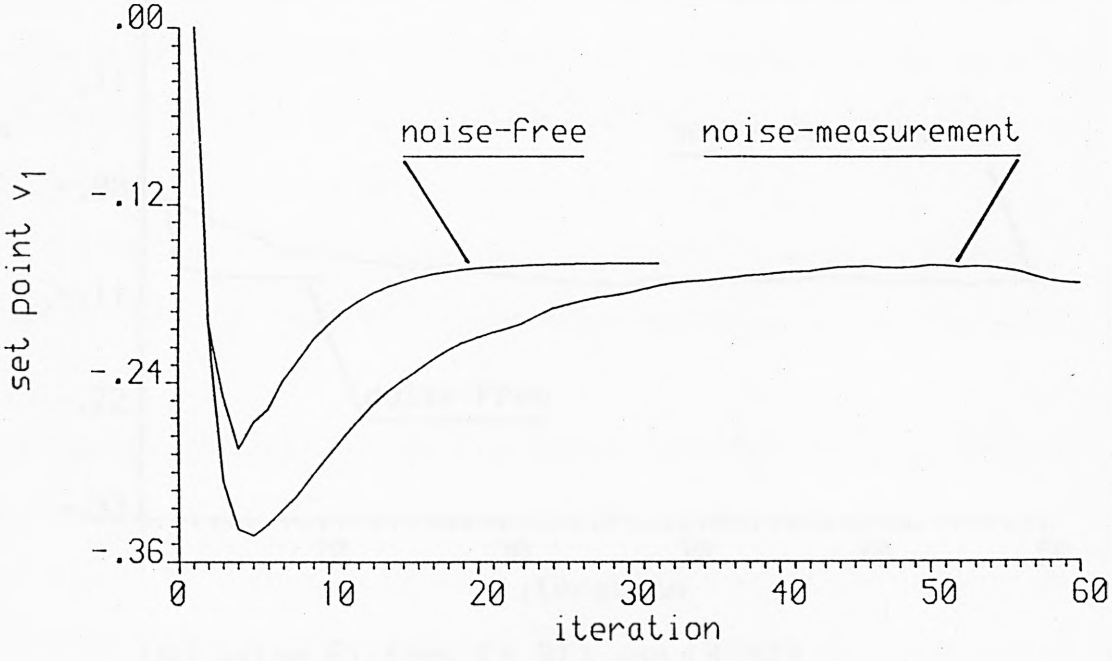


(B) using filter (4.31)

Fig. 4.4. Performance of Example three in the presence of noise

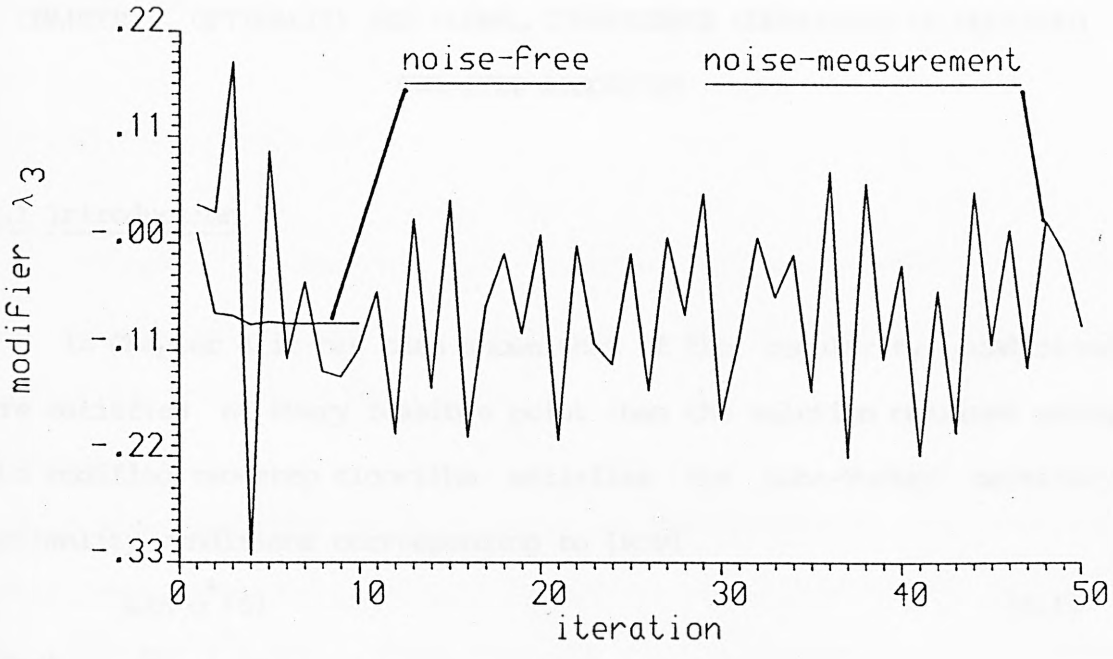


(A) no filter

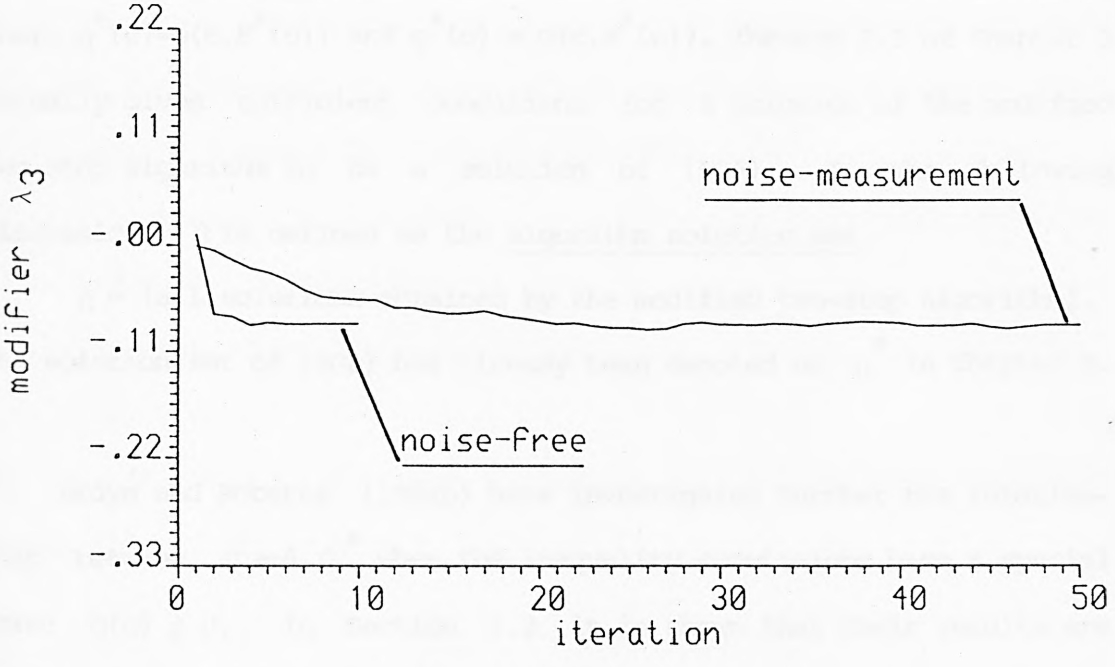


(B) using filters (4.31) and (4.32)

Fig. 4.5. A typical control set point trajectory of Example two



(A) no filter



(B) using filters (4.31) and (4.32)

Fig. 4.6. A typical modifier trajectory of Example one

CHAPTER 5. OPTIMALITY AND GLOBAL CONVERGENCE CONDITIONS OF MODIFIED TWO-STEP ALGORITHM

5.1 Introduction

In Chapter 4 it has been shown that if the regularity conditions are satisfied at every feasible point then the solution obtained using the modified two-step algorithm satisfies the Kuhn-Tucker necessary optimality conditions corresponding to (ROP)

$$\min_{c} q^*(c) \tag{5.1}$$

(ROP)

$$\text{s.t. } g^*(c) \leqslant 0 \tag{5.2}$$

where $q^*(c)=Q(c,F^*(c))$ and $g^*(c) = G(c,F^*(c))$. Theorem 3.3 of Chapter 3 actually gives sufficient conditions for a solution of the modified two-step algorithm to be a solution of (ROP). In the following discussion, Ω is defined as the algorithm solution set

$$\Omega \overset{\Delta}{=} \{\text{all solutions obtained by the modified two-step algorithm}\}.$$

The solution set of (ROP) has already been denoted as Ω^* in Chapter 3.

Brdys and Roberts (1984b) have investigated further the relationship between Ω and Ω^* when the inequality constraints have a special form: $G(c) \leqslant 0$. In Section 5.2, it is shown that their results are readily valid for a more general form of inequality constraints: $G(c,y) \leqslant 0$. In Section 5.3 the modified two-step algorithm is considered as an algorithmic mapping, and its component mappings are defined. This gives an inside view of the algorithm and provides a basis for convergence analysis. Global convergence conditions given in Section 5.4 are due to Brdys and Roberts (1984b). Only small changes

have been made.

5.2 Optimality conditions

In order to simplify the analysis it is assumed that the derivatives are calculated accurately. The following theorem gives sufficient conditions that $\Omega \subset \Omega^*$.

Theorem 5.1

Let v be an algorithm solution and ξ be the associated Lagrange multiplier vector. Assume (a) G(c,y) and Q(c,y) are differentiable on $R^n x R^m$, F(c, α) and $F^*(c)$ are differentiable on R^n. (b) For any $c \in C$, where $C \overset{\Delta}{=} \{c \in R^n: g^*(c) \leq 0\}$ is the feasible control set, the regularity conditions are satisfied. Then v satisfies the Kuhn-Tucker necessary optimality conditions for (ROP). If, additionally, C is convex and q^* is convex on C then v is a solution of (ROP).

Proof. Obviously, v is feasible, $\xi \geq 0$ and the following conditions are satisfied (see Section 4.2):

$$\bar{c}(v, \alpha(v), \xi) = v$$

$$\bar{\xi}(v, \alpha(v), \xi) = \xi .$$

By the definitions of $\bar{c}$ and $\bar{\xi}$ (Section 4.2) and according to assumptions (a) and (b) we have

$$\frac{\partial^T q(\bar{c}, \alpha(v))}{\partial c} - \lambda(v, \alpha(v), \xi) + \frac{\partial^T g(\bar{c}, \alpha(v))}{\partial c} \bar{\xi} = 0,$$

$$g(\bar{c}, \alpha(v)) \leq 0,$$

$$\bar{\xi}^T g(\bar{c}, \alpha(v)) = 0 \text{ and } \bar{\xi} \geq 0.$$

Notice that $F(v, \alpha(v)) = F^*(v)$ and using the definition of modifier vector

λ (equation (4.16) or (4.17)) we obtain

$$\frac{\partial^T q^*(v)}{\partial c} + \frac{\partial^T g^*(v)}{\partial c}\,\xi = 0,$$

$$g^*(v) \lesssim 0,$$

$$\xi^T g^*(v) = 0 \text{ and } \xi \gtrsim 0,$$

which is precisely the Kuhn-Tucker conditions at v for (ROP). The Kuhn-Tucker conditions become sufficient for optimality if q^* is convex (Luenberger, 1973). Q.E.D.

The above theorem, which is similar to Theorem 3.3, shows that the points produced by the modified two-step algorithm satisfy strong necessary conditions for optimality. If these necessary conditions are also sufficient $\Omega \subset \Omega^*$ is guaranteed. If, however, the problem is such that there is no equivalence between the necessary and sufficient conditions for optimality, it is preferred that at least one solution of (ROP) belongs to Ω. The following theorem provides sufficient conditions that $\Omega^* \subset \Omega$.

Theorem 5.2

Assume (i) For every $\alpha \in R^t$, the set C_α is convex, $g_i(., \alpha)$ i=1,...,p are quasi-convex on C_α and $q(., \alpha)$ is pseudo-convex on C_α, where $C_\alpha \overset{\Delta}{=} \{c \in R^n: g(c, \alpha) \lesssim 0\}$, $g(c, \alpha) = (g_1(c, \alpha), \ldots, g_p(c, \alpha))^T = G(c, F(c, \alpha))$ and $q(c, \alpha) = Q(c, F(c, \alpha))$. (ii) Every point of the set $\{(c, v, \alpha) \in R^n \times R^n \times R^t: F(v, \alpha) = F^*(v), g(c, \alpha) \lesssim 0 \text{ and } v = c\}$

satisfies regularity conditions. (iii) $\left[\frac{\partial^T F(v, \alpha(v))}{\partial \alpha}\right]^{-1}$ exists for every

$v \in C$. (iv) The model is point-parametric on C. Then every solution of (ROP) belongs to the algorithm solution set Ω.

Proof. Let c^* be a solution of (ROP). Due to assumption (iv) every triplet (c^*, v^*, α^*), where $v^* = c^*$ and α^* is any solution of the equation $F(v^*, \alpha) = F^*(v^*)$, solves (OP2) (see Lemma 3.1 and Section 4.2)

$$\begin{aligned} &\min_{c,v,\alpha} q(c, \alpha) \\ (\text{OP2}) \quad &\text{s.t. } F(v, \alpha) = F^*(v) \\ &\quad g(c, \alpha) \lesssim 0 \\ &\quad v = c. \end{aligned}$$

According to assumption (ii) there exist multiplier vectors $\lambda \in R^n$, $\eta \in R^m$ and $\xi \in R^p$ such that

$$\frac{\partial^T q(c^*, \alpha^*)}{\partial c} - \lambda + \frac{\partial^T g(c^*, \alpha^*)}{\partial c} \xi = 0, \tag{5.3}$$

$$\lambda + \left[\frac{\partial^T F(v^*, \alpha^*)}{\partial v} - \frac{T F^*(v^*)}{\partial v}\right] \eta = 0, \tag{5.4}$$

$$\frac{\partial^T q(c^*, \alpha^*)}{\partial \alpha} + \frac{\partial^T g(c^*, \alpha^*)}{\partial \alpha} \xi + \frac{\partial^T F(v^*, \alpha^*)}{\partial \alpha} \eta = 0, \tag{5.5}$$

$$g(c^*, \alpha^*) \lesssim 0, \tag{5.6}$$

$$\xi^T g(c^*, \alpha^*) = 0 \text{ and } \xi \gtrsim 0. \tag{5.7}$$

Assumption (iii) ensures that η can be solved from (5.5) and substituted into (5.4) to obtain

$$\lambda = \left[\frac{\partial^T F(v^*, \alpha^*)}{\partial v} - \frac{\partial^T F^*(v^*)}{\partial v}\right]\left[\frac{\partial^T F(v^*, \alpha^*)}{\partial \alpha}\right]^{-1} \left[\frac{\partial^T q(c^*, \alpha^*)}{\partial \alpha}\right]$$

$$+ \frac{\partial^T g(c^*, \alpha^*)}{\partial \alpha} \xi].$$

Because of $v^* = c^*$

$$\lambda = \lambda(v^*, \alpha^*, \xi) = [\frac{\partial^T F(v^*, \alpha^*)}{\partial v} - \frac{\partial^T F^*(v^*)}{\partial v}][\frac{\partial^T F(v^*, \alpha^*)}{\partial \alpha}]^{-1}$$

$$\cdot [\frac{\partial^T q(v^*, \alpha^*)}{\partial \alpha} + \frac{\partial^T g(v^*, \alpha^*)}{\partial \alpha} \xi] . \qquad (5.8)$$

(5.3), (5.6), (5.7) and (5.8) imply that c^* with associated Lagrange multiplier ξ satisfies the Kuhn-Tucker necessary optimality condition for (MMOP)

$$\text{(MMOP)} \quad \min_c \{q(c, \alpha^*) - \lambda^T(v^*, \alpha^*, \xi)c\}$$
$$\text{s.t. } g(c, \alpha^*) \leq 0. \qquad (5.9)$$

Due to assumption (i) c^* is a solution of (5.9) (see Mangasarian, 1965). Recall the definition of an algorithm solution $\bar{v}$ and associated Lagrange multiplier $\bar{\xi}$ is such that (Section 4.2)

$$\hat{c}(\bar{v}, \alpha(\bar{v}), \bar{\xi}) = \bar{v}$$

$$\hat{\xi}(\bar{v}, \alpha(\bar{v}), \bar{\xi}) = \bar{\xi},$$

where $\alpha(\bar{v})$ is a solution of $F(\bar{v}, \alpha) = F^*(\bar{v})$, $\hat{c}$ and $\hat{\xi}$ are a solution and corresponding Lagrange multiplier of (5.9) with given $\bar{v}$ and $\bar{\xi}$. We conclude that $c^* \in \Omega$. Q.E.D.

The conditions to ensure that the algorithm solution set includes all real optimum points require neither C is convex nor q^* is convex

(or pseudo-convex) on C. This indicates that the modified two-step algorithm has an advantage when using a simpler linear model to solve a non-linear steady-state optimisation control problem.

5.3 The algorithmic mapping of modified two-step algorithm

We will restrict the discussion of global convergence conditions only on a class of problems where inequality constraints do not depend on the system outputs y. There are two reasons for this restriction. Firstly, any C_α becomes identical to $C \stackrel{\Delta}{=} \{c \in R^n: G(c) \lesssim 0\}$. This ensures that it is possible to choose some iterative strategy that guarantees the feasibility of every points generated during the iterative procedure. Secondly, it is extremely difficult if not impossible to formulate the global convergence conditions when the inequality constraints involve y. The following discussion is an extension to the original work of Brdys and Roberts (1984b).

It is assumed that the modifier vector is calculated according to (4.17), which, in the absence of y in the inequality constraints, can be written as

$$\lambda = \lambda(v, \alpha) = \left[\frac{\partial^T F(v, \alpha)}{\partial v} - \frac{\partial^T F^*(v)}{\partial v}\right] \frac{\partial^T Q(v, F(v,\alpha))}{\partial y} . \qquad (5.10)$$

This ensures that the algorithmic mapping is autonomous, i.e., at kth iteration, the mapping depends only on v^k (see Appendix B, Definition B7.). If (4.16) is employed, the algorithmic mapping will become non-autonomous, i.e., it depends on $v^0, \ldots, v^k$. To prove the convergence of a non-autonomous algorithm is, in general, very difficult. It is worth

emphasizing that the optimality of the solution obtained remains the same using either (4.16) or (4.17).

At each iteration, the modified two-step algorithm consists of three stages as follows:

(a) Parameter estimation: For $v^k \in C$ determine the parameter vector α^k by solving $F(v^k, \alpha) = F^*(v^k)$. Clearly, the solution is not required to be unique.

(b) Modified model optimisation:

$$\min_{c \in C} \{q(c, \alpha^k) - (\lambda^k)^T c\},$$

where $\lambda^k = \lambda(v^k, \alpha^k)$ is calculated according to (5.10). Let c^k be a solution.

(c) Updating v: If $c^k = v^k$ the procedure is terminated. Otherwise $v^{k+1} = v^k + K_v(c^k - v^k)$ and the procedure is repeated.

In the following, the mappings representing these stages are defined. The overall algorithmic mapping then can be expressed as a composite of these mappings.

For $v \in C$ if we define the sets

$$\hat{\alpha}(v) = \{\alpha \in R^t : F(v, \alpha) = F^*(v)\}, \tag{5.11}$$

and

$$A \triangleq \bigcup_{v \in C} \hat{\alpha}(v). \tag{5.12}$$

Then a point-to-set mapping is given as

$$\hat{\alpha} : C \to A, \tag{5.13}$$

and Stage (a) can be described as: For a given $v^k \in C$ find α^k such that $\alpha^k \in \hat{\alpha}(v^k)$.

Let us define another point-to-set mapping

$$\hat{c}: C \times A \to C, \tag{5.14}$$

$$\text{as } \hat{c}(v, \alpha) \overset{\Delta}{=} \text{Arg} \min_{c \in C}\{q(c, \alpha) - \lambda^T(v, \alpha)c\}, \tag{5.15}$$

where the capital A of Arg indicates that the optimisation solutions are not required to be unique and

$$\lambda : C \times A \to R^n, \tag{5.16}$$

is determined by the relation (5.10). Then Stage (b) is described as: For a given $(v^k, \alpha^k) \in C \times A$ find c^k such that $c^k \in \hat{c}(v^k, \alpha^k)$.

We will limit the gain matrix K_v to be $K_v = \theta I$ and allow the positive scalar θ to change during the iterations, i.e.,

$$v^{k+1} = v^k + \theta^k(c^k - v^k) \text{ with } 0 < \tau \leq \theta^k \leq B(v^k)$$

where $B: C \to R$ is an appropriately defined function. Now let us define

$$\hat{\omega} : C \times (C-C) \to C, \tag{5.17}$$

$$\text{with } \hat{\omega}(v, d) \overset{\Delta}{=} \{v + \theta d \in C: \ \tau \leq \theta \leq B(v)\}. \tag{5.18}$$

Notice that the appropriate choice of $B(v)$ is of great importance for guaranteeing $v + \theta d \in C$. Stage (c) can be stated as: For a given v^k and c^k find v^{k+1} such that $v^{k+1} \in \hat{\omega}(v^k, c^k - v^k)$. If $c^k = v^k$, then $v^{k+1} = v^k$ and the iterative procedure is terminated.

Next, the following compositions of $\hat{\alpha}$, $\hat{c}$ and $\hat{\omega}$ are given. The Cartesian product of $\hat{\alpha}$ and the identity mapping I:

$$\hat{\delta}: C \to C \times A, \tag{5.19}$$

$$\text{with } \hat{\delta}(v) \overset{\Delta}{=} \{(v, \alpha) \in C \times A: \ \alpha \in \hat{\alpha}(v)\}. \tag{5.20}$$

The composite mapping of $\hat{\delta}$ and $\hat{c}$:

$$\hat{c}\,\hat{\delta}: C \to C. \tag{5.21}$$

Stages (a) and (b) together can be expressed as: For a given $v^k \in C$ find

$c^k \in \hat{C}\hat{\delta}(v^k)$. The sum mapping of $\hat{C}$ $\hat{\delta}$ and -I:

$$\hat{C}\hat{\delta} - I: C \to C-C, \tag{5.22}$$

with $(\hat{C}\hat{\delta} - I)(v) \overset{\Delta}{=} \{d = c - v \in C-C: c \in \hat{C}\hat{\delta}(v)\}$. (5.23)

The Cartesian product of $\hat{C}$ $\hat{\delta}$ -I and I:

$$\tilde{\nu}: C \to C \times (C-C), \tag{5.24}$$

with $\tilde{\nu}(v) \overset{\Delta}{=} \{(v, d) \in C \times (C-C): d \in (\hat{C}\hat{\delta} - I)(v)\}$. (5.25)

Finally, the overall algorithmic mapping of the modified two-step algorithm is defined as the composition of $\tilde{\nu}$ and $\hat{\omega}$, i.e.,

$$\Phi = \hat{\omega}\tilde{\nu}, \tag{5.26}$$

and the kth iteration of the algorithm is expressed as: For a given $v^k \in C$ find v^{k+1} such that $v^{k+1} \in \Phi(v^k)$. In a more general sense, the set of all fixed points of Φ can be defined as the algorithmic solution set

$$\Omega \overset{\Delta}{=} \{v \in C: v \in \Phi(v)\}. \tag{5.27}$$

If the solutions at Stages (a) and (b) are unique and a fixed scalar θ is chosen the above mappings are all point-to-point mappings.

5.4 Global convergence conditions

Mathematical preliminaries are given in Appendix B to help understanding the following theorem.

Theorem 5.3 (global convergence theorem) (Brdys and Roberts,1984b) Assume (1) The set C is bounded and the set A is compact. (2) The functions G_j, j=1,...,p are convex and continuous on R^n. (3) The function q^* is differentiable on C and its derivative is Lipschitz continuous on C with Lipschitz constant $\rho > 0$, i.e.,

$$\| \nabla q^*(c + h) - \nabla q^*(c) \| \leq \rho \|h\|, \text{ for all } c,\ c + h \in C,$$

where $\|.\|$ denotes the Euclidean norm. (4) The function F^* is continuous on R^n and continuously differentiable on C. (5) The function F is continuous on $R^n \times R^t$ and $F(., \alpha)$ is continuously differentiable on C for all $\alpha \in A$. (6) F is point-parametric on C, i.e., the set $\hat{\alpha}(v)$ is not empty for all $v \in C$. (7) The function Q is continuous on $R^n \times R^m$ and $Q(c,.)$ is continuously differentiable on R^m for all $c \in C$. (8) For every $\alpha \in A$, the function $q(., \alpha)$ is differentiable on C and its derivative is uniformly monotone on C with constant $b(\alpha) > 0$, i.e.,

$$[\nabla_c q(c + h, \alpha) - \nabla_c q(c, \alpha)]^T h \geq b(\alpha)\|h\|^2, \text{ for all } c,\ c + h \in C,$$

where $b: A \to R$ is upper semicontinuous on A and the following holds:

$$\inf_{\alpha \in A} b(\alpha) > 0. \tag{5.28}$$

(9) The point-to-set mapping $\hat{\alpha}$ is open on C. Then

(i) If τ and ε are chosen such that $1 \geq \tau > 0$, $\varepsilon > 0$ and

$$B(v) \overset{\Delta}{=} \min\{1, \frac{2}{\rho} \inf_{\alpha \in \hat{\alpha}(v)} b(\alpha) - \varepsilon\}, \tag{5.29}$$

$$\tau + \varepsilon \leq \frac{2}{\rho} \inf_{\alpha \in A} b(\alpha), \tag{5.30}$$

then the algorithmic mapping Φ is well-defined on C to C and closed outside the set Ω.

(ii) There exists at least one cluster point of the sequence $\{v^k\}$ generated by the algorithm $v^{k+1} \in \Phi(v^k)$. Each cluster point belongs to Ω and, moreover, each point v^k satisfies the real process constraints and corresponding process performance values satisfy the condition:

$$q^*(v^{k+1}) < q^*(v^k) \text{ if } v^k \notin \Omega \text{ for } k=1,2,\ldots . \tag{5.31}$$

We employ Zangwill's convergence theorem (see Appendix B) to prove this theorem. The proof is given in the following two subsections.

5.4.1 Proof of assertion (i) of Theorem 5.3

a) $\tilde{\nu}$ is closed on C.

Assumption (6) ensures that $\hat{\alpha}$ is well-defined. The continuity of F^* and F (assumptions (4) and (5)) implies that $\hat{\alpha}$ is closed on C. In fact, for $v \in C$, let $\{v^k\} \subset C$, $v^k \to v$ and $\alpha^k \to \alpha$, $\alpha^k \in \hat{\alpha}(v^k)$. We need to prove

$$\alpha \in \hat{\alpha}(v) \overset{\Delta}{=} \{ \bar{\alpha} \in R^t : F(v, \bar{\alpha}) = F^*(v)\}.$$

But $F(v^k, \alpha^k) = F^*(v^k)$ hence, $F(v, \alpha) = F^*(v)$, i.e., $\alpha \in \hat{\alpha}(v)$.

According to assumptions (4), (5) and (7), the function $\lambda(v, \alpha)$ is apparently continuous on C x A, therefore, the function

$$q(c, \alpha) - \lambda^T(v, \alpha)c, \tag{5.32}$$

is continuous on C x(C x A). Assumptions (1) and (2) imply that C is compact and convex and hence, CxA and Cx(C x A) are compact. Therefore, the mapping $\hat{c}$ is well-defined (a continuous function achieves its minimums on a compact set). Moreover, assumption (8) implies that $\nabla_c q(c, \alpha) - \lambda(v, \alpha)$ is uniformly monotone on C with the same constant $b(\alpha)$ for all $(v, \alpha) \in C \times A$. This ensures that function (5.32) is strictly convex on the convex set C for any $(v, \alpha) \in C \times A$, and $\hat{c}(v, \alpha)$ consists of a single point for a given $(v, \alpha) \in CxA$. Thus $\hat{c}$ is a point-to-point continuous mapping on C x A (Appendix B, Theorem B5).

Proving the closedness of $\tilde{\nu}$ given in (5.24) is now straightforward. The mapping $\hat{\delta}$ defined in (5.19) is closed on C due to the closedness of $\hat{\alpha}$ and identical mapping I (Theorem B3). Next, the mapping $\hat{c}\,\hat{\delta}$ given

in (5.21) is closed on C (Theorem B1) and $\hat{c}\ \hat{\delta}$ - I defined in (5.22) is closed on C (Theorem B2). Finally $\tilde{\nu}$ is closed on C because it is the Cartesian product of $\hat{c}\ \hat{\delta}$ - I and I.

b) $\hat{\omega}$ is closed at (v, d), where $d \neq 0$ and $(v, d) \in C \times (C - C)$.
First it is easy to check that $\hat{\omega}$ is well-defined, i.e., $\hat{\omega}(v,d)$ is not empty for all $(v,d) \in Cx(C - C)$. Let $(v,d) \in C \times (C - C)$, we need to prove that the gain θ defined in (5.18) exists, i.e., to prove $\tau \leq B(v)$. Due to (5.30) the following is true:

$$\tau + \varepsilon \leq \frac{2}{\rho} \inf_{\alpha \in A} b(\alpha) \leq \frac{2}{\rho} \inf_{\alpha \in \hat{\alpha}(v)} b(\alpha) \quad \text{and} \quad \tau \leq 1,$$

or $\tau \leq \frac{2}{\rho} \inf_{\alpha \in \hat{\alpha}(v)} b(\alpha) - \varepsilon$ and $\tau \leq 1$, consequently, $\tau \leq B(v)$.

Next we show that $B: C \to R$ is upper semicontinuous on C. Consider the function from C to R:

$$\inf_{\alpha \in \hat{\alpha}(v)} b(\alpha) .$$

$\hat{\alpha}$ is open on C (assumption (9)) and b is upper semicontinuous on A (assumption (8)) imply that the above function is upper semicontinuous on C (Theorem B4). Now one can easily check that B is upper semicontinuous from the definition of upper semicontinuous (Appendix B, definition B9).

For any $(v,d) \in Cx(C - C)$ safisfying $d \neq 0$, let $(v^k,d^k) \in Cx(C - C)$ and $(v^k,d^k) \to (v,d)$, and let $y^k = v^k + \theta^k d^k \to \bar{y}$, $y^k \in \hat{\omega}(v^k,d^k)$. We need to prove $\bar{y} \in \omega(v, d)$. Define

$$\bar{\theta} \overset{\Delta}{=} \frac{\|\bar{y} - v\|}{\|d\|} .$$

We have $\bar{\theta} = \lim_{k \to \infty} \frac{\|y^k - v^k\|}{\|d^k\|} = \lim_{k \to \infty} \theta^k$. Obviously $\bar{y} = v + \bar{\theta} d$. Since

$\tau \leq \theta^k \leq B(v^k)$ and because B is upper semicontinuous, then

$$\tau \leq \bar{\theta} \leq \limsup_{k \to \infty} B(v^k) \leq B(v).$$

Therefore, $\bar{y} \in \hat{\omega}(v,d)$. This proves that $\hat{\omega}$ is closed at $(v,d) \in C \times (C - C)$, where $d \neq 0$.

c) Assertion (i) holds.

Clearly Φ is well-defined because $\hat{\omega}$ and $\tilde{\nu}$ are both well-defined. To show $\Phi(C) \subset C$, notice that for a given $v \in C$ we have $\hat{c}(v, \alpha) \in C$ for all $\alpha \in \hat{\alpha}(v)$. Because $B(v) \leq 1$ and $\tau > 0$, the convexity of C implies $v + \theta(\hat{c}(v, \alpha) - v) \in C$ for all $\tau \leq \theta \leq B(v)$. Thus $\Phi(v) \subset C$.

Finally, for a $v \in C$ satisfying $v \notin \Omega$, any $(v,d) \in \tilde{\nu}(v)$ satisfies $d \neq 0$. The closedness of $\tilde{\nu}$ and $\hat{\omega}$ ensures that Φ is closed at v (Theorem B1). The proof of assertion (i) is complete.

5.4.2 Proof of assertion (ii) of Theorem 5.3

The feasiblity of a sequence $\{v^k\}$ generated by the algorithm has already been proved. What remains to be proved is that a continuous Zangwill function Z: $C \to R$ exists. We will show that Z can be chosen as q^*.

For v, $c \in C$, because of assumption (3) and the convexity of C, the following holds (Appendix B, Theorem B7):

$$q^*(v) - q^*(c) \geq \frac{\partial q^*(v)}{\partial v}(v - c) - \frac{\rho}{2}\|v - c\|^2.$$

Let $c \in \Phi(v)$, i.e., $c = v + \theta[\hat{c}(v, \alpha) - v]$ for some $\alpha \in \hat{\alpha}(v)$ and $\tau \leq \theta \leq B(v)$, then

$$q^*(v) - q^*(c) \geq \theta \frac{\partial q^*(v)}{\partial v}(v - \hat{c}(v, \alpha)) - \frac{\theta^2 \rho}{2}\|v - \hat{c}(v, \alpha)\|^2. \quad (5.33)$$

For any $(v, \alpha) \in C \times A$, we already know that function (5.32) is strictly convex on the convex set C with $\hat{c}(v, \alpha)$ as its unique global minimum point over C. Therefore, the following holds (Theorem B6):

$$[\nabla_c q(\hat{c}(v, \alpha), \alpha) - \lambda(v, \alpha)]^T (v - \hat{c}(v, \alpha)) \geq 0. \quad (5.34)$$

According to (5.10)

$$\lambda(v, \alpha) = \nabla_c F(v, \alpha) \nabla_y Q(v, F(v, \alpha)) - \nabla F^*(v) \nabla_y Q(v, F(v, \alpha))$$

$$= \nabla_c q(v, \alpha) - \nabla_c Q(v, F(v, \alpha)) - \nabla F^*(v) \nabla_y Q(v, F(v, \alpha))$$

$$- \nabla q^*(v) + \nabla_c Q(v, F^*(v)) + \nabla F^*(v) \nabla_y Q(v, F^*(v)).$$

Because $F(v, \alpha) = F^*(v)$ we have

$$\lambda(v, \alpha) = \nabla_c q(v, \alpha) - \nabla q^*(v). \quad (5.35)$$

Substitute (5.35) into (5.34) we obtain

$$\frac{\partial q^*(v)}{\partial c}(v - \hat{c}(v, \alpha)) \geq \left(\frac{\partial q(v, \alpha)}{\partial c} - \frac{\partial q(\hat{c}(v, \alpha), \alpha)}{\partial c}\right)(v - \hat{c}(v, \alpha)). \quad (5.36)$$

According to assumption (8) (5.36) can be rewritten as

$$\frac{\partial q^*(v)}{\partial c}(v - \hat{c}(v, \alpha)) \geq b(\alpha)||v - \hat{c}(v, \alpha)||^2. \tag{5.37}$$

Combining (5.33) and (5.37) we have

$$q^*(v) - q^*(c) \geq \theta\,(b(\alpha) - \frac{\theta\rho}{2})||v - \hat{c}(v, \alpha)||^2. \tag{5.38}$$

Finally, due to (5.29) the following holds:

$$0 < \tau \leq \theta \leq B(v) \leq \frac{2}{\rho}\inf_{\alpha \in \hat{\alpha}(v)} b(\alpha) - \varepsilon \leq \frac{2}{\rho} b(\alpha) - \varepsilon,$$

or $b(\alpha) - \frac{\theta\rho}{2} \geq \frac{\varepsilon\rho}{2} > 0$ and $\theta \geq \tau > 0$. Hence,

$$q^*(v) - q^*(c) \geq \frac{\tau\varepsilon\rho}{2}||v - \hat{c}(v, \alpha)||^2 \geq 0. \tag{5.39}$$

Because $\frac{\tau\varepsilon\rho}{2} > 0$ and if $v \notin \Omega$, $||v - \hat{c}(v, \alpha)|| \neq 0$, (5.39) indicates

$q^*(c) < q^*(v)$ if $v \in \Omega$ and $q^*(c) < q^*(v)$ if $v \notin \Omega$, where $c \in \Phi(v)$.

Now all the conditions in Zangwill's convergence theorem have been verified. We conclude that assertion (ii) is true. Q.E.D.

5.5 Discussion of assumptions in Theorem 5.3

Assumptions (1) and (2) are mild. C is often bounded in practical problems. An unconstrained case has also been considered by Brdys' and

Roberts (1984b). If q^* is twice continuously differentiable on C assumption (3) is satisfied. Assumptions (4) to (7) are standard and are usually fulfilled.

The crucial assumption is assumption (8). It requires that the composition of the performance index and the process mathematical model is a uniformly convex function on the feasible control set. However, no assumption requires that q^* is convex. The modified two-step algorithm can be applied to highly nonlinear processes using a simple linear model as the following Lemma shows.

Lemma 5.1

Suppose the process mathematical model has the form:

$$F(c, \alpha) = Dc + P(\alpha), \tag{5.40}$$

where D is a mxn constant matrix and P is a vector function of α . If Q is differentiable and uniformly convex on C x R^m, then assumption (8) is verified.

Proof: Because Q is differentiable and uniformly convex on C x R^m, for any $c, c+h_1 \in C$ and $y, y+h_2 \in R^m$, there exists a constant $\sigma > 0$. such that the following holds (Ortega and Rheinboldt, 1970):

$$[\nabla_c Q(c+h_1, y+h_2) - \nabla_c Q(c,y)]^T h_1 + [\nabla_y Q(c+h_1, y+h_2) - \nabla_y Q(c,y)]^T h_2 \geq \sigma (||h_1||^2 + ||h_2||^2). \tag{5.41}$$

Notice that for any $c \in C$, $\alpha \in A$,

$$\nabla_c q(c, \alpha) = \nabla_c Q(c, F(c, \alpha)) + \nabla_c F(c, \alpha) \nabla_y Q(c, F(c, \alpha))$$
$$= \nabla_c Q(c, F(c, \alpha)) + D^T \nabla_y Q(c, F(c, \alpha)).$$

Therefore

$$[\nabla_c q(c+h_1, \alpha) - \nabla_c q(c, \alpha)]^T h_1 = [\nabla_c Q(c+h_1, F(c+h_1, \alpha)) - \nabla_c Q(c, F(c, \alpha))]^T h_1$$

$$+ [\nabla_y Q(c+h_1, F(c+h_1, \alpha)) - \nabla_y Q(c, F(c, \alpha)]^T D h_1. \tag{5.42}$$

Let $y = F(c, \alpha)$ and $y + h_2 = F(c+h_1, \alpha)$, then $h_2 = Dh_1$. Combining (5.42) and (5.41) we have:

$$[\nabla_c q(c+h_1, \alpha) - \nabla_c q(c, \alpha)]^T h_1 \geq \sigma \ (\|h_1\|^2 + \|Dh_1\|^2)$$

$$= \sigma(\|h_1\|^2 + h_1^T(D^TD)h_1). \qquad (5.43)$$

Let $r_{min}(D^TD)$ denote the minimal eigenvalue of D^TD, then for any $c \in R^n$, $c^T(r_{min}(D^TD))c \leq c^T(D^TD)c$. Because D^TD is at least positive semidefinite, $r_{min}(D^TD) \geq 0$ and (5.43) can be written as

$$[\nabla_c q(c+h_1, \alpha) - \nabla_c q(c, \alpha)]^T h_1 \geq \sigma \ (1 + r_{min}(D^TD))\|h_1\|^2.$$

We conclude that assumption (8) is satisfied with

$$b(\alpha) \stackrel{\Delta}{=} \sigma(1 + r_{min}(D^TD)) > 0, \text{ for any } \alpha \in A. \qquad (5.44)$$

Q.E.D.

One immediate observation of (5.44) is that $b(\alpha)$ is a positive constant. Therefore, B(v) is a positive constant for any $v \in C$ and a fixed gain θ can be chosen for every iteration. This has a benefit of simplifying the iterative procedure. It is, of course, desirable to weaken assumption (8). Unfortunately, this can be achieved only at the cost of strengthening the requirements on the constraint functions G_j, $j=1,\ldots p$ (Brdys and Roberts, 1984b).

Sufficient condition for the openness of $\hat{\alpha}$ (assumption (9)) can be found in Hogan (1973), and Greenberg and Pierskalla (1972). However, if the result of the parameter estimation is unique, i.e., $\hat{\alpha}$ is a point-to-point mapping, other assumptions have sufficiently guaranteed that $\hat{\alpha}$ is a continuous function on C as shown in Section 5.4.1. This will be enough to prove the upper semicontinuity of B(v). Therefore, assumption (9) can be omitted.

Finally, consider an application of Theorem 5.3 and Lemma 5.1 to the following simple example.

Simple example 2

The model and reality equations are

$$y = F(c, \alpha) = c + \alpha$$

$$y = F^*(c) = c + c^2. \tag{5.45}$$

The performance index and constraint set are

$$Q(c,y) = c^2 + (y - 2)^2$$

$$C_Y \overset{\Delta}{=} \{(c,y) \in R^2: -1 \leq c \leq 1\}.$$

A simple check shows that assumptions (1) to (8) in Theorem 5.3 are all valid and assumption (9) is not needed. We, therefore, have confidence in that the iterative procedure will converge and an iterative strategy can be chosen such that all points generated during the procedure are feasible and the real value of the performance index is decreased at each iteration until convergence is achieved. In fact, with the aid of Lemma 5.1, the iterative strategy can be chosen as (see Appendix F)

$$v^{k+1} = v^k + 0.4(c^k - v^k). \tag{5.46}$$

Furthermore, because q^* is convex, according to Theorem 5.1 the iterative procedure will converge to the correct real optimum. The iterative results are presented in TABLE 5.1 where $v^0 = 0$.

In practice, the real equation (5.45) is not available and the Lipschitz constant ρ is not known. The gain in (5.46) then has to be found by experiment.

5.6 Summary

Theoretical analysis of the modified two-step algorithm has been

presented. Under mild assumptions it has been shown that an iterative strategy can be chosen for the algorithm such that each point generated by the iterative procedure is feasible, the value of the real process performance index is improved at each iteration and the algorithm converges to a real process optimal operating condition.

Although, in order to guarantee convergence, the composition of the performance index and the mathematical model is required to be a uniformly convex function on the feasible control set, the real process input-output mapping can be nonlinear and the composition of the performance index and the real process input-output mapping can be non-convex. The analysis result shows that there are advantages of using simple linear models when solving non-linear steady-state optimisation control problems by the modified two-step algorithm.

Assumptions in Lemma 5.1 are sufficient conditions. Some practical applications (Roberts, 1979; Ellis and Roberts, 1985) involving nonlinear models have exhibited excellent convergent behaviour.

TABLE 5.1 Iterative results of Simple example 2

iteration	v	q^*	α	λ	c
1	0.00000	4.00000	0.00000	0.00000	1.00000
2	0.40000	2.23360	0.16000	2.30400	1.00000
3	0.64000	1.31286	0.40960	2.43302	1.00000
4	0.78400	0.97627	0.61466	1.88581	1.00000
5	0.87040	0.89598	0.75760	1.29517	0.94499
6	0.90024	0.89414	0.81043	1.04188	0.85526
7	0.88224	0.89355	0.77836	1.19773	0.91025
8	0.89345	0.89330	0.79825	1.10180	0.87632
9	0.88660	0.89321	0.78606	1.16089	0.89719
10	0.89084	0.89318	0.79359	1.12450	0.88433
11	0.88823	0.89316	0.78896	1.14691	0.89225
12	0.88984	0.89316	0.79181	1.13311	0.88737
13	0.88885	0.89316	0.79006	1.14161	0.89037
14	0.88946	0.89316	0.79114	1.13637	0.88852
15	0.88909	0.89316	0.79047	1.13960	0.88966
16	0.88932	0.89316	0.79088	1.13761	0.88896
17	0.88917	0.89316	0.79063	1.13884	0.88939
18	0.88926	0.89316	0.79079	1.13808	0.88913
19	0.88921	0.89316	0.79069	1.13855	0.88929
20	0.88924	0.89316	0.79075	1.13826	0.88919
21	0.88922	0.89316	0.79071	1.13844	0.88925
22	0.88923	0.89316	0.79074	1.13833	0.88921
23	0.88923	0.89316	0.79072	1.13839	0.88924
24	0.88923	0.89316	0.79073	1.13835	0.88922
25	0.88923	0.89316	0.79073	1.13838	0.88923
26	0.88923	0.89316	0.79073	1.13836	0.88923
27	0.88923	0.89316	0.79073	1.13837	0.88923
28	0.88923	0.89316	0.79073	1.13837	0.88923

CHAPTER 6. TWO-LEVEL TYPE METHOD

6.1 Introduction

The modified two-step algorithm requires the real output derivatives with respect to controller set points for the purpose of calculating the modifiers. This produces a serious practical consideration particularly when the measurements are contaminated by noise. For this reason it is often advisable to apply simple software filtering to the computation of the modifiers λ as shown in Chapter 4. However, in order to achieve a robust performance under noisy measurement conditions it is preferred that an algorithm requires no process derivatives.

In this chapter, an alternative modification of the two-step approach is investigated in which an additional level is introduced to prescribe modifiers by minimizing the performance index. This idea was first suggested by Brdys in a manuscript. If some appropriate strategy is chosen the resulted algorithm does not require process derivatives. On the other hand, the second level only has information on the values of the performance function and, consequently, the computational requirement is increased as the payment. Another advantage of the proposed algorithm is that even when the inequality constraint functions G(c,y) are not differentiable with respect to y the algorithm may still be used while the modified two-step algorithm will have difficulty because it needs the derivatives of G(c,y) with respect to y for calculating the modifiers.

6.2 Formulation of two-level type method

6.2.1 Preliminaries

Let us examine an alternative way to solve (OP2) of (4.4) to (4.7) given in Section 4.2. Assume, as in Section 4.2, that the regularity conditions are satisfied and consider first (MMOP) of (4.18) and (4.19):

$$\min_{c}\{q(c, \alpha) - \lambda^{T}c\}, \tag{6.1}$$

$$\text{s.t. } g(c,\alpha) \lesssim 0. \tag{6.2}$$

If, unlike in Section 4.2, λ is not evaluated according to (4.17) (or (4.16)), rather, it is regarded as an independent variable vector, then the solution of (6.1) and (6.2) is dependent on α and λ. clearly, the solution and associated Lagrange multiplier $\hat{c}(\alpha, \lambda)$ and $\hat{\xi}(\alpha, \lambda)$ of (6.1) and (6.2) satisfy (4.9), (4.14) and (4.15) with given α and λ.

As in Section 4.2, let $\hat{\alpha}(v)$ be the solution of (4.13) with given v, then for a given λ any solution of

$$\hat{c}(\hat{\alpha}(v),\lambda) = v, \tag{6.3}$$

together with corresponding Lagrange multiplier $\hat{\xi}(\hat{\alpha}(v), \lambda)$ satisfies conditions (4.9), (4.12) to (4.15). Denote the solution of (6.3) as $\hat{v}(\lambda)$. If $\hat{v}(\lambda)$ exists it is always feasible, i.e.,

$$\hat{v}(\lambda) \in C \overset{\Delta}{=} \{c \in R^{n}: g^{*}(c) \lesssim 0\}.$$

Conditions (4.10) and (4.11) are not necessarily satisfied if λ is arbitrarily chosen. Remember that our final aim is to find a c_{opt}

such that

$$q^*(c_{opt}) = \min_{c \in C} q^*(c). \tag{6.4}$$

If such a c_{opt} can be found, whether or not we know the precise values of associated Lagrange multipliers η and ξ is not particularly important. Let $v_{opt} = c_{opt}$ and α_{opt} be a solution of (4.13) with given v_{opt} then (c_{opt}, v_{opt}, α_{opt}) solve (OP2) of (4.4) to (4.7) with associated Lagrange multipliers $\lambda_{opt} \in R^n$, $\eta_{opt} \in R^m$ and $\xi_{opt} \in R^p$. If, for example, the solution of (4.13) is unique and the optimisation problem (6.1) and (6.2) is convex for given α and λ, the following holds:

$$c_{opt} = \hat{v}(\lambda_{opt}).$$

Let us define the set:

$$\mathcal{A} \overset{\Delta}{=} \{\lambda \in R^n: \hat{v}(\lambda) \text{ exists}\}, \tag{6.5}$$

and $$\bar{q}(\lambda) \overset{\Delta}{=} q(\hat{v}(\lambda), \hat{\alpha}(\hat{v}(\lambda))). \tag{6.6}$$

Since $\hat{v}(\lambda)$ is always feasible $\bar{q}(\lambda) \geq q^*(c_{opt})$. This indicates that a proper $\bar{\lambda}$ can be found by solving the minimization problem

$$\min_{\lambda \in \mathcal{A}} \bar{q}(\lambda), \tag{6.7}$$

such that $\hat{v}(\bar{\lambda})$ is a global minimal point of (6.4). At least one $\bar{\lambda}$ exists (λ_{opt} is a obvious candidate).

6.2.2 Description of the two-level type algorithm

Based on the above discussion, the following two-level algorithm is proposed. The infimal level is precisely a two-step procedure as given in Section 3.3 except that the model optimisation problem (3.8) has been modified into

$$\min_{c}\{q(c, \alpha^k) - \lambda^T c\}, \qquad (6.8)$$

with λ given by the supremal level. The solution of the infimal problem is denoted as $\hat{v}(\lambda)$.

The supremal level's task is to find a λ_{opt} such that

$$\bar{q}(\lambda_{opt}) = \min_{\lambda \in \Lambda} \bar{q}(\lambda).$$

where $\bar{q}(\lambda)$ is defined by the relation (6.6). It often occurs that Λ is a whole space and the supremal task is an unconstrained optimisation problem. This may be because λ does not affect the constraint functions. The information structure of the algorithm is shown in Fig. 6.1.

6.2.3 Some comments

The functional relation $\bar{q}(\lambda)$ is not explicitly known. It is well-known from mathematical programming theory that $\bar{q}(\lambda)$ may not be differentiable. Basically any technique of minimizing without calculating derivatives can be used to solve the supremal problem. For example, some coordinate descent techniques, especially those which adopt approximately conjugate directions, can be employed as the supremal algorithm. If the number of the controller set points is not large (e.g. less than 6), the simplex search method may be adequate.

It is worth emphasising that λ_{opt} is not necessarily unique even when the modified two-step algorithm has a unique solution of the modifier vector, in another words, even when the Lagrange multiplier λ

associated with (OP2) is unique. To show this consider Simple example 1 introduced in Section 3.3. Applying the two-level type algorithm to this example obtains the following results (see Appendix C):

For any $\lambda \leq -2$, the infimal solution is the real optimum $\hat{v}(\lambda) = -1$ and $\bar{q}(\lambda) = 1$. Fig. 6.2. illustrates the infimal solutions as a function of λ. Because of this, the stop criterion for the supremal level algorithm should be based on the values of the performance function instead of on the modifers.

6.3 Applicability conditions

The two-level type algorithm can be efficiently used providing that

a) There exist the modifiers λ_{opt}.

b) Infimal iterations are convergent.

c) There is an efficient strategy, i.e., algorithm for adjusting λ to achieve λ_{opt}.

Let us define the set $A \overset{\Delta}{=} \bigcup_{v \in C} \hat{\alpha}(v)$ and the mapping $\hat{\alpha} : C \to A$. The following theorem gives sufficient conditions for existence of λ_{opt}.

Theorem 6.1 (existence of supremal solution)

Assume (i) Q is continuous on $R^n \times R^m$. (ii) F^* is continuous on R^n, F is continuous on $R^n \times R^t$ and $\hat{\alpha}$ is continuous on C. (iii) $\hat{v}(\lambda)$ is continuous on Λ. (iv) Λ is compact. Then there exists at least one λ_{opt} which solves the supremal problem, i.e.,

$$\bar{q}(\lambda_{opt}) = \min_{\lambda \in \Lambda} \bar{q}(\lambda).$$

Proof. Assumptions (i) to (iii) ensure that the mapping

$$\bar{q}: \mathcal{A} \to R,$$

$$\bar{q}(\lambda) \stackrel{\Delta}{=} Q(\hat{v}(\lambda), F(\hat{v}(\lambda), \hat{\alpha}(\hat{v}(\lambda)))) = Q(\hat{v}(\lambda), F^*(\hat{v}(\lambda))),$$

is continuous. Because a continuous function on a compact set attains its infimum, the assertion of Theorem 6.1 holds. Q.E.D.

Sufficient conditions for the continuity of $\hat{\alpha}$ can be derived based on the implicit function theorem (e.g. Bertsekas, 1982; Ortega and Rheinboldt, 1970). The key assumption is assumption (iii). Lemma 6.1 gives sufficient conditions to satisfy assumption (iii).

Lemma 6.1

Assume (a) The model is point-parametric on C. (b) $\hat{\alpha}$ is Lipschitz continuous on C with constant $\rho_1 > 0$, i.e.,

$$||\hat{\alpha}(c_1) - \hat{\alpha}(c_2)|| \leq \rho_1 ||c_1 - c_2||, \text{ for any } c_1, c_2 \in C.$$

(c) $\hat{c}$ is continuous on $A \times \mathcal{A}$. (d) For any $\lambda \in \mathcal{A}$, $\hat{c}(., \lambda)$ is Lipschitz continuous on A with constant $\rho_2 > 0$, i.e.,

$$||\hat{c}(\alpha_1, \lambda) - \hat{c}(\alpha_2, \lambda)|| \leq \rho_2 ||\alpha_1 - \alpha_2||, \text{ for any } \alpha_1, \alpha_2 \in A.$$

(e) C is closed. Then, if $\rho_1 \rho_2 < 1$ there exists a unique infimal solution $\hat{v}(\lambda)$ continuous on $\mathcal{A}$.

Proof. Define the infimal algorithmic mapping of (6.1) and (6.2)

$$T: C \times \mathcal{A} \to C,$$

$$T(c, \lambda) \stackrel{\Delta}{=} \hat{c}(\hat{\alpha}(c), \lambda).$$

Due to assumptions (a) to (c), T is well-defined continuous on $C \times \mathcal{A}$. For any $\lambda \in \mathcal{A}$, according to assumptions (b) and (d)

$$||T(c_1, \lambda) - T(c_2, \lambda)|| \leq \rho_1 \rho_2 ||c_1 - c_2||, \text{ for any } c_1, c_2 \in C.$$

Finally, because C is closed and $\rho_1 \rho_2 < 1$, there exists a unique fixed point $\hat{v}(\lambda)$ of $T(., \lambda)$, which is continuous on $\mathcal{A}$ (see Appendix A, A2). Q.E.D.

It is, in general, difficult to form the sufficient conditions which guarantee the global convergence of the infimal iterations for any $\lambda \in \Lambda$, especially when the constraints involve y. Therefore, we are unable to discuss point b) given in the beginning of this section.

In the next section using the simplex search method and Powell's method as the supremal algorithm is considered. The convergence conditions of these two methods have well been established (e.g. Luenberger, 1973, 1984; Nelder and Mead, 1965; Zangwill, 1969).

6.4 Techniques for solving the supremal problem

6.4.1 Simplex search method

The simplex method is a search procedure for minimizing a function of n variables. Its strengths are that it requires no derivatives, so it can cope with functions which are not easily written as analytic expressions, and that it always increases the information available concerning the function by reporting its values at a number of points. Its weakness is primarily that it does not use this information very effectively, so may take an unnecessarily large number of function evaluations to locate a solution.

A simplex is the structure formed by (n+1) points in an n-dimensional space. These points should not be in the same plane. The essence of the algorithm is as follows: The function is evaluated at each point (vertex) of the simplex and the vertex having highest function value is replaced by a new point with a lower function value.

This allows the simplex to adapt itself and, finally, the procedure contracts on to a final minimum. There are four main operations which are made on the simplex: reflection, expansion, reduction and contraction. The technical details can be found in the literature (Avriel, 1976; Nash, 1979; Nelder and Mead, 1965).

A program listing of a simplex algorithm written in FORTRAN is given in Appendix D. To use this routine (MIFSIM) as the supremal algorithm of the two-level type method, the user must supply the subroutine FUN. The purpose of FUN is to solve the infimal problem iteratively for any given λ and to report the resulted value of the performance index to MIFSIM. Suitable values of the reflection factor ALPHA, the contraction factor BETA and the extension factor GAMMA depend on the particular problem to be solved. In practice, they can be chosen to try to improve the rate of convergence of the procedure. The following values:

ALPHA = 1, BETA = 0.5 and GAMMA = 2

have been found to be effective in many tests. The desired tolerance TOL is generally chosen to be larger than that used in the infimal loop.

6.4.2 Powell's method

This method (Avriel, 1976; Powell, 1964, 1972; Zangwill, 1967), from the theoretical point of view, belongs to the category of coordinate descent methods. The basic idea of the method can be described as follows: Each stage of the procedure consists of n+1 successive one-dimensional linear searches, first along n linearly

independent directions and then along the direction connecting the best point (obtained at the end of the n-dimensional linear searches) with the starting point of that stage. After these searches, a judgement is made to decide whether or not one of the first n directions is to be replaced by the (n+1)th direction, and then a new stage begins.

The Powell's method has a fine property of quadratic termination, i.e., using the method to find the minimum of a quadratic function on R^n

$f(x) = a + bx + x^T Qx$, where Q is positive definite,

it will terminate after at most n successive one-dimensional linear searches along n mutually conjugate directions with respect to Q. The method is recognized as an effective technique for minimizing functions without using derivatives.

A FORTRAN program listing of a Powell's method is presented in Appendix E. A parabolic interpolation linear search technique is incorporated with the algorithm to perform one-dimensional linear searches. Efforts have been made to reduce the number of function evaluations as much as possible since each function evaluation corresponds to an infimal iterative loop. Coordinate directions can be chosen as the initial search directions.

6.4.3 Some remarks

These two techniques, in general, can only find a local minimum. Therefore, if $\mathcal{A}$ is not convex, the solution obtained may depend on the initial value of the λ. Simplex search method is very robust.

However, for more than about five-dimensional problems, the procedure appears to become inefficient. Powell's method, on the other hand, is more efficient.

From the structure point of view, the two-level type method seems to be a natural modification of the two-step method. If a suboptimum solution is acceptable the algorithm may be operated such that the supremal procedure can be terminated at any stage when the infimal solution has been improved enough compared with that obtained using the two-step method. Since the infimal solution is always feasible it causes no difficulty. Such an operating strategy is especially suitable if the simplex technique is employed as the supremal algorithm since a simplex can quickly reduce itself to a reasonable small size which contains a minimum but takes a long time to contract finally on to this minimum.

6.5 Simulation study

In the following simulation study, convergence of infimal loops is assumed when all elements of c satisfy $| c_i^{k+1} - c_i^{k} | < 5 \times 10^{-5}$. The supremal problem is solved using either the subroutine MIFSIM (Appendix D) or the subroutine MIFNOD (Appendix E). The actual values of the arguments used in these two subroutines are listed in TABLE 6.3 and TABLE 6.4.

First consider a simple example with nondifferentiable inequality constraints with respect to y.

Simple example 3

Model and reality are:

$$y_1 = 2c_1 + y_2 + \alpha_1$$

$$y_2 = c_2 + 2y_1 + \alpha_2,$$

$$y_1^* = 2c_1 + y_2^* + 0.5c_1^2$$

$$y_2^* = c_2 + 2y_1^* + 0.5c_2^2.$$

Constraints and performace index are:

$$CY \overset{\Delta}{=} \{(c,y) \in R^4 : |c_i| \leq 1,\ i=1,2,\ y_1 \geq -1,\ y_2 \geq 0,$$

$$c_1 + 2y_1 + y_2 \leq 1 \text{ and } |c_2| + |y_1 + y_2| \leq 1\},$$

$$Q(c,y) = 0.5\{c_1^2 + c_2^2 + (y_1 - 1)^2 + (y_1 - 2)^2 + (y_2 - 1)^2 + (y_2 - 2)^2\}.$$

Using the simplex technique as the supremal level algorithm with $\lambda_1^0 = \lambda_2^0 = 0$, the two-level type algorithm successfully obtained the real optimum $Q = 2.760897$, $c_1 = -0.191094$ and $c_2 = 0.084627$ after a total of 8 iterations at the infimal level. The final modifiers were $\lambda_1 = \lambda_2 = 0.3$.

Examples 1, 2 and 3 given in Section 4.4 were used again as the testing examples. Simulation results are shown in TABLE 6.1. As expected, the two-level type algorithm is less efficient compared with the modified two-step algorithm. Unlike the latter which achieves a unique final modifier vector, the supremal solution can be non-unique. If any one of the supremal solutions is found by the supremal level algorithm, the procedure will terminate at the real system optimum condition. TABLE 6.2 gives some illustration on this point. On the other hand, if A is not convex, starting from an arbitrary initial point the supremal level algorithm may not be able to attain a supremal

solution (see results of Example 2).

6.6 Summary

An alternative modification of the two-step method has been examined in this chapter. This two-level type method does not require real process derivatives in searching for the real optimum operating condition, which is an important advantage over the modified two-step method. The benefit is obtained at the cost of introducing a second level. Every time the supremal level prescribes a modifier vector, a two-step like iterative loop is performed at the infimal level. This two-loop feature is inefficient from the computational point of view.

At present properties of the supremal function $\bar{q}(\lambda)$ and the supremal feasible set $\mathcal{A}$ are not very clear. More theoretical analysis is needed in order to understand the method better. Further work is required to develop some more powerful algorithms for solving the supremal level task.

TABLE 6.1 Simulation results of two-level type algorithm

example	supremal algorithm	initial λ	infimal iterations (set point changes)	real performance	real optimum
one	Powell	0	403	6.3266	6.3266
two	Powell	0	425	2.1452	2.1405
	Powell	*	325	2.1405	
three	Powell	0	268	5.9261	5.9261
	Simplex	0	410	5.9261	

* $\lambda^0 = [0.6\ 0.0\ 0.0\ -0.6\ 0.2\ 0.2]^T$.

TABLE 6.2 Final modifiers of Example 3

	λ_1	λ_2	λ_3	λ_4	λ_5
MTS (new version)	-0.5163	-0.3037	0.4076	0.5805	0.7113
TL (Powell)	-0.7361	0.2619	0.1833	0.7500	0.7198
TL (Simplex)	-0.4210	-0.5613	0.5399	-0.0461	0.6971

TABLE 6.3 Values of arguments used in MIFSIM

	TOL	STEP	ALPHA	BETA	GAMMA
simple example 3	10^{-4}	0.2	1.0	0.5	2.0
example 3	10^{-4}	0.5	1.0	0.5	2.0

TABLE 6.4 Values of arguments used in MIFNOD

example	ALPHA	EPSIL	BETA	SK	A1	A2
one	10^{-4}	0.5	0.01	1.0	1.5	-0.25
two	10^{-4}	0.5	0.01	1.0	1.5	-0.25
three	10^{-4}	0.5	0.01	1.0	1.5	-0.25

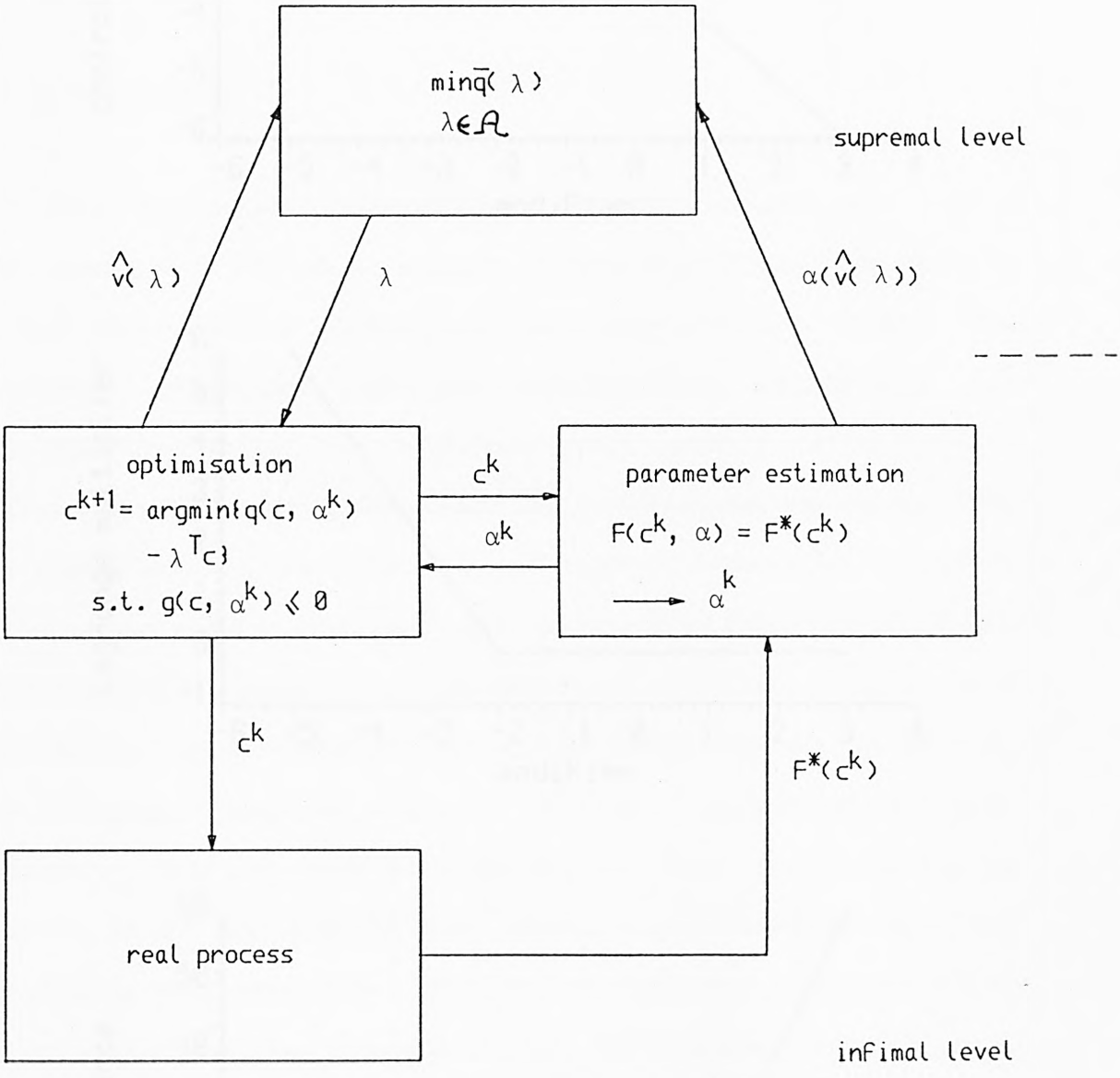


Fig. 6.1. Two-level type algorithm

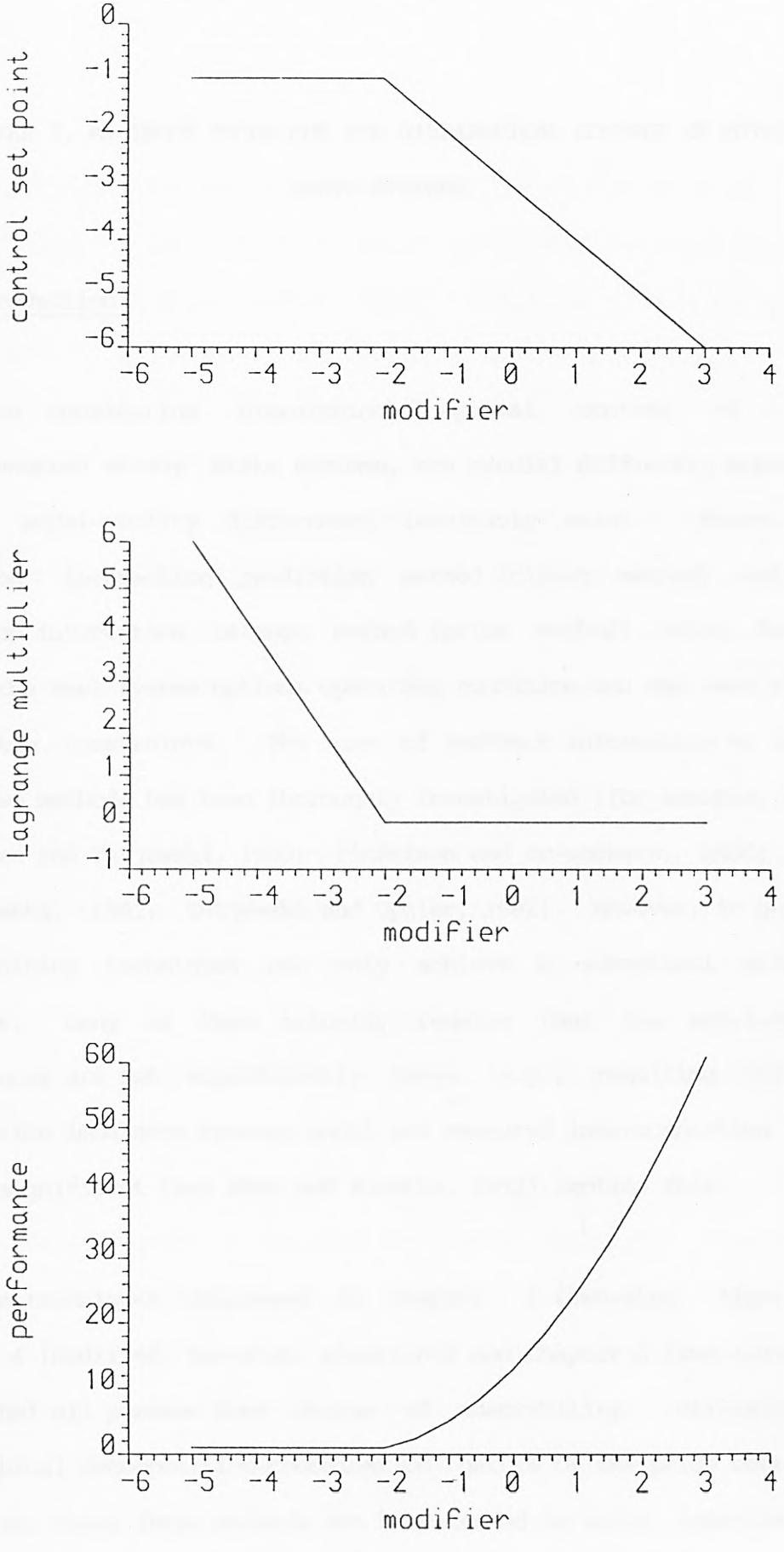


Fig. 6.2. Infimal solutions of Simple example 1

CHAPTER 7. AN ISOPE TECHNIQUE FOR HIERARCHICAL CONTROL OF STEADY STATE SYSTEMS

7.1 Introduction

When considering hierarchical optimal control of large interconnected steady state systems, the crucial difficulty appears to be that model-reality differences inevitably exist. Hence, the open-loop interaction prediction method (direct method) and the open-loop interaction balance method (price method) often fail to attain the real system optimum operating condition and may even violate the system constraints. The use of feedback information to improve these two methods has been thoroughly investigated (for example, Brdyś, Findeisen and Tatjewski, 1980; Findeisen and co-workers, 1980; Shao and Roberts, 1983; Tatjewski and Cygler, 1981). However, in general, the resulting techniques can only achieve a suboptimal solution. Moreover, many of them actually require that the model-reality differences are not significantly large, e.g., requiring that the interaction imbalance between model and measured interconnection inputs is not significant (see Shao and Roberts, 1983) implies this.

The techniques discussed in Chapter 3 (two-step algorithm), Chapter 4 (modified two-step algorithm) and Chapter 6 (two-level type algorithm) all possess some degree of adaptability. Utilizing the hierarchical decomposition-coordination nature of the price correction mechanism, these three methods can be extended to solve interconnected large-scale system problems. A method presented by Brdyś (1983) is an example of combining the two-step and price techniques. Although it

has better adaptability than those techniques mentioned above, it is a suboptimal method because of the suboptimality of the two-step method. In an attempt to incorporate the price and modified two-step mechanisms the joint coordination method (JCM) (Michalska, Ellis and Roberts, 1985) has been introduced. However, because the old version of the modified two-step method was employed, JCM can only achieve the steady state optimal condition when the system inequality constraints do not depend on the system outputs. Brdys and Roberts (1984a) have shown how a group of optimal adaptive structures can be formulated by integrating the modified two-step method or the two-level type method with the price correction mechanism. In these structures, the measurement feedback is used in different manners.

The weakness of the structures obtained by combining the two-level type method and the price method is primarily that they are very inefficient and may take an unacceptable long time to locate an optimal solution. This is because of their three level (loop) structure and because of lacking effective algorithms for solving the supremal task as discussed in Chapter 6.

In this chapter a hierarchical ISOPE technique obtained by combining the modified two-step method with the price method is studied. The technique requires the real system interaction output and input information feedback and updates modifiers λ and prices p at each iteration. As shown in Chapter 4, if (4.29) is employed to approximate the real output derivatives, each time λ is updated the set of real control inputs needs to be perturbed n times (where n is the total number of the control inputs) and the computational requirement is

dominated by on-line waiting time and information exchange among the coordinator, local control units and real subsystems. Therefore, advantage would be gained if it is possible to design an alternative method which updates λ less frequently. In this chapter it will be shown that this aim can be achieved by employing a double iterative mechanism. The basic idea is similar to that employed by Shao and Roberts (1983) where an interior iterative procedure was introduced to achieve model interaction balance while shift variables computed using real system feedback information were updated at less frequent intervals in an outer loop.

7.2 Description of the system

It is assumed that external disturbances which affect the system are varying slowly and can be considered as constant in the considered time interval of determining optimum control and, hence, the system is described by a set of subsystem input-output mappings

$$F_i^*: \mathcal{C}_i \times \mathcal{U}_i \rightarrow \mathcal{Y}_i, \quad i=1,\ldots,N,$$

as follows

$$y_i = F_i^*(c_i, u_i), \quad i=1,\ldots,N,$$

where N denotes the number of subsystems, $\mathcal{C}_i$, $\mathcal{U}_i$ and $\mathcal{Y}_i$ are finite dimensional spaces, c_i, u_i and y_i are the ith subsystem's control, interaction input and interaction output vectors, respectively. The subsystems are interconnected through the coupling equations

$$u_i = H_i y = \sum_{j=1}^{N} H_{ij} y_j, \quad i=1,\ldots,N,$$

where H_i and H_{ij} are interconnection matrices. The local constraint

set takes the form

$$(c_i, u_i, y_i) \in CUY_i \triangleq \{(c_i, u_i, y_i) \in \mathcal{C}_i \times \mathcal{U}_i \times \mathcal{Y}_i : G_i(c_i, u_i, y_i) \leq 0\}, \quad i=1,\ldots,N,$$

where G_i is the ith subsystem constraint function vector. The system as a whole can be written as

$$y = F^*(c, u), \tag{7.1}$$

$$u = Hy, \tag{7.2}$$

$$(c, u, y) \in CUY \triangleq \{(c, u, y) \in \mathcal{C} \times \mathcal{U} \times \mathcal{Y} : G(c, u, y) \leq 0\}, \tag{7.3}$$

where $c \triangleq (c_1^T, \ldots, c_N^T)^T \in \mathcal{C}_1 \times \ldots \times \mathcal{C}_N = \mathcal{C}$,

$$u \triangleq (u_1^T, \ldots, u_N^T)^T \in \mathcal{U}_1 \times \ldots \times \mathcal{U}_N = \mathcal{U},$$

$$y \triangleq (y_1^T, \ldots, y_N^T)^T \in \mathcal{Y}_1 \times \ldots \times \mathcal{Y}_N = \mathcal{Y},$$

$$H \triangleq (H_{ij}), \quad i,j=1,\ldots N,$$

$$G(c,u,y) \triangleq ((G_1(c_1,u_1,y_1))^T, \ldots, (G_N(c_N,u_N,y_N))^T)^T.$$

Next, we require that for each $c \in \mathcal{C}$, the equation $y = F^*(c, Hy)$ has a unique solution

$$y = K^*(c), \tag{7.4}$$

where $K^* : \mathcal{C} \to \mathcal{Y}$ and $K^*(c) \triangleq ((K_1^*(c))^T, \ldots, (K_N^*(c))^T)^T$.

In general the above real system relations are not known exactly and, consequently, we have only their approximate models

$$F_i : \mathcal{C}_i \times \mathcal{U}_i \times \bar{\mathcal{A}}_i \to \mathcal{Y}_i,$$

$$y_i = F_i(c_i, u_i, \alpha_i), \quad i=1,\ldots,N,$$

where $\bar{\mathcal{A}}_i$ is a finite dimensional space and α_i is the ith subsystem model parameter vector. As before, the global model equations can be

written as

$$y = F(c,\ u,\ \alpha), \tag{7.5}$$

where $\alpha \in \bar{\mathcal{A}}$ and $\bar{\mathcal{A}} \triangleq \bar{\mathcal{A}}_1 x \ldots x \bar{\mathcal{A}}_N$. The interconnection matrix and the local constraint set are assumed to be known exactly.

Finally, a known local performance index, associated with each subsystem

$$Q_i : \mathcal{C}_i x \mathcal{U}_i x \mathcal{Y}_i \to R,\ i=1,\ldots,N,$$

is required to be minimized. The overall performance function

$$Q : \mathcal{C} x \mathcal{U} x \mathcal{Y} \to R,$$

is assumed to be

$$Q(c,u,y) = \sum_{i=1}^{N} Q_i(c_i,\ u_i,\ y_i). \tag{7.6}$$

7.3 Formulation of the algorithms

7.3.1 Formulation of the optimal control problem

The task of determining the optimum controller set points c for a real system can be defined as the following real steady state optimizing control problem

$$\begin{aligned} &\min_{c,u}\ Q(c,\ u,\ y), \\ (\text{ROCP})\quad &\text{s.t. } y = K^{*}(c), \\ &\qquad u = H\,y, \\ &\qquad G(c,\ u,\ y) \lesssim 0. \end{aligned} \tag{7.7}$$

An equivalent optimal control problem of (ROCP) (see Brdys' and Roberts, 1984a) is

$$\min_{c,v,u,\alpha} q(c, u, \alpha),$$

$$\text{(EOCP)} \quad \text{s.t. } F(v, HK^*(v), \alpha) = K^*(v),$$

$$u = HF(c, u, \alpha), \tag{7.8}$$

$$g(c, u, \alpha) \leq 0,$$

$$v = c,$$

where $q(c,u, \alpha) = Q(c,u, F(c,u, \alpha))$ and $g(c,u, \alpha) = G(c,u, F(c,u, \alpha))$. The Lagrangian associated with (EOCP) can then be written as

$$L(c,u,v, \alpha,p, \lambda, \eta, \xi) = q(c,u, \alpha) + p^T[u-HF(c,u, \alpha)] + \lambda^T(v-c) \tag{7.9}$$

$$+ \eta^T[F(v,HK^*(v), \alpha)-K^*(v)] + \xi^T g(c,u, \alpha),$$

where p, λ, η and ξ are Lagrange multipliers. In particular, p is known as the price vector and λ is known as the modifier vector. Assuming that all required derivatives exist and regularity conditions are satisfied then the Kuhn-Tucker necessary optimality conditions (see, for example, Luenberger, 1973) of (EOCP) are:

$$\nabla_c L = \frac{\partial^T q(c,u, \alpha)}{\partial c} - \frac{\partial^T F(c,u, \alpha)}{\partial c} H^T p - \lambda + \frac{\partial^T g(c,u, \alpha)}{\partial c} \xi = 0, \tag{7.10}$$

$$\nabla_u L = \frac{\partial^T q(c,u, \alpha)}{\partial u} + \frac{\partial^T [u-HF(c,u,\alpha)]}{\partial u} p + \frac{\partial^T g(c,u,\alpha)}{\partial u} \xi = 0, \tag{7.11}$$

$$\nabla_v L = \lambda + \left[\frac{\partial^T F(v,HK^*(v), \alpha)}{\partial v} - \frac{\partial^T K^*(v)}{\partial v}\right] \eta = 0, \tag{7.12}$$

$$\nabla_\alpha L = \frac{\partial^T q(c,u,\alpha)}{\partial \alpha} - \frac{\partial^T F(c,u,\alpha)}{\partial \alpha} H^T p + \frac{\partial^T F(v,HK^*(v),\alpha)}{\partial \alpha} \eta +$$

$$\frac{\partial^T g(c,u,\alpha)}{\partial \alpha}\,\xi = 0, \tag{7.13}$$

$$\nabla_p L = u - HF(c,u,\alpha) = 0, \tag{7.14}$$

$$\nabla_\lambda L = v - c = 0, \tag{7.15}$$

$$\nabla_\eta L = F(v,HK^*(v),\alpha) - K^*(v) = 0, \tag{7.16}$$

$$\nabla_\xi L = g(c,u,\alpha) \leqslant 0, \tag{7.17}$$

$$\xi^T \nabla_\xi L = \xi^T g(c,u,\alpha) = 0, \quad \xi \geqslant 0. \tag{7.18}$$

Let us consider how to obtain a solution of (7.10) to (7.18). Assuming that the inverse $\left[\frac{\partial^T F(v,HK^*(v),\alpha)}{\partial \alpha}\right]^{-1}$ exists, η can be derived from (7.13) and substituted into (7.12) to yield the modifiers

$$\lambda = \lambda(c,u,v,\alpha,p,\xi) = \left[\frac{\partial^T F(v,HK^*(v),\alpha)}{\partial v} - \frac{\partial^T K^*(v)}{\partial v}\right]\left[\frac{\partial^T F(v,HK^*(v),\alpha)}{\partial \alpha}\right]^{-1}$$

$$\cdot\left[\frac{\partial^T q(c,u,\alpha)}{\partial \alpha} + \frac{\partial^T g(c,u,\alpha)}{\partial \alpha}\,\xi - \frac{\partial^T F(c,u,\alpha)}{\partial \alpha} H^T p\right]. \tag{7.19}$$

Because of $v = c$ and $u = HK^*(v)$ for any solution of (7.10) to (7.18), (7.19) can also be written as

$$\lambda = \lambda(v,HK^*(v),v,\alpha,p,\xi) = \left[\frac{\partial^T F(v,HK^*(v),\alpha)}{\partial v} - \frac{\partial^T K^*(v)}{\partial v}\right]\left[\frac{\partial^T F(v,HK^*(v),\alpha)}{\partial \alpha}\right]^{-1}$$

$$\cdot\left[\frac{\partial^T q(v,HK^*(v),\alpha)}{\partial \alpha} + \frac{\partial^T g(v,HK^*(v),\alpha)}{\partial \alpha}\,\xi - \frac{\partial^T F(v,HK^*(v),\alpha)}{\partial \alpha} H^T p\right]. \tag{7.20}$$

Secondly, for given α, p and λ equations (7.10), (7.11), (7.17) and (7.18) can be satisfied by solving the modified optimisation problem

$$\min_{c,u} \{q(c,u,\alpha) - \lambda^T c + p^T[u - HF(c,u,\alpha)]\} ,$$
$$\text{s.t. } g(c,u,\alpha) \leqslant 0 . \tag{7.21}$$

Finally let us denote the solution of (7.16) $\alpha(v)$ for a given v, calculate λ by formula (7.20) with given v, $\alpha(v)$, p and ξ, and denote the solution of (7.21) and corresponding Lagrange multiplier $\hat{c}(v, \alpha(v), p, \xi)$, $\hat{u}(v, \alpha(v), p, \xi)$ and $\hat{\xi}(v, \alpha(v), p, \xi)$, respectively. Then any solution of the following equations

$$\hat{c}(v, \alpha(v), p, \xi) = v , \tag{7.22}$$

$$\hat{u}(v, \alpha(v), p, \xi) = HF(\hat{c}, \hat{u}, \alpha(v)), \tag{7.23}$$

$$\hat{\xi}(v, \alpha(v), p, \xi) = \xi , \tag{7.24}$$

is a solution of equations (7.10) to (7.18). An iterative algorithm for solving (7.22), (7.23) and (7.24) is formulated in the next subsetion.

7.3.2 A single iterative algorithm (SIA)

Computing $\alpha(v)$ from equation (7.16) involves the operation of matching the real system output and the model output, and hence is viewed as parameter estimation. Notice that (7.16) can be decomposed into N independent parts, one for each subsystem,

$$F_i(v_i, H_i K_i^*(v), \alpha_i) = K_i^*(v), \quad i=1,\ldots,N. \tag{7.25}$$

On the other hand, when α, p and λ are given, (7.21) is purely a model-based optimisation problem and is also separable. It can be

rewritten as

$$\min \sum_{i=1}^{N} L_i(c_i, u_i, \alpha_i, \lambda_i, p), \tag{7.26}$$

$$\text{s.t. } g_i(c_i, u_i, \alpha_i) \leq 0, \quad i=1,\ldots N,$$

$$\text{where } L_i(c_i, u_i, \alpha_i, \lambda_i, p) = q_i(c_i, u_i, \alpha_i) - \lambda_i^T c_i + p_i^T u_i - \sum_{j=1}^{N} p_j^T H_{ji} F_i(c_i, u_i, \alpha_i). \tag{7.27}$$

A single iterative algorithm can readily be designed. The ith local control problem which consists of the ith local parameter estimation and the ith local optimisation problems is defined:

LCP_i, $i=1,\ldots,N$:

(i) For given v_i and the measurements $K_i^*(v)$ and $H_iK^*(v)$ find the model parameters α_i which satisfy

$$F_i(v_i, H_iK^*(v), \alpha_i) - K_i^*(v) = 0,$$

(ii) For given α_i, p and λ_i find the controller set points $\hat{c}_i$ and interconnection inputs $\hat{u}_i$ such that

$$(\hat{c}_i, \hat{u}_i) = \arg\min_{c_i, u_i} L_i(c_i, u_i, \alpha_i, \lambda_i, p),$$

$$\text{s.t. } g_i(c_i, u_i, \alpha_i) \leq 0 .$$

The task of the coordinator is to ensure that the solution which satisfies (7.22), (7.23) and (7.24) is obtained in an efficient manner. This task can be stated as:

CP: Find p^* and λ^* such that

the conditions (7.12) to (7.15) are satisfied.

The overall problem is suitable for employing an iterative procedure where v and ξ are improved based on $\hat{c}$ and $\hat{\xi}$ given by the local units, p

is improved based on the interaction imbalance $\Delta\hat{u} = \hat{u} - HF(\hat{c},\hat{u}, \alpha)$ and λ is updated by formula (7.19) (or (7.20)).

In order to regulate convergence and stability properties, the iterative strategies for solving equations (7.22), (7.23) and (7.24) are chosen as

$$v_i^{k+1} = \Psi_{v_i}(\hat{c}_i^k, v_i^k) = v_i^k + K_{v_i}(\hat{c}_i^k - v_i^k), \quad i=1,\ldots,N, \tag{7.28}$$

$$\xi_i^{k+1} = \Psi_{\xi_i}(\hat{\xi}_i^k, \xi_i^k) = \xi_i^k + K_{\xi_i}(\hat{\xi}_i^k - \xi_i^k), \quad i=1,\ldots,N, \tag{7.29}$$

$$p^{k+1} = \Psi_p(p^k, \Delta\hat{u}^k) = p^k + K_p(\hat{u}^k - HF(\hat{c}^k, \hat{u}^k, \alpha^k)), \tag{7.30}$$

where K_{v_i}, K_{ξ_i} and K_p are diagonal gain matrices with diagonal elements usually positive and less than or equal to unity. Then the kth iteration is described as follows:

(a) The local control units update v_i, i=1,...,N, then apply v_i^k to the real subsystems and obtain the corresponding steady-state measurements of local outputs $K_i^*(v^k)$ and interconnection inputs $H_iK^*(v^k)$. Determine α_i^k by solving

$$F_i(v_i^k, H_iK^*(v^k), \alpha_i) = K_i^*(v^k), \quad i=1,\ldots,N.$$

Perform additional perturbations about v_i^k and take the corresponding real measurements. The values of α_i^k, v_i^k and all real measurements are sent to the coordinator for computing finite difference approximations of the derivatives

$$\frac{\partial^T F(v^k, HK^*(v^k), \alpha^k)}{\partial v} - \frac{\partial^T K^*(v^k)}{\partial v}$$

(b) The coordinator computes ξ^k and p^k, calculates modifiers according to (7.19)

$$\lambda^k = \lambda\,(\hat{c}^{k-1},\ \hat{u}^{k-1},\ v^k,\ \alpha^k,\ p^k,\ \xi^k),$$

and sends λ_i^k and p^k to the ith local unit. The modifiers may also be computed according to (7.20) similar to the centralized case discussed in Chapter 4.

(c) The local units perform their own optimisation task

$$(\hat{c}_i^k,\ \hat{u}_i^k) = \arg\min_{c_i,u_i} L_i(c_i,\ u_i,\ \alpha_i^k,\ \lambda_i^k,\ p^k)$$

$$\text{s.t. } g_i(c_i,\ u_i,\ \alpha_i^k) \leq 0,\ i=1,\ldots,N.$$

The values of $\hat{c}_i^k$, $\hat{u}_i^k$ and associated Lagrange multiplier $\hat{\xi}_i^k$ are then sent to the coordinator.

It is noted that the requirement to obtain real process derivatives produces a serious practical consideration particularly when the measurements are contaminated by noise. For this reason it is often advisable to apply simple software filtering to the computation of the modifiers as shown in Chapter 4. Fig. 7.1. shows the information structure of the algorithm. The overall convergence is achieved when successive solutions of v and ξ are unchanged and interaction balance is satisfied. In practice, the overall process may be terminated when every elements of v, $\hat{u}$ and ξ satisfy

$$|\ v_{ij}^k - \hat{c}_{ij}^k\ | < \beta_v, \tag{7.31}$$

$$|\ \hat{\xi}_{il}^k - \xi_{il}^k\ | < \beta_\xi, \tag{7.32}$$

$$|\ \hat{u}_{is}^k - (H_iF(\hat{c}^k,\ \hat{u}^k,\ \alpha^k))_s\ | < \beta_p, \tag{7.33}$$

where β_v, β_ξ and β_p are some desired tolerances. $(H_iF)_s$ denotes the sth element of H_iF.

A special situation occurs when the system constraints do not depend on the outputs. In this case $\frac{\partial^T g(c, u, \alpha)}{\partial \alpha} \xi = 0$, and the iterative strategy (7.29) and the criterion (7.32) are not needed. In the following discussion we will assume that (4.29) is used to approximate the real output derivatives. Therefore, the number of set point changes in step (a) (i.e., at each iteration) is n, where n is the total number of the controller set points.

7.3.3 A double iterative algorithm (DIA)

Let us examine the local control problem LCP_i and coordination problem CP in more detail. Notice that both problems are composed of two parts, one requiring real system information and the other not, and hence, the problem is very suitable for restructuring into a double iterative mechanism such as that employed by Shao and Roberts (1983). The key step is separating the function of improving the modifier λ and local parameter estimation from that of improving the price p and local optimisation. The latter gives rise to an interior procedure during which information is interchanged only between local optimisation units and the coordinator, while the former, which requires the real system information feedback, is solved at less frequent intervals in an outer iterative loop. The overall iterative procedure is described as follows:

With α and λ fixed, the interior procedure is the entirely model-based problem

$$\text{ILOP}_i: \quad i=1,\ldots,N \begin{cases} \text{Find } \hat{c}_i \text{ and } \hat{u}_i \text{ such that} \\ (\hat{c}_i, \hat{u}_i) = \arg\min_{c_i, u_i} L_i(c_i, u_i, \alpha_i, \lambda_i, p), \\ \text{s.t. } g_i(c_i, u_i, \alpha_i) \leq 0, \end{cases}$$

$$\text{ICP}: \begin{cases} \text{Find } \hat{p} \text{ such that} \\ \hat{u} - HF(\hat{c}, \hat{u}, \alpha) = 0, \end{cases}$$

and it can readily be solved iteratively using an open-loop interaction balance method. When the interaction balance has been achieved, α and λ are adjusted by the outer iterative loop and then the interior procedure restarts with new α and λ. The outer loop iterates α and λ similarly to the single iterative algorithm as described in steps (a) and (b) except that $\hat{c}^{k-1}, \hat{u}^{k-1}$ and p^k are the solution and corresponding price vector of the interior procedure. The overall process is terminated when v and ξ determined in the outer procedure remain sufficiently unchanged between successive iterations.

The information structure of the mechanism is shown in Fig. 7.2., where it is observed that only when α and λ are adjusted within the outer iterative procedure that the real system information is required. Bearing in mind that calculating α and λ requires perturbing the controller set points n times, it is expected that this mechanism will reduce on-line computing time significantly.

7.4 Information interchange requirement

Two types of information interchange which occur during the

iterative procedure are defined. The first type occurs between the local control units and the coordinator after each model-based optimisation iteration, which we will denote as off-line information interchange. The second type occurs between the real subsystems and the coordinator (through the local control units) after each time that the controller set points have been changed, which we will refer to as on-line information interchange. It is assumed that the total number of information exchanges is roughly equal to the sum of the off-line and on-line information interchanges.

Because the process dynamic characteristic is often slow compared with the computing time required for model optimisation and because of the necessity to approximate the process output derivatives, the time required to locate an optimal solution by the single iterative algorithm is dominated by the time spent on on-line information interchange. The aim of the double iterative algorithm is to reduce on-line information interchange and hence, to reduce the time required for determining the optimal steady state control, even at the cost of increasing off-line information exchange.

7.5 Simulation study

7.5.1 Simulation examples

Four examples have been used in computer simulation studies.

Example 4. The model and reality equations are:

$$y_{11} = c_{11} - c_{12} + 2u_{11} + \alpha_{11}$$

$$y_{21} = c_{21} - c_{22} + u_{21} - 3u_{22} + \alpha_{21}$$

$$y_{22} = 2c_{22} - c_{23} - u_{21} + u_{22} + \alpha_{22}$$

$$y_{31} = c_{31} + 2.5c_{32} - 4u_{31} + \alpha_{31},$$

$$y_{11}^{*} = 1.3c_{11} - c_{12} + 2u_{11} + 0.15u_{11}c_{11}$$

$$y_{21}^{*} = c_{21} - c_{22} + 1.2u_{21} - 3u_{22} + 0.1c_{22}^{2}$$

$$y_{22}^{*} = 2c_{22} - 1.25c_{23} - u_{21} + u_{22} + 0.25c_{22}c_{23} + 0.1$$

$$y_{31}^{*} = 0.8c_{31} + 2.5c_{32} - 4.2u_{31}.$$

The subsystem performance indices are:

$$Q_1(c_1, u_1) = (u_{11} - 1)^4 + 5(c_{11} + c_{12} - 1)^2,$$

$$Q_2(c_2, u_2) = 4u_{21}^2 + u_{22}^2 + 2(c_{21} - 2)^2 + c_{22}^2 + 3c_{23}^2,$$

$$Q_3(c_3, u_3) = (u_{31} - 1)^2 + (c_{31} + 1)^2 + 2.5c_{32}^2,$$

where $c_1 = (c_{11}, c_{12})^T$, $c_2 = (c_{21}, c_{22}, c_{23})^T$, $c_3 = (c_{31}, c_{32})^T$,

$u_1 = u_{11}$, $u_2 = (u_{21}, u_{22})^T$, $u_3 = u_{31}$,

$y_1 = y_{11}$, $y_2 = (y_{21}, y_{22})^T$, $y_3 = y_{31}$.

The system constraints and the coupling equation are:

$$CUY_1 \triangleq \{(c_1, u_1, y_1) \in R^4 : c_{11}^2 + c_{12}^2 \leq 1 \text{ and } 0 \leq u_{11} \leq 0.5\},$$

$$CUY_2 \triangleq \{(c_2, u_2, y_2) \in R^7 : 0.5c_{21} + c_{22} + 2c_{23} \leq 1 \text{ and } 4c_{21}^2 + 2c_{21}u_{21} + 0.4u_{21} + c_{21}c_{23} + 0.5c_{23}^2 + u_{21}^2 \leq 4\},$$

$$CUY_3 \triangleq \{(c_3, u_3, y_3) \in R^4 : c_{31} + u_{31} + 0.5 \geq 0 \text{ and } 0 \leq c_{32} \leq 1\},$$

$$\begin{pmatrix} u_{11} \\ u_{21} \\ u_{22} \\ u_{31} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} y_{11} \\ y_{21} \\ y_{22} \\ y_{31} \end{pmatrix}.$$

Example 5. The model and reality equations are:

$$y_{11} = 1.4375c_{11} - 0.1875c_{12} + 1.5u_{11} + \alpha_{11}$$

$$y_{21} = 0.5c_{21} - 1.5c_{22} + u_{21} - 2u_{22} + \alpha_{21}$$

$$y_{22} = 2.5c_{22} - 0.5c_{23} - u_{21} + 1.5u_{22} + \alpha_{22}$$

$$y_{31} = 1.25c_{31} - 3u_{31} + \alpha_{31},$$

$$y_{11}^{*} = c_{11} - c_{12} + 2u_{11} - 0.5c_{11}^{2} + 0.5(c_{11} + c_{12} - 2)u_{11}$$

$$y_{21}^{*} = c_{21} - c_{22} + u_{21} - 3u_{22}$$

$$y_{22}^{*} = 2c_{22} - c_{23} - u_{21} + u_{22}$$

$$y_{31}^{*} = c_{31} - 4u_{31} + 0.5c_{31}u_{31}.$$

The subsystem performance indices are:

$$Q_1(c_1, u_1) = (u_{11} - 1)^2 + c_{11}^2 + (c_{12} - 2)^2,$$

$$Q_2(c_2, u_2) = 4u_{21}^2 + u_{22}^2 + 2(c_{21} - 2)^2 + c_{22}^2 + 3c_{23}^2,$$

$$Q_3(c_3, u_3) = (u_{31} - 1)^2 + (c_{31} + 1)^2.$$

The system constraints are:

$$CUY_1 \stackrel{\Delta}{=} \{(c_1, u_1, y_1) \in R^4 : 1.006 - c_{11} - u_{11} \geq 0\},$$

$$CUY_2 \stackrel{\Delta}{=} \{(c_2, u_2, y_2) \in R^7\},$$

$$CUY_3 \stackrel{\Delta}{=} \{(c_3, u_3, y_3) \in R^3 : 0.375 + 2.25c_{31} - 2.75u_{31} - y_{31} \geq 0\},$$

and the coupling equation is the same as in Example 4.

Example 6. The model and reality equations and performance indices are:

$$y_{11} = c_{11} - c_{12} + 2u_{11} + \alpha_{11}$$

$$y_{21} = c_{21} - c_{22} + u_{21} + \alpha_{21}$$

$$y_{22} = 2c_{22} - c_{23} - u_{21} + \alpha_{22},$$

$$y_{11}^{*} = 1.4c_{11} - 0.6c_{12} + 1.8u_{11}$$
$$y_{21}^{*} = 1.3c_{21} - 1.1c_{22} + 1.1u_{21}$$
$$y_{22}^{*} = 2.3c_{22} - 0.7c_{23} - 1.1u_{21},$$

$$Q_1(c_1, y_1) = (y_{11} - 1)^2 + c_{11}^2 + c_{12}^2,$$
$$Q_2(c_2, y_2) = 2(y_{21} - 2)^2 + (y_{22} - 3)^2 + c_{21}^2 + c_{22}^2 + c_{23}^2.$$

The system constraints and the coupling equation are:

$$CUY_1 \triangleq \{(c_1, u_1, y_1) \in R^4: |c_{11}| \leq 1, |c_{12}| \leq 1, y_{11} \geq 0 \text{ and } 0.8 - c_{12} - 0.6u_{11} \geq 0\},$$
$$CUY_2 \triangleq \{(c_2, u_2, y_2) \in R^6: (|c_{2i}| \leq 1, i=1,2,3), y_{21} \geq 0, y_{22} \geq 0 \text{ and } 2.04 + 1.05u_{21} - c_{21}^2 - c_{22}^2 - c_{23}^2 \geq 0\},$$

$$\begin{pmatrix} u_{11} \\ u_{21} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} y_{11} \\ y_{21} \\ y_{22} \end{pmatrix}.$$

Example 7. This is a five subsystem example (Fig. 7.3.).

Subsystem 1. Model and reality are:

$$y_{11} = c_{11} - c_{12} + c_{13} - 0.45c_{14}^2 + 2.1u_{11} + \alpha_{11}$$
$$y_{12} = c_{11} + c_{12} + 1.1c_{13}c_{14} - 0.65u_{11} + \alpha_{12},$$

$$y_{11}^{*} = c_{11} - c_{12} + c_{13} - 0.5c_{14}^2 + 2u_{11}$$
$$y_{12}^{*} = c_{11} + c_{12} + c_{13}c_{14} - (1 + c_{13})u_{11}.$$

Performance index and constraint set are:

$$Q_1(c_1, u_1) = (u_{11} - 1)^2 + \sum_{i=1}^{4} c_{1i}^2,$$

$$CUY_1 \triangleq \{(c_1, u_1, y_1) \in R^7: (|c_{1i}| \leq 1, i=1,2,3,4), u_{11} \geq 0 \text{ and } c_{11} + 1.5c_{12} + c_{13} + u_{11} \leq 1\}.$$

Subsystem 2. Model and reality are:

$$y_{21} = 1.15c_{21} - c_{22} + u_{21} - 2.75u_{22} + \alpha_{21},$$

$$y_{21}^{*} = c_{21} - c_{22} + u_{21} - 3u_{22}.$$

Performance index and constraint set are:

$$Q_2(c_2, u_2) = 2u_{21}^2 + (u_{22} - 1)^2 + 2(c_{21} - 2)^2 + c_{22}^2,$$

$$CUY_2 \overset{\Delta}{=} \{(c_2, u_2, y_2) \in R^5 : (|c_{2i}| \leq 1,\ u_{2i} \geq 0,\ i=1,2) \text{ and}$$

$$c_{21}^2 + c_{22}^2 \leq u_{21} + 3u_{22}\}.$$

Subsystem 3. Model and reality are:

$$y_{31} = c_{31} + 2.5c_{32} - c_{33} + c_{34} + 1.93u_{31} + \alpha_{31}$$

$$y_{32} = c_{33} + c_{34} - 1.1c_{31}^2 + 0.9u_{32} + \alpha_{32}$$

$$y_{33} = 1.15c_{32}c_{33} + 0.9c_{35}^2 - u_{31} + u_{32} + \alpha_{33},$$

$$y_{31}^{*} = c_{31} + 2.5c_{32} - c_{33} + c_{34} + (2 - c_{35})u_{31}$$

$$y_{32}^{*} = c_{33} + c_{34} - c_{31}^2 + u_{32}$$

$$y_{33}^{*} = c_{32}c_{33} + c_{35}^2 - u_{31} + u_{32}.$$

Performance index and constraint set are:

$$Q_3(c_3, u_3) = (u_{31} - 1)^2 + (u_{32} - 1)^2 + c_{31}^2 + 2c_{32}^2 + c_{33}^2 + 0.5c_{34}^2 + c_{35}^2,$$

$$CUY_3 \overset{\Delta}{=} \{(c_3, u_3, y_3) \in R^{10} : (|c_{3i}| \leq 1,\ i=1,\ldots,5),\ (u_{3i} \geq 0,\ i=1,2) \text{ and}$$

$$c_{33} + 0.5c_{35} + u_{32} \leq 2\}.$$

Subsystem 4. Model and reality are:

$$y_{41} = c_{41} + 0.5c_{42} + c_{43} + 2.5u_{41} + 0.894u_{42} - 2.417u_{43} + \alpha_{41},$$

$$y_{41}^{*} = c_{41} + 0.5c_{42} + c_{43} + 2.5u_{41} + (1 - c_{41})u_{42} - (3 - c_{43})u_{43}.$$

Performance index and constraint set are:

$$Q_4(c_4, u_4) = (u_{41} - 1)^2 + (u_{42} - 2)^2 + (u_{43} - 1)^2 + (c_{41} + c_{42} + 1)^2 + (c_{43} - 1)^2$$

$$CUY_4 \overset{\Delta}{=} \{(c_4, u_4, y_4) \in R^7 : (|c_{4i}| \leq 1,\ u_{4i} \geq 0,\ i=1,2,3) \text{ and}$$

$$2c_{41} + c_{42} + c_{43} \leq u_{41} + 1.5u_{42} + 2u_{43}\}.$$

Subsystem 5. Model and reality are:

$$y_{51} = c_{51} + 2.1c_{52} + c_{55} - 3.15u_{51} + \alpha_{51}$$

$$y_{52} = c_{52} + c_{53} - 1.981u_{52} + \alpha_{52}$$

$$y_{53} = 2c_{53} + c_{54} - c_{56} + 0.8u_{51} + 1.2u_{52} + \alpha_{53},$$

$$y_{51}^{*} = c_{51} + 2c_{52} + c_{55} - 3u_{51}$$

$$y_{52}^{*} = c_{52} + c_{53} - (2 + c_{54})u_{52}$$

$$y_{53}^{*} = 2c_{53} + c_{54} - c_{56} + u_{51} + u_{52}.$$

Performance index and constraint set are:

$$Q_5(c_5, u_5) = (u_{51} - 2)^2 + (u_{52} - 1)^2 + \sum_{i=1}^{3} c_{5i}^2 + 3c_{54}^2 + c_{55}^2 + (c_{56} - 1)^2,$$

$$CUY_5 \overset{\Delta}{=} \{(c_5, u_5, y_5) \in R^{11}: (|c_{5i}| \leq 1,\ i=1,\ldots,6),\ (u_{5i} \geq 0,\ i=1,2) \text{ and } \sum_{i=1}^{6} c_{5i} \leq 1.5\}.$$

Finally, the coupling equation is:

$$\begin{bmatrix} u_{11} \\ u_{21} \\ u_{22} \\ u_{31} \\ u_{32} \\ u_{41} \\ u_{42} \\ u_{43} \\ u_{51} \\ u_{52} \end{bmatrix} = \begin{bmatrix} 0&0&0&0&0&0&0&0&0&1 \\ 1&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0 \\ 0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0 \end{bmatrix} \begin{bmatrix} y_{11} \\ y_{12} \\ y_{21} \\ y_{31} \\ y_{32} \\ y_{33} \\ y_{41} \\ y_{51} \\ y_{52} \\ y_{53} \end{bmatrix}.$$

In each example, the aim is to use the two versions of the hierarchical ISOPE technique formulated in this chapter to determine the optimum steady state control for the real system.

7.5.2 Simulation results

The simulation work was performed on a Prime 550 computer. The local optimisation problems were solved using the NAG library subroutine E04UAF.

The simulation results are given in TABLE 7.1. Initial conditions and suitable coefficients chosen for (7.28) to (7.33) can be found in TABLE 7.2. Fig. 7.4. to Fig. 7.9. illustrate the convergence behaviour of performance, some prices, a typical controller set point and a typical modifier during the course of iterations for updating λ.

7.5.3 Discussion of results

The optimality of both algorithms is demonstrated by the results shown in TABLE 7.1. Of particular interest is to compare the efficiency and communication cost of the two algorithms.

From TABLE 7.1 it is observed that although the double iterative algorithm (DIA) requires a considerably greater total number of model optimisation iterations than that required by the single iterative algorithm (SIA), it significantly reduces the total number of set point changes applied to the real process. Following the same arguments as used by Shao and Roberts (1983), we denote M_1 as the total number of

model-based optimisation iterations and M_2 as the total number of set point changes. The time needed to determine the optimal steady state condition may be given in the form

$$t \approx M_1 \times t_1 + M_2 \times t_2,$$

where t_1 is the computing time required for each model optimisation iteration and t_2 is the system settling time period required to reach a new steady state after the set point changes have been applied. Since, in many industrial processes, $t_1 << t_2$ and, moreover, M_2 of SIA is comparable to M_1 of DIA, a significant time saving by using DIA is apparent.

As mentioned before, the purpose of various double iterative procedures is to reduce the on-line information exchange and hence, to reduce the time for determining the optimum solution, even at the cost of increasing the off-line information exchange. If it is accepted that the total number of information exchanges is roughly equal to the sum of the off-line and on-line information exchanges, it often occurs that double iterative procedures increase the total number of information exchanges (for example, Shao and Roberts, 1983). However, from TABLE 7.1, this has not occured in the new DIA designed in this chapter. In fact, the total number of information exchanges is significantly reduced in some of the examples. This benefit is apparently due to the fact that each iteration for calculating the modifiers λ requires n items of on-line information exchange and also to the fact that the DIA reduces significantly the iterations required for calculating λ. It appears from the simulation results that for larger systems such benefit is more apparent (compare examples 6, 5, 4 and 7).

7.6 Summary

Two versions of an optimal adaptive procedure for combining the price coordination technique and the modified two-step method have been presented. The procedure has an important advantage over several other existing methods (such as the price method with feedback and the direct method with feedback) in that, in an efficient manner and based only on an approximate model, it attains the optimal steady state operating condition rather than a suboptimal one. This benefit is obtained at the cost of demanding more information from the real process in the form of measurement derivatives with respect to the controller set points. A further advantage occurs, however, in that due to its adaptive nature, the procedure enables simpler model to be used, and consequently, has a great advantage in the situation where process knowledge is highly uncertain.

The double iterative version is preferred in that it can significantly reduce the time for determining the optimum steady state condition of a real system. An interesting result is that, in the particular examples investigated, the double iterative version of the presented technique does not increase the total number of information exchanges within the overall iterative procedure and may even reduce it significantly.

Using the same technique of Chapter 5 and the stationarity conditions (Lasdon, 1970), optimality conditions similar to those given in Chapter 5 can be formulated for this hierarchical ISOPE technique.

Brdys and Roberts (1985) has discussed sufficient global convergence conditions for another hierarchical ISOPE technique under a linear quadratic assumption. Employing their tactics, global convergence conditions for linear quadratic problems can be derived for the technique given in this chapter. Further work needs to be conducted to investigated more general convergence conditions.

The robustness of the procedure under noisy measurement conditions is investigated in a real-time simulation study given in the next chapter where simple techniques are employed to reduce the influence of the noise.

TABLE 7.1 Comparison of results

example	algorithm	iterations for λ	total iterations	set point changes	real performance	real optimum
4	SIA	42	42	294	6.3266	6.3266
	DIA	6	113	42	6.3266	
5	SIA	95	95	570	2.1405	2.1405
	DIA	13	673	78	2.1405	
6	SIA	36	36	180	5.9261	5.9261
	DIA	20	501	100	5.9261	
7	SIA	62	62	1240	8.5404	8.5404
	DIA	27	520	540	8.5413	

TABLE 7.2 Initial conditions and chosen coefficients

example	algorithm	ξ^0	v^0	p^0	β_v	β_ξ	β_p	K_v	K_ξ	K_p
4	SIA	-	0	$\hat{p}$ *	10^{-4}	-	10^{-4}	0.9I	-	**
	DIA	-	0	$\hat{p}$	10^{-4}	-	10^{-4}	0.9I	-	**
5	SIA	0	0	$\hat{p}$	10^{-4}	10^{-3}	10^{-4}	0.6I	0.8I	0.2I
	DIA	0	0	$\hat{p}$	10^{-4}	10^{-3}	10^{-4}	0.6I	0.8I	0.2I
6	SIA	0	0	$\hat{p}$	10^{-4}	10^{-3}	10^{-4}	0.4I	0.8I	+
	DIA	0	0	$\hat{p}$	10^{-4}	10^{-3}	10^{-4}	0.4I	0.8I	+
7	SIA	-	0	0	5×10^{-4}	-	5×10^{-4}	0.3I	-	++
	DIA	-	0	0	5×10^{-4}	-	5×10^{-4}	0.3I	-	++

* $\hat{p}$: open loop optimal price with $\alpha = 0$.

** K_p= diag{0.2, 0.25, 0.15, 0.2}.

\+ K_p= diag{0.8, 0.9}.

++ K_p= diag{0.2, 0.2, 0.2, 0.1, 0.2, 0.2, 0.2, 0.2, 0.15, 0.2}.

I: identity matrix.

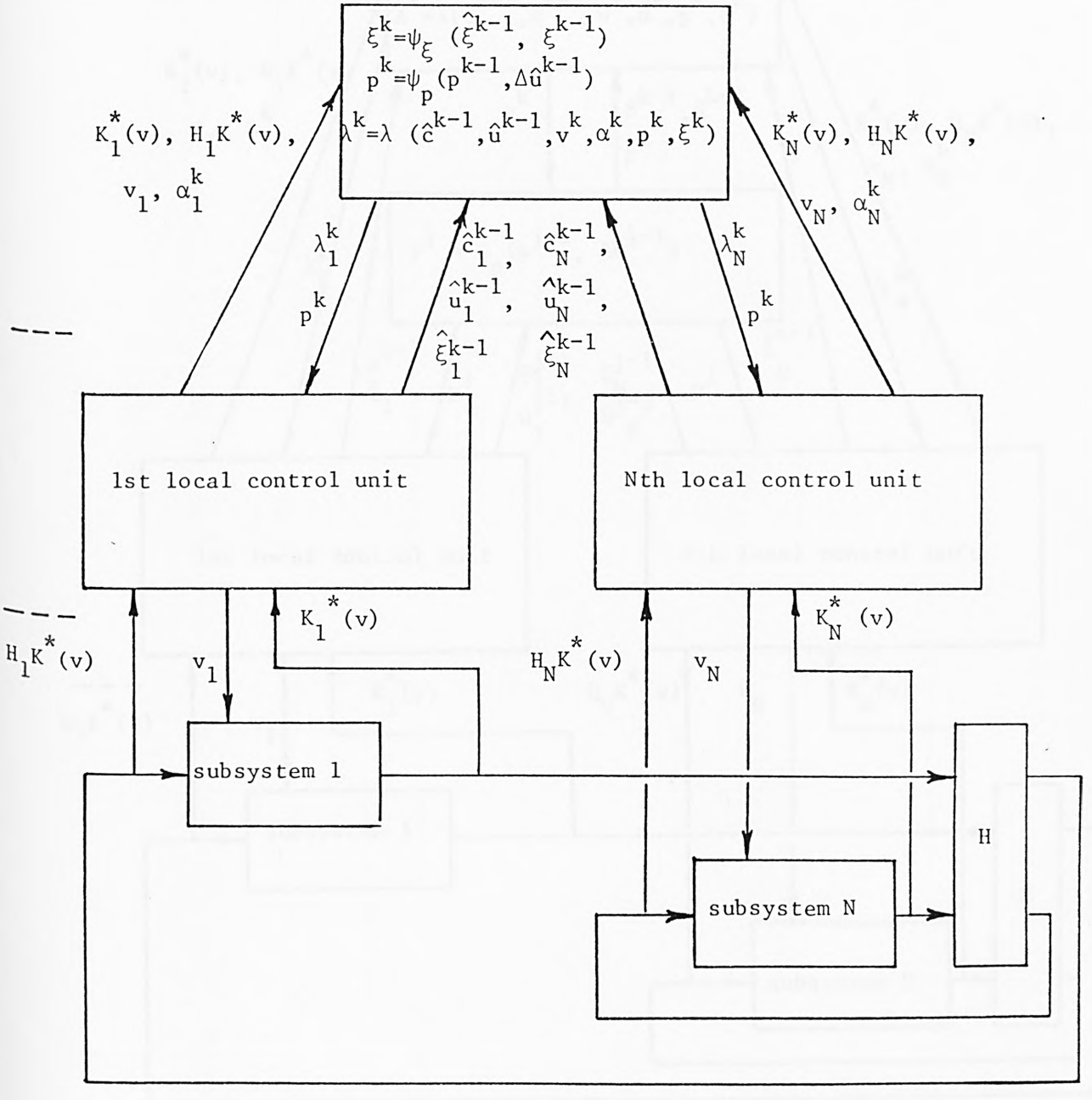


Fig. 7.1. Information structure of SIA

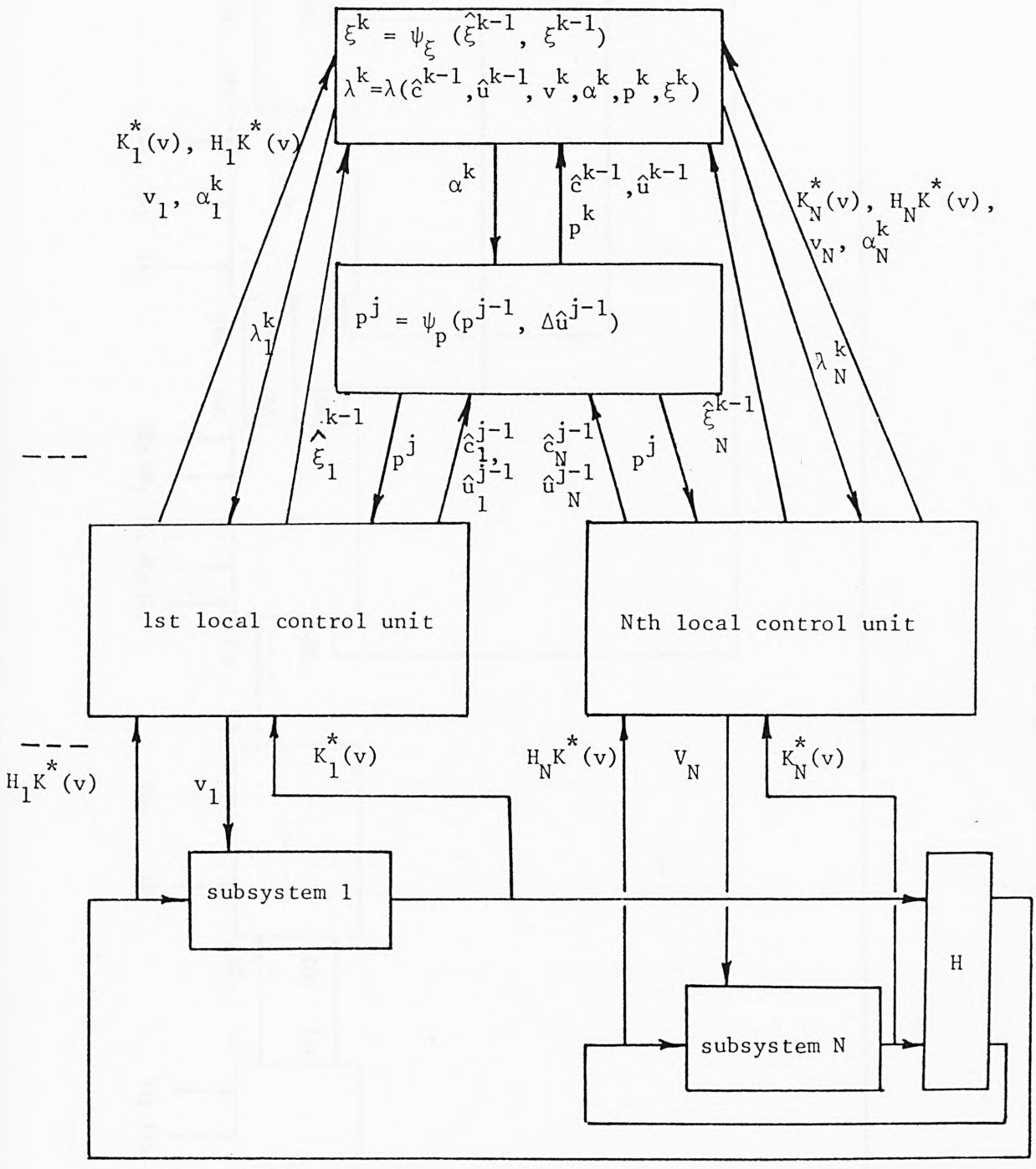


Fig. 7.2. Information structure of DIA

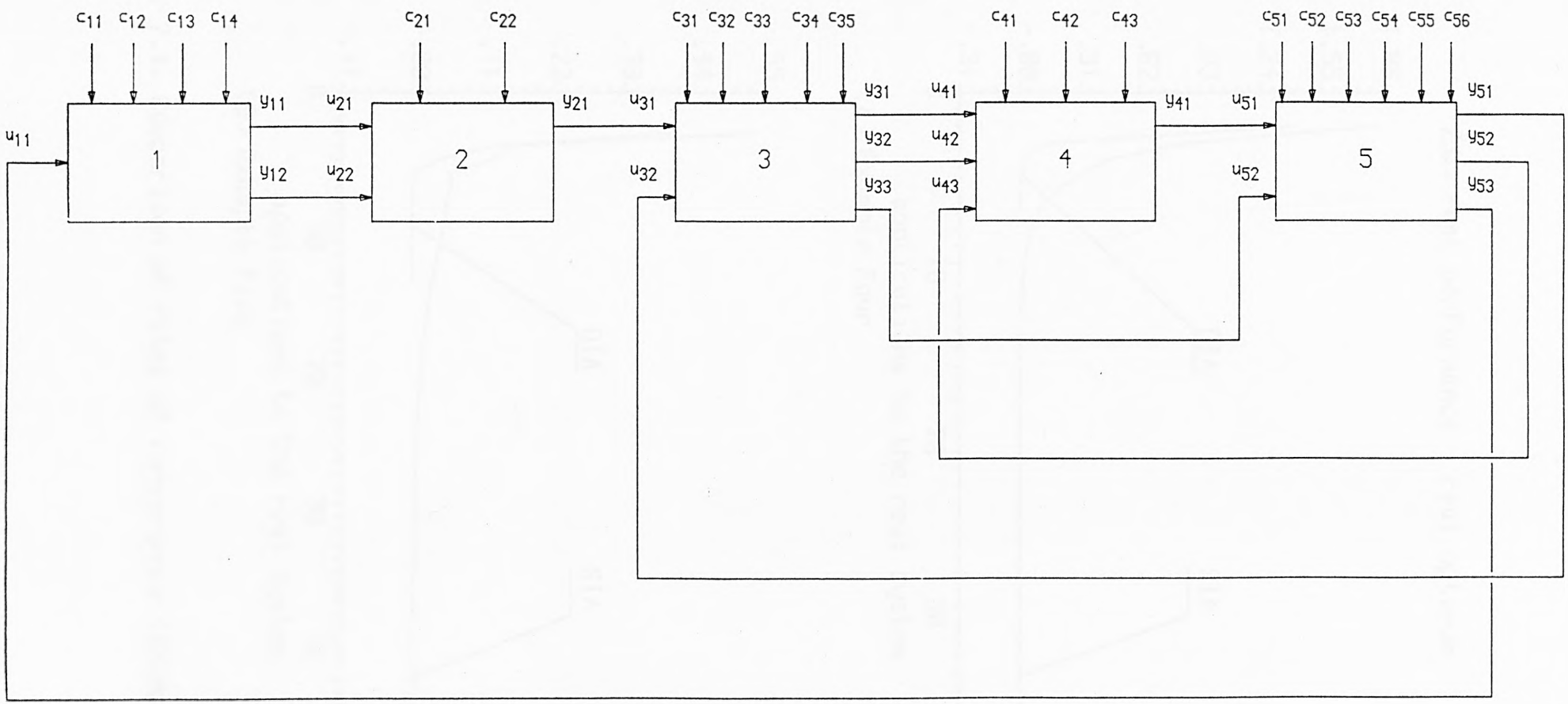
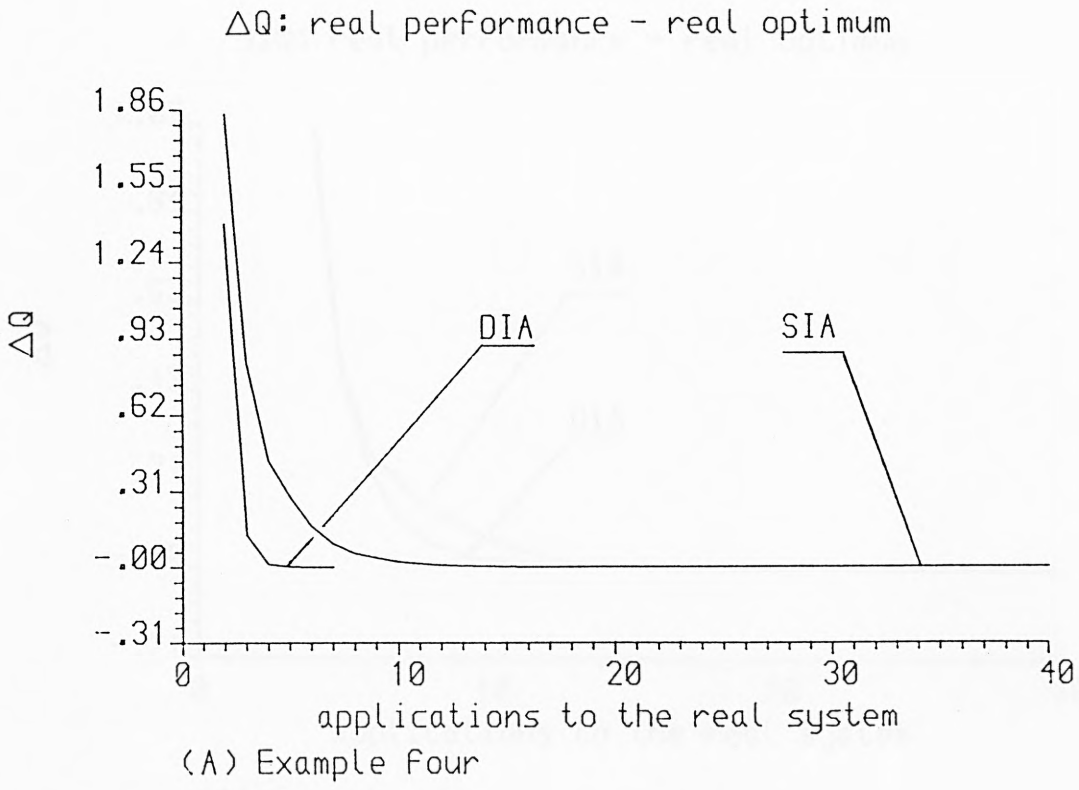


Fig. 7.3. Structure of Example 7



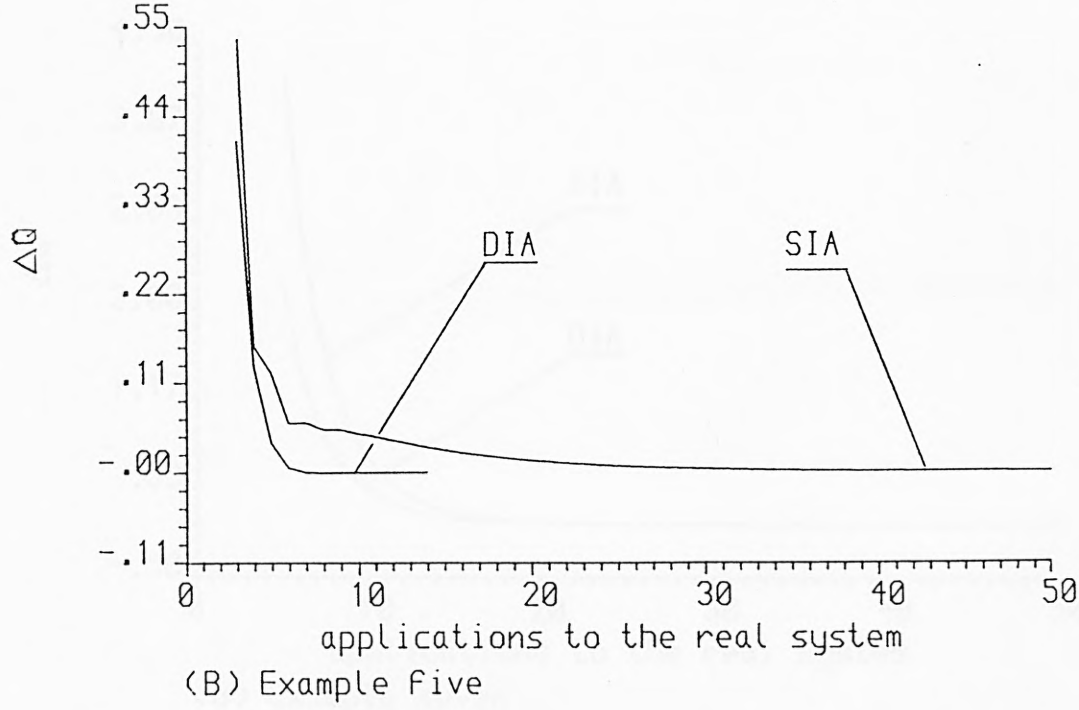
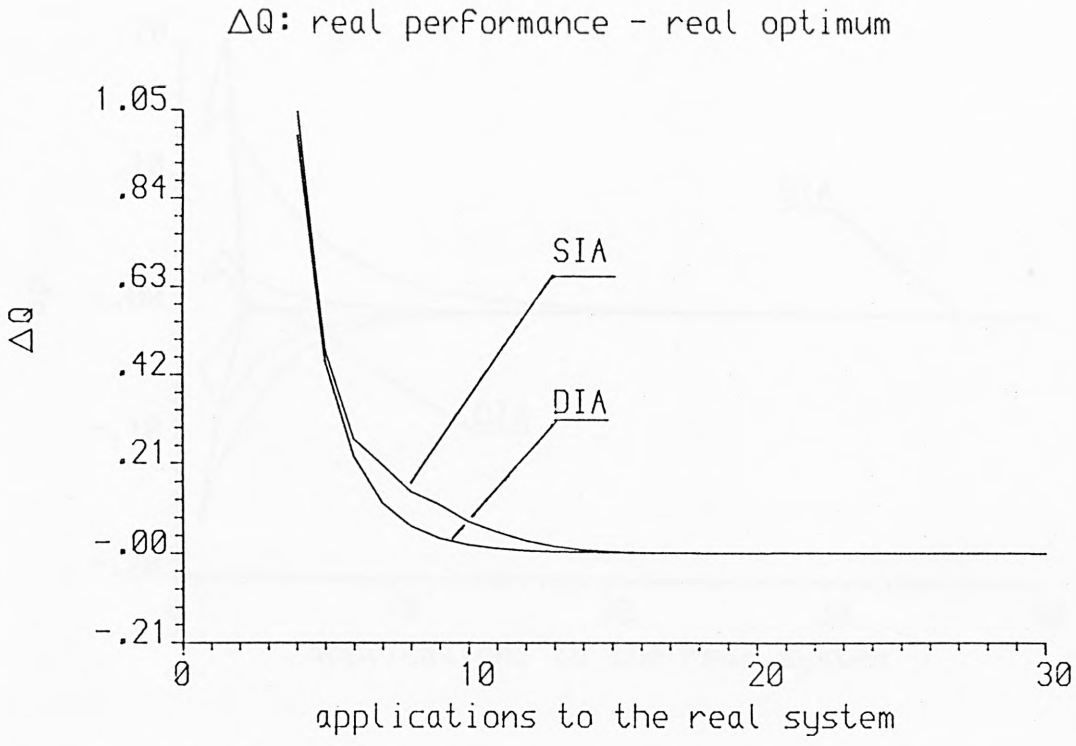
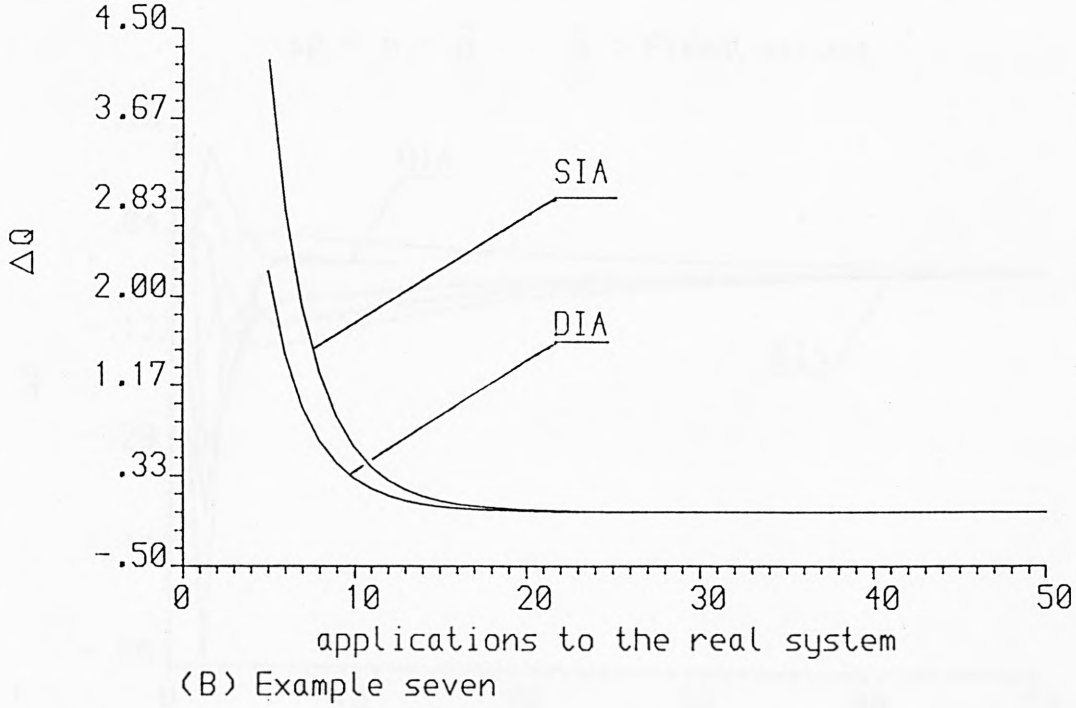


Fig. 7.4. Comparison of rates of convergence (Examples 4 and 5)



(A) Example six



(B) Example seven

Fig. 7.5. Comparison of rates of convergence (Examples 6 and 7)

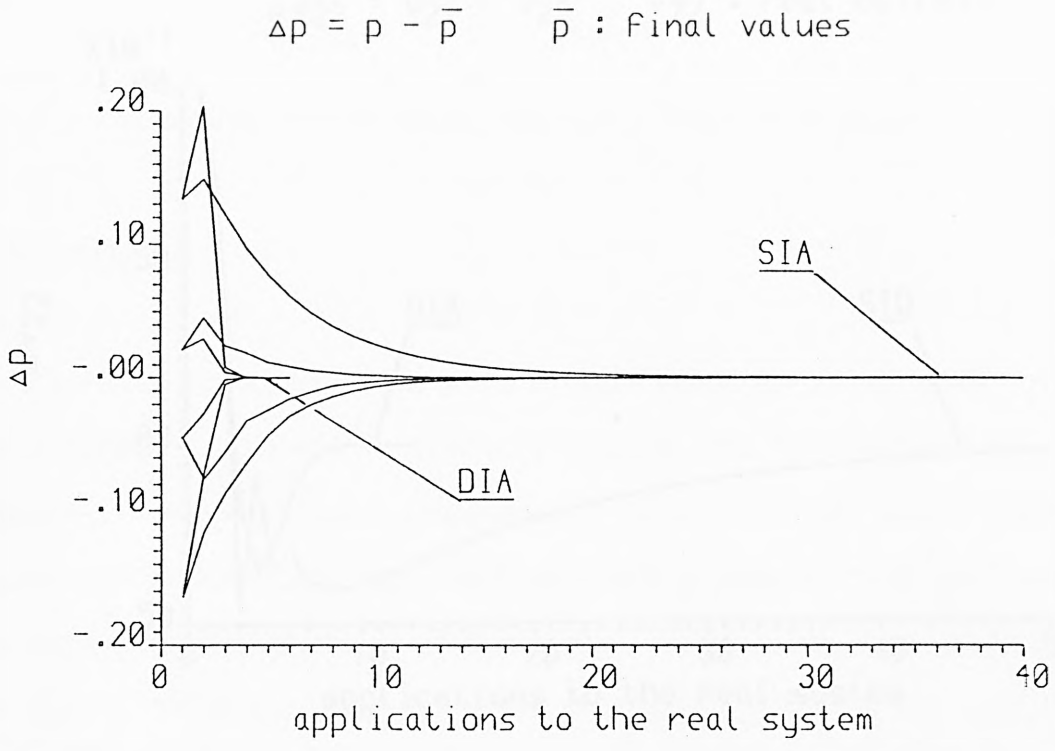


Fig. 7.6. Comparison of price trajectories of Example 4

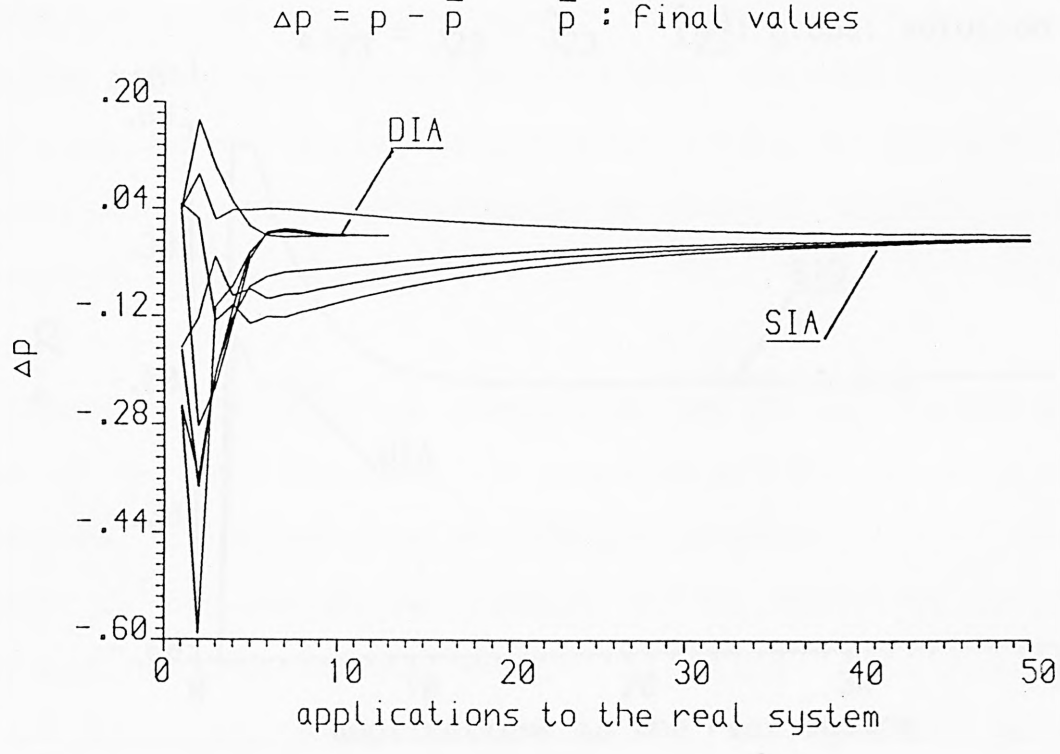


Fig. 7.7. Comparison of price trajectories of Example 5

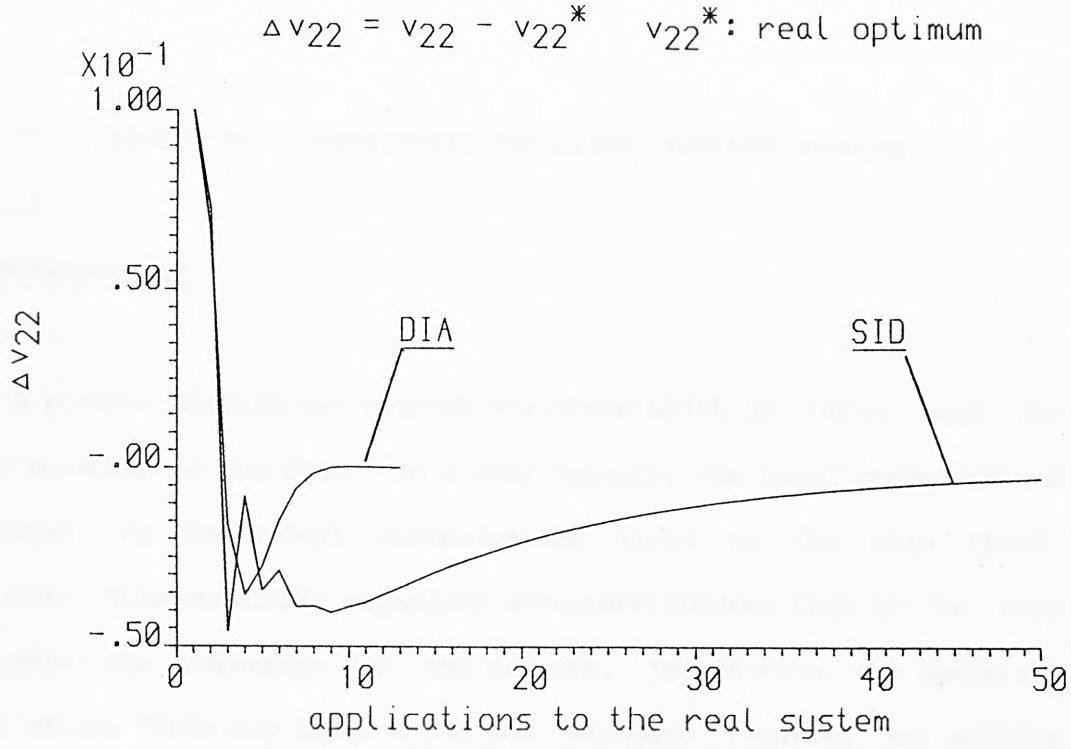


Fig. 7.8. A typical control set point of Example 5

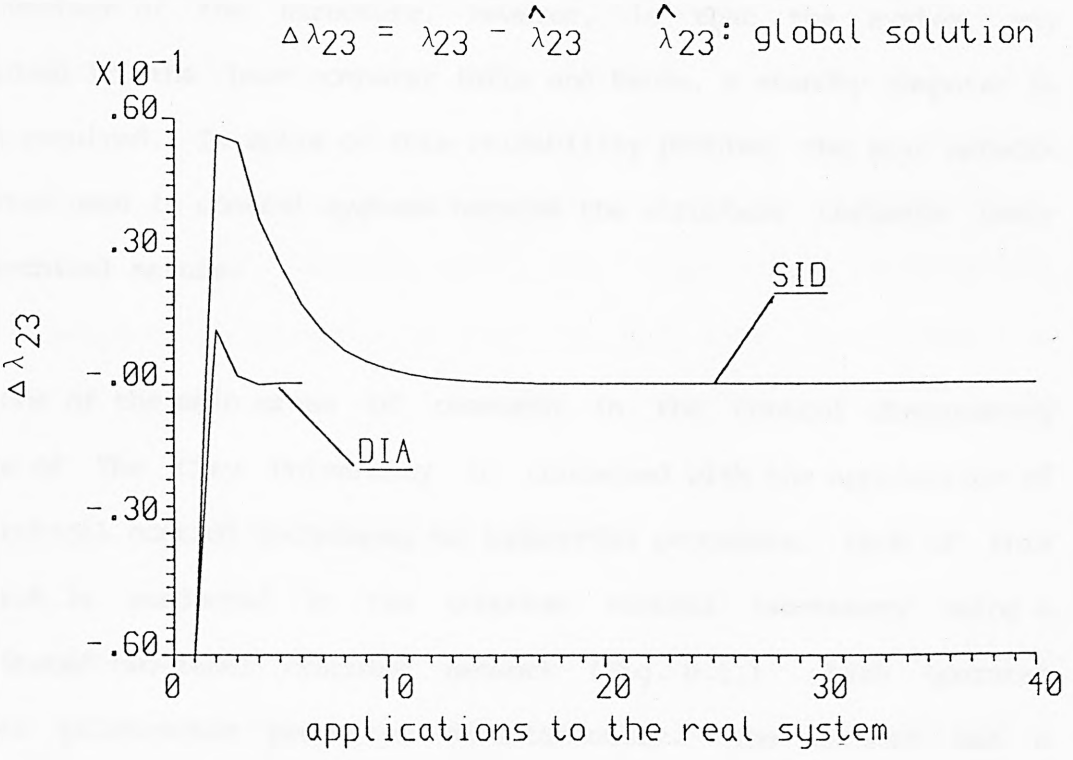


Fig. 7.9. A typical modifier of Example 4

CHAPTER 8. A DISTRIBUTED TWO-LEVEL COMPUTER NETWORK

8.1 Introduction

A popular distributed network structure which is often used for small networks is the star. In a star network, the local computers are connected by individual communication links to the main (host) computer. This modularly organized structure ensures that it is easy to handle the expansion of the network. In addition, the operating speed of the links may be slow and the hardware required to support them can be very simple. This structure is efficient provided that there is little need for communication between local computers (since all such communication must be routed via the host). The major disadvantage of the structure, however, is that the system may breakdown if the host computer fails and hence, a standby computer is often required. In spite of this reliability problem, the star network is often used in control systems because the structure reflects their hierarchical nature.

One of the main areas of research in the Control Engineering Centre of The City University is concerned with the application of hierarchical control techniques to industrial processes. Much of this research is conducted in the computer control laboratory using a distributed two-level computer network (Fig. 8.1.) which operates several pilot-scale processes simultaneously. The network has a typical star structure.

Previously, several Intel 8085 based I-MIC microcomputers occupied

the current positions of Acorn BBC model B microcomputers. In some applications, the speed of the communication links between the I-MICs and the LSI11/23 proved to be insufficient. For example, in the application of the hierarchical ISOPE technique given in Chapter 7 to determine the optimum steady-state operating condition of an interconnected two-subsystem process it was found that considerable amount of time was spent in data communication (Chen, Roberts and Wadhwani, 1985). In order to improve the efficiency of the network and to accommodate new research activities, these I-MICs have gradually been replaced by BBCs with a more efficient inter-communication system.

In the remainder of the chapter the distributed hierarchical computer system and its applications are briefly described. The main focus, however, is on asynchronous serial data communication between an Acorn BBC microcomputer and a DEC LSI11/23 minicomputer. The development of software for such serial data communication is essential to enable the operation of the network which includes a LSI11/23 and several BBCs. The configuration and supporting software of inter-communication between the LSI11/23 and BBCs is discussed in detail.

8.2 Distributed hierarchical computer system

The host computer, a LSI11/23 which is used to coordinate and monitor the computers at the infimal level, has a full complement of 256Kbyte of RAM and runs the TSX-PLUS time-shared operating system. Programmes and data are stored on twin 20Mbyte Winchester discs and an 1.2Mbyte floppy disc. Other peripherals available at the supremal

level include a Tektronix graphics terminal, a hard copy unit, a 180 cps line printer and an Intecolor 8001G graphics terminal. The LSI11/23 is programmed mainly in FORTRAN IV because the real-time support routines are written in FORTRAN or are FORTRAN callable.

The infimal level of the network contains two I-MICs, a DEC LSI11/02 minicomputer and four BBCs. The I-MIC has 16Kbyte of RAM and its software is mostly written in CONTROL BASIC with a few routines, such as the link communication routines, written in Intel 8085 machine code in situations where the execution speed of the high level interpretive language is insufficient. Each BBC has 32Kbyte of RAM and a floppy disc system, and is programmed in BBC BASIC with the exception of the link routines which are written in 6502 assembly language. The LSI11/02 is programmed entirely in FORTRAN IV.

At present an I-MIC controls a hot and cold water mixing rig, a BBC and an I-MIC control a pilot-scale freon vaporiser which is viewed as an interconnected two-subsystem plant. A BBC is used in the control of a TQ 2000 robot manipulator. The rest of the BBCs control an analogue computer simulating interconnected plants. The LSI11/02 controls a pilot-scale travelling load furnace.

The network is used as a development tool for designing and testing control algorithms in a real-time situation. Three types of data communication link exist. The communication between the LSI11/02 and the LSI11/23 takes place over a 16 bit optically isolated parallel link. The data transfer rate for the parallel link is limited primarily by the response speed of the time-shared operating system to

interrupt requests. The communication between an I-MIC and the LSI11/23 takes place over a 20mA current loop serial line at 1200bd, whilst the communication between a BBC and the LSI11/23 takes place via RS423 (RS232) interfaces at 9600bd, i.e., eight times faster than that of the I-MIC and the LSI11/23 link. Stevenson (1984, 1985) has described the configurations and supporting softwares for the parallel link and the serial link between the I-MIC and the LSI11/23. The serial link between the BBC and the LSI11/23 is considered in Section 8.4.

8.3 Applications

8.3.1 Mixing rig

The hot and cold water mixing rig consists of two 0.12m square and 0.45m high vessels in cascade. Streams of hot and cold water enter the first vessel. The local I-MIC controller performs supervised DDC and obtains measurements of levels and mixed temperatures in the two tanks. The LSI11/23 is used to display the measurements and to select the control of the level and temperature in either of the two tanks. Whichever tank is selected, the level is controlled by providing analogue control signals to an electro-pneumatic valve actuator on the cold water line, and the temperature is controlled in a similar way by adjusting a valve on the hot water line.

Due to the inevitable interactions occurring between the two control loops, the process is a valuable vehicle for the demonstration of multivariable control techniques, particularly, the non-interacting

control method and the Inverse Nyquist Array method (Rosenbrock, 1974).

8.3.2 Freon vaporiser

The pilot-scale vaporiser process contains two subsystems. The first subsystem, the freon vaporiser, is a vertical cylindrical tube, 0.075m diameter and 0.69m high. Liquid freon enters the base of the vaporiser tube to be heated by hot water flowing through an inner heating tube which runs the length of the vaporiser tube. On its travel through the tube, the liquid freon absorts sufficient heat to undergo a phase change. The second subsystem is a hot water reservoir tank which supplies the heat to freon liquid during vaporisation.

Previously the process was considered as a centralized system with an I-MIC as its controller. Applications to this process included on-line determining the optimum operating condition of the process using the modified two-step technique (Ellis and Roberts, 1985). Recently the process has been decomposed into two subsystems and a BBC has been employed as the second local decision unit. Application of various hierarchical optimisation techniques to the process is being investigated by a research student.

8.3.3 Robot

A TQ2000 robot manipulator with a BBC microcomputer as its controller has been connected to the system recently. An alternative approach for modelling industrial robots and dynamic simulation of a robot involving decentralized control technique have been studied

(Zainol Anuar, 1985). These methods may be modified and applied to the TQ2000 robot system in the future.

8.3.4 Analogue computer system

This small computer system consists of the LSI11/23, which serves as the coordinator, two BBCs, which are used as two local control units, and an EAL Pace TR48 general purpose analogue computer, which simulates interconnected two-subsystem dynamic processes. The system is an ideal tool for investigating various hierarchical optimisation techniques. Problems associated with the on-line implementation of the techniques can be studied and methods for dealing with these problems can be tested under real time operation. In the next chapter using this system to implement the algorithms given in Chapter 7 will be discussed.

Previously, two I-MICs were used as the local control units. Research work performed in the old analogue computer system can be found in the papers (Roberts and co-workers,1984; Chen, Roberts and Wadhwani, 1985).

8.3.5 Travelling load furnace

The pilot-scale travelling load furnace consists of eight separately controllable heating zones through which are propelled aluminium block loads measuring 202x198x102 millimetres. Each heating zone contains six radiant heaters, the power to which is controlled by phase controllers. The loads are propelled by an adjustable speed

conveyor system. Measurements of the load temperatures and the position of the loads within the furnace are taken. The local controller interfacing the furnace is a DEC LSI11/02 minicomputer which is connected to the LSI11/23 at a higher level using an 16 bit parallel optically isolated link. Data received by the LSI11/23 during an experiment can be transferred to a Prime 550 minicomputer for analysis.

This complex heating process is fitted into a category of continuous dynamic processes which may be split, in time, into batches, and its mathematical model is highly uncertain. Stevenson, Brdys and Roberts (1985), and Stevenson (1985) have extended the use of the modified two-step technique originally developed for steady state processes to batch type dynamic processes and have reported a successful application to on-line adaptive control of the travelling load furnace.

8.4 BBC and LSI11/23 serial data link

8.4.1 Hardware configuration

The link connecting a BBC and the LSI11/23 is shown in Fig. 8.2. The connection requires only three lines: a DATA-IN, a DATA-OUT and a GROUND. This is the barest minimum for a bidirectional link. The two handshaking lines of the BBC, Request To Send (RTS) and Clear To Send (CTS), are not required.

The major component of the RS423 interface in a BBC model B microcomputer is a 6850 Asynchronous Communication Interface Adaptor

(ACIA) which is responsible for changing 8 bit data from a parallel format to a serial format and vice versa. A SERial Uncommitted Logic Array (SERULA) allows the input to the 6850 ACIA to be switched between a cassette and the RS423 interface and defines the baud rate. The programmer can gain access to the RS423 interface through three registers. The register at address &FE08 acts as the ACIA Control Register (ACIACR) when it is written to, and acts as the ACIA Status Register (ACIASR) when it is read from. In the same manner, the register at address &FE09 functions as the ACIA Transmit and Receive Data Registers (ACIATDR and ACIARDR) depending on whether it is written to or read from. The third register at address &FE10 is the SERULA control register. The detailed information of these registers can be found in the books (Bray, Dickens and Holmes, 1983; James, 1983).

The LSI11/23 uses the DLV11-J 4-channel asynchronous serial unit to interface BBCs. Each channel can be independently configured for a RS232 link and is connected to a BBC RS423 interface. The transmit and receive baud rates have been preset to 9600bd but it is possible to select various baud rates using software control. Every channel is assigned with four registers and two vectors. These registers are all memory mapped and are: Receive Control/Status Register (RCSR), Receive BUFfer register (RBUF), Transmit Control/Status Register (XCSR) and Transmit BUFfer register (XBUF). The two vectors are the receive vector and transmit vector. The DEC microcomputer interfaces handbook gives full detailed information.

8.4.2 Communication protocol

The following communication scheme has been used:

1) Data transfer between the LSI11/23 and a BBC occurs asynchronously and is initiated by the BBC.

2) The communication data are organized into blocks where the length of the blocks can vary.

3) After a block of data has been received, the LSI11/23 must devote an amount of time to process the data.

4) Before transmitting a block of data to a BBC, the LSI11/23 must issue a ready-to-send flag to the BBC.

At present only integer data are considered. Since the link can only transfer data in the form of bytes (8 bit), integer data must first be converted into bytes and then sent to the destination where they are restored into integers. However, there is a further consideration. Fig. 8.3. shows how the LSI11/23 and the BBC store the value of an integer. Because the LSI11/23 can only accommodate two-byte integers ranging from -32767 to 32767, integer data to be transferred over the link should be limited to the same range. It is clear from Fig. 8.3. that to transfer an integer from a BBC it is enough to transmit two lower bytes to the LSI11/23. However, to transfer an integer from the LSI11/23 to a BBC the corresponding values of two higher bytes in the BBC must be decided by bit7 of received byte 2, which is the sign bit of the integer in the LSI11/23. If this bit is 1 the received integer is negative, therefore, the bits of the two higher bytes must be filled with ones. Otherwise, they should be filled with zeros.

8.4.3 Software

8.4.3.1 General description

The link programmes for the LSI11/23 are written in FORTRAN. Since the communication load on each of the links is light the operating speed of the FORTRAN routines is adequate. The software to operate the link inside a BBC is written in 6502 assembly language. The link programme flowchart is shown in Fig. 8.4.

The LSI11/23 invokes a particular link by calculating the RCSR address and the receive vector location for the required channel of the DLV11-J serial interface card. Once these two addresses are known the addresses of the remaining registers and the transmit vector can easily be calculated. The TSXLIB routine ICNINT is then used to connect the receive handler and the transmit handler to the appropriate vectors. This is followed by setting up the receive interrupt vector, and initializing the 6850 ACIA and the SERULA by writing appropriate values into the ACIACR and the SERULA control register inside the BBC. As soon as the above two steps have been completed the link involved is ready for operation.

The communications are always initiated by the BBC because this simplifies the link software. The 6850 ACIA transmit interrupt is disabled in the initialization. Each time that the BBC needs to transmit a block of data a non-interrupt-driven routine simply writes the data one by one into the ACIATDR. The BBC receives data using an

interrupt-driven mechanism. Using an indirect addressing mode all maskable interrupts on a BBC are accessed through an Interrupt ReQuest Vector 1 (IRQ1V) at addresses &204 and &205, and the 6850 ACIA chip has the highest priority among all hardware which generate maskable interrupts. Interrupts are intercepted at the IRQ1V level and passed to a receive interrupt service routine. If an ACIA receive interrupt occurs this routine performs proper receive service. If the interrupt was not caused by an ACIA receive it simply hands over the control to the BBC system service routines. It is very convenient using the resident integer variables (@%, A% to Z%) to transfer data between the link routines and BASIC programmes because they have fixed locations (starting from @% at &0400 with four bytes for each variable) and they can be changed or cleared only through explicit assignment or switching the machine off and on.

In the LSI11/23 each time a data transfer request over the link is made the TSX-PLUS operating system calls the appropriate handler which then either sends data from a buffer to the link or receives data from the link and stores it in a buffer. The information is passed between the link programmes and the FORTRAN main programme using COMMON statements.

A complete communication cycle involves the follows: At the start of the communication the BBC transmits the length of the data to be sent to the LSI11/23. It is then followed by the actual data. After the LSI11/23 has processed the received data, a block of data is transmitted to the BBC. Again the first data is the length of the data to be sent to the BBC. A description of the link programmes is given

in the successive subsections. Program listing can be found in (Chen, Wadhwani and Roberts, 1985, 1986).

8.4.3.2 BBC link software

A total of eight routines written in 6502 assembly language form the supporting software for the serial data link in the BBC. Six of these routines together perform the non-interrupt-driven transmit and the interrupt-driven receive using the 6850 ACIA and a single-byte receive buffer. The other two routines support the task of transmitting and receiving a whole integer. Except RECBSR, these routines can be called in BASIC programmes by the instruction CALL. The link routines reserve two resident integer variables R% and T% for operation. A brief summary of the tasks carried out by them is given below.

LININI

This routine sets up the receive interrupt vector, and initializes the ACIA and the SERULA by writing appropriate values to the ACIACR and the SERULA control register.

The main task of LININI is to save the original content of IRQ1V to OLDSRV and to write the location of the link receive interrupt service routine (RECBSR) to IRQ1V so that whenever a maskable interrupt occurs RECBSR will be called first. It is absolutely necessary to disable interrupts during the course of changing the two bytes of IRQ1V in case an interrupt occurs during the process.

The reason for disabling the ACIA transmit interrupt is very simple. If not disabled the transmit interrupt would occur immediately since the ACIA starts out empty and therefore is ready to transmit data but at this stage no trasmit data is available. However, if we have disabled the transmit interrupts the system cannot learn from the interrupts that the ACIA is ready to transmit. The solution to this problem is to create a non-interrupt-driven routine that writes a transmit byte to the ACIA.

RECBSR

The RECBSR identifies whether the interrupt was an ACIA receive interrupt (bit0 of ACIASR=1) or the interrupt of some other source. If an ACIA receive interrupt occurs, the routine reads the data from the ACIARDR, saves it in the software buffer (RXBYTE) and sets up the receive data flag (RXBFLA). If the interrupt was caused by some other source, a JMP to OLDSRV instruction in the routine redirects the interrupt to the system service routines.

Interrupts must not have any effect on the interrupted programme, i.e., the interrupted programme will expect the processor registers and flags to be exactly the same after the return from an interrupt routine as they were before the interrupt occurred. Thus an interrupt routine must either not alter any registers (which is difficult) or restore all register contents to their original values before returning. In a BBC the interrupt routines are entered with interrupts disabled, and the following conditions apply on entry to an interrupt routine:

(i) The original processor status byte and return address are already

stacked ready for a RTI instruction at exit.

(ii) The original X and Y states are still in their registers.

(iii) The original A register contents are in location &FC. However, this location is only used temporarily at the very beginning of an interrupt routine.

The first action of the RECBSR, therefore, is to push the contents of &FC onto the stack. Notice that because the RECBSR has no effect on the X and Y registers, there is no need to save them. If the interrupt was not caused by an ACIA receive, before a JMP to the system interrupt routines, the conditions (i), (ii) and (iii) must be restored. Since (i) and (ii) are still true the only action needed is to pull the contents of the A register out of the stack and to restore them in &FC.

RXBGET and RXBSTA

The RXBGET waits for a receive byte to become available, clears the receive software flag and loads the byte to the second least significant byte of R%.

The RXBSTA sets the CARRY flag from the receive software flag so that the RXBGET can examine the status of the receive software buffer by testing CARRY. If CARRY=1 a receive byte is ready in the software buffer. If CARRY=0 no receive byte is available.

TRNBSR

The TRNBSR writes a transmit byte (assumed to be the least

significant byte of T%) to the ACIATDR and hence, forces it to be sent out. This routine is the non-interrupt-driven entry to the link, i.e., whenever a byte is to be transmitted out it must be called (compare with the interrupt-driven RECBSR).

BYTEOP and BITTES

The BYTEOP shifts the contents of a memory location to the one byte lower location. The original location is then filled with zero. The memory locations treated must be in page 4 (resident integer variables are in page 4). This enables multibytes to be received or transmitted using a fixed byte in R% or T%.

As mentioned early, an integer in the LSI11/23 consists of two bytes, while an integer in the BBC has four bytes (see Fig. 8.3.). After the BBC has received the two bytes of an integer from the LSI11/23 and has stored them in the two lower bytes of R%, BITTES can be used to check bit7 of the second byte, which is the sign bit of the integer in the LSI11/23, and to decide the values of the two higher bytes of R% according to the value of this bit. To set the X register to the required low byte of an address, assignment

X% = low byte of address

can be used before CALL the routines in BASIC programmes.

TIDYOU

This routine restores IRQ1V to the original place. It should be called at the end of execution of the programme so that the BBC

microcomputer can function normally after the termination of the link operation.

8.4.3.3 LSI11/23 software

The LSI11/23 link subroutines written in FORTRAN have been obtained by modifying the original subroutines written for the I-MIC and the LSI11/23 serial link (Stevenson, 1984, 1985). The changes are mainly in the interrupt handlers RXIHAN and TXIHAN, and the completion routine RK, which simplify the programming and improve the efficiency of the network.

INIT

This subroutine accepts an argument which is the link number required. It then calculates the addresses of the RCSR, RBUF, XCSR and XBUF, and the locations of the receive vector and the transmit vector. The TSXLIB routine ICNINT is used to connect the handlers RXIHAN and TXIHAN to the appropriate vectors. The receive interrupt is enabled on exit.

RXIHAN and TXIHAN

The receive interrupt handler RXIHAN and the transmit interrupt handler TXIHAN perform actual data transfer over the link. RXIHAN leaves the transmit interrupts enabled on exit. TXIHAN disables the transmit interrupts after it is entered, it also echoes a zero to the BBC while the BBC is transmitting. These two handlers, together with

software flags FLAG1 (RXFREE), FLAG11 (TXFREE) and FLAG21, ensure that correct data transfer is achieved.

RK

On the termination of receiving data, the completion routine RK is called to store and/or process the newly received data. Notice that it sets the flag (INFLAG) to inform the LSI11/23 main programme.

RXFORM and TXFORM

The two accessory subroutines RXFORM and TXFORM are used to convert the received data from a BYTE array into an INTEGER*2 format and to convert the data to be transmitted from an INTEGER*2 format to a BYTE format.

8.4.3.4 How to use the link programmes

The simplest way to use the BBC link routines is to join the BASIC main programme after the link programmes. Each time the programme is executed the BBC first assembles the link routines into machine code and then enters the main BASIC programme.

A better method is to assemble the link routines into machine code separately and to hide them somewhere in the memory. There are several methods for hiding a piece of machine code. Unfortunately none of them proves to be really satisfactory. One technique involves altering the pseudo-variable PAGE to make space for machine code, which proves

useful. For detailed discussion see (Chen, Wadhwani and Roberts, 1985).

8.5 Concluding remarks

It has been shown that, by adopting a star network configuration, the expansion of the distributed two-level computer system has easily been achieved. A more powerful local computer network has been established and a more efficient inter-communication system has been developed.

The data communication scheme discussed in this chapter is designed such that it can easily be expanded to enable the LSI11/23 to communicate with more than one BBC simultaneously which is required in the application of hierarchical decomposition-coordination techniques (Chen, Wadhwani and Roberts, 1986).

Although, at present, only the transfer of integer data has been incorporated it is expected that the link software can be improved in order to transfer real data over the serial data link. However, the data formats for a real number in the BBC and the LSI11/23 have significant differences, which is the main difficulty and must first be overcome.

It is possible to organize the link programmes so that the data communication can be initiated randomly either by the BBC or by the LSI11/23, where the software for such a communication scheme is more complex. Previous research applications conducted using the I-MIC and

the LSI11/23 serial link have suggested that the simple link strategy given in this chapter is often adequate.

TABLE 8.1 Device RCSR addresses and Receive Interrupt vectors

DQ614	Winchester controller	174400-160
M7941	LP parallel interface	177514-200
MXV21	Floppy controller	177170-264
MLSI DRV11	LSI11/02	167770-270
M8043 DLV11-J	Port 0 TSX line 1	176540-300
	1 2	176550-310
	2 3	176560-320
	3 (console) 4	177560-60
M8028 DLV11-F	I-MIC	176520-330
- -	I-MIC (Vaporiser)	176530-340
- -	Intercolor	175610-350
- -	I-MIC (Mixing rig)	176500-360
- -	I-MIC	176510-370
MLSI DLV11-J	Line 1 BBC (Analogue	176600-500
	2 BBC computer)	176610-510
	3 BBC	176620-520
	4 BBC	176630-530
M8028 DLV11-F	Prime 550 link	176570-770

All numbers are octal number.

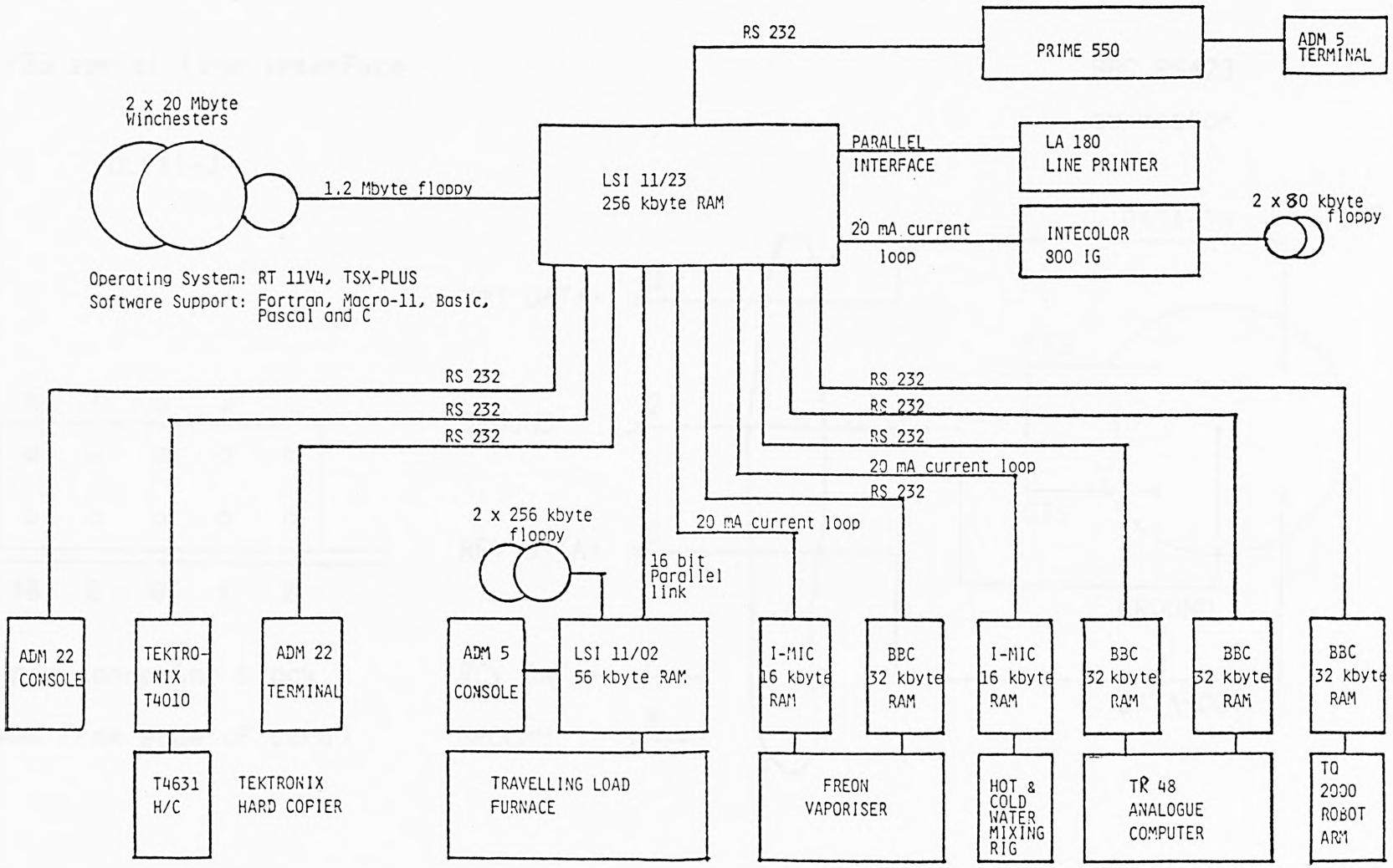


Fig. 8.1. Distributed computer network

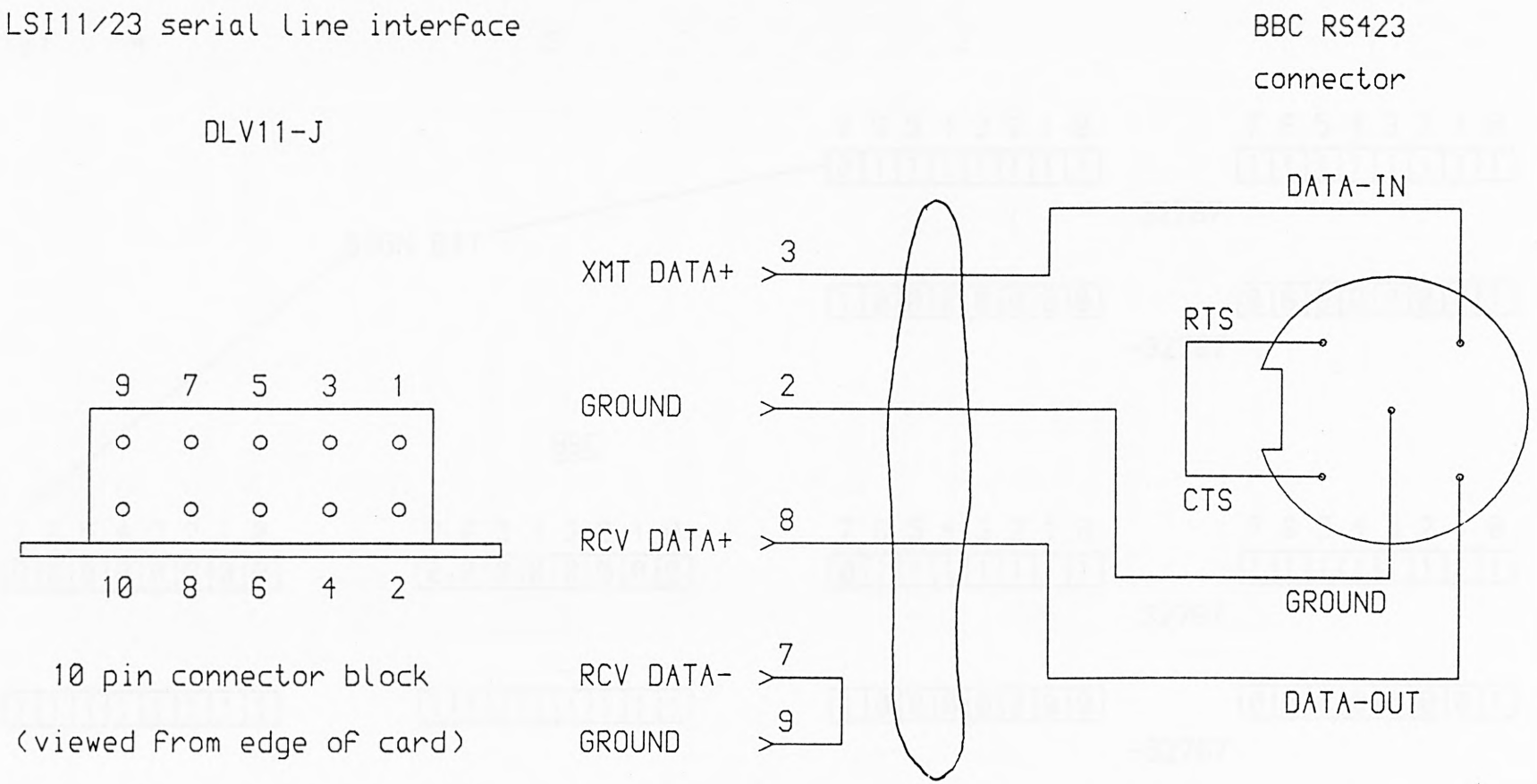


Fig. 8.2. LSI11/23 and BBC serial connections

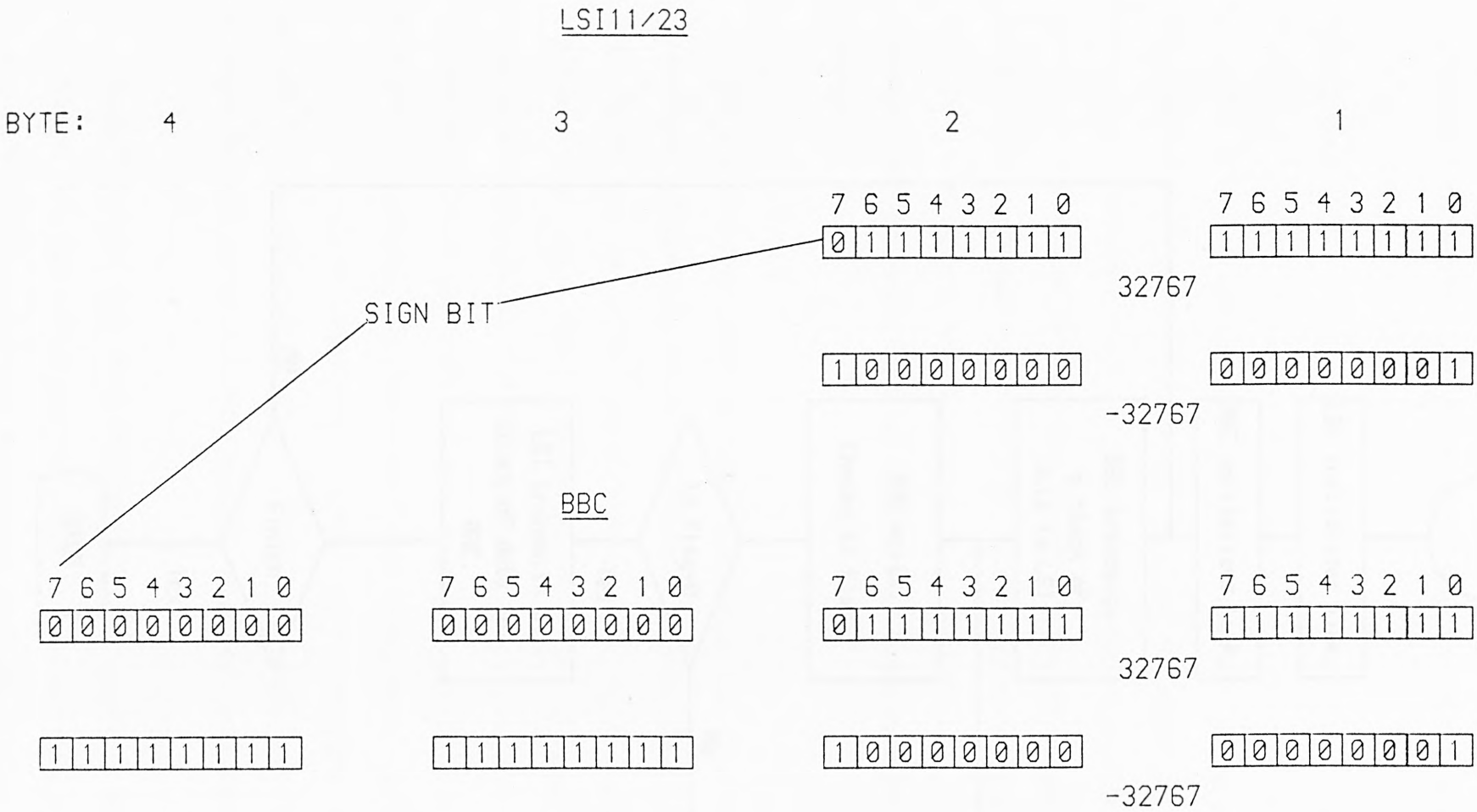


Fig. 8.3. LSI11/23 and BBC integer formats

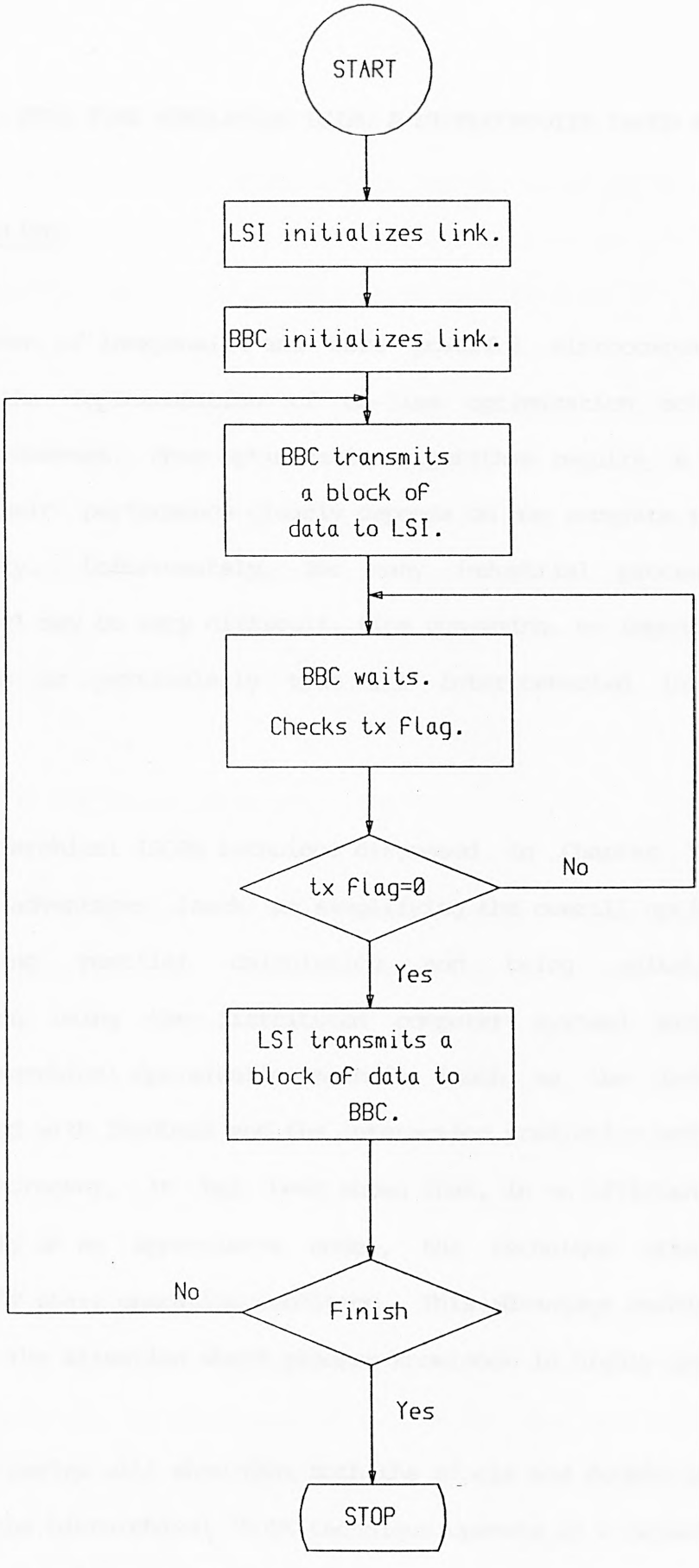


Fig. 8.4. Link programme flowchart

CHAPTER 9. REAL TIME SIMULATION USING A MICROCOMPUTER BASED SYSTEM

9.1 Introduction

The advent of inexpensive and more powerful microcomputers has facilitated the implementation of on-line optimization schemes to industrial processes. Most optimization algorithms require a process model and their performance clearly depends on how accurate the model is to reality. Unfortunately, for many industrial processes an accurate model may be very difficult, time consuming, or impractical to obtain. This is particularly true for interconnected large-scale processes.

The hierarchical ISOPE technique discussed in Chapter 7 shares many common advantages (such as simplifying the overall optimisation task, enabling parallel calculation and being suitable for implementation using the distributed computer system) with other existing hierarchical optimisation methods (such as the interaction balance method with feedback and the interaction prediction method with feedback). Moreover, it has been shown that, in an efficient manner and based only on an approximate model, the technique attains the optimal steady state operating condition. This advantage enables it to be used in the situation where process knowledge is highly uncertain.

A quick review will show that both the single and double iterative versions of the hierarchical ISOPE technique operate in a decentralized manner at the local level, and there is no need for communication among the local units. They can easily be implemented using hierarchical

(star) computer networks such as the one discussed in Chapter 8, with a central computer (supremal level) serving as the coordinator and local computers (infimal level) serving as the local control units. Parallel local optimisation and parameter estimation calculations are achieved.

In this chapter, we implement the hierarchical ISOPE technique using part of the network shown in Chapter 8, where the aim is to investigate the behaviour of the technique in a real-time environment and to study methods for dealing with some problems associated with on-line application. The efficiency of the two versions of the technique is also compared.

9.2 On-line implementation aspects

Using the distributed computer system, parallel computation can be performed at the local level once coordination variables have been received from the supremal level. However, synchronisation and interprocess communication problems arise at the local units because the iteration in each unit is finished within a different time interval.

Because steady state measurements are needed in order to estimate parameters and, particularly, to compute finite difference approximations of the derivatives with respect to the controller set points, bad quality measurements will affect the iterative procedure. For this reason synchronisation is recommended to enable controls to be sent to the real process simultaneously. Roberts and co-workers (1984) considered two methods for synchronising the local units -- elapsed

time and semaphore.

The 'elapsed time' method requires an estimate of the time taken for each local unit to finish one iteration. Each unit then waits a sufficient time to enable the slowest unit to complete its task before sending the controls to the real subprocesses. Although decentralisation can be obtained at the local level, it is difficult to determine the waiting time for every iteration of each unit. This arises because the computation time required at each iteration is, in general, unpredicable and often varies due to differing communication delays caused by the time-shared operating system used at the supremal level.

The second method is the 'semaphore' method. After each optimization computation (or other actions such as taking measurements) each local unit sends a completion flag to the coordinator and waits. A task of the coordinator is to check that all the completion flags from the local units have been received before transmitting a start flag to each unit, all of which then send controls simultaneously to the real subprocesses. In real implementation, data sent to the coordinator may well serve as completion flags and data sent from the coordinator may be used as a start flag. This synchronisation scheme has been shown to work efficiently in previous research and, therefore, is used in the implementation of the hierarchical ISOPE technique investigated in the present chapter.

After controls have been applied to the real subprocesses, all local control units should wait a sufficient time until the plant has

settled down, at which point steady state measurements are taken. It is difficult to reduce this on-line waiting time interval without risking the quality of the steady state measurements. However, if the number of controller set point changes can be reduced the time for determining the optimal solution may be less. The double iterative algorithm is aimed at achieving this purpose.

A usual way to speed up the iterative procedure involving multi-processors is to introduce some kind of asynchronization to the procedure. Because of the complexity of the hierarchical ISOPE technique (mainly calculating real output derivatives) it is difficult to choose a suitable asynchronized operating scheme for it, which achieves the purpose of faster convergence. Experience also shows that if the measurements are taken long before the process reaches its steady state the performance of the technique will deteriorate. Therefore, only the synchronized scheme mentioned above is employed to implement the technique.

The requirement to evaluate real process derivatives represents a significant drawback of the technique. In the situation where the noise level becomes intolerable filter techniques should be applied to reduce the influence of noise. A simple technique involves using a first-order low-pass filter to smooth each element of the modifier vector which has already been proposed in Chapter 4.

9.3 Configuration of the simulation system

Part of the computer network used to implement the algorithms

given in Chapter 7 includes the LSI11/23, which serves as the coordinator, two BBC microcomputers, which are used to perform local parameter estimation and optimisation tasks, and an EAL Pace general purpose analogue computer, which simulates an inteconnected two-subsystem dynamic process (Fig.9.1.).

Measurements are taken using the BBC analogue port, which is a 4 channel 12-bit A to D convertor. This is really a poor performance convertor. In order to reduce the thermo-effect of the port and to improve the quality of measurements, the reference diodes of the port have been replaced to increase the range of voltages from 0v to 1.8v to 0v to 2.5v and a simple averaging procedure is performed.

Control signals applied to the plant are generated using two 3 channel 8-bit D to A convertors. Data are outputed to the D/A convertor through the BBC user port by means of software control. The principle is as follows: The output/input register B of the user port at address &FE60 is divided into two halfs. The lower half contains 4-bit data, and the other half acts as control bits to decide which channel and which half (lower or upper half) of the channel data register the 4-bit data will be assigned to (see Fig.9.2. and TABLE 9.1). The user port must be set up for 8-bit output and control signal CB2 set to pulse mode before the above application can be realized. This can be achieved by setting all bits of the Data Direction Register B (DDRB) at address &FE62 to one and the upper half of the Peripheral Control Register (PCR) at address &FE6C to binary 1010. In a similar way a 4 channel 12-bit D/A convertor can also be obtained at a cost of decreasing converting speed.

To end this section a summary of digital data and analogue signal ranges is provided. Control inputs to the two D/A convertors are limited in the range of -1.0 to 1.0 (real numbers). The corresponding output voltages are -2.5v to 2.5v. The minimum and maximum voltages of the analogue computer are -10v and 10v, which correspond to real numbers -4.0 and 4.0, respectively. Voltages ranging from -10v to 10v must first be transferred to 0v to 2.5v before they can be inputed to the A/D convertor of the BBC microcomputer whose outputs are integers from 0x16 to 4095x16. A simple software operation can then be applied to change these integers to real numbers in the range of -4.0 to 4.0.

9.4 Study example

An interconnected two-subsystem plant (Fig. 9.3.) is simulated using a TR48 analogue computer. The values of the time constants T_{11} to T_{26} can be chosen to achieve a required overall plant settling time period. The steady state of the process is governed by the following equations

$$y_{11}^{*} = 1.4c_{11} - 0.6c_{12} + 1.8u_{11},$$

$$y_{21}^{*} = 1.3c_{21} - 1.1c_{22} + 1.1u_{21}$$

$$y_{22}^{*} = 2.3c_{22} - 0.7c_{23} - 1.1u_{21}.$$

The model equations and coupling equation are

$$y_{11} = c_{11} - c_{12} + 2u_{11} + \alpha_{11},$$

$$y_{21} = c_{21} - c_{22} + u_{21} + \alpha_{21}$$

$$y_{22} = 2c_{22} - c_{23} - u_{21} + \alpha_{22}.$$

$$\begin{bmatrix} u_{11} \\ u_{21} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} y_{11} \\ y_{21} \\ y_{22} \end{bmatrix}.$$

The performance indices and constraint sets are

$$Q_1(c_1, y_1) = (y_{11} - 1)^2 + c_{11}^2 + c_{12}^2,$$
$$Q_2(c_2, y_2) = 2(y_{21} - 2)^2 + (y_{22} - 3)^2 + c_{21}^2 + c_{22}^2 + c_{23}^2.$$

$$CUY_1 \overset{\Delta}{=} \{(c_1, u_1, y_1) \in R^4 : |c_{11}| \leq 1,\ |c_{12}| \leq 1 \text{ and } 0.8 - c_{12} - 0.6u_{11} \geq 0\},$$
$$CUY_2 \overset{\Delta}{=} \{(c_2, u_2, y_2) \in R^6 : |c_{2i}| \leq 1,\ i=1,2,3\}.$$

where $c_1 = (c_{11}\ c_{12})^T$, $c_2 = (c_{21}\ c_{22}\ c_{23})^T$, $u_1 = u_{11}$, $u_2 = u_{21}$, $y_1 = y_{11}$, $y_2 = (y_{21}\ y_{22})^T$.

The two local optimization problems (which are much simpler than the overall optimisation problem) are simple enough to be solved analytically. In this case there is no need to load some standard numerical optimisation algorithm into the BBC's RAM, which is a difficult task due to the limited memory capacity of the microcomputer.

Local optimization problem 1:

$$\min_{c_{11}, c_{12}, u_{11}} \{(c_{11} - c_{12} + 2u_{11} + \alpha_{11} - 1)^2 + c_{11}^2 + c_{12}^2 - \lambda_{11}c_{11} - \lambda_{12}c_{12} + p_{11}u_{11} - p_{21}(c_{11} - c_{12} + 2u_{11} + \alpha_{11})\},$$

s.t. $|c_{11}| \leq 1$, $|c_{12}| \leq 1$ and $0.8 - c_{12} - 0.6u_{11} \geq 0$.

Local optimization problem 2:

$$\min_{c_{21}, c_{22}, c_{23}, u_{21}} \{2(c_{21} - c_{22} + u_{21} + \alpha_{21} - 2)^2 + (2c_{22} - c_{23} - u_{21} + \alpha_{22} - 3)^2 + c_{21}^2 + c_{22}^2 + c_{23}^2 - \lambda_{21}c_{21} - \lambda_{22}c_{22} - \lambda_{23}c_{23} + p_{21}u_{21} - p_{11}(c_{21} - c_{22} + u_{21} + \alpha_{21})\},$$

s.t. $|c_{2i}| \leq 1$, $i=1,2,3$.

Because both problems are convex the first-order necessary optimality

conditions are also sufficient. The Kuhn-Tucker necessary conditions, therefore, can be employed to solve these optimization problems. There are eighteen possible regions for problem 1 and twenty-seven possible regions for problem 2, depending upon the values of the parameters α_{11}, α_{21} and α_{22}, the modifiers λ_{1i}, i=1,2 and λ_{2j}, j=1,2,3, and the prices p_{11} and p_{21} (see Appendix G). The optimization computation in each local decision unit is reduced to deciding which solution region is applicable according to the values of the given parameters, modifiers and prices.

It is observed that measurement noise is near 1% on average about normal operating conditions. Sources of noise include the low quality of the A/D convertors, the long age of the analogue computer and other unknown factors. However, such a noise level is small in an industrial environment and should not cause difficulties to the algorithms if they are to have any practical usefulness. When the process settling time period is set at 20 seconds, an item of on-line information interchange, begining when the local control units apply the controls to the real process until the start flag of the next action is received, will take about 21.5 seconds. It is observed that it takes approximately 1.5 seconds for the system to complete a typical off-line information interchange (each local unit performs its optimisation calculation, sends its results to the coordinator and then waits until it is informed by the coordinator that all other units have completed their tasks). A notable improvement in data communication speed has been achieved (The previous I-MIC - LSI11/23 system took nearly 10 seconds to complete the same item of off-line information interchange).

9.5 Experimental results and discussion

The derivatives required to compute λ were approximated using formula (4.28) given in Subsection 4.3.2. Suitable values of the gain matrices K_v, K_p were found by experiment as $K_v = 0.4I$ and $K_p = \text{diag}\{0.8, 0.9\}$, where I is an identity matrix. Desired tolerances were chosen as $\beta_v = \beta_p = 0.01$ according to measurement accuracy. Starting from zero initial controls, typical results obtained using the technique formulated in Chapter 7 are shown in TABLE 9.2 where the results are compared with the real optimum values. To compare the efficiency of the single iterative algorithm and the double iterative algorithm, the time required to determine the real optimum solution and the communication requirement of both versions are listed in TABLE 9.3. Computer software can be found in (Chen, Wadhwani and Roberts, 1986).

In practice the technique can be applied such that after the optimal operating condition has been found, the plant is maintained at this condition and the control system is switched into a 'supervising mode', ready to track the new optimum if the process conditions are changed. Fig.9.4. demonstrates this adaptability, where at time instant 0, a disturbance occurred and real subsystem 1 became:

$$y_{11}^{*} = 1.4c_{11} - 0.6c_{12} + 1.8u_{11} + 1.0.$$

Because of the interconnection all real measurements were affected. A sudden deterioration in the performance was reported to the host computer who restarted the procedure. It can be seen from Fig.9.4. that, in this case, the DIA agian took only half of the time required by the SIA to converge to the new optimum.

To study the behaviour of the algorithms under a stochastic situation additional random noise sources, ranging from -0.05 to 0.05, generated using the BBC function RND were added to all real measurements. The given noise was about 5% of the normal operating conditions. To see how the differentiation amplifies measurement errors let the minimum perturbation on controller set points be 0.06; using (4.28) to calculate the derivatives under such a noise condition the maximum error would approximately be 1.7, which is about 150% of the average real derivative. To enable the application of the DIA, the inner loop tolerance was increased to 0.06 and a maximum limitation of 20 iterations was imposed on the inner loop. Fig.9.5. and Fig.9.6. illustrate the serious effect of the additional noise on real system performance during the course of the iterations and the improvement achieved by the use of the smoothing technique (4.31) proposed in Subsection 4.3.3. For the example tested, the DIA seemed to yield more robust results.

9.6 Some concluding remarks

Real-time simulation results demonstrate that both versions of the hierarchical ISOPE technique can be used to overcome model-reality differences and to track slowly changing optimal operating conditions in on-line application. The aim of the DIA is to reduce on-line information interchange and hence, to reduce the time required for determining the optimum steady state control, even at the cost of increasing off-line information exchange. This feature is well demonstrated by the results shown in TABLE 9.3.

The DIA is particularly desirable for processes which have long settling time periods. While the real process settling time period is often beyond our control using more powerful computers at the local level and increasing the efficiency of the communication link will reduce the time required for off-line information interchange, which also favours the DIA.

As with the majority of decentralised methods, to guarantee good performance, the hierarchical ISOPE technique requires synchronised operation. The requirement to measure first-order derivatives of real outputs imposes an important practical limitation to the technique. To enable the application to stochastic processes, further research is needed to investigate the possibility of incorporating filter theory with the hierarchical ISOPE technique.

TABLE 9.1 D/A channel control using BBC user port

bit 7 6 5 4	data bits 0 to 3 assigned to
0 0 0 0	lower half of the register of D/A channel 0
0 0 0 1	upper half of the register of D/A channel 0
0 0 1 0	lower half of the register of D/A channel 1
0 0 1 1	upper half of the register of D/A channel 1
0 1 0 0	lower half of the register of D/A channel 2
0 1 0 1	upper half of the register of D/A channel 2

All bits of the register at &FE62 must be set to 1.

Upper half of the register at &FE6C must be assigned with binary 1010.

TABLE 9.2 Comparison of results

	c_{11}	c_{12}	c_{21}	c_{22}	c_{23}	$Q(c,y^*)$
real optimum	-0.72	0.12	0.90	1.00	-0.83	5.93
SIA	-0.72	0.12	0.89	1.00	-0.85	5.93
DIA	-0.71	0.12	0.90	1.00	-0.83	5.93

TABLE 9.3 Comparison of efficiency

algorithm	iterations of modifiers	optimisation iterations	set point changes	total information interchanges	computation time(minutes)
SIA	23	24	138	162	50.4
DIA	11	113	66	179	27.2

Plant settling time period = 20 seconds.

θ: COORDINATION ACTION

i: LOCAL INFORMATION

m: MEASUREMENT FEEDBACK

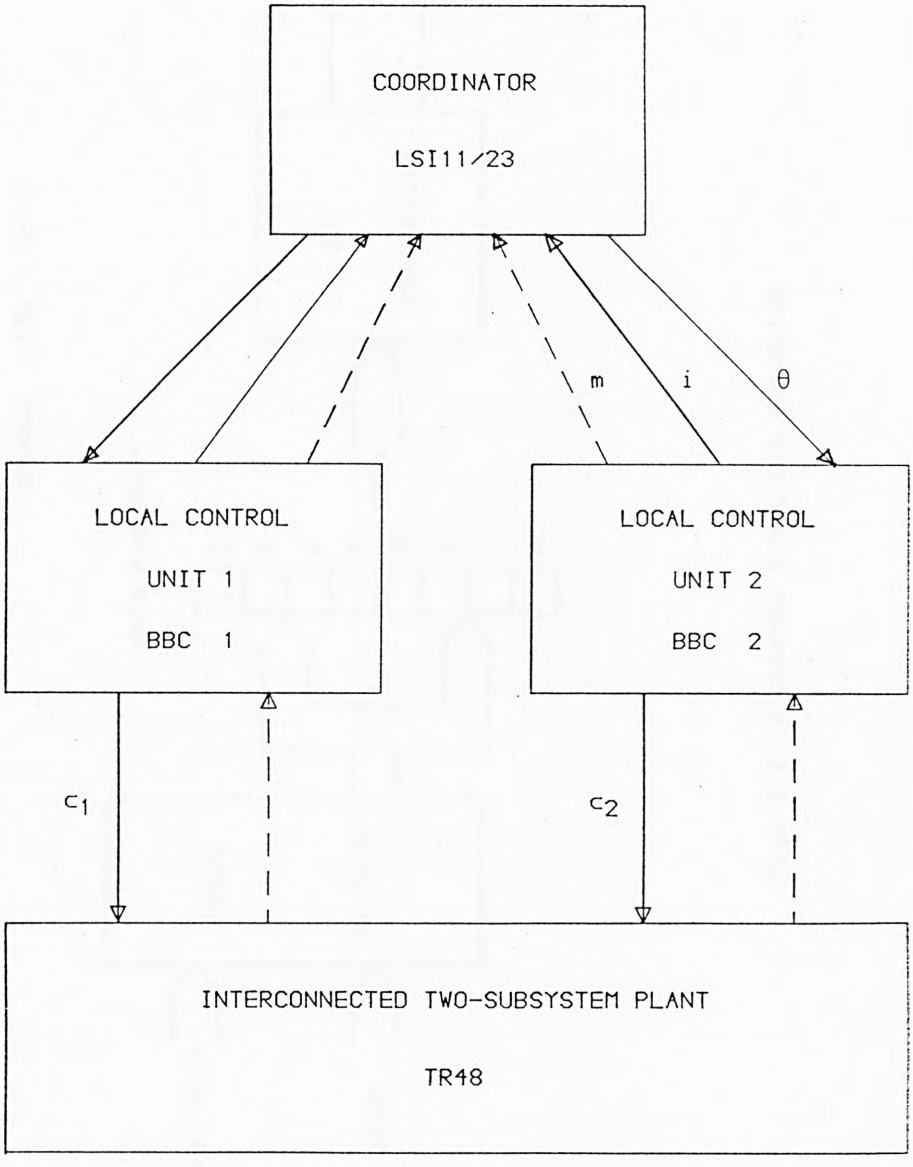


Fig. 9.1. Configuration for analogue computer simulation

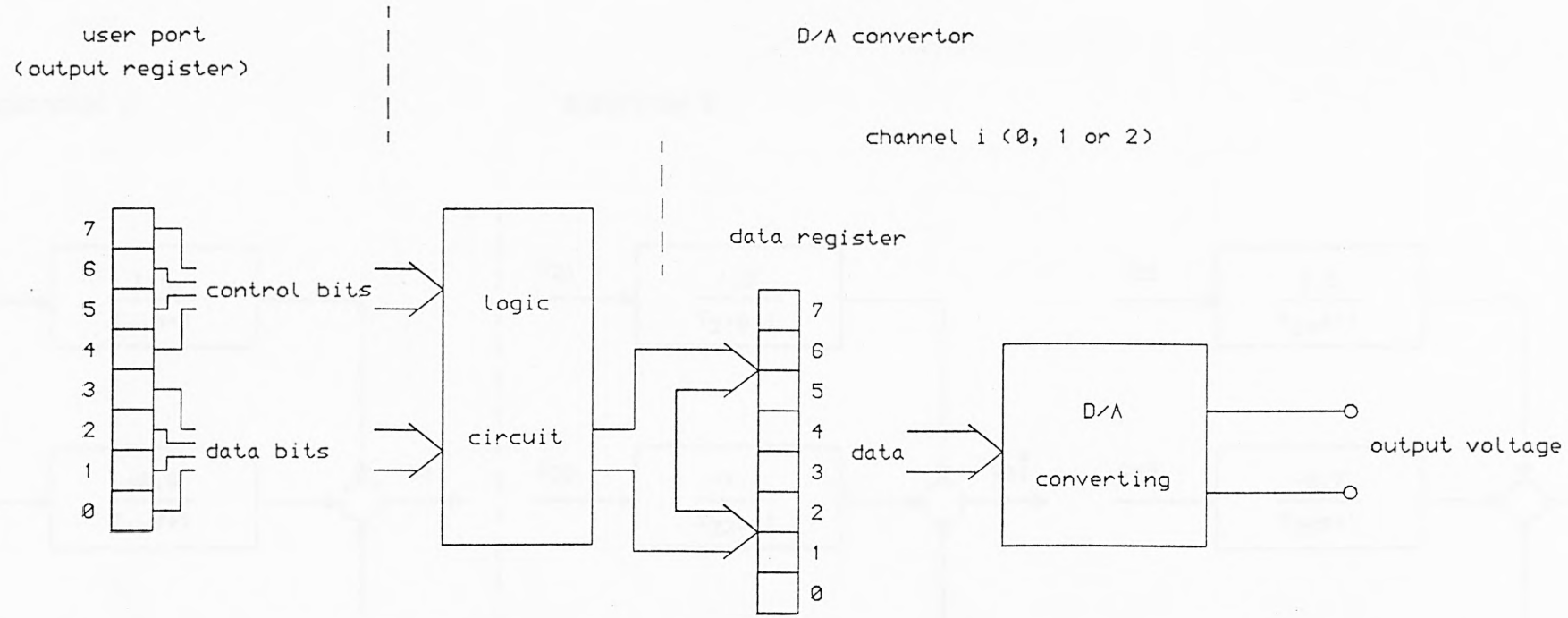


Fig. 9.2. 3 channel 8-bit D/A convertor

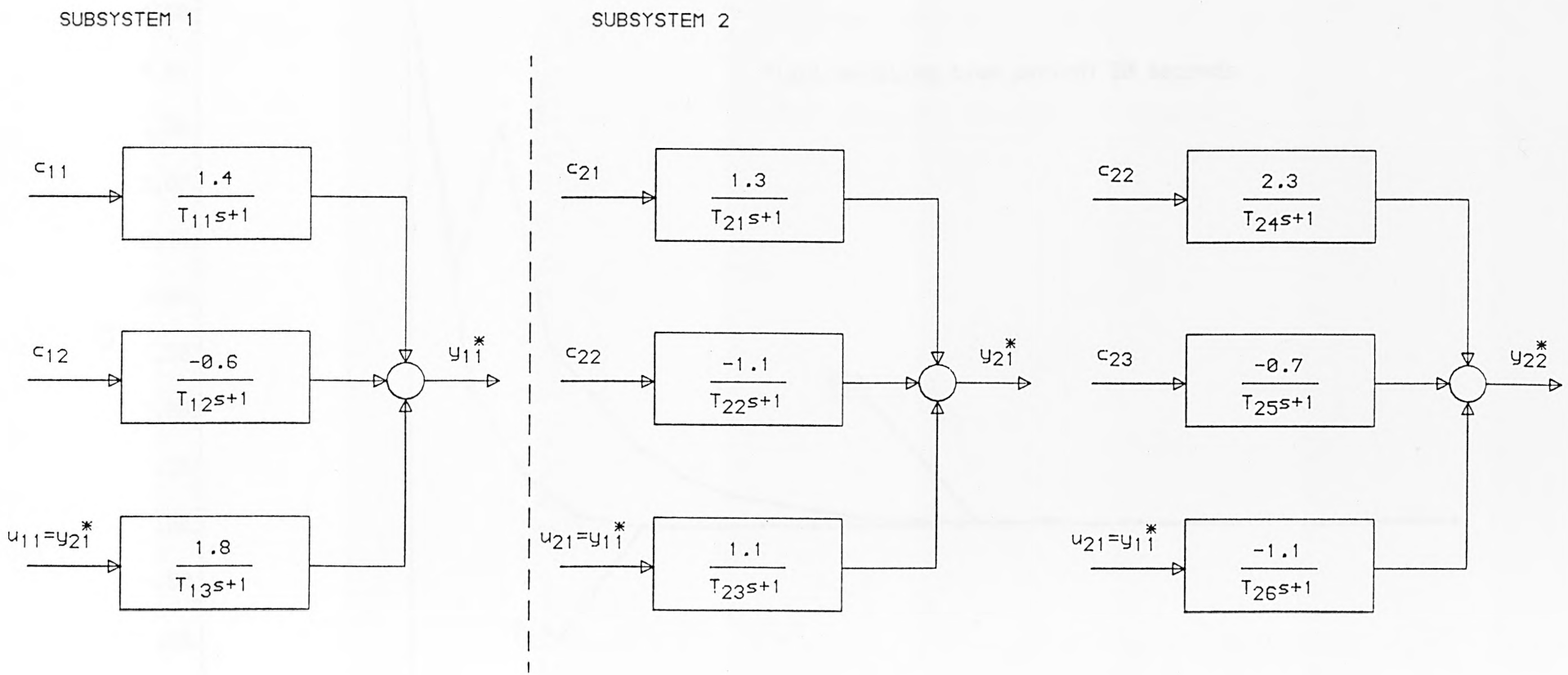


Fig. 9.3. Interconnected two-subsystem plant

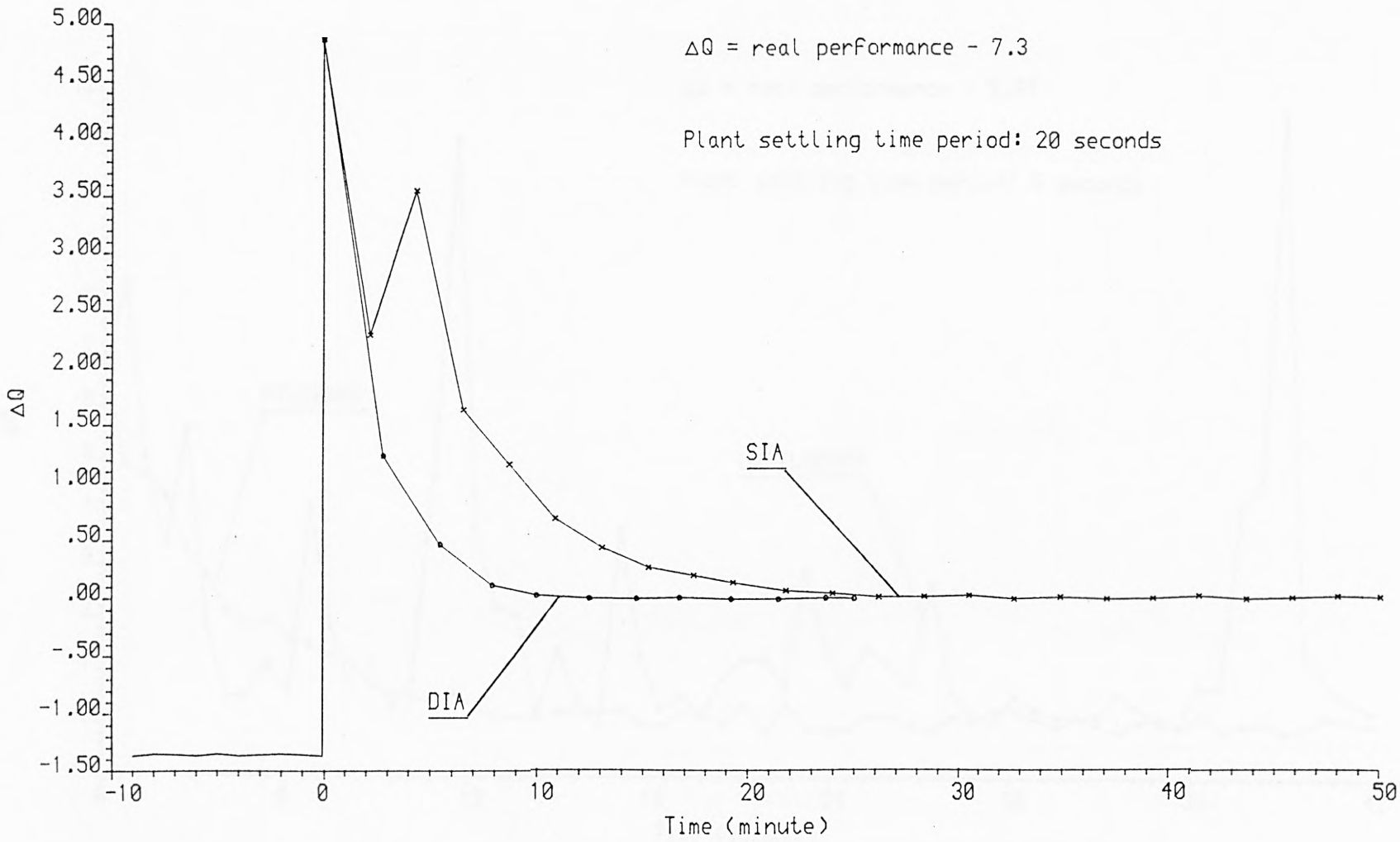


Fig. 9.4. Adaptability of SIA and DIA

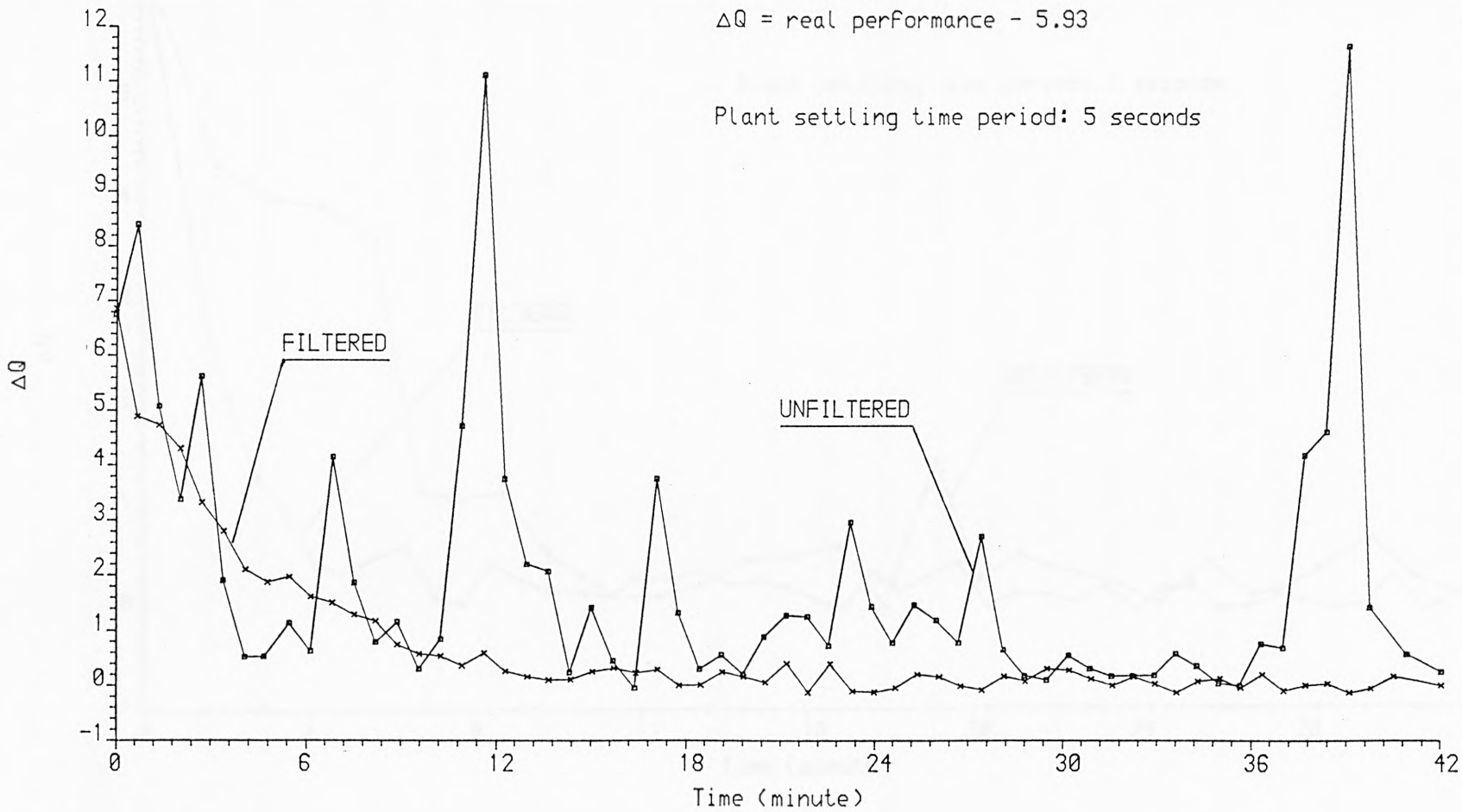


Fig. 9.5. Performance of SIA in stochastic situation

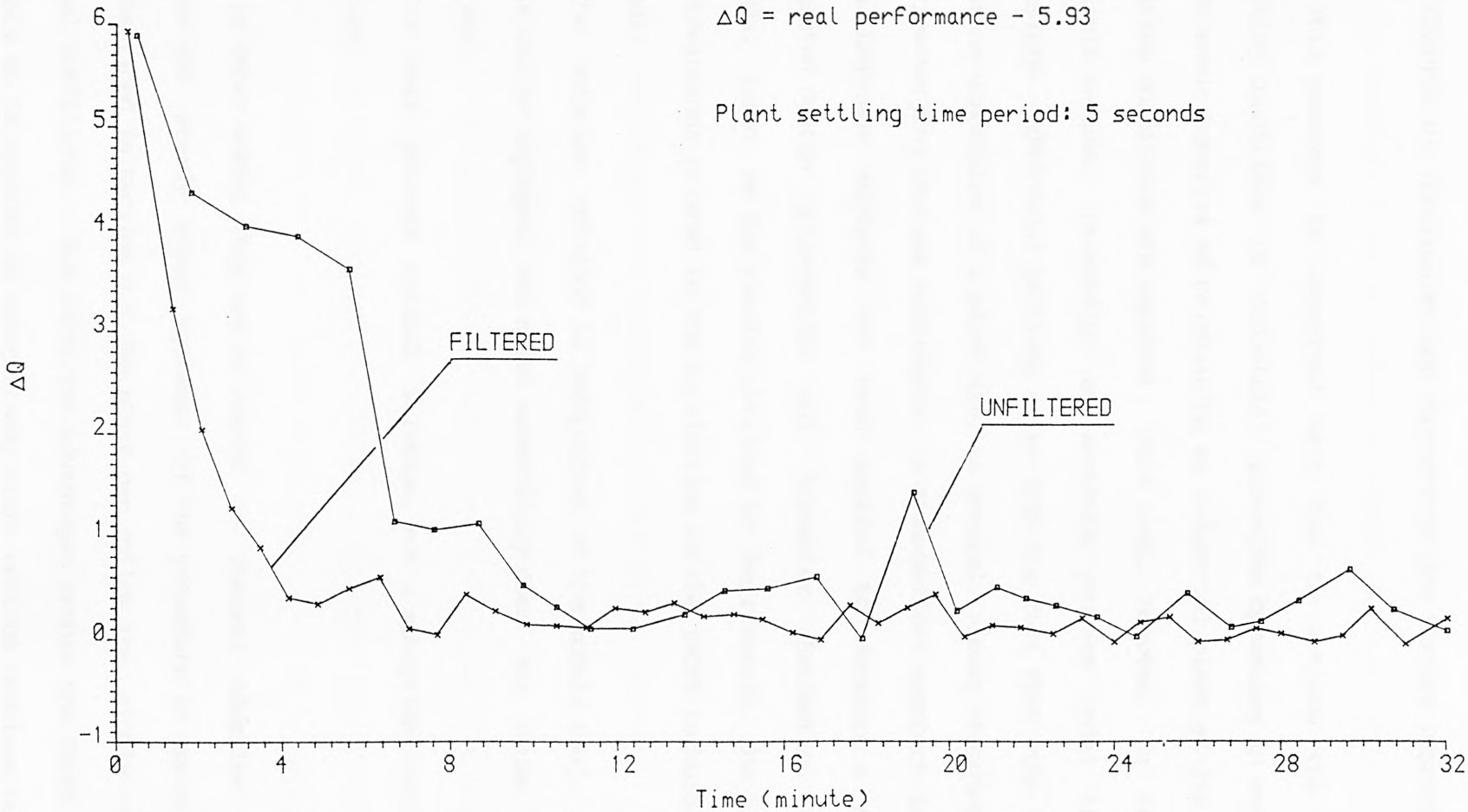


Fig. 9.6. Performance of DIA in stochastic situation

CHAPTER 10. CONCLUSIONS AND SUGGESTIONS FOR FURTHER RESEARCH

This research is concerned with how to achieve the optimal operating conditions of industrial processes operating in real time. The economic benefits of maintaining an industrial plant at its optimal operating conditions are apparent. This task, however, is extremely difficult because, primarily, an accurate process model is often impossible. Additional problems arise from the fact that the optimal operating conditions of a plant are, in general, slowly changing due to the time-varying process conditions. A systematical approach involving normal Lagrange analysis has been applied to develop a class of Integrated System Optimisation and Parameter Estimation (ISOPE) methods. Based on the results obtained in the research, the following two conclusions related to the application of the ISOPE techniques are obvious:

(1) The solution obtained is independent of the models used. Simpler models can be employed, and model uncertainty does not cause serious problems.

(2) The real process optimal solution, not a suboptimal one, can be attained.

In other words, they can be viewed as optimal adaptive control schemes for steady state systems. If the procedure is operated in a way described in Section 9.5 the plant can follow the slowly changing optimal conditions. The above two advantages enable the ISOPE methods not only to be applied to solve steady state control problems but also to be used to solve complex non-convex mathematical programming problems. Given a complex non-convex mathematical programming problem

involving highly nonlinear equality constraints, we may consider these equality constraints as 'real process equations' and construct some simpler linear equations as their 'models'. Then the ISOPE methods are readily applicable to solve the problem iteratively. In some cases this may be an easier and faster way to obtain a solution than solving the original problem directly. In this application, there exists no uncertainty. Even the information of derivatives could be perfect. General convergence conditions and the rate of convergence can be derived, and some advanced iterative strategies, instead of a fixed gain parameter, can be chosen. Further research is suggested to investigate using the modified two-step algorithm as a mathematical programming method. Although the optimality of the ISOPE techniques, in general, is in a local sense, simulation involving non-convex problems has shown that, provided that the convergence can be achieved, the ISOPE techniques tend to converge to the global optimum.

It should be noted that the modified two-step algorithm and its extensions require knowledge of real output derivatives with respect to the inputs during the course of the iterations. This imposes an important practical limitation. These algorithms can obtain good results provided that the measurement noise level is tolerable and a deterministic approach is feasible. In this research, it has been shown that averaging output measurements and using simple filter techniques to filter the modifier vector can significantly improve the performance of the modified two-step algorithm under noisy measurement situations. Further research is suggested to investigate how to incorporate filter theory with the ISOPE methods in order to extend their application to stochastic processes.

Using a stochastic approach to incorporate system optimisation and parameter estimation has been studied in the Control Engineering Centre of The City University. For the linear quadratic problem with unknown parameters under a classical information structure, a stochastic modified two-step algorithm has been formulated which gives excellent results in computer simulation (Han and co-workers, 1986). However, attempts to extend the idea to nonlinear stochastic processes will meet enormous difficulties and may not be possible.

An algorithm has been proposed and studied in the research which is more robust to measurement noise because it does not require the real measurement derivatives. This is achieved not at the cost of the optimality but at the cost of introducing an additional hierarchical level which, in this situation, results in a substantial cost of computational efficiency. Although, like the modified two-step algorithm, it is originally derived by analyzing the Kuhn-Tucker necessary optimal conditions the algorithm seems to belong to a special case of two-level decision making problems. This two-level type algorithm has been observed to perform well when using coordinate descent techniques, such as Powell's conjugate direction method, to solve the supremal problem. However, the speed of convergence still needs to be improved so that it can meet the requirements of real-time application. Further research can be conducted in this direction to investigate more powerful techniques for solving the supremal problem. More theoretical study is also required to analyze the mathematical properties of the algorithm.

In this research the modified two-step technique has been extended

for application to interconnected large scale systems by incoporating a two-level hierarchical structure employing the price correction mechanism. Because there is no need to interchange the information within the local level the hierarchical ISOPE technique is particularly suitable for implementation using a distributed computer network connected as a star structure. In the research considerable emphasis has been given to information exchange requirements between the hierarchical units with real time application in mind. A double iterative coordination strategy has been shown to have important practical advantages under real time operation.

In this thesis normal Lagrange analysis is used as a basic tool to develop on-line ISOPE techniques. It is suggested that the augmented Lagrangian approach can be employed in a similar manner to derive a group of augmented ISOPE algorithms. It is thought that two advantages can be gained by employing augmented Lagrange analysis. Firstly, the resulted algorithms will have faster convergence rate. Secondly, the applicability of the algorithms will be wider than that of the algorithms derived from normal Lagrange analysis. Because of the similarities between these two mathematical tools the results reported in this thesis will benefit the development of the algorithms based on augmented Lagrange analysis, which is currently being conducted in the Control Engineering Centre of The City University. Further efforts can be directed to investigate practical implementation aspects, to compare them with normal ISOPE methods and to study the benefits obtained using this new approach.

In this research we have restricted our interest to steady state

optimisation. Further work is suggested to consider how to extend the ISOPE methods to dynamic processes. For general dynamic processes it is not clear how this can be achieved, and the task may be impossible to accomplish. However, a successful extension of the modified two-step algorithm to on-line control of a pilot scale travelling load furnace, which is a batch dynamic plant, has been performed (Stevenson, 1985). This suggests that at least it is feasible to apply the ISOPE methods to batch type dynamic processes. Extending the results of Han and co-workers (1986) to the stochastic linear quadratic dynamic problem with unknown parameters is also feasible.

The final concern is the application of the ISOPE techniques to industry. These techniques are quite sophisticated. Many practical implementation aspects need further investigation. People from industry still need to be convinced that the application of ISOPE methods is feasible, that it can bring them profits and that the techniques are reliable. Some computer software packages have to be developed which can conveniently be used by the engineers working in industry.

REFERENCES

[1] Avriel, M.(1976). Nonlinear Programming Analysis and Methods. Prentice-Hall, Inc., Englewood Cliffs, N.J..

[2] Bertsekas, D.P. (1982). Constrained Optimization and Lagrange Multiplier Methods. Academic Press, New York.

[3] Bray, A.C., A.C. Dickens, and M.A. Holmes (1983). The Advanced User Guide for the BBC Microcomputer. The Cambridge Microcomputer Centre.

[4] Brdys, M.(1983). Hierarchical optimising control of steady-state large scale systems under model-reality differences of mixed type-- a mutually interacting approach. Proc. of the 3rd IFAC Symposium on Large Scale System: Theory and Application, Warsaw, Poland, pp49-57.

[5] Brdys, M., S. Chen, and P.D. Roberts (1984). An extension to the modified two-step algorithm for steady state system optimisation and parameter estimation. Research Memorandum CEC/MB/SC/PDR-14, Control Eng. Centre, The City University, London. (To be published in Int. J. Systems Sci.).

[6] Brdys, M., W. Findeisen, and P. Tatjewski (1980). Hierarchical control for systems operating in steady-state. Large Scale Systems, Vol.1, pp193-213.

[7] Brdys, M., and P.D. Roberts (1984a). Optimal structures for steady-state adaptive optimizing control of large scale industrial processes. Research Memorandum CEC/MB-PDR/1, Control Eng. Centre, The City University, London.

[8] Brdys, M., and P.D. Roberts (1984b). Convergence and optimality of modified two-step algorithm for integrated system optimization and parameter estimation. Research Memorandum CEC/MB-PDR/12, Control Eng.

Centre, The City University, London.

[9] Brdys, M., and P.D. Roberts (1985). Hierarchical single and double iterative techniques for optimizing control of large scale industrial processes; derivation, applicability and convergence. IEE Conference, Control 85, Publication No.252, pp89-94.

[10] Chen, S., M. Brdys, and P.D. Roberts (1985). An integrated system optimisation and parameter estimation technique for hierarchical control of steady state systems. Research Memorandum CEC/SC-BM-PDR/19, Control Eng. Centre, The City University, London. (To be published in Int. J. Systems Sci.).

[11] Chen, S., P.D. Roberts, and M. Brdys (1984). Comparison of several on-line integrated system optimisation and parameter estimation methods for steady state systems. Research Memorandum CEC/SC-PDR-MB/15, Control Eng. Centre, The City University, London.

[12] Chen, S., P.D. Roberts, and D.S. Wadhwani (1985). Integrated system optimisation and parameter estimation technique using a distributed hierarchical computer system. Presented at the 1st European Workshop on Parallel Processing Techniques for Simulation, UMIST, Machester, England, Oct.28-29, 1985.

[13] Chen, S., D.S. Wadhwani, and P.D. Roberts (1985). Asynchronous serial data communication between a BBC microcomputer and a DEC LSI11/23 minicomputer. Research Memorandum CEC/SC/DSW/PDR-26, Control Eng. Centre, The City University, London.

[14] Chen, S., D.S. Wadhwani, and P.D. Roberts (1986). Hierarchical integrated system optimisation and parameter estimation using a microcomputer based system. Research Memorandum CEC/SC-DSW-PDR/29, Control Eng. Centre, The City University, London.

[15] Coll, J., and D. Allen (ed.) (1984). The BBC Microcomputer User

Guide. British Broadcasting Corporation.

[16] Durbeck, R.C.(1965). Principles for Simplification of Optimising Control Models. Ph.D. Thesis, Case Western Reserve University, Cleveland, Ohio, U.S.A..

[17] Ellis, J.E., and P.D. Roberts (1981). Simple model for integrated optimization and parameter estimation. Int. J. Systems Sci., Vol.12, No.4, pp455-472.

[18] Ellis, J.E., and P.D. Roberts (1982). Measurement and modelling trade-off for integrated system optimization and parameter estimation. Large Scale Systems, Vol.3. pp191-204.

[19] Ellis, J.E., and P.D. Roberts (1985). On the practical viability of integrated system optimisation and parameter estimation. IEE International Conference 'Control 85', University of Cambridge, England, Publication No.252, pp281-285.

[20] Findeisen, W., F.N. Bailey, M. Brdyś, K. Malinowski, P. Tatjewski, and A. Woźniak (1980). Control and Coordination in Hierarchical Systems. Wiley, London.

[21] Foord, A.G.(1974). On-line Optimisation of a Petrochemical Complex. Ph.D. Thesis, University of Cambridge, England.

[22] Greenberg, H.J., and W.P. Pierskalla (1972). Extensions of the Evans-Gould stability theorems for mathematical programs. Operations Research, Vol.20, pp143-153.

[23] Haimes, Y.Y., and D.D. Wismer (1972). A computational approach to the combined problem of optimization and parameter identification. Automatica, Vol.8, pp337-347.

[24] Han, C., J.E. Ellis, S. Chen, and P.D. Roberts (1986). An integrated algorithm for stochastic steady state system optimisation and parameter estimation. Research Memorandum, Control Eng. Centre,

The City University, London.

[25] Hogan, W.W.(1973). Point-to-set maps in mathematical programming. SIAM Review, Vol.15, No.3, pp591-603.

[26] Hutson, V., and J.S. Pym (1980). Applications of Functional Analysis and Operator Theory. Academic Press, London.

[27] James, M.(1983). The BBC Micro: an expert guide. Granada.

[28] Kantorovich, L.V., and G.P. Akilov (1982). Functional Analysis. (Second edition) Pergamon Press, Oxford.

[29] Lasdon, L.S.(1970). Optimization Theory for Large Systems. MacMillan, London.

[30] Lefkowitz, I.(1966). Multilevel approach applied to control system design. Trans. ASME, Series D, J. of Basic Eng., Vol.88, No.2, pp392-398.

[31] Lefkowitz, I.(1977). Integrated control of industrial systems. Royal Society Philosophical Trans., Series A, Vol.287, pp443-465.

[32] Luenberger, D.G.(1973). Introduction to Linear and Nonlinear Programming. Addison-Wesley, Reading, Massachusetts.

[33] Luenberger, D.G.(1984). Linear and Nonlinear Programming. (Second edition), Addison-Wesley, Reading, Massachusetts.

[34] Mangasarian, O.L.(1965). Pseudo-convex functions. SIAM J. on Control, Series A, Vol.3, No.2, pp281-290.

[35] Michalska, H., J.E. Ellis, and P.D. Roberts (1985). Joint coordination method for the steady-state control of large scale systems. Int. J. Systems Sci., Vol.16, No.5, pp605-618.

[36] Nash, J.C.(1979). Compact Numerical Mehtods for Computers: linear algebra and function minimisation. Adam Hilger Ltd, Bristol.

[37] Nelder, J.A., and R. Mead (1965). A simplex method for function minimization. Computer J., Vol.7, pp308-313.

[38] Ortega, J.M., and W.C. Rheinboldt (1970). Iterative Solution of Nonlinear Equations in Several Variables. Academic Press, New York.

[39] Powell, M.J.D.(1964). An efficient method for finding the minimum of a function of several variables without calculating derivatives. Computer J., Vol.7, pp155-162.

[40] Powell, M.J.D.(1972). Unconstrained minimization algorithms without computation of derivatives. Report T.P.483, A.E.R.A., Harwell, U.K.

[41] Roberts, P.D.(1977a). Multilevel approaches to the combined problem of system optimization and parameter identification. Int. J. Systems Sci., Vol.8, No.3, pp273-299.

[42] Roberts, P.D.(1977b). Integrated system optimisation and parameter estimation of a chemical reactor using a multilevel approach. Preprints of the 5th IFAC/IFIP Conference on Digital Computer Applications to Process Control, Hague, Netherlands, pp551-557.

[43] Roberts, P.D.(1978). Algorithms for integrated system optimisation and parameter estimation. Electronics Letters, Vol.14, No.6, pp196-197.

[44] Roberts, P.D.(1979). An algorithm for steady-state system optimization and parameter estimation. Int. J. Systems Sci., Vol.10, No.7, pp719-734.

[45] Roberts, P.D., and J.E. Ellis (1981). Refinements to an algorithm for combined system optimisation and parameter estimation. Proc. of the 1981 UKSC Conference on Computer Simulation, Harrogate, England.

[46] Roberts, P.D., C.W. Li, I.A. Stevenson, and D.S. Wadhwani (1984). On-line distributed hierarchical control and optimisation of large scale processes using a micro-computer based system. Preprints

of the 1st European Workshop on the "Real Time Control of Large Scale Systems", University of Patras, Greece, pp432-441.

[47] Roberts, P.D., and T.W.C. Williams, (1981). On an algorithm for combined system optimisation and parameter estimation. Automatica, Vol.17, pp199-209.

[48] Rosenbrock, H.H.(1974). Computer Aided Control System Design. Academic Press, New York.

[49] Shao, F.Q., and P.D. Roberts (1983). A price correction mechanism with global feedback for hierarchical control of steady state systems. Large Scale Systems, Vol.4, pp67-80.

[50] Stevenson, I.A.(1984). Recent improvements to the distributed computer control system and a survey on-line discrete-time parameter estimation algorithms for use on the travelling-load furnace. Research Memorandum CEC/IAS/5, Control Eng. Centre, The City University, London.

[51] Stevenson, I.A.(1985). On-line Adaptive Computer Control of a Travelling Load Furnace. Ph.D. Thesis, Control Eng. Centre, The City University, London.

[52] Stevenson, I.A., M. Brdys, and P.D. Roberts (1985). Integrated system optimisation and parameter estimation for travelling load furnace. Preprints of the 7th IFAC/IFORS Symposium on Identification and System Parameter Estimation, University of York, England, pp1641-1646.

[53] Tatjewski, P., and M. Cygler (1981). Completely decentralized output control based on an approximate mathematical model. Large Scale Systems, Vol.2, pp243-255.

[54] Youle, P.V., and L.A. Duncanson (1970). On-line control of olefine Plant. Chemical and Process Eng., Vol.51, pp49-52.

[55] Zainol Anuar, B.M.S.(1985). Modelling and Control of Industrial Robots. Ph.D. Thesis, Control Eng. Centre, The City University, London.

[56] Zangwill, W.I.(1967). Minimizing a function without calculating derivatives. Computer J., Vol.10, pp293-296.

[57] Zangwill, W.I.(1969). Nonlinear Programming A Unified Approach. Prentice-Hall, Inc., Englewood Cliffs, N.J..

[58] Digital Equipment Corporation. Microcomputer Interfaces Handbook.

[59] Digital Equipment Corporation. RT-11 Advanced Programmer's Guide.

[60] NAG FORTRAN Library Manual, Mark10, 1983.

[61] S & H Computer Systems Inc. TSX-PLUS FORTRAN Library.

[62] S & H Computer Systems Inc. TSX-PLUS Manual.

BIBLIOGRAPHY

[1] Abatzoglou, T., and B. O'Donnell (1982). Minimization by coordinate descent. J. Optz. Theory and Applications, Vol.36, No.2, pp163-174.

[2] Baudet, G.M.(1978). Asynchronous iterative methods for multiprocessors. J. of the Association for Computing Machinery, Vol.25, No.2, pp226-244.

[3] Bertsekas, D.P.(1979). Convexification procedure and decomposition methods for nonconvex optimization problems. J. Optz. Theory and Applications, Vol.29, No.2, pp169-197.

[4] Bibbero, R.J., and D.M. Stern (1982). Microprocessor Systems: Interfacing and Applications. John Wiley and Sons, Inc., New York.

[5] Brdys, M., P. Michalak, and B. Ulanicki (1982). Optimizing control of large scale systems under time-varying disturbances by price mechanism with local feedback. Large Scale Systems, Vol.3, pp123-142.

[6] Brent, R.P.(1973). Algorithms for Minimization without Derivatives. Prentice-Hall, Inc., Englewood Cliffs, N.J..

[7] Cohen, G.(1978). Optimization by decomposition and coordination: a unified approach. IEEE Trans. on Automatic Control, Vol.AC-23, No.2, pp222-232.

[8] Cohen, G.(1980). Auxiliary problem principle and decomposition of optimization problems. J. Optz. Theory and Applications, Vol.32, No.3, pp277-305.

[9] Ellis, J.E., H. Michalska, and P.D. Roberts (1984). An integrated system optimisation and parameter estimation technique for the steady-state control of large scale systems. Research Memorandum CEC/JEE-HM-PDR/3, Control Eng. Centre, The City University, London.

[10] Findeisen, W., M. Brdys, K. Malinowski, P. Tatjewski, and A. Wozniak (1978). On-line hierarchical control for steady-state systems. IEEE Trans. on Automatic Control, Vol.AC-23, No.2, pp189-208.

[11] Geoffrion, A.M., and W.W. Hogan (1972). Coordination of two-level organization with multiple objectives. In: Techniques of Optimization, ed. by A.V. Balakrishnan, pp455-466, Academic Press, New York.

[12] Gill, P.E., W. Murray, and M.H. Wright (1981). Practical Optimization. Academic Press.

[13] Guddat, J., and D. Klatte (1976). Qualitative stability in non-linear parametric optimization. Proc. of IX International Symposium on Mathematical Programming, Budapest, pp217-229.

[14] Hanus, R., I. Cohen, E. Le Moine, and F. Tolfo (1983). The simplex technique as a general optimization method. Proc. of the 4th IFAC Workshop on Applications of Nonlinear Programming to Optimization and Control, San Francisco, U.S.A., pp73-79.

[15] Held, M., P. Wolfe, and H.P. Crowder (1974). Validation of subgradient optimization. Mathematical Programming, Vol.6, No.1, pp62-88.

[16] Hogan, W.W.(1973) Directional derivatives for extremal-value functions with applications to the complete convex case. Operations Research, Vol.21, pp188-209.

[17] Huard, P.(1975). Optimization algorithms and point-to-set maps. Mathematical Programming, Vol.8, No.3, pp308-331.

[18] Lemarechal, C., and R. Mifflin (eds.) (1978). Nonsmooth Optimization. Pergamon Press.

[19] Leventhal, L.A., and W. Saville (1982). 6502 Assembly Language Subroutines. McGraw-Hill, Inc., Berkeley, California, U.S.A..

[20] Li, R.H., and M.G. Singh (1981). An on-line hierarchical approach to the joint problem of identification and optimization. Large Scale Systems, Vol.2, pp205-216.

[21] Malinowski, K., and A. Ruszczynski (1975). Application of interaction balance method to real process coordination. Control and Cybernetics, Vol.4, No.2, pp15-31.

[22] Mesarovic, M.D., D. Macko, and Y. Takahara (1970). Theory of Hierarchical, Multilevel Systems. Academic Press, New York.

[23] Mihara, M., A. Ogasawara, C. Imamichi, and A. Inamoto (1983). Large scale control system for the most advanced hot strip mill. In: IFAC Distributed Computer Control Systems, ed. by M.G. Rodd, pp45-56.

[24] Nash, P.(ed.) (1981). Systems Modelling and Optimization. IEE Control Eng. Series 16, Peter Peregrinus Ltd.

[25] Ostrowski, A.(1966). Solution of Equations and Systems of Equations. Academic Press, New York.

[26] Rall, L.B.(1969). Computational Solution of Nonlinear Operator Equations. John Wiley and Sons, Inc., New York.

[27] Roberts, P.D., and L. Roussias (1980). Overcoming model uncertainties in optimising supervisory control. Preprints of the 6th IFAC/IFIP Conference on Digital Computer Applications to Process Control, Dusseldorf, F.D. Germany, pp383-388.

[28] Rockafellar, R.T.(1981). The Theory of Subgradients and its Applications to Problems of Optimization. Convex and Nonconvex Functions. Heldermann Verlag, Berlin.

[29] Ruszcynski, A.(1976) Convergence conditions for the interaction balance algorithm based on approximate mathematical model. Control and Cybernetics, Vol.5, No.4, pp27-43.

[30] Sanchez, E., and C. Foulard (1983). Energy saving in crude dis-

tillation process. Large Scale Systems, Vol.4, pp81-90.

[31] Sandell, N.R., P. Varaiya, M. Athans, and M.G. Safonov (1978). Survey of decentralized control methods for large scale systems. IEEE Trans. on Automatic Control, Vol.AC-23, No.2, pp108-128.

[32] Shao, F.Q., and P.D. Roberts (1985). Augmented decentralized control with local feedback for steady-state systems. Int. J. Systems Sci., Vol.16, No.6, pp649-665.

[33] Shimizu, K., and E. Aiyoshi (1980). Necessary conditions for min-max problems and algorithms by a relaxation procedure. IEEE Trans. on Automatic Control, Vol.AC-25, No.1, pp62-66.

[34] Shimizu, K.(1981). Two-level decision problems and their new solution mehtods by a penalty method. Proc. of The 8th Triennial World Congress of IFAC, Kyoto, Japan, pp1303-1308.

[35] Shimizu, K., and E. Aiyoshi (1981a). A new computational method for stackelberg and min-max problems by use of a penalty method. IEEE Trans. on Automatic Control, Vol.AC-26, No.2, pp460-466.

[36] Shimizu, K., and E. Aiyoshi (1981b). Hierarchical multi-objective decision systems for general resoure allocation problems. J. Optz. Theory and Applications, Vol.35, No.4, pp517-533.

[37] Siljak, D.D.(1978). Large Scale Dynamic Systems Stability and Structure. North-Holland, New York.

[38] Siljak, D.D.(1983). Complex dynamic systems: dimensionality, structure, and uncertainty. Large Scale Systems, Vol.4, pp279-294.

[39] Singh, M.G.(1980). Dynamical hierarchical control. (Revised edition) North-Holland.

[40] Singh, M.G.(1981). Decentralised Control. North-Holland.

[41] Singh, M.G., and A. Titli (eds.) (1979). Handbook of large Scale Systems Engineering Applications. North-Holland.

[42] Smart, D.R.(1974). Fixed Point Theorems. Cambridge University Press.

[43] Stephanopoulos, G., and A.W. Westerberg (1975). The use of Hestene's method of multipliers to resolve dual gaps in engineering system optimization. J. Optz. Theory and Applications, Vol.15, No.3, pp285-309.

[44] Stevenson, I.A., D.S. Wadhwani, J.E. Ellis, C.W. Li, and P.D. Roberts (1984). A distributed computer network and its application to systems control. 'Eurocon 84' Computers in Communication and Control, Brighton, England, PPL Conference Publication No.22, pp189-193.

[45] Szymanowski, J., and A. Ruszcyzynski (1979). Convergence analysis for two-level algorithms of mathematical programming. Mathematical Programming Study, Vol.10, pp158-171.

[46] Takahara, Y.(1982). Decomposability conditions for general systems. Large Scale Systems, Vol.3, pp57-65.

[47] Tanino, T., and T. Ogawa (1984). An algorithm for solving two-level convex optimization problems. Int. J. Systems Sci., Vol.15, No.2, pp163-174.

[48] Tatjewski, P.(1983). On-line steady-state control of large scale nonlinear interconnected systems using augmented interaction balance method with feedback. Proc. of the 3rd IFAC Symposium on Large Scale System: Theory and Application, Warsaw, Poland, pp526-531.

[49] Wan Bai-Wu, and P.D. Roberts (1981). Some improvements in techniques for hierarchical control of steady-state systems. Research Memorandum DSS/B-WW-PDR/229, Department of Systems Science, The City University, London.

[50] Wilson, I.D.(1979a). Foundations of hierarchical control. Int. J. of Control, Vol.29, No.6, pp899-933.

[51] Wilson, I.D.(1979b). Three applications of decomposition method for designing hierarchical control systems. Int. J. of Control, Vol.29, No.6, pp935-947.

[52] Zoutendijk, G.(1960). Methods of Feasible Directions. Elsevier, Amsterdam.

Appendix A Fixed point theorems

A1. Schauder's theroem (e.g. Kantorovich and Akilov, 1982)

A continuous mapping that maps a compact convex set in a Banach space into itself has a fixed point.

A2. (e.g. Hutson and Pym, 1980; Kantorovich and Akilov, 1982)

Let B and C be Banach spaces. Take closed $D \subset B$, and an arbitrary $E \subset C$, and assume $A: D \times E \to D$ is continuous. Suppose that there exists a $q < 1$ such that $A(.,g)$ is a contraction with Lipschitz constant q independent on g. Then there exists a unique fixed point $f(g)$ of $A(.,g)$. Moreover, $f(g)$ is continuous on E.

Appendix B Mathematical preliminaries for Chapter 5

Definition B1. (point-to-set mapping)

A mapping $F: X \to Y$ is said to be a point-to-set mapping if for any $x \in X$, $F(x)$ is a set in Y.

Definition B2. (open mapping)

A point-to-set mapping $F: X \to Y$ is open at a point $x \in X$ if $\{x^k\} \subset X$, $x^k \to x$, and $y \in F(x)$ imply the existence of an integer m and a sequence $\{y^k\} \subset Y$ such that $y^k \in F(x^k)$ for $k \geq m$ and $y^k \to y$.

Definition B3. (closed mapping)

A point-to-set mapping $F: X \to Y$ is closed at a point $x \in X$ if $\{x^k\} \subset X$, $x^k \to x$, $y^k \in F(x^k)$, and $y^k \to y$ imply that $y \in F(x)$.

Definition B4. (fixed point)

A fixed point for a point-to-set mapping $F: X \to Y$ is a point $x \in X$ such that $x \in F(x)$, where X and Y are subsets in a same space.

Definition B5. (composition)

Let $F: X \to Y$ and $G: Y \to Z$ be point-to-set mappings. The composite mapping $H = GF$ is defined as the point-to-set mapping $H: X \to Z$ with

$$H(x) \triangleq \bigcup_{y \in F(x)} G(y).$$

Definition B6. (Cartesian product)

Let $H_i: X \to Y_i$, $i=1,\ldots,n$ be point-to-set mappings. The Cartesian product $H = H_1 \times \ldots \times H_n$ is defined as the point-to-set mapping

$H: X \to Y_1 x \ldots x Y_n$ with $H(x) \overset{\Delta}{=} \{(y_1,\ldots,y_n): y_i \in H_i(x),\ i=1,\ldots,n\}$.

Definition B7. (algorithm)

An algorithm is an iterative process consisting of a sequence of point-to-set mappings $\Phi_k: X \to X$. Given a point $x^0 \in X$, a sequence $\{x^k\} \subset X$ is generated by use of the recursion $x^{k+1} \in \Phi_k(x^k)$. The mappings Φ_k are called the algorithmic mappings. A special case is $\Phi_k = \Phi$ for all k. Remark: Generally, the mappings Φ_k depend on the iteration number k as well as previous points $x^0, \ldots, x^{k-1}$. Such mappings are termed as nonautonomous mappings. Algorithmic mapping Φ does not depend on previous points $x^0,\ldots,x^{k-1}$. Such a mapping is referred to as an autonomous mapping.

Definition B8. (monotone)

A mapping $F: D \subset X \to Y$ is monotone on $D_0 \subset D$ if

$$[F(x) - F(y)]^T(x - y) \geq 0, \quad \forall x,y \in D_0. \tag{B.1}$$

F is strictly monotone on D_0 if strict inequality holds in (B.1) whenever $x \neq y$ and uniformly monotone if there is a $r > 0$ such that

$$[F(x) - F(y)]^T(x - y) \geq r(x - y)^T(x - y), \quad \forall x,y \in D_0.$$

Definition B9. (upper semicontinuous)

A real-valued function $f:X \to R$ is upper semicontinuous at $x \in X$ if $x^k \to x$, $k \to \infty$ implies $\overline{\lim_{k \to \infty}} f(x^k) \leq f(x)$, where

$$\overline{\lim_{k \to \infty}} f(x^k) \overset{\Delta}{=} \limsup_{k \to \infty} f(x^k).$$

Definition B10. (pseudo-convex)

Let the real-valued function $h: C \to R$ be differentiable on C. h(c) is

said to be pseudo-convex on C if for any c_1, $c_2 \in C$

$$(c_2 - c_1)^T \nabla h(c_1) \geq 0 \text{ implies } h(c_2) \geq h(c_1)$$

where $\nabla h(c)$ denotes $\dfrac{\partial^T h(c)}{\partial c}$.

Definition B11. (quasi-convex)

Let $h: C \to R$ be a real-valued function and let C be convex. h(c) is said to be quasi-convex on C if for any c_1, $c_2 \in C$

$$h(c_1) \leq h(c_2) \text{ implies } h(\lambda c_1 + (1 - \lambda) c_2) \leq h(c_2)$$

for every λ such that $0 \leq \lambda \leq 1$.

Theorem B1 (Zangwill, 1969; Luenberger, 1973)

Let $F: X \to Y$ and $G: Y \to Z$ be point-to-set mappings. Suppose F is closed at x and G is closed on F(x). Then the composite mapping H=GF is closed at x if any one of the following three conditions holds:

(a) If $x^k \to x$ and $y^k \in F(x^k)$, then there is a y such that, for some subsequence $\{y^{k_i}\}$, $y^{k_i} \to y$.

(b) Y is compact.

(c) F is a point-to-point mapping continuous at x.

Theorem B2 (Zangwill,1969)

Let $F: X \to Y$ and $G: X \to Y$ be point-to-set mappings closed at $x \in X$. Then the sum mapping $H = F + G: X \to Y$ is closed at x if any one of the following three conditions holds:

(a) If $x^k \to x$ and $y^k \in F(x^k)$, then there is a y such that, for some subsequence $\{y^{k_i}\}$, $y^{k_i} \to y$.

(b) Y is compact.

(c) Either F or G is a point-to-point mapping continuous at x.

Theorem B3 (Hogan,1973)

If $H_i: X \to Y_i$, i=1,...,n are closed point-to-set mappings on X. Then the Cartesian product $H_1 x \ldots x H_n: X \to Y_1 x \ldots x Y_n$ is closed on X.

Define a real-valued function $f: Z \times Y \to R$ and a point-to-set mapping $H: Z \to Y$. Let

$$c(z) \overset{\Delta}{=} \inf\{f(z,y): y \in H(z)\} \text{ and } M(z) \overset{\Delta}{=} \{y \in H(z): c(z) = f(z,y)\}.$$

The function c is an infimal value function and the point-to-set mapping M describes its solution set. The following two theorems are due to Hogan (1973).

Theorem B4

If H is open at z and f is upper semicontinuous on z x H(z), then c is upper semicontinuous at z.

Theorem B5

If f is continuous on Z x Y, H is continuous on Z, Z is connected, Y is compact and M is single-valued at z. Then M is continuous at z.

Remarks: The definition of connected is as follows.

Z is connected if for any z_1, $z_2 \in Z$ there exists a polygonal line (a curve obtained by joining a finite number of straight line segments) joint z_1 and z_2 which lies entirely in Z. Thus if Z is convex then Z must be connected. If H(z) is identical to Y for all $z \in Z$, without Z being connected, the assertion of Theorem B5 still holds.

Theorem B6 (Luenberger, 1973)

Let the real-valued function h be convex on a convex set C and assume that h is continuously differentiable. If x^* is a global minimum point

of h over C, then for all $y \in C$

$$h'(x^*)(y - x^*) \geq 0,$$

where $h'(x)$ denotes $\dfrac{\partial h(x)}{\partial x}$.

Theorem B7

Let the real-valued function $h: C \to R$ be differentiable on C and its derivative h' be Lipschitz continuous on C with constant $\rho > 0$, i.e.,

$$\| \nabla h(v) - \nabla h(c) \| \leq \rho \| v - c \| \quad \text{for all } v, c \in C,$$

and let C be convex. Then for all $v, c \in C$ the following holds:

$$h(v) - h(c) \geq h'(v)(v - c) - \frac{\rho}{2}\| v - c \|^2.$$

Proof: Owing to the convexity of C, $c + \alpha(v - c) \in C$ for all $0 \leq \alpha \leq 1$. Hence we have

$$h(v) - h(c) = \int_0^1 h'(c + \alpha(v - c))(v - c)\, d\alpha = h'(v)(v - c) +$$

$$\int_0^1 [h'(c + \alpha(v - c)) - h'(v)](v - c)\, d\alpha .$$

Using Schwarz inequality $a^T b \geq -\|a\| \|b\|$,

$$h(v) - h(c) \geq h'(v)(v - c) - \int_0^1 \| h'(c + \alpha(v - c)) - h'(v) \| \| v - c \|\, d\alpha$$

$$\geq h'(v)(v - c) - \int_0^1 \rho \| (\alpha - 1)(v - c) \| \| v - c \|\, d\alpha = h'(v)(v - c) -$$

$$\int_0^1 \rho|\alpha - 1|\,||v - c||^2 d\alpha = h'(v)(v - c) - \rho\,||v - c||^2 \int_0^1 (1 - \alpha)\, d\alpha$$

$$= h'(v)(v - c) - \frac{\rho}{2}||v - c||^2. \quad \text{Q.E.D.}$$

Zangwill's convergence theorem (Zangwill, 1969; Luenberger, 1973)

Let the point-to-set mapping $\Phi : C \to C$ determine an algorithm that given a point $v^0 \in C$ generates the sequence $\{v^k\}$ by use of the recursion $v^{k+1} \in \Phi(v^k)$. Also let a solution set $\Omega \subset C$ be given. Suppose

(1) All points v^k are in a compact set $X \subset C$.

(2) There is a continuous function $Z: C \to R$ such that

(a) if $v \notin \Omega$, then $Z(c) < Z(v)$ for all $c \in \Phi(v)$.

(b) if $v \in \Omega$, then either the algorithm terminates or $Z(c) \leq Z(v)$ for all $c \in \Phi(v)$.

(3) The mapping Φ is closed at points outside Ω.

Then either the algorithm stops at a solution or the limit of any convergent subsequence is a solution.

Appendix C Analytical solutions of Simple example 1

Model: $y = F(c, \alpha) = -1.5c + \alpha$.

Reality: $y = F^*(c) = c + 1$.

Performance index: $Q(c,y) = c^2 + y^2$. Constraint: $G(c,y) = c + y + 1 \leq 0$.

Let $q(c, \alpha) = c^2 + (-1.5c + \alpha)^2$, $q^*(c) = c^2 + (c + 1)^2$,

$g(c, \alpha) = -0.5c + \alpha + 1$ and $g^*(c) = 2(c + 1)$.

Notice that the real system constraint set is $c \leq -1$.

C1. Real optimum

Consider $L^*(c, \xi) = q^*(c) + \xi\, g^*(c)$. This is a strictly convex problem.

Let $\nabla_c L^* = 0$, $\nabla_\xi L^* \leq 0$, $\xi \nabla_\xi L^* = 0$ and $\xi \geq 0$, i.e.,

$2c + 1 + \xi = 0$, $c \leq -1$, $\xi(c + 1) = 0$ and $\xi \geq 0$.

Suppose $\xi = 0$, i.e., $c_{opt} = -0.5$ but this violates the real constraint $c \leq -1$. Hence,

$c_{opt} = -1$, $\xi_{opt} = 1$ and $Q_{opt} = 1$.

C2. Two-step method

Parameter estimation: Solve $-1.5v + \alpha = v + 1$ we have $\alpha = 2.5v + 1$.

Consider $L_1(c, \bar{\xi}) = q(c, \alpha) + \bar{\xi}\, g(c, \alpha)$. From $\nabla_c L_1 = 0$ we have

$13\bar{c} = 6\alpha + \bar{\xi}$.

Let $\bar{c} = v$ we obtain $13\bar{c} = 6(2.5\bar{c} + 1) + \bar{\xi}$ or

$\bar{c} = -3 - 0.5\bar{\xi}$ and $\bar{\alpha} = -6.5 - 1.25\bar{\xi}$.

Check: $g(\bar{c}, \bar{\alpha}) = -0.5\bar{c} + \bar{\alpha} + 1 = -4 - \bar{\xi} < 0$ for $\bar{\xi} \geq 0$, hence,

$\bar{c} = -3$, $\bar{\xi} = 0$ and $\bar{Q} = 13$.

C3. Previous version of modified two-step method

$$\text{Modifier:}\quad \lambda = \left(\frac{\partial F}{\partial c} - \frac{\partial F^*}{\partial c}\right)\frac{\partial Q}{\partial y}\Bigg|_v = 7.5v - 5\,\alpha.$$

Consider $L_2(c, \hat{\xi}) = q(c, \alpha) - \lambda c + \hat{\xi}\, g(c, \alpha)$. From $\nabla_c L_2 = 0$ we have

$$13\hat{c} = 6\,\alpha + \hat{\xi} + 2\,\lambda.$$

Let $\hat{c} = v$ we obtain $13\hat{c} = 6(2.5\hat{c} + 1) + \hat{\xi} + 2(7.5\hat{c} - 5(2.5\hat{c} + 1))$ or

$$\hat{c} = -0.5 + 0.125\hat{\xi}.$$

But $\hat{\xi} \geq 0$ and $\hat{c}$ violates the real system constraint $c \leq -1$. Hence, the solution is not feasible.

C4. New version of modified two-step method

$$\text{Modifier:}\quad \lambda = \left(\frac{\partial F}{\partial c} - \frac{\partial F^*}{\partial c}\right)\left(\frac{\partial Q}{\partial y} + \frac{\partial G}{\partial y}\,\xi\right)\Bigg|_v = 7.5v - 5\,\alpha - 2.5\,\xi.$$

Solve $\nabla_c L_2 = 0$ we have $13\hat{c} = 6\,\alpha + \hat{\xi} + 2\,\lambda$.

Let $\hat{c} = v$ and $\hat{\xi} = \xi$ we obtain

$$\hat{c} = -0.5 - 0.5\hat{\xi}.$$

Check $g(\hat{c}, \alpha) = 0$ we find $\hat{\xi} = 1$, hence,

$$\hat{c} = c_{opt} = -1,\quad \hat{\xi} = \xi_{opt} = 1 \text{ and } \hat{Q} = Q_{opt} = 1.$$

$\lambda = -2.5$ which is unique.

C5. Two-level type method

Again from $\nabla_c L_2 = 0$ we have

$$13\hat{c} = 6\alpha + \hat{\xi} + 2\lambda.$$

Notice that here λ is an independent variable. Let $\hat{c} = v$ and substitute $\alpha = 2.5v + 1$ into the above equation

$\hat{c} = -3 - \lambda - 0.5\hat{\xi}$. Also $\hat{\alpha} = -6.5 - 2.5\lambda - 1.25\hat{\xi}$ and $g(\hat{c}, \hat{\alpha}) = -4 - 2\lambda - \hat{\xi}$.

a) $g(\hat{c}, \hat{\alpha})$ is not active: $g(\hat{c}, \hat{\alpha}) < 0$ and $\hat{\xi} = 0$.

i.e., for $\lambda > -2$, $\hat{c} = -3 - \lambda$ and $\hat{Q} = 2\lambda^2 + 10\lambda + 13$.

b) $g(\hat{c}, \hat{\alpha})$ is degenerate: $g(\hat{c}, \hat{\alpha}) = 0$ and $\hat{\xi} = 0$.

i.e., for $\lambda = -2$, $\hat{c} = -1$ and $\hat{Q} = 1$.

c) $g(\hat{c}, \hat{\alpha})$ is active: $g(\hat{c}, \hat{\alpha}) = 0$ and $\hat{\xi} > 0$.

i.e., for $\lambda < -2$, $\hat{c} = -1$, $\hat{\xi} = -4 - 2\lambda$ and $\hat{Q} = 1$.

Appendix D Programme listing of simplex algorithm

```
C
      SUBROUTINE MIFSIM(N, X, F, FUN, STEP, ALPHA, BETA, GAMMA, TOL,
     * JCOUNT, INMON, XS)
C
C       EXPLANATION----
C     PURPOSE: Minimizing a function without calculating derivatives
C     (Simplx search method).
C       PARAMETERS----
C     Inputs:
C     N: The number of independent variables. Unchange on exit.
C     STEP: Increment for generatig initial simplex. Unchange on exit.
C     ALPHA: The reflection factor. Unchange on exit.
C     BETA: The contraction factor. Ucchange on exit.
C     GAMMA: The extension fator. Unchange on exit.
C     TOL: The criterion for stopping the search. Denote VL the lowest
C     function value and VH the highest function value in the current
C     simplex. If | VH - VL | < TOL is satisfied the procedure will
C     be terminated. Unchange on exit.
C     INMON: On entry if INMON >0 MONISI will be called, if INMON <0
C     MONISI will not be called. Unchange on exit.
C     Inputs/Outputs:
C     X: On entry X specifies a guess of optimal point. On a success
C     exit X contains the optimal point found.
C     F: On a success exit F contains the function value at the
C     optimal point.
C     JCOUNT: On exit JCOUNT specifies number of calling FUN.
C       WORKING SPACE----
C     XS.
C       ASSOCIAXTED SUBROUTINES----
C     MONISI: Subroutine to present the result during MIFSIM. It must
C     not change any parameter in MIFSIM.
C     FUN: User must supply this subroutine to calculate the function
C     value at any given point and declare it as EXTERNAL. The
C     specification is:
C           SUBROUTINE FUN(N, X, F)
C           INTEGER N
C           DOUBLE PRECISION X(N), F
C
      DOUBLE PRECISION X(N), F, STEP, ALPHA, BETA, GAMMA, TOL,
     * XS(N+1,N+2)
      INTEGER N, JCOUNT, INMON
      DOUBLE PRECISION SIZE, SS, VL, VH, VN, T, DABS, DFLOAT
      INTEGER IC, IL, IN, IH, I, J
      EXTERNAL FUN
C
      IC=N+2
C
C      Initial function evaluation and saving vertex 1
C
      JCOUNT=1
      CALL FUN(N, X, F)
```

```
      IL=1
      XS(N+1, 1)=F
      DO 20 I=1,N
      XS(I,1)=X(I)
20    CONTINUE
C
C       Generate rest of simplex and compute a measure of size
C
      SIZE=0.0
      DO 30 J=2,N+1
      DO 40 I=1,N
      XS(I,J)=X(I)
40    CONTINUE
      XS(J-1,J)=X(J-1)+STEP
      SIZE=SIZE+DABS(STEP)
30    CONTINUE
C--**--
C       9000 IS THE POINT AT WHICH THE ALGORITHM CONTINUES AFTER
C       A GENERAL CONTRACTION OF THE SIMPLEX.
C
C       Generate function values at each vertex
C
9000  SS=SIZE
      DO 50 J=1,N+1
      IF (J.EQ.IL) GO TO 50
      DO 60 I=1,N
      X(I)=XS(I,J)
60    CONTINUE
      JCOUNT=JCOUNT+1
      CALL FUN(N, X, F)
      XS(N+1,J)=F
50    CONTINUE
C--**--
C       14000 IS THE POINT AT WHICH THE ALGORITHM CONTINUES IN
C       ALL INSTANTS EXCEPT A GENERAL CONTRACTION.
C
C       Order of function values
C
14000 IL=1
      IH=1
      IN=1
      VL=XS(N+1,1)
      VH=VL
      VN=VL
      DO 70 J=2,N+1
      T=XS(N+1,J)
      IF (T.GE.VL) GO TO 100
      VL=T
      IL=J
      GO TO 70
100   IF (T.LT.VH) GO TO 200
      IN=IH
      VN=VH
      IH=J
```

```
      VH=T
      GO TO 70
200   IF (T.LT.VN) GO TO 70
      IN=J
      VN=T
70    CONTINUE
C
C       Monitoring
C
      IF(INMON.GT.0) CALL MONISI(JCOUNT, N, VL, VH, XS)
C--**--
C         DETERMINE WHETHER THE PROCEDURE CAN BE TERMINATED.
C
      IF (DABS(VL-VH).LT.TOL) GO TO 40000
C
C       Find the centroid
C
      DO 80 I=1,N
      T=-XS(I,IH)
      DO 90 J=1,N+1
      T=T+XS(I,J)
90    CONTINUE
      XS(I,IC)=T/DFLOAT(N)
80    CONTINUE
C
C       Reflection
C
      DO 15 I=1,N
      X(I)=(1.0+ALPHA)*XS(I,IC)-ALPHA*XS(I,IH)
15    CONTINUE
C
C       Calculate new function value
C
      JCOUNT=JCOUNT+1
      CALL FUN(N, X, F)
C
C       Test new function value
C
      IF (F.LT.VL) GO TO 35000
      IF (F.LT.VN) GO TO 39000
      IF (F.GE.VH) GO TO 27000
C
C       Replacement
C
      DO 25 I=1,N
      XS(I,IH)=X(I)
25    CONTINUE
      XS(N+1,IH)=F
C
C       Reduction
C
27000 DO 35 I=1,N
      X(I)=(1.0-BETA)*XS(I,IC)+BETA*XS(I,IH)
35    CONTINUE
```

```
C
C       Calculate new function value
C
      JCOUNT=JCOUNT+1
      CALL FUN(N, X, F)
C
C       Test new function value
C
      IF (F.LT.XS(N+1,IH)) GO TO 39000
C
C       General contraction
C
      SIZE=0.0
      DO 45 J=1,N+1
      IF (J.EQ.IL) GO TO 45
      DO 55 I=1,N
      XS(I,J)=BETA*(XS(I,J)-XS(I,IL))+XS(I,IL)
      SIZE=SIZE+DABS(XS(I,J)-XS(I,IL))
55    CONTINUE
45    CONTINUE
C--**--
C       TEST SIMPLEX SIZE.
C
      IF (SIZE.LT.SS) GO TO 9000
      WRITE (1,99994)
99994 FORMAT(28H -*-THIS IS NOT NORMAL EXIT./
     * 61H SIZE OF SIMPLEX HAS NOT BEEN REDUCED BY GENERAL CONTRACTION.)
      GO TO 40000
C--**--
C       EXTENSION AND SAVING BR IN THE RIGHT-MOST COLUMN.
C
35000 DO 65 I=1,N
      T=GAMMA*X(I)+(1.0-GAMMA)*XS(I,IC)
      XS(I,IC)=X(I)
      X(I)=T
65    CONTINUE
      XS(N+1,IC)=F
C
C       Calculate new function value
C
      JCOUNT=JCOUNT+1
      CALL FUN(N, X, F)
C
C       Test new function value
C
      IF (F.LT.XS(N+1,IC)) GO TO 39000
C
C       Reset X to BR
C
      DO 75 I=1,N
      X(I)=XS(I,IC)
75    CONTINUE
      F=XS(N+1,IC)
C
```

```
C       Tansfering
C
39000 DO 85 I=1,N
      XS(I,IH)=X(I)
85    CONTINUE
      XS(N+1,IH)=F
      GO TO 14000
40000 DO 95 I=1,N
      X(I)=XS(I,IL)
95    CONTINUE
      F=XS(N+1,IL)
      RETURN
      END
C
C
C
      SUBROUTINE MONISI(JCOUNT, N, FMIN, FMAX, SIM)
C       EXPLANATION----
C     PURPOSE: Subroutine to present the result during MIFSIM.
      DOUBLE PRECISION FMIN, FMAX, SIM(N+1,N+2)
      INTEGER JCOUNT, N,  I, J
C
C     Inset statements here to show the results, for example, in a order:
C
C        JCOUNT, FMIN, FMAX
C        ((SIM(I,J), J=1,N+1), I=1,N+1)
C
      RETURN
      END
```

Appendix E Programme listing of Powell's method

```
C
      SUBROUTINE MIFNOD(N, X, F, FUN, ALPHA, EPSIL, ITERAT, BETA, SK,
     * A1, A2, INMON, XI, X0, T, LAMBDA, BL)
C
C       EXPLANATION----
C     PURPOSE: Minimizing a function without calculating derivatives
C     (Powell's conjugate direction method--Zangwill's simplified version).
C       PARAMETERS----
C     Inputs:
C     N: The number of independent variables. Unchange on exit.
C     ALPHA: The criterion for stopping the procedure. Denote kTH F
C     the optimal value of the function found at kTH iteration, then if
C     | kTH F - (k-1)TH F | < ALPHA the procedure will be terminated.
C     Unchange on exit.
C     EPSIL: A positive number of less or equate to 1. Unchange on
C     exit.
C     BETA, SK, A1, A2: See LINSE1.
C     INMON: If on entry INMON>0, MONITO will be called; if INMON<0,
C     MONITO will not be called. Unchange on exit.
C     Inputs/Outputs:
C     X: On entry X specifies a initial guess of optimal point. On a
C     success exit X contains the optimal point found.
C     F: On a success exit F contains the function value at the
C     optimal point.
C     ITERAT: On exit ITERAT specifies the iteration number performed.
C     Here "iteration" means a whole set of direction searches.
C       WORKING SPACE----
C     XI, X0, T, LAMBDA, BL.
C       ASSOCIAXTED SUBROUTINES----
C     LINSE1/LINSE2: Subroutine(s) to perform linear search.
C     MONITO: Subroutine to present the result during MIFNOD.
C     FUN: User must supply this subroutine to calculate the function
C     value at any given point and declare it as EXTERNAL. The
C     specification is:
C           SUBROUTINE FUN(N, X, F)
C           INTEGER N
C           DOUBLE PRECISION X(N), F
C
      DOUBLE PRECISION X(N), F, ALPHA, EPSIL, BETA, SK, A1,
     * A2, XI(N,N), X0(N), T(N), LAMBDA(N+1), BL(N)
      INTEGER N, ITERAT, INMON
      DOUBLE PRECISION FOM, DELTA, SUMM, DSQRT, DABS, LAMS, TEST
      INTEGER I, J, IS
      EXTERNAL FUN
C
C       Set coordinate directions
C
      DO 10 I=1,N
      DO 20 J=1,N
      IF (I.EQ.J) XI(I,J)=1.0
      IF (I.NE.J) XI(I,J)=0.0
```

```
20    CONTINUE
      X0(I)=X(I)
10    CONTINUE
      CALL FUN(N, X0, F)
      F0M=F
      ITERAT=0
      DELTA=1.0
C--**--
C       DIRECTION SEARCHES.
C
10000 ITERAT=ITERAT+1
C
C       Search loop
C
      DO 30 I=1,N
C
C       Linear search along each direction
C
      DO 40 J=1,N
      T(J)=XI(J,I)
40    CONTINUE
      CALL LINSE1(N,X,T,LAMBDA(I),F,FUN,BETA,SK,A1,A2,BL)
30    CONTINUE
C--**--
C       CHOOSING A NEW POINT.
C
C       TEST WHETHER THE PROCEDURE CAN BE TERMINATED.
C
      IF(DABS(F0M-F).LT.ALPHA) GO TO 20000
C
C       Set a new direction
C
      SUMM=0.0
      DO 50 I=1,N
      SUMM=SUMM+(X(I)-X0(I))**2
50    CONTINUE
      SUMM=DSQRT(SUMM)
      DO 60 I=1,N
      T(I)=(X(I)-X0(I))/SUMM
60    CONTINUE
C
C       Search a new optimum point along this new direction
C
      CALL LINSE1(N,X,T,LAMBDA(N+1),F,FUN,BETA,SK,A1,A2,BL)
C
C       Monitoring
C
      IF (INMON.GT.0) CALL MONITO(N, XI, T, X0, X, F0M, F, ITERAT)
C
C       TEST WHETHER THE PROCEDURE CAN BE TERMINATED.
C
      IF(DABS(F-F0M).LT.ALPHA) GO TO 20000
C--**--
C       SET NEXT SET OF DIRECTIONS.
```

```
C
      FOM=F
      DO 65 I=1,N
      XO(I)=X(I)
65    CONTINUE
      LAMS=LAMBDA(1)
      IS=1
      DO 70 I=2,N
      IF (LAMBDA(I).LE.LAMS) GO TO 70
      LAMS=LAMBDA(I)
      IS=I
70    CONTINUE
      TEST =LAMS*DELTA/SUMM
      IF (TEST.LT.EPSIL) GO TO 10000
      DO 80 I=1,N
      XI(I,IS)=T(I)
80    CONTINUE
      DELTA=TEST
      GO TO 10000
C
C       Procedure is terminated
C
20000 RETURN
      END
C
C
C
      SUBROUTINE MONITO(N, XI, T, XO, X, FO, F, ICOUNT)
C       EXPLANATION----
C     PURPOSE: Subroutine to present the result during MIFNOD. It must
C     not change any parameter in MIFNOD.
C
      DOUBLE PRECISION XI(N,N), T(N), XO(N), X(N), FO, F
      INTEGER ICOUNT,  I, J
C
C       Inset statements here to show the results, for example,
C       in a order:
C                 ICOUNT, XO, FO
C                 (J, (XI(I,J), I=1,N), J=1,N)
C                 X, F
C                 T
C
      RETURN
      END
C
C
C
      SUBROUTINE LINSEl(N, X, T, LAMB, F, FUN, BETA, SK, Al, A2, B)
C
C       EXPLANATION----
C     PURPOSE: Subroutine to perform linear search using the success-
C     failure and parabolic inverse interpolation.
C       PARAMETERS----
C     Inputs:
```

```
C     N: The order of X. Unchange on exit.
C     T: The search direction. Unchange on exit.
C     BETA: The criterion for stopping the search, i.e. when
C     ¦ kTH LAMB - (k-1)TH LAMB ¦ < BETA is satisfied, the search
C     is terminated and returned to where it was called. Unchange
C     on exit.
C     SK: On entry SK specifies the initial increment step. Unchange
C     on exit.
C     A1, A2: Section factors. A1 >1, 0< -A2 <1 and A1 * (-A2) <1.
C     Unchange on exit.
C     Inputs/Outputs:
C     X: On entry X specifies a initial guess of optimal point. On
C     exit X contains the optimal point along the direction T so
C     far been found, i.e. X = X + LAMB * T.
C     LAMB: On exit LAMB contains the optimal step along the search
C     direction.
C     F: On entry F specifies the function value at initial point X.
C     On exit F contains the function value at the optimal point.
C       WORKING SPACE----
C     B.
C       ASSOCIATED SUBROUTINES----
C     FUN: Subroutine to evaluate the function value at any given
C     point. The specification is:
C            SUBROUTINE FUN(N, X, F)
C            INTEGER N
C            DOUBLE PRECISION X(N), F
C
      DOUBLE PRECISION X(N), T(N), LAMB, F, BETA, SK, A1, A2, B(N)
      INTEGER N
      DOUBLE PRECISION SK0, SSTEP, LAM1, FL1,
     * STEP, LAM2, FL2, LAM3, FL3, TEP1, TEP2, DABS
      INTEGER INF, I
      EXTERNAL FUN
C
      SK0=SK
      SSTEP=0.0
      FL2=F
      LAM2=0.0
      INF=0
C--**--
C       SEARCH BEGINS.
C
C       Form a V shape by success-failure performance
C
10000 STEP=SSTEP+SK0
      DO 10 I=1,N
      B(I)=X(I)+STEP*T(I)
10    CONTINUE
      CALL FUN(N, B, F)
      IF (F.GE.FL2) GO TO 100
      FL1=FL2
      LAM1=LAM2
      FL2=F
      LAM2=STEP
```

```
      SSTEP=STEP
      SK0=A1*SK0
      INF=2
      GO TO 10000
100   IF(INF.EQ.2) GO TO 200
      FL1=F
      LAM1=STEP
      INF=2
      SK0=A2*SK0
      GO TO 10000
200   FL3=F
      LAM3=STEP
C
C       Arrange order: LAM1 < LAM2 < LAM3
C
      IF (LAM1.LT.LAM3) GO TO 20000
      TEP1=LAM1
      TEP2=FL1
      LAM1=LAM3
      FL1=FL3
      LAM3=TEP1
      FL3=TEP2
C--**--
C       PARABOLIC INVERSE INTERPOLATION.
C
20000 TEP1=(LAM2-LAM3)*FL1+(LAM3-LAM1)*FL2+(LAM1-LAM2)*FL3
      TEP2=(LAM2**2-LAM3**2)*FL1+(LAM3**2-LAM1**2)*FL2+(LAM1**2-
     * LAM2**2)*FL3
C
C       If TEMP1 is not equal to zero, quadratic approximation
C       can be performed
C
      IF(TEP1.NE.0.0) GO TO 90
      WRITE(1,99999)
99999 FORMAT(' 0 DIVISION, LINEAR SEARCH FAIL!')
      STOP
90    LAMB=0.5*TEP2/TEP1
C--**--
C       TEST WHETHER LINEAR SEARCH CAN BE TERMINATED.
C
      IF (DABS(LAMB-LAM2).LT.BETA) GO TO 40000
C
C       Evaluate the function value at the minimum of the quadratic
C       function
C
      DO 20 I=1,N
      B(I)=X(I)+LAMB*T(I)
20    CONTINUE
      CALL FUN(N, B, F)
C
C       Replace one of old points by the new point
C
      IF ((LAMB.GT.LAM2).AND.(F.GT.FL2)) GO TO 500
      IF ((LAMB.GT.LAM2).AND.(F.LE.FL2)) GO TO 510
```

```
      IF ((LAMB.LT.LAM2).AND.(F.GT.FL2)) GO TO 520
      LAM3=LAM2
      FL3=FL2
      LAM2=LAMB
      FL2=F
      GO TO 20000
500   LAM3=LAMB
      FL3=F
      GO TO 20000
510   LAM1=LAM2
      FL1=FL2
      LAM2=LAMB
      FL2=F
      GO TO 20000
520   LAM1=LAMB
      FL1=F
      GO TO 20000
C
C       Linear search is terminated
C
40000 LAMB=LAM2
      F=FL2
      DO 30 I=1,N
      X(I)=X(I)+LAMB*T(I)
30    CONTINUE
      RETURN
      END
```

Appendix F Iterative strategy for Simple example 2

Simple example 2

The model and reality equations are

$$y = F(c, \alpha) = c + \alpha$$

$$y = F^*(c) = c + c^2.$$

The performance index and constraint are

$$Q(c,y) = c^2 + (y - 2)^2$$

$$CY \overset{\Delta}{=} \{(c,y) \in R^2 : -1 \leqslant c \leqslant 1\}.$$

Firstly, $[\nabla_c Q(c+\Delta c, y+\Delta y) - \nabla_c Q(c,y)]\Delta c + [\nabla_y Q(c+\Delta c, y+\Delta y) - \nabla_y Q(c,y)]\Delta y$

$$= 2(\Delta c^2 + \Delta y^2).$$

Therefore, $\sigma = 2$. According to (5.44)

$b(\alpha) = \sigma(1 + r_{min}(D^T D)) = 2(1 + 1) = 4$, which implies

$$\inf_{\alpha \in A} b(\alpha) = \inf_{\alpha \in \hat{\alpha}(v)} b(\alpha) = 4.$$

Next, we note that $q^*(c) = c^2 + (c + c^2 - 2)^2$, hence,

$$\nabla q^*(c+h) - \nabla q^*(c) = 2h(2h^2 + 6c^2 + 6ch + 3h + 6c - 2).$$

Define $f(c,h) \overset{\Delta}{=} 2h^2 + 6c^2 + 6ch + 3h + 6c - 2$

where $-1 \leqslant c + h \leqslant 1$ and $-1 \leqslant c \leqslant 1$. It can easily be shown that

$$\max f(c,h) = f(1,0) = 10 \text{ and}$$

$$\min f(c,h) = f(-0.5,0) = -3.5.$$

We then have

$$|\nabla q^*(c+h) - \nabla q^*(c)| \leqslant 20|h|.$$

That is the Lipschitz constant ρ can be chosen as 20. Finally, consider (5.29) and (5.30). If we set $\varepsilon = 0^+$, then $B(v) = 0.4$. We conclude that the iterative strategy can be chosen as

$$v^{k+1} = v^k + \theta(c^k - v^k)$$

with the gain θ satisfying $0 < \theta \leqslant 0.4$.

Appendix G Solution regions of the example in Chapter 9

Local optimisation problem 1

[1] $G_1 > 0$, $-1 < c_{11} < 1$ and $-1 < c_{12} < 1$

$c_{11} = 0.5\,\lambda_{11} + 0.25p_{11}$, $c_{12} = 0.5\,\lambda_{12} - 0.25p_{11}$,

$u_{11} = 0.25(-\lambda_{11} + \lambda_{12} - 1.5p_{11} + p_{21} - 2\,\alpha_{11}) + 0.5$,

$G_1 = 0.1(1.5\,\lambda_{11} - 6.5\,\lambda_{12} + 4.75p_{11} - 1.5p_{21} + 3\,\alpha_{11}) + 0.5$.

[2] $G_1 > 0$, $-1 < c_{11} < 1$ and $c_{12} = -1$

$c_{11} = 0.5\,\lambda_{11} + 0.25p_{11}$, $u_{11} = 0.25(-\lambda_{11} - p_{11} + p_{21} - 2\,\alpha_{11})$,

$G_1 = 0.05(3\,\lambda_{11} + 3p_{11} - 3p_{21} + 6\,\alpha_{11}) + 1.8$.

[3] $G_1 > 0$, $-1 < c_{11} < 1$ and $c_{12} = 1$

$c_{11} = 0.5\,\lambda_{11} + 0.25p_{11}$, $u_{11} = 0.25(-\lambda_{11} - p_{11} + p_{21} - 2\,\alpha_{11}) + 1$,

$G_1 = 0.05(3\,\lambda_{11} + 3p_{11} - 3p_{21} + 6\,\alpha_{11}) - 0.8$.

[4] $G_1 > 0$, $c_{11} = -1$ and $-1 < c_{12} < 1$

$c_{12} = 0.5\,\lambda_{12} - 0.25p_{11}$, $u_{11} = 0.25(\lambda_{12} - p_{11} + p_{21} - 2\,\alpha_{11}) + 1$,

$G_1 = 0.05(-13\,\lambda_{12} + 8p_{11} - 3p_{21} + 6\,\alpha_{11}) + 0.2$.

[5] $G_1 > 0$, $c_{11} = -1$ and $c_{12} = -1$

$u_{11} = (-p_{11} + 2p_{21} - 4\,\alpha_{11})/8 + 0.5$, $G_1 = (3p_{11} - 6p_{21} + 12\,\alpha_{11})/40 + 1.5$.

[6] $G_1 > 0$, $c_{11} = -1$ and $c_{12} = 1$

$u_{11} = (-p_{11} + 2p_{21} - 4\,\alpha_{11})/8 + 1.5$, $G_1 = (3p_{11} - 6p_{21} + 12\,\alpha_{11})/40 - 1.1$.

[7] $G_1 > 0$, $c_{11} = 1$ and $-1 < c_{12} < 1$

$c_{12} = 0.5\,\lambda_{12} - 0.25p_{11}$, $u_{11} = 0.25(\lambda_{12} - p_{11} + p_{21} - 2\,\alpha_{11})$,

$G_1 = 0.05(-13\,\lambda_{12} + 8p_{11} - 3p_{21} + 6\,\alpha_{11}) + 0.8$.

[8] $G_1 > 0$, $c_{11} = 1$ and $c_{12} = -1$

$u_{11} = (-p_{11} + 2p_{21} - 4\,\alpha_{11})/8 - 0.5$, $G_1 = (3p_{11} - 6p_{21} + 12\,\alpha_{11})/40 + 2.1$.

[9] $G_1 > 0$, $c_{11} = 1$ and $c_{12} = 1$

$u_{11} = (-p_{11} + 2p_{21} - 4\,\alpha_{11})/8 + 0.5$, $G_1 = (3p_{11} - 6p_{21} + 12\,\alpha_{11})/40 - 0.5$.

[10] $G_1 = 0$, $-1 < c_{11} < 1$ and $-1 < c_{12} < 1$

$c_{11}= (89\,\lambda_{11}+ 19.5\,\lambda_{12}+ 32.5p_{11}+ 4.5p_{21}- 9\,\alpha_{11}- 15)/187,$

$c_{12}= (19.5\,\lambda_{11}+ 9\,\lambda_{12}+ 15p_{11}- 19.5p_{21}+ 39\,\alpha_{11}+ 65)/187,$

$u_{11}= (-32.5\,\lambda_{11}- 15\,\lambda_{12}- 25p_{11}+ 32.5p_{21}- 65\,\alpha_{11}+ 141)/187.$

[11] $G_1= 0$, $-1< c_{11}< 1$ and $c_{12}=-1$

$c_{11}= 0.25(\lambda_{11}+ p_{21}- 2\,\alpha_{11}) - 3,\ u_{11}= 3.$

[12] $G_1= 0$, $-1< c_{11}< 1$ and $c_{12}= 1$

$c_{11}= 0.25(\lambda_{11}+ p_{21}- 2\,\alpha_{11}) + 4/3,\ u_{11}=-1/3.$

[13] $G_1= 0$, $c_{11}=-1$ and $-1< c_{12}< 1$

$c_{12}= (9\,\lambda_{12}+ 15p_{11}- 39p_{21}+ 78\,\alpha_{11}+ 52)/356,$

$u_{11}= (-15\,\lambda_{12}- 25p_{11}+ 65p_{21}- 130\,\alpha_{11}+ 388)/356.$

[14] $G_1= 0$, $c_{11}=-1$ and $c_{12}=-1$

$u_{11}= 3.$

[15] $G_1= 0$, $c_{11}=-1$ and $c_{12}= 1$

$u_{11}=-1/3.$

[16] $G_1= 0$, $c_{11}= 1$ and $-1< c_{12}< 1$

$c_{12}= (9\,\lambda_{12}+ 15p_{11}- 39p_{21}+ 78\,\alpha_{11}+ 208)/356,$

$u_{11}= (-15\,\lambda_{12}- 25p_{11}+ 65p_{21}- 130\,\alpha_{11}+ 128)/356.$

[17] $G_1= 0$, $c_{11}= 1$ and $c_{12}=-1$

$u_{11}= 3.$

[18] $G_1= 0$, $c_{11}= 1$ and $c_{12}= 1$

$u_{11}=-1/3.$

Local optimisation problem 2

[1] $-1< c_{21}< 1$, $-1< c_{22}< 1$ and $-1< c_{23}< 1$

$c_{21}= (3.5\,\lambda_{21}- \lambda_{22}+ \lambda_{23}+ 0.5p_{11}+ 4p_{21}- 2\,\alpha_{21}- 2\,\alpha_{22}+ 10)/9,$

$c_{22}= (-\lambda_{21}+ 3.5\,\lambda_{22}+ \lambda_{23}+ 0.5p_{11}- 5p_{21}- 2\,\alpha_{21}- 2\,\alpha_{22}+ 10)/9,$

$c_{23}= (\lambda_{21}+ \lambda_{22}+ 3.5\,\lambda_{23}- 0.5p_{11}+ 0.5p_{21}+ 2\,\alpha_{21}+ \alpha_{22}- 10)/9,$

$u_{21}= (-4\,\lambda_{21}+ 5\,\lambda_{22}- 0.5\,\lambda_{23}+ 2p_{11}- 11p_{21}- 8\,\alpha_{21}+ \alpha_{22}+ 13)/9.$

[2] $-1 < c_{21} < 1$, $-1 < c_{22} < 1$ and $c_{23} = -1$

$c_{21} = (2.5\lambda_{21} - \lambda_{22} + 0.5p_{11} + 3p_{21} - 2\alpha_{21} - 2\alpha_{22} + 8)/7,$

$c_{22} = (-\lambda_{21} + 2.5\lambda_{22} + 0.5p_{11} - 4p_{21} - 2\alpha_{21} - 2\alpha_{22} + 8)/7,$

$u_{21} = (-3\lambda_{21} + 4\lambda_{22} + 1.5p_{11} - 8.5p_{21} - 6\alpha_{21} + \alpha_{22} + 10)/7.$

[3] $-1 < c_{21} < 1$, $-1 < c_{22} < 1$ and $c_{23} = 1$

$c_{21} = (2.5\lambda_{21} - \lambda_{22} + 0.5p_{11} + 3p_{21} - 2\alpha_{21} - 2\alpha_{22} + 12)/7,$

$c_{22} = (-\lambda_{21} + 2.5\lambda_{22} + 0.5p_{11} - 4p_{21} - 2\alpha_{21} - 2\alpha_{22} + 12)/7,$

$u_{21} = (-3\lambda_{21} + 4\lambda_{22} + 1.5p_{11} - 8.5p_{21} - 6\alpha_{21} + \alpha_{22} + 8)/7.$

[4] $-1 < c_{21} < 1$, $c_{22} = -1$ and $-1 < c_{23} < 1$

$c_{21} = (2.5\lambda_{21} + \lambda_{23} + 0.5p_{11} + 2p_{21} - 2\alpha_{21} - 2\alpha_{22} + 12)/7,$

$c_{23} = (\lambda_{21} + 2.5\lambda_{23} - 0.5p_{11} + 1.5p_{21} + 2\alpha_{21} + 2\alpha_{22} - 12)/7,$

$u_{21} = (-2\lambda_{21} - 1.5\lambda_{23} + p_{11} - 3p_{21} - 4\alpha_{21} + 3\alpha_{22} - 11)/7.$

[5] $-1 < c_{21} < 1$, $c_{22} = -1$ and $c_{23} = -1$

$c_{21} = 0.1(3\lambda_{21} + p_{11} + 2p_{21} - 4\alpha_{21} - 4\alpha_{22}) + 2,$

$u_{21} = 0.1(-2\lambda_{21} + p_{11} - 3p_{21} - 4\alpha_{21} + 6\alpha_{22}) - 2.$

[6] $-1 < c_{21} < 1$, $c_{22} = -1$ and $c_{23} = 1$

$c_{21} = 0.1(3\lambda_{21} + p_{11} + 2p_{21} - 4\alpha_{21} - 4\alpha_{22}) + 2.8,$

$u_{21} = 0.1(-2\lambda_{21} + p_{11} - 3p_{21} - 4\alpha_{21} + 6\alpha_{22}) - 3.2.$

[7] $-1 < c_{21} < 1$, $c_{22} = 1$ and $-1 < c_{23} < 1$

$c_{21} = (2.5\lambda_{21} + \lambda_{23} + 0.5p_{11} + 2p_{21} - 2\alpha_{21} - 2\alpha_{22} + 8)/7,$

$c_{23} = (\lambda_{21} + 2.5\lambda_{23} - 0.5p_{11} + 1.5p_{21} + 2\alpha_{21} + 2\alpha_{22} - 8)/7,$

$u_{21} = (-2\lambda_{21} - 1.5\lambda_{23} + p_{11} - 3p_{21} - 4\alpha_{21} + 3\alpha_{22} + 9)/7.$

[8] $-1 < c_{21} < 1$, $c_{22} = 1$ and $c_{23} = -1$

$c_{21} = 0.1(3\lambda_{21} + p_{11} + 2p_{21} - 4\alpha_{21} - 4\alpha_{22}) + 1.2,$

$u_{21} = 0.1(-2\lambda_{21} + p_{11} - 3p_{21} - 4\alpha_{21} + 6\alpha_{22}) + 1.2.$

[9] $-1 < c_{21} < 1$, $c_{22} = 1$ and $c_{23} = 1$

$c_{21} = 0.1(3\lambda_{21} + p_{11} + 2p_{21} - 4\alpha_{21} - 4\alpha_{22}) + 2,$

$u_{21} = 0.1(-2\lambda_{21} + p_{11} - 3p_{21} - 4\alpha_{21} + 6\alpha_{22}).$

[10] $c_{21}=-1$, $-1< c_{22}< 1$ and $-1< c_{23}< 1$

$c_{22}= (2.5\lambda_{22}+ \lambda_{23}+ 0.5p_{11}- 3p_{21}- 2\alpha_{21}- 2\alpha_{22}+ 12)/7,$

$c_{23}= (\lambda_{22}+ 2.5\lambda_{23}- 0.5p_{11}- 0.5p_{21}+ 2\alpha_{21}+ 2\alpha_{22}- 12)/7,$

$u_{21}= (3 \lambda_{22}+ 0.5\lambda_{23}+ 2p_{11}- 5p_{21}- 8\alpha_{21}- \alpha_{22}+ 27)/7.$

[11] $c_{21}=-1$, $-1< c_{22}< 1$ and $c_{23}=-1$

$c_{22}= 0.1(3\lambda_{22}+ p_{11}- 4p_{21}- 4\alpha_{21}- 4\alpha_{22}) + 2,$

$u_{21}= 0.1(4\lambda_{22}+ 3p_{11}- 7p_{21}- 12\alpha_{21}- 2\alpha_{22}) + 4.$

[12] $c_{21}=-1$, $-1< c_{22}< 1$ and $c_{23}= 1$

$c_{22}= 0.1(3\lambda_{22}+ p_{11}- 4p_{21}- 4\alpha_{21} - 4\alpha_{22}) + 2.8,$

$u_{21}= 0.1(4\lambda_{22}+ 3p_{11}- 7p_{21}- 12\alpha_{21}- 2\alpha_{22}) + 4.4.$

[13] $c_{21}=-1$, $c_{22}=-1$ and $-1< c_{23}< 1$

$c_{23}= 0.1(3 \lambda_{23}- p_{11}+ p_{21}+ 4\alpha_{21}+ 4\alpha_{22}) - 2.8,$

$u_{21}= 0.1(-\lambda_{23}+ 2p_{11}- 2p_{21}- 8\alpha_{21}+ 2\alpha_{22}) + 0.6.$

[14] $c_{21}=-1$, $c_{22}=-1$ and $c_{23}=-1$

$u_{21}= (p_{11}- p_{21}- 4\alpha_{21}+ 2\alpha_{22})/6.$

[15] $c_{21}=-1$, $c_{22}=-1$ and $c_{23}= 1$

$u_{21}= (p_{11}- p_{21}- 4\alpha_{21}+ 2\alpha_{22}- 4)/6.$

[16] $c_{21}=-1$, $c_{22}= 1$ and $-1< c_{23}< 1$

$c_{23}= 0.1(3 \lambda_{23}- p_{11}+ p_{21}+ 4\alpha_{21}+ 4\alpha_{22}) - 2,$

$u_{21}= 0.1(-\lambda_{23}+ 2p_{11}- 2p_{21}- 8\alpha_{21}+ 2\alpha_{22}) + 3.$

[17] $c_{21}=-1$, $c_{22}= 1$ and $c_{23}=-1$

$u_{21}= (p_{11}- p_{21}- 4\alpha_{21}+ 2\alpha_{22}+ 16)/6.$

[18] $c_{21}=-1$, $c_{22}= 1$ and $c_{23}= 1$

$u_{21}= (p_{11}- p_{21}- 4\alpha_{21}+ 2\alpha_{22})/6 + 2.$

[19] $c_{21}= 1$, $-1< c_{22}< 1$ and $-1< c_{23}< 1$

$c_{22}= (2.5\lambda_{22}+ \lambda_{23}+ 0.5p_{11}- 3p_{21}- 2\alpha_{21}- 2\alpha_{22}+ 8)/7,$

$c_{23}= (\lambda_{22}+ 2.5\lambda_{23}- 0.5p_{11}- 0.5p_{21}+ 2\alpha_{21}+ 2\alpha_{22}- 8)/7,$

$u_{21}= (3 \lambda_{22}+ 0.5\lambda_{23}+ 2p_{11}- 5p_{21}- 8\alpha_{21}- \alpha_{22}+ 11)/7.$

[20] $c_{21}= 1,\ -1< c_{22}< 1$ and $c_{23}=-1$

$c_{22}= 0.1(3\lambda_{22}+ p_{11}- 4p_{21}- 4\alpha_{21} - 4\alpha_{22}) + 1.2,$

$u_{21}= 0.1(4\lambda_{22}+ 3p_{11}- 7p_{21}- 12\alpha_{21}- 2\alpha_{22}) + 1.6.$

[21] $c_{21}= 1,\ -1< c_{22}< 1$ and $c_{23}= 1$

$c_{22}= 0.1(3\lambda_{22}+ p_{11}- 4p_{21}- 4\alpha_{21}- 4\alpha_{22}) + 2,$

$u_{21}= 0.1(4\lambda_{22}+ 3p_{11}- 7p_{21}- 12\alpha_{21}- 2\alpha_{22}) + 2.$

[22] $c_{21}= 1,\ c_{22}=-1$ and $-1< c_{23}< 1$

$c_{23}= 0.1(3\lambda_{23}- p_{11}+ p_{21}+ 4\alpha_{21}+ 4\alpha_{22}) - 2,$

$u_{21}= 0.1(-\lambda_{23}+ 2p_{11}- 2p_{21}- 8\alpha_{21}+ 2\alpha_{22}) - 1.$

[23] $c_{21}= 1,\ c_{22}=-1$ and $c_{23}=-1$

$u_{21}= (p_{11}- p_{21}- 4\alpha_{21}+ 2\alpha_{22}- 8)/6.$

[24] $c_{21}= 1,\ c_{22}=-1$ and $c_{23}= 1$

$u_{21}= (p_{11}- p_{21}- 4\alpha_{21}+ 2\alpha_{22})/6 - 2.$

[25] $c_{21}= 1,\ c_{22}= 1$ and $-1< c_{23}< 1$

$c_{23}= 0.1(3\lambda_{23}- p_{11}+ p_{21}+ 4\alpha_{21}+ 4\alpha_{22}) - 1.2,$

$u_{21}= 0.1(-\lambda_{23}+ 2p_{11}- 2p_{21}- 8\alpha_{21}+ 2\alpha_{22}) + 1.4.$

[26] $c_{21}= 1,\ c_{22}= 1$ and $c_{23}=-1$

$u_{21}= (p_{11}- p_{21}- 4\alpha_{21}+ 2\alpha_{22}+ 8)/6.$

[27] $c_{21}= 1,\ c_{22}= 1$ and $c_{23}= 1$

$u_{21}= (p_{11}- p_{21}- 4\alpha_{21}+ 2\alpha_{22}+ 4)/6.$