Optimal Control Synthesis
by
Hill-Climbing in Function-Space

John Clifford Allwright

A thesis submitted for the degree of
Doctor of Philosophy
in the Faculty of Engineering

Control Systems Group
Centre for Computing and Automation
Imperial College of Science and Technology
University of London

July, 1969

Abstract

In this thesis, control function optimisation for linear dynamical systems with quadratic performance indices is mainly considered. Control functions are synthesised from components of linearly-independent basis-functions. Hill-climbing in function-space is then achieved by optimising the components of the basis-functions which are present in the control function.

The basis-functions may be chosen before optimisation; combining their use with the convolution-description of linear dynamical systems then has considerable potential advantages computationally. A new algorithm, based on Dynamic Programming, is developed for optimising the components of the basis-functions present in the control function for linear convolution-described dynamical systems. The optimal components are achieved in one iteration, as a linear function of initial conditions.

The basis-functions used for control function synthesis may also be defined during iterative optimisation, from calculated gradient functions. The map from the linear manifold of the control space which is spanned by the basis-functions to the similarly defined manifold of the gradient space can be simultaneously determined. This enables new algorithms to be developed which yield faster convergence than the steepest-descent algorithm. Control functions can be determined as a function of initial conditions using our approach, which is not possible using the conventional conjugate-gradient algorithm. A new technique is used for determining when, in non-quadratic environments, data deduced from past iterations should be discarded.

To ascertain whether the performance achievable would be appreciably better were the control function to be synthesised from a larger number of basis-functions, methods are developed for calculating lower-bounds for the minimal performance index for a large class of optimisation problems when only non-optimal control functions are available. The lower-bound calculations are not expensive computationally and enable the stage to be determined at which iterative optimisation should cease.

Acknowledgements

I should like to thank my supervisor, Professor J.H. Westcott, for his interest in, and support for, my work.

Thanks are also due to Dr. D.Q. Mayne for many useful discussions throughout the course of the research which led to this thesis, as well as to Mr. R.W. Wilde for his continuous interest and encouragement.

My fellow research students have contributed through many enjoyable arguments, in particular:
Stuart Cumming, David Jacobson, David Mee, David Norris, Ian Rowe and David Stone.

The financial support of the Science Research Council for a large part of the research is gratefully acknowledged.

J.C. Allwright

Notation

Those notations which are not common or are not explained when first used are introduced here.

All quantities considered in this thesis are real.

Vectors are usually denoted by lowercase italic characters, such as x, although some are denoted by lowercase script characters, such as $\mathcal{u}$. Matrices are denoted by uppercase italic characters, such as A. The scalar elements of vectors x and matrices A are usually denoted by x_i, a_{ij}, etc. Scalar-valued functions are usually denoted by characters such as V, F and G. Spaces and sets are denoted by uppercase script characters, such as $\mathcal{R}$.

The set of integers $\{m, m+1, .., n\}$, $n > m$, is denoted by $I(m,n)$, and by $I(n)$ if $m = 1$.

Transposition is denoted by ^{T}. The set of all $n \times m$ matrices of real scalars associated with linear maps from $\mathcal{R}^m$ to $\mathcal{R}^n$ is denoted by $M(\mathcal{R}^m \to \mathcal{R}^n)$. The unit matrix of any order is denoted by I, and the $n \times n$ unit matrix in particular by $I(n,n)$. The zero matrix or vector of any order is denoted by O, and the $n \times m$ zero matrix by $O(n,m)$. $\delta(t,\tau)$ is a matrix of appropriate order which is a unit matrix when $t = \tau$ and is a zero matrix otherwise. If $x \in \mathcal{R}^n$, we interpret $\begin{pmatrix} O(m,1) \\ x \end{pmatrix}$, $\begin{pmatrix} x \\ O(m,1) \end{pmatrix} \in \mathcal{R}^{m+n}$ as $x \in \mathcal{R}^n$ if $m = 0$, and similarly for matrices. The square symmetric matrix which is obtained by replacing every element a_{ij} of a square matrix A by $a_{ij}^{\dagger} = (a_{ij} + a_{ji})$ is denoted by $A^{\dagger}$.

Positive definite, negative definite and non-negative definite are abbreviated to p.d., n.d. and n.n.d.

Maximum and minimum eigenvalues of any real, symmetric and n.n.d. matrix Z are denoted by $\lambda_{max}(Z)$ and $\lambda_{min}(Z)$, respectively.

Partial derivatives such as $(\partial h/\partial x)$ are sometimes written as h_x.

We assume that $\sum_{j=i}^{i} a_j = a_i$, and similarly for products.

Open intervals of the real line are denoted by (t_1,t_2) and closed intervals by $\{t_1,t_2\}$, etc. The segment of a time function u on an interval $\{t_1,t_2\}$ is denoted by $u\{t_1,t_2\}$, etc. Integration on $\{t_1,t_2)$ is denoted by $\int_{\{t_1}^{t_2)} d\cdot$, etc. When properties of a function on some subset (perhaps not proper) of its domain are mentioned, we refer to properties of the image under the function of each element belonging to the considered subset. When no confusion is likely to arise, different functions are sometimes denoted by the same character with different arguments - to reduce notational complexity.

A set A containing only $b(c)$ for every c belonging to a set C is written as $\{a : a = b(c); \forall c \in C\}$, etc. By $\forall t \geqq \tau$, $\tau \in T$ we mean for all $\tau \in T$ and, for each such τ, for all $t \in T$ such that $t \geqq \tau$.

By the x_s-optimal control function we mean the optimal control function for the initial condition x_s, and similarly for the x_s-minimal performance index.

The number 1.58E±08, say, is to be interpreted as $1.58 \times 10^{\pm 8}$.

The main results of this thesis are often summarised in Remarks. The beginning and end of Remarks, Comments, Definitions, Assumptions, Lemmas and Proofs are indicated in the left-hand margin by the symbols ∇ and Δ, for clarity. By RHS (LHS) we mean right (left) hand side.

Contents

Chapter 1 : Introduction

1.1 :	Motivation	10
1.2 :	Background	11
1.3 :	Outline of the Thesis	14
1.4 :	Major Contributions of the Thesis	19

Chapter 2 : Optimal Control Function Synthesis for Linear Convolution-Described Dynamical Systems

2.1 :	Summary	21
2.2 :	Convolution-Description of Linear Dynamical Systems	21
	2.2.1 The Convolution-Description	22
	2.2.2 Effective Convolution State	23
	2.2.3 Potential Advantages of the Convolution-Description	24
2.3 :	Gradient Function Determination for Linear Convolution-Described Dynamical Systems	26
	2.3.1 The Optimisation Problem	26
	2.3.2 Gradient Function Determination	27
	2.3.3 Concluding Comments	33
2.4 :	Second-Order, Dynamic Programming Based, Optimisation on $L(pN)$	34
	2.4.1 The Linear Manifold $L(pN)$	34
	2.4.2 An Optimisation Problem Defined on $L(pN)$	36
	2.4.3 Algorithm Development	36
	2.4.4 Statement of the Second-Order Optimisation Algorithm	46
	2.4.5 The Optimal Performance Index on $L(pN)$	48
	2.4.6 Comments Concerning Computation	49
	2.4.7 Extensions	51
	2.4.8 Discrete-Time Performance Indices	53
	2.4.9 Concluding Comments	54

2.5 : ε-Approximations to Optimal Control Functions and Optimal Control Laws 57

2.5.1 Lower-Bounds for the Minimal Performance Index on $L(pN)$ 57
2.5.2 ε-Approximations 69
2.5.3 Concluding Comments 72

2.6 : A Computed Example 73

Figures Fig. 2.1 73
Fig. 2.2 77

Chapter 3 : Optimal Control Function Synthesis using a Gradient-Decomposition Approach

3.1 : Summary 78

3.2 : An Optimisation Problem 79

3.3 : Optimisation on Translated Linear Manifolds 80

3.4 : $\varepsilon(x_s)$-Approximations to the x_s-Optimal Control Function 86

3.4.1 Lower-Bounds for the x_s-Minimal Performance Index on U 86
3.4.2 $\varepsilon(x_s)$-Approximations 95

3.5 : Gradient Decomposition 100

3.6 : An Iterative, Gradient-Decomposition Based, Algorithm for Control Function Optimisation 104

3.7 : Comparative Effectiveness of the Gradient-Decomposition Based and Steepest-Descent Algorithms 117

3.8 : Computed Examples 132

3.8.1 A Boiler 132
3.8.2 A Heat-Exchanger 137

3.9 : Optimal Control Function Determination as a Function of Initial Conditions 139

3.10 : Computed Results 151

3.11	:	A Gradient-Decomposition Based Algorithm for Optimising in Non-Quadratic Environments	157
3.12	:	A Computed Example	167
3.13	:	Concluding Comments	171
<u>Figures</u>		Fig. 3.1	136
		Fig. 3.2	138
		Fig. 3.3	169
		Fig. 3.4	170
Chapter 4	:	<u>Final Remarks</u>	173
References			175

Chapter 1 : Introduction

1.1 Motivation

The solution of optimal control problems defined for dynamical systems is of great theoretical and practical importance, and has been the subject of widespread interest in recent years. The aim is to choose the control function so that the system considered behaves optimally with respect to a prescribed performance functional, which may be chosen to give a numerical measure of how well the controlled system fulfils its design objectives.

One of the important abstract problems which has been studied is that of trajectory optimisation on a fixed and finite time interval; a realisation of this problem which is of considerable engineering significance is:

For the linear dynamical system described by

$$\dot{x}(t) = f\left(x(t),u(t),t\right)$$

$$y(t) = C(t)x(t)$$

on the time interval $T = \{t_s,t_f\}$ with the initial condition

$$x(t_s) = x_s,$$

the problem is to minimise the (scalar) performance index

$$V\left(x_s,u\right) = \int_T F\left(y(t),u(t),t\right)dt + G\left(y(t_f)\right)$$

with respect to the control function u on T,

where F and G are quadratic.

The application of conventional optimisation techniques to problems of the above type which are formulated for complex dynamical systems is itself complex and computationally expensive: in this thesis are studied approaches to control function optimisation for such problems which can be more efficient computationally than the conventional techniques.

In the next section we review briefly the background to the developments of this thesis.

1.2 Background

The optimisation of the control function u for the optimisation problem considered in 1.1 is more complex than the choice of the control function on each infinitesimal sub-interval of T in such a way as to extremise the contribution to the performance index on that sub-interval; the dynamic effects which relate the contribution to the performance index on each following sub-interval to the control function on the considered sub-interval must also be considered.

The Calculus of Variations, Pontryagin's Maximum Principle, Halkin's Principle of Optimal Evolution, Bellman's Dynamic Programming and the methods of Functional Analysis all utilise theoretically the dynamic features of control function optimisation problems. Some salient attributes of these are next discussed.

The Calculus of Variations

The Calculus of Variations is concerned with problems which "require the determination of the form of an unknown quantity, or of unknown quantities, as a function or functions of a variable (being a dependent variable or dependent variables) so that some integral may assume a maximum or minimum value" {1}. This, essentially, is our problem; however the theory is developed so that, in our context,

$\dot{x}$ is chosen optimally rather than the manipulable control variable u which affects $\dot{x}$. More convenient results, next discussed, have been produced during the last decade.

<u>Pontryagin's Maximum Principle</u>

The optimal control function is generated when the correct boundary conditions are obtained for a two-point boundary value problem {2-6}. Halkin's work is parallel to this and uses reachable-set theory {7-10}. The unknown boundary conditions can be obtained by iteratively adjusting initial guesses in a systematic way {11-12}. The system behaviour is often very sensitive to the boundary conditions because the optimality of the entire control function is dependent only on the boundary conditions at the start of the time interval considered. This helps to ensure that ultimately an exactly optimal control function is obtained, but it can cause severe numerical difficulties. The optimal control function is not obtained from an optimal feedback control law. Perturbation analysis has, however, been used to allow compensation for initial condition changes {13-14}.

<u>Dynamic Programming</u>

Dynamic Programming can, theoretically, be applied to yield globally extremal feedback control laws {15-17}, but the computational and storage requirements become vast for even relatively small dynamic problems, although techniques have been developed for reducing high-speed storage requirements {18}.

Nevertheless, the philosophy of Dynamic Programming, embodied in the Principle of Optimality {15}, can be applied to yield algorithms which utilise first- and second-order expansions about non-optimal control functions and the associated system responses {19-23}. The resulting algorithms generate information which allows non-optimal control functions to be varied so that the performance index is improved.

The first-order algorithms iteratively generate the gradient of the performance index with respect to the control function and improve the performance index by making a sufficiently small change in the negative gradient direction (i.e. downhill). Convergence to the optimal control function may, however, be very slow.

The second-order algorithms calculate actual control function changes, possibly large, from which the optimal control function can be determined. For optimisation problems of the type considered in 1.1, the optimal control function can be achieved in one iteration. The second-order algorithms, however, require extra theoretical complexity, programming effort and extra computation per iteration compared with the first-order algorithms - these effects becoming rapidly more pronounced as the number of differential equations describing the considered dynamical system becomes larger. For nonlinear systems with non-quadratic but nonlinear performance indices, the control function changes may have to be limited to preserve the validity of the second-order expansions used, and many iterations may be required. An optimal feed-back control law is yielded by the second-order algorithms, which is perhaps especially useful for implementation purposes. This is a result of the fundamental approach used (Principle of Optimality) by which the control chosen at each time instant is functionally related to the state of the system at that time.

Functional Analysis

The techniques of functional analysis, discussed in Porter {24}, may be used for linear dynamical systems having norms as performance indices. These fundamental restrictions allow the use of powerful but specialised theoretical results to select the optimal control function from the class of admissible functions.

1.3 Outline of the Thesis

In this thesis we mainly consider determining optimal control functions for optimisation problems of the same kind as that mentioned in 1.1 when $x_s \in R^n$ for either an (arbitrary) nominal initial condition $\tilde{x}_s$ or for all initial conditions x_s belonging to $X(q)$, where

∇ *Definition 1.3.1* $\tilde{x}_s \in R^n$ and has bounded Euclidean norm,

$\hat{X}(q) = \{x_s : x_s = X^q \delta x^q;\ \forall \delta x^q \in R^q\}$,

$X(q) = \{x_s : x_s = \tilde{x}_s + s;\ \forall s \in \hat{X}(q)\}$,

$X^q = (X_1 \dots X_q) \in M(R^q \to R^n)$ and has rank q,

$X_i \in R^n,\ \forall i \in I(q)$, and has bounded Euclidean norm,

$\delta x^q = (\delta x^q_1 \dots \delta x^q_q)^T$,

Δ $q \in I(n)$.

We also consider initial conditions belonging to a closed and bounded neighbourhood $\overline{X}(q)$ of $\tilde{x}_s$, defined by

∇ *Definition 1.3.2*

$$\overline{X}(q) = \{x_s : x_s = \tilde{x}_s + \sum_{j=1}^{q} X_j \delta x^q_j;\ \forall |\delta x^q_j| \leqq |\delta\hat{x}^q_j|,\ \forall j \in I(q)\},$$

Δ where $|\delta\hat{x}^q_j| < \infty,\ \forall j \in I(q)$.

Choice of q, $\hat{X}(q)$, $\tilde{x}_s$ and $\overline{X}(q)$ may be made in an essentially arbitrary way, depending on the particular control problem considered. Consideration of initial conditions belonging to $X(q)$ is not particularly restrictive since if q is set equal to n, $X(q) = R^n$ – the space of all possible initial conditions for the considered optimisation problem. It is, however, desirable that the smallest q be chosen such that $X(q)$ contains all those initial conditions of interest since, as will become apparent later, such a choice reduces the computational effort involved in optimal control function synthesis. Consideration of initial conditions belonging to the bounded neighbourhood $\overline{X}(q)$ is also not particularly restrictive since initial conditions for practical problems are bounded and $\delta\hat{x}_j^q$, $\forall j \in I(q)$, can be chosen so that $\overline{X}(q)$ contains all those bounded initial conditions of interest. Choice of the smallest suitable neighbourhood $\overline{X}(q)$ by choice of the smallest suitable $\delta\hat{x}_j^q$, $\forall j \in I(q)$, is desirable since it may well reduce the computational effort required.

The optimisation techniques of 1.2 require the numerical integration of differential equations when they are applied to optimisation problems such as that of 1.1. It is necessary to discretise time to permit the numerical integration of the differential equations. A potential disadvantage is that the discretisation required for the stability of the integration procedure used may well be far finer than that required to allow the discretised control function to fluctuate rapidly enough in time for effective optimisation to be possible, so that far more numerical work may be performed than is really necessary. The integrations which are required are perhaps the most expensive feature involved in the application of conventional optimisation techniques from the computational point of view. It would therefore be desirable to eliminate the need for the direct numerical solution of differential system equations during optimisation.

A representation of linear dynamical systems in non-differential form is given by the convolution-description. The optimisation of control functions for linear convolution-described dynamical systems is mainly considered in this thesis, although the algorithms of Chapter 3 may also be used for differentially-described dynamical systems.

In Chapter 2 we briefly examine the potential advantages of the convolution-description and then develop a procedure for gradient function determination for convolution-described linear dynamical systems using an approach closely related to Dynamic Programming. Because algorithms using first-order gradient functions do not necessarily achieve convergenc to the optimal control function belonging to the control space considered in one (or even many) iterations, we continue to develop a second-order, Dynamic Programming based, optimisation algorithm for choosing, in one iteration, the optimal control function belonging to a pre-chosen linear manifold $L(pN)$ of the control space in terms of components of the basis-functions which span $L(pN)$. Optimal component choice is achieved as a function of initial conditions x_s belonging to $X(q)$. A simple result is then obtained for calculating, for any particular initial condition x_s, a lower-bound for the x_s-minimal performance index on a linear manifold for a large class of optimisation problems when only a non-x_s-optimal control function belonging to the linear manifold is available. The lower-bound calculation usually requires far less numerical work than that which would be required to determine the x_s-minimal performance index on the linear manifold. The lower-bound result can be used to help overcome the arbitrariness associated with the choice of $L(pN)$ because it enables us to determine, in a simple and computationally inexpensive way, whether the optimal control law which determines x_s-optimal control functions belonging to $L(pN)$ as a function of initial conditions $x_s \in X(q)$ yields x_s-optimal control functions which are a certain desired $\varepsilon(x_s)$-approximation (defined in the body of the thesis) to the x_s-optimal control function belonging to any particular larger linear manifold $L(pN)$ for all initial conditions x_s belonging to the bounded neighbourhood $\overline{X}(q)$. A control law yielding such an approximation for all x_s belonging to $\overline{X}(q)$ is referred to as an $\varepsilon\{\overline{X}(q)\}$-approximation to the optimal control law which determines x_s-optimal control functions belonging to $L(pN)$ as a function of initial conditions x_s belonging to $X(q)$.

The second-order algorithm of Chapter 2 is rather complex and has considerable computer storage requirements, in common with other second-order algorithms. Thus it would seem to be advantageous to develop an algorithm which, for any particular initial condition x_s, causes the control function to converge more rapidly to the x_s-optimal control function belonging to the control space than does the steepest-descent algorithm while only using the relatively simple first-order gradient equations. Such an algorithm is developed in Chapter 3. The algorithm depends on the decomposition of calculated gradient functions into components of non-arbitrary basis-functions, and is therefore referred to as a gradient-decomposition based optimisation algorithm. Lower-bounds for the x_s-minimal performance index on the control space are obtained in this gradient-decomposition context for a large class of optimisation problems. The lower-bounds can be computed with little computational effort. The gradient-decomposition based algorithm achieves optimisation on linear manifolds of the control space (perhaps translated along some non-zero initial control function) which are spanned by basis-functions defined by that algorithm. The dimension of the (perhaps translated) linear manifold on which optimisation is possible is increased by each iteration of the algorithm. The lower-bound results are used to determine when sufficient iterations have been used for the x_s-optimal control function belonging to the (perhaps translated) linear manifold on which optimisation is possible to be some pre-chosen $\varepsilon(x_s)$-approximation to the x_s-optimal control function belonging to the control space, even though the x_s-optimal control function belonging to the control space is not known.

An efficient procedure is presented for determining an optimal control law such that for each initial condition x_s belonging to $\overline{X}(q)$, the resulting control function is some pre-chosen $\varepsilon(x_s)$-approximation to the x_s-optimal control function belonging to the control space, even though the latter control function is not known.

A new algorithm is also developed for use with problems which are the same as that of 1.1 save in that f is non-linear and (or) F and (or) G are non-linear and non-quadratic. For a computed non-linear example, more rapid reduction of the performance index has been achieved using our algorithm than was obtained using the steepest-descent or conjugate-gradient algorithms.

Chapter 4 contains our conclusions and a brief discussion of those areas in which further work might be profitable.

Computed examples which illustrate the theoretical discussions are presented where appropriate.

It will be seen that the techniques developed in this thesis involve optimisation on (perhaps translated) linear manifolds of the control space for the optimisation problem considered. The optimal control function belonging to each (perhaps translated) linear manifold is synthesised by hill-climbing with respect to the components of the basis-functions spanning the linear manifold which are present in the control function. This justifies our title:
Optimal Control Synthesis by Hill-Climbing in Function Space.

1.4 Major Contributions of the Thesis

The major contributions of this thesis, which are believed to be original, are:

1) Our application of the philosophy of Dynamic Programming to optimisation problems defined for linear convolution-described dynamical systems, in Chapter 2.

Balakrishnan and Hsieh have considered the optimisation of control functions for linear convolution-described dynamical systems using first-order algorithms {25-26}. A different approach is used here which is more computation oriented and which permits a second-order algorithm to be developed.

2) The gradient-decomposition approach of Chapter 3 and the associated optimisation algorithms.

Lasdon, Mitter and Warren have recently published an account of the use of the conjugate-gradient algorithm in a control context {27}. Their algorithm can be shown to be related to our gradient-decomposition based algorithms. Optimal control functions cannot, however, be obtained as a function of initial conditions using their algorithm, but can be so obtained using our results. For a computed example, our algorithm for optimising in non-quadratic environments has yielded more rapid optimisation of the performance index as a function of iterations than did their conjugate-gradient algorithm while only using the same number (one) of gradient calculations and optimisations in calculated search directions per iteration as did their algorithm.

3) The results of Chapters 2 and 3 for lower-bounds for the minimal performance index, and the use of the lower-bound results for optimisation purposes through our $\varepsilon\left(x_s\right)$-approximations to the x_s-optimal control function and our $\varepsilon\left(\overline{X}(q)\right)$-approximations to the optimal control law which determines x_s-optimal control functions as a function of initial conditions x_s belonging to the bounded neighbourhood $\overline{X}(q)$.

Pearson has considered the computation of lower-bounds for the minimal performance index for differentially-described dynamical systems {28-30}. Our approach is different and is developed for linear convolution-described dynamical systems, although the results of Chapter 3 are also valid for linear differentially-described dynamical systems.

Chapter 2 : Optimal Control Function Synthesis for Linear Convolution-Described Dynamical Systems

2.1 Summary

In this chapter the philosophy underlying Dynamic Programming is first applied to optimisation problems defined for linear convolution-described dynamical systems. Some potential advantages of the convolution-description over the differential-description are discussed in 2.2. A new approach to gradient determination for linear convolution-described systems is used in 2.3. To facilitate computation, the considered control functions are then constrained to belong to a pre-chosen finite-dimensional linear manifold $L(\mathrm{pN})$ of the control space. A second-order optimisation algorithm is developed in 2.4 for determining the optimal control function belonging to $L(\mathrm{pN})$ as a function of initial conditions belonging to $X(q)$. We develop in 2.5 results which enable us to compute a lower-bound for the x_s-minimal performance index on $L(\mathrm{pN})$ for any initial condition x_s when only a non-x_s-optimal control function belonging to $L(\mathrm{pN})$ is available. The lower-bound can be computed with much less effort than that which would be needed to determine the x_s-minimal performance index on $L(\mathrm{pN})$. We also show how a similar result can be obtained for all initial conditions belonging to $\overline{X}(q)$ when only a non-optimal control law is available. The results can help us to overcome the arbitrariness associated with the choice of $L(\mathrm{pN})$ in that they enable us to compare the effectiveness of optimisation on $L(\mathrm{pN})$ and on a larger linear manifold $L(pN)$ for all initial conditions belonging to $\overline{X}(q)$ in a simple and computationally inexpensive way without having to determine optimal control functions belonging to $L(pN)$. A numerical example is presented in 2.6 which demonstrates the application of the main lower-bound result.

2.2 Convolution-Description of Linear Dynamical Systems

The convolution-description is introduced in 2.2.1 and an effective convolution state is defined in 2.2.2. Some potential advantages of the convolution-description are considered in 2.2.3.

2.2.1 The Convolution-Description

Many linear systems of engineering interest, with control u and output y, have the following form of differential-description:

$$\begin{aligned}(d/dt)x(t) &= A(t)x(t) + B(t)u(t) \;:\; x(t_s) = x_s,\\ y(t) &= C(t)x(t), \end{aligned} \tag{2.1}$$

where t	ε	T	the independent (time) variable,
T	$=$	$\{t_s, t_f\}$	the interval on which (2.1) is defined,
x_s, $x(t)$	ε	R^n	the initial condition and the state at time t,
$y(t)$	ε	R^r	the system output at time t,
$u(t)$	ε	R^m	the system control at time t,

$A(t) \;\varepsilon\; M(R^n \to R^n)$, $\quad B(t) \;\varepsilon\; M(R^m \to R^n)$,

$C(t) \;\varepsilon\; M(R^n \to R^r)$.

If A and Bu are continuous almost everywhere on T and $\int_T \| A(t) \| dt < \infty$, $\int_T \| B(t)u(t) \| dt < \infty$, $\| x_s \| < \infty$, which are fairly unrestrictive conditions from the practical point of view, a unique solution x exists for (2.1) which satisfies (2.1) in that x is continuous on T and satisfies $x(t) = x_s + \int_{t_s}^{t} \big(A(\tau)x(\tau) + B(\tau)u(\tau)\big)d\tau$, $\forall t \;\varepsilon\; T$.

The output y can then be written as:

$$y(t) = \psi(t,t_s)x_s + \int_{t_s}^{t} W(t,\tau)u(\tau)d\tau, \;\forall t \;\varepsilon\; T, \tag{2.2}$$

where $\Phi(t,\tau) \;\varepsilon\; M(R^n \to R^n)$, $\forall t \geqq \tau$, $\tau \;\varepsilon\; T$,

$$\begin{aligned} \psi(t,t_s) &= C(t)\Phi(t,t_s) \;\varepsilon\; M(R^n \to R^r), \;\forall t \;\varepsilon\; T,\\ W(t,\tau) &= C(t)\Phi(t,\tau)B(\tau) \;\varepsilon\; M(R^m \to R^r), \;\forall t \geqq \tau, \;\tau \;\varepsilon\; T,\\ (d/dt)\Phi(t,\tau) &= A(t)\Phi(t,\tau) \;:\; \Phi(\tau,\tau) = I(n,n), \;\forall t \geqq \tau, \;\tau \;\varepsilon\; T, \end{aligned}$$

$\| \quad \|$ denotes Euclidean norm.

∇ *Definition 2.2.1* We refer to (2.2) as the convolution-description of the linear dynamical system with the differential-description of (2.1), Δ and to W as the convolution kernel.

Further discussions are available in {24} and {31}. Similar convolution-descriptions can be obtained for discrete-time (see 2.4.8) and distributed-parameter systems {32}.

2.2.2 Effective Convolution State

The state of a dynamical system at any time is intuitively defined as the minimal information which summarises the entire past history of the system at that time as far as its effect on the future behaviour of the system is concerned {33}. The state of differentially-described dynamical systems is usually considered in the literature. We next define an effective state for convolution-described systems.

Suppose that the dynamical system considered in 2.2.1 actually exists on $T'' = \{t_s, t_f''\}$, $t_f'' > t_f$. Then:

∇ *Definition 2.2.2* For convolution-description (2.2), sufficient information regarding the system history up to, and including, any time $t \in T$, as far as that history affects the future behaviour on T'', is given by the pair $(x_s, u\{t_s, t\})$. We refer to this pair as the Δ effective convolution state at time t, and denote it by S_t.

For all $t \in T$, our effective convolution state at time t can be considered to be a minimal description of the history of the considered convolution-described system, as far as the future behaviour on T'' is concerned, if S_t contains no contribution which does not affect the future response on T'', i.e. if, for all $\xi \in (t, t_f'')$, x_s and $u\{t_s, t\}$

belong to the orthogonal complements of the null-spaces of $\psi(\xi,t_s)$ and the linear map on $u\{t_s,t\}$ which is defined by $\int_{t_s}^{t} W(\xi,\tau)u(\tau)d\tau$, respectively.

S_t will not usually be a minimal description of the system history in the above sense, but it will serve our purpose.

2.2.3 Potential Advantages of the Convolution-Description

Consider calculating the output function y of a linear dynamical system with the differential-description of (2.1) and the convolution-description of (2.2).

∇ *Comment 2.2.1* When differential-description (2.1) is used to calculate $y(t)$, the behaviour of every element of $x(t)$ has to be calculated, even though $y(t)$ may have considerably fewer elements than $x(t)$. Therefore, if only y and u are costed in the performance index (as in 1.1) and $r < n$, an advantage of convolution-description (2.2) over differential-description (2.1) is that, by using the convolution-description, x does not have to be calculated to obtain y. This is especially important if the linear dynamical system which is actually considered has distributed dynamics but only non-distributed input u and output y, since such a system can often be viewed as a version of (2.1) with n infinite. Δ

∇ *Comment 2.2.2* A most unpleasant feature of the direct numerical integration of even time-invariant versions of (2.1) occurs when the matrix A has a wide range of eigenvalues. The time interval on which the output function y is of interest (probably determined by the system mode yielding the least-rapidly varying contribution to the time-response of x) is then likely to be large relative to the integration step-length require for the stability of the integration algorithm employed (probably determined by the system mode yielding the most-rapidly varying contribution to the time response of x), so that many integration steps and much computational effort are likely to be required. The computational difficulties associated with arrays consisting of elements with a wide spread in value have been noted by Kalman {34}.

The integration step-length needed (and thus the computational effort involved) when convolution-description (2.2) is used to calculate the output y depends on the smoothness of W and u. These may be fairly smooth functions even if A has a wide range of eigenvalues. A potential advantage, therefore, of the convolution-description is that the integrations involved in calculating y to specified accuracy when the convolution-description is used may require less computational effort than that which would be required were the differential-description used, even if
Δ x(t) and y(t) are both n-vectors.

∇ *Comment 2.2.3* A potential advantage of the convolution-description over the differential-description for system modelling purposes is that the convolution kernel and ψ can be estimated from observed input-output data using only linear regression {35-36}, while the parameters of the differential-description should be chosen using nonlinear regress-
Δ ion, due to the highly nonlinear way in which y depends on A.

Once ψ and W of (2.2) have been determined, we see from the above comments that the convolution-description has potential advantages over the differential-description for calculating outputs y for given input functions u and initial conditions x_s. Since control function optimisation may require several, perhaps many, computations of outputs for different control functions, these potential advantages could be of considerable computational significance. This is the practical justification for studying control function optimisation for linear convolution-described dynamical systems.

2.3 Gradient Function Determination for Linear Convolution-Described Dynamical Systems

An idea closely connected with the use of Dynamic Programming for control function optimisation is that the contribution to an integral performance index following any time belonging to the domain of integration depends only on the state of the system at that time and the following control function segment. This idea is used in this section to develop in a novel way a (novel) procedure for calculating the sensitivity of the performance index on the control function, which we refer to as the (first-order) gradient function, for linear convolution-described dynamical systems. The optimisation problem is stated in 2.3.1 and gradient function determination is considered in 2.3.2. Control function optimisation can then be achieved using the steepest-descent algorithm or the new algorithms which are developed in Chapter 3. Some concluding comments are contained in 2.3.3.

2.3.1 The Optimisation Problem

For the linear dynamical system described by

$$y(t) = \psi(t,t_s)x_s + \int_{t_s}^{t} W(t,\tau)u(\tau)d\tau, \ \forall t \in T, \tag{2.3}$$

minimise with respect to the control function u on T the scalar performance index

$$V(x_s,u) = \int_T F\big(y(t),u(t),t\big)dt + G\big(y(t_f)\big), \tag{2.4}$$

where $\mathcal{T} = \{t_s, t_f\},\ \ t_f - t_s < \infty,$

ψ, W and u are bounded and continuous,

$x_s \ \epsilon\ R^n,\ \ y(t)\ \epsilon\ R^r,\ \ u(t)\ \epsilon\ R^m,\ \forall t\ \epsilon\ \mathcal{T}$, and

for any bounded initial condition x_s and considered control function u,

F, F_y and F_u are bounded and continuous on $\mathcal{T}$,

G and G_y are bounded.

∇ *Comment 2.3.1* Convolution-description (2.3) with bounded and continuous ψ and W is that for the system with the differential-description of (2.1) with $t_f - t_s < \infty$ if A, B and C of (2.1) are bounded and continuous on $\mathcal{T}$. Note that W being bounded and continuous refers to $W(t,\tau)$ being bounded and continuous on the triangle $t_s \leq \tau \leq t \leq t_f$. The above restrictions on ψ and W are not serious from the computational point of view and can easily be relaxed in the following discussions to ψ and W
Δ being continuous almost everywhere and being bounded.

2.3.2 Gradient Function Determination

Consider the optimisation problem of 2.3.1 for some nominal initial condition x_s. Suppose that a nominal control function u is applied to the dynamical system with the convolution-description of (2.3) and that the resulting output function and effective convolution state (of Definition 2.2.2) at time t are $y(t)$ and S_t, $\forall t\ \epsilon\ \mathcal{T}$.

For all $t\ \epsilon\ \mathcal{T}$, define $V\big(t;S_t,u(t,t_f\}\big)$ to be the contribution following time t to performance index (2.4) given the effective convolution state S_t at time t and the following control function segment $u(t,t_f\}$. Then:

$$V\big(t;S_t,u(t,t_f\}\big) = \int_{(t}^{t_f\}} F\big(y(t),u(t),t\big)dt + G\big(y(t_f)\big),\ \forall t\ \epsilon\ \mathcal{T}, \qquad (2.5)$$

where $V\{t;S_t,u(t,t_f\}\}$ with $t = t_f$ is interpreted as

$$V\{t_f;S_{t_f},u(t_f,t_f\}\} = G\{y(t_f)\}. \tag{2.6}$$

The gradient function $\{\partial V(x_s,u)/\partial u\}$ is defined so that, for sufficiently small arbitrary δu:

$$\begin{aligned} V\{x_s,(u+\delta u)\} &= V\{x_s,u\} + \delta V\{x_s,\delta u\} \\ \simeq V\{x_s,(u+\delta u)\}^1 &= V\{x_s,u\} + \delta V\{x_s,\delta u\}^1, \end{aligned} \tag{2.7}$$

where

$$\delta V\{x_s,\delta u\}^1 = \int_T <\{\partial V(x_s,u)/\partial u(\xi)\},\{\delta u(\xi)\}> d\xi \tag{2.8}$$

and the superscript ^{1} denotes first-order expansion.

For all $t \in T$, define $\delta V\{t;\delta S_t,\delta u(t,t_f\}\}$ to be the change in the contribution following time t to performance index (2.4) which is caused by a change from S_t to $(S+\delta S)_t$ in the effective convolution state at time t and a change from $u(t,t_f\}$ to $(u+\delta u)(t,t_f\}$ in the following control function segment. Then:

$$V\{t;(S+\delta S)_t,(u+\delta u)(t,t_f\}\} = V\{t;S_t,u(t,t_f\}\} + \delta V\{t;\delta S_t,\delta u(t,t_f\}\},$$
$$\forall t \in T. \tag{2.9}$$

A first-order expansion of $V\{t;(S+\delta S)_t,(u+\delta u)(t,t_f\}\}$ can then be written as $V\{t;(S+\delta S)_t,(u+\delta u)(t,t_f\}\}^1 = V\{t;S_t,u(t,t_f\}\} + \delta V\{t;\delta S_t,\delta u(t,t_f\}\}^1_; \ \forall t \in T.$

For the non-anticipatory system considered, $\delta u(t,t_f\}$ has no effect on the variables y and u costed in performance index (2.4) before time t. Its effect on $V\{t;(S+\delta S)_t,(u+\delta u)(t,t_f\}\}$ is therefore the same as its effect on the performance index. A first-order expansion of $\delta V\{t;\delta S_t,\delta u(t,t_f\}\}$ with respect to $\delta u(t,t_f\}$ can therefore be written as $\int_{t)}^{t_f\}} <\{\partial V(x_s,u)/\partial u(\xi)\},\{\delta u(\xi)\}> d\xi$, where $\{\partial V(x_s,u)/\partial u(\xi)\}$ is the sensitivity of the performance index on the control at time ξ and is the gradient which we wish to determine for all $\xi \in T$. Since we wish to determine the gradient for the initial condition x_s, no initial condition change is considered here. The change δS_t in the effective convolution state at time t can, from Definition 2.2.2, therefore only be caused by a control function change on $\{t_s,t\}$, which we denote by $\delta u\{t_s,t\}$. Hence:

$$\delta S_t = \delta u\{t_s,t\}, \; \forall t \in T. \tag{2.10}$$

A first-order expansion of $\delta V(t;\delta S_t,\delta u(t,t_f\})$ with respect to δS_t can therefore be written as $\int_{\{t_s}^{t\}} <(\partial V(t)/\partial u(\Xi)),(\delta u(\Xi))>d\Xi$. Combining the above two first-order contributions leads to the following first-order expansion for $\delta V(t;\delta S_t,\delta u(t,t_f\})$:

$$\delta V(t;\delta S_t,\delta u(t,t_f\})^1 = \int_{\{t_s}^{t\}} <(\partial V(t)/\partial u(\Xi)),(\delta u(\Xi))>d\Xi + \int_{(t}^{t_f\}} <(\partial V(x_s,u)/\partial u(\xi)),(\delta u(\xi))>d\xi, \; \forall t \in T, \tag{2.11}$$

where $\delta V(t;\delta S_t,\delta u(t,t_f\})^1$ with $t = t_f$ is interpreted as

$$\delta V(t;\delta S_{t_f},\delta u(t_f,t_f\})^1 = \int_{\{t_s}^{t_f\}} <(\partial V(t_f)/\partial u(\Xi)),(\delta u(\Xi))>d\Xi. \tag{2.12}$$

The changes δS_t (of (2.10)) and $\delta u(t,t_f\}$ on which $\delta V(t;\delta S_t,\delta u(t,t_f\})$ depend will change $y(\tau)$ and S_τ to, say, $(y+\delta y)(\tau)$ and $(S+\delta S)_\tau$, respectively, $\forall\tau \in (t,t_f\}$. Now F and G of performance index (2.4) can be expanded to first-order in δy and δu as follows

$$F((y+\delta y)(\tau),(u+\delta u)(\tau),\tau)^1 = F(y(\tau),u(\tau),\tau) + \delta F(\delta y(\tau),\delta u(\tau),\tau)^1, \; \forall\tau \in T,$$

$$G((y+\delta y)(t_f))^1 = G(y(t_f)) + \delta G(\delta y(t_f))^1,$$

where

$$\delta F(\delta y(\tau),\delta u(\tau),\tau)^1 = <F_y(\tau),\delta y(\tau)> + <F_u(\tau),\delta u(\tau)>, \; \forall\tau \in T, \tag{2.13}$$

$$\delta G(\delta y(t_f))^1 = <G_y,\delta y(t_f)>, \tag{2.14}$$

and F_y, F_u and G_y are all evaluated for y and u.

From (2.5) and the definition of $\delta V(t;\delta S_t,\delta u(t,t_f\})$, we therefore see that a first-order expansion of $\delta V(t;\delta S_t,\delta u(t,t_f\})$ should satisfy:

$$\delta V\big(t;\delta S_t,\delta u(t,t_f\}\big)^1 = \int_{(t}^{t_f\}} \delta F\big(\delta y(\tau),\delta u(\tau),\tau\big)^1 d\tau + \delta G\big(\delta y(t_f)\big)^1,\ \forall t \in T, \tag{2.15}$$

where $V\big(t;\delta S_t,\delta u(t,t_f\}\big)^1$ with $t = t_f$ is interpreted as

$$\delta V\big(t_f;\delta S_{t_f},\delta u(t_f,t_f\}\big)^1 = \delta G\big(\delta y(t_f)\big)^1. \tag{2.16}$$

On splitting the domain of integration of the integral involved in (2.15):

$$\delta V\big(t;\delta S_t,\delta u(t,t_f\}\big)^1 = \int_{(t}^{t+\delta t\}} \delta F\big(\delta y(\tau),\delta u(\tau),\tau\big)^1 d\tau + \delta V\big(t+\delta t;\delta S_{t+\delta t},\delta u(t+\delta t,t_f\}\big)^1,\ \forall t+\delta t>t,\ t \in T. \tag{2.17}$$

From convolution-description (2.3) and (2.10), the output function change at time $\tau \geqq t$ caused by δS_t and $\delta u(t,t_f\}$ is given by

$$\delta y(\tau) = \int_{\{t_s}^{\tau\}} W(\tau,\eta)\delta u(\eta)d\eta,\ \forall \tau \geqq t,\ t \in T. \tag{2.18}$$

By combining (2.16), (2.14), (2.18) and (2.12), we see that:

$$\int_{\{t_s}^{t_f\}} <\big(\partial V(t_f)/\partial u(\Xi)\big),\big(\delta u(\Xi)\big)>d\Xi = \int_{\{t_s}^{t_f\}} <\big(W(t_f,\eta)^T G_y\big),\big(\delta u(\eta)\big)>d\eta. \tag{2.19}$$

It is clear from (2.13) and (2.18) that the integrand of (2.17) can be written as

$$\delta F\big(\delta y(\tau),\delta u(\tau),\tau\big)^1 = <\big(F_u(\tau)\big),\big(\delta u(\tau)\big)> + \int_{\{t_s}^{t\}} <\big(W(\tau,\eta)^T F_y(\tau)\big),\big(\delta u(\eta)\big)>d\eta + \int_{(t}^{\tau\}} <\big(W(\tau,\eta)^T F_y(\tau)\big),\big(\delta u(\eta)\big)>d\eta,$$
$$\forall \tau \in (t,t+\delta t\},\ \forall t+\delta t>t,\ t \in T. \tag{2.20}$$

On using the first-order expansions of (2.11) and (2.20) in (2.17), it can be seen that

$$\int_{\{t_s}^{t\}} <\big(\partial V(t)/\partial u(\Xi)\big),\big(\delta u(\Xi)\big)>d\Xi + \int_{(t}^{t_f\}} <\big(\partial V(x_s,u)/\partial u(\xi)\big),\big(\delta u(\xi)\big)>d\xi$$
$$= \int_{(t}^{t+\delta t\}} <\big(F_u(\tau)\big),\big(\delta u(\tau)\big)>d\tau$$
$$+ \int_{(t}^{t+\delta t\}} d\tau \int_{\{t_s}^{t\}} <\big(W(\tau,\eta)^T F_y(\tau)\big),\big(\delta u(\eta)\big)>d\eta$$
$$+ \int_{(t}^{t+\delta t\}} d\tau \int_{(t}^{\tau\}} <\big(W(\tau,\eta)^T F_y(\tau)\big),\big(\delta u(\eta)\big)>d\eta +$$
$$\int_{\{t_s}^{t+\delta t\}} <\big(\partial V(t+\delta t)/\partial u(\Xi)\big),\big(\delta u(\Xi)\big)>d\Xi + \int_{(t+\delta t}^{t_f\}} <\big(\partial V(x_s,u)/\partial u(\xi)\big),\big(\delta u(\xi)\big)>d\xi,$$
$$\forall t+\delta t>t,\ t \in T. \tag{2.21}$$

For arbitrary $\delta u\{t_s,t\}$, we see from (2.21) that

$$\int_{\{t_s}^{t\}} <(\partial V(t)/\partial u(\Xi)),(\delta u(\Xi))>d\Xi = \int_{(t}^{t+\delta t\}} d\tau \int_{\{t_s}^{t\}} <(W(\tau,\eta)^T F_y(\tau)),(\delta u(\eta))>d\eta$$
$$+ \int_{\{t_s}^{t\}} <(\partial V(t+\delta t)/\partial u(\Xi)),(\delta u(\Xi))>d\Xi, \quad \forall t+\delta t > t,\ t \in T.$$

For sufficiently small δt, this reveals that

$$\int_{\{t_s}^{t\}} <(\partial V(t)/\partial u(\Xi)),(\delta u(\Xi))>d\Xi \simeq \delta t\int_{\{t_s}^{t\}} <(W(t,\eta)^T F_y(t)),(\delta u(\eta))>d\eta$$
$$+ \int_{\{t_s}^{t\}} <(\partial V(t+\delta t)/\partial u(\Xi)),(\delta u(\Xi))>d\Xi, \quad \forall t+\delta t,\ t \in T. \tag{2.22}$$

For arbitrary $\delta u(t,t+\delta t\}$, we see from (2.21) that

$$\int_{(t}^{t+\delta t\}} <(\partial V(x_s,u)/\partial u(\xi)),(\delta u(\xi))>d\xi = \int_{(t}^{t+\delta t\}} <(F_u(\tau)),(\delta u(\tau))>d\tau$$
$$+ \int_{(t}^{t+\delta t\}} d\tau \int_{(t}^{\tau\}} <(W(\tau,\eta)^T F_y(\tau)),(\delta u(\eta))>d\eta$$
$$+ \int_{(t}^{t+\delta t\}} <(\partial V(t+\delta t)/\partial u(\Xi)),(\delta u(\Xi))>d\Xi, \quad \forall t+\delta t > t,\ t \in T. \tag{2.23}$$

We can now prove

∇ *Remark 2.3.1* For the optimisation problem of 2.3.1, the gradient function $(\partial V(x_s,u)/\partial u)$ can be determined using the following reverse-time equations:

$$-(d/dt)(\partial V(t)/\partial u(\Xi)) = W(t,\Xi)^T F_y(t), \ \forall \Xi \in \{t_s,t\}: \tag{2.24}$$
$$(\partial V(t_f)/\partial u(\Xi)) = W(t_f,\Xi)^T G_y, \ \forall \Xi \in T, \tag{2.25}$$
$$(\partial V(x_s,u)/\partial u(t)) = F_u(t) + (\partial V(t)/\partial u(t)), \quad \forall t \in T, \tag{2.26}$$

where F_u, F_y and G_y are evaluated for the control function u and the
Δ associated response y of (2.3) for the initial condition x_s.

∇ *Proof of Remark 2.3.1* Under the assumptions of 2.3.1, we see that:

(a) reverse-time equation (2.24) comes from (2.22) as $\delta t \to 0$,

(b) starting condition (2.25) comes from (2.19) for arbitrary $\delta u\{t_s,t_f\}$,

Δ (c) (2.26) comes from (2.23) as $\delta t \to 0$.

∇ *Comment 2.3.2* Note that the interval $\{t_s,t\}$ on which

$\{\partial V(t)/\partial u(\Xi)\}$ of (2.24) has to be stored at time t decreases as t decreases from t_f to t_s at the same rate that the interval $\{t,t_f\}$ increases on which the gradient function has been determined using (2.26) and needs to be stored, for future use. The main storage needed for gradient function determination using (2.24) and (2.26), other than that needed to store W, F_u, F_y and G_y, is therefore only that needed to store starting condition (2.25), i.e. that needed to store

Δ $\{\partial V(t_f)/\partial u(\Xi)\}$ for all $\Xi \in T$.

∇ *Comment 2.3.3* It is clear from Remark 2.3.1 that the gradient function $\{\partial V(x_s,u)/\partial u\}$ is bounded and continuous on T when the boundedness

Δ and continuity assumptions of 2.3.1 hold.

∇ *Comment 2.3.4* It is interesting to compare the result of Remark 2.3.1 with the equivalent result for an optimisation problem which is the same as that of 2.3.1 save in that differential-description (2.1) is used in place of convolution-description (2.3). It is well-known that the gradient function $\{\partial V(x_s,u)/\partial u\}$ is then given by the following reverse-time equations:

$$-(d/dt)V_x(t) = C(t)^T F_y(t) + A(t)^T V_x(t) \quad : \quad V_x(t_f) = C(t_f)^T G_y,$$
$$\{\partial V(x_s,u)/\partial u(t)\} = F_u(t) + B(t)^T V_x(t), \ \forall t \in T,$$

where F_u, F_y and G_y are evaluated for the control function u and the associated response y of (2.1) for the initial condition x_s. Note the similarity in structure of the equations of Remark 2.3.1 for gradient function determination for linear convolution-described dynamical systems and the above equations for linear differentially-described dynamical

Δ systems. .

∇ *Comment 2.3.5* It is clear from (2.7) and (2.8) that a necessary condition for a control function u to be the x_s-optimal control function belonging to the linear space of bounded and continuous control functions with domain T and range R^m is that $\left(\partial V(x_s,u)/\partial u\right)$ should be zero almost everywhere on T. This arises since otherwise, for sufficiently small $\Omega > 0$, the control function $u - \Omega\left(\partial V(x_s,u)/\partial u\right)$ would give a lower (i.e. better) performance index value for the initial condition x_s than would the control function u. Δ

2.3.3 Concluding Comments

The determination of the first-order gradient function for convolution-described dynamical systems has also been considered by Balakrishnan {25} and Hsieh {26}, in a different way. Our approach of 2.3.2, which is believed to be original, yields additional insight into gradient function determination for such systems.

In this chapter we do not consider optimisation algorithms which use first-order gradient functions (although we develop new algorithms of this type in Chapter 3) but continue to develop a second-order, Dynamic Programming based, algorithm for choosing the x_s-optimal control function belonging to a finite-dimensional linear manifold of the considered control space, as a function of initial conditions x_s belonging to $X(q)$.

2.4 Second-Order, Dynamic Programming Based, Optimisation on $L(pN)$

In 2.4.1 we define the finite-dimensional linear manifold $L(pN)$ of the linear space of all bounded control functions $u: T \to R^m$. An optimisation problem on $L(pN)$ is formulated in 2.4.2. A second-order, Dynamic Programming based, algorithm for choosing the optimal control function belonging to $L(pN)$ as a function of initial conditions belonging to $X(q)$ is developed in 2.4.3 for linear convolution-described dynamical systems. The algorithm is stated in 2.4.4. The optimal performance index on $L(pN)$ is considered in 2.4.5. Some comments concerning computation and some extensions are the subjects of 2.4.6 and 2.4.7. Another type of optimisation problem to which the approach used herein can be applied is mentioned in 2.4.8. Some concluding comments are contained in 2.4.9.

2.4.1 The Linear Manifold $L(pN)$

∇ *Definition 2.4.1* The pN-dimensional linear manifold $L(pN)$ of the linear space of all bounded control functions $u: T \to R^m$ is defined to be that linear manifold which is spanned by the following pN, m-vector valued, linearly independent and bounded basis-functions which are continuous almost everwhere on their domain T:

$$f_{s,j}, \quad \forall s \in I(p), \ \forall j \in I(N),$$

where, for all $j \in I(N)$,

$$f_{s,j}(t) = O(m,1) \ \text{ if } \ t \notin T^j, \ \forall s \in I(p),$$

$$f_{s,j}(t) = f''_{s,j}(t) \ \text{ if } \ t \in T^j, \ \forall s \in I(p),$$

and

$$T^1 = \{t_1, t_2\}, \quad T^j = (t_j, t_{j+1}\}, \ \forall j \in I(2,N), \quad T = \{t_s, t_f\},$$

Δ $$t_s = t_1 < t_2 < t_3 < \ldots < t_N < t_{N+1} = t_f.$$

The number N of disjoint sub-intervals T^j which cover T, the partition $P(N) = (T^1\ T^2\ \dots\ T^N)$ of T, the number p of basis-functions which are not necessarily zero on each sub-interval T^j and the functions f'' define $L(pN)$, and can all be freely chosen within the constraints imposed in Definition 2.4.1.

Any control function u belonging to $L(pN)$ can be uniquely decomposed into components of the basis-functions f of Definition 2.4.1, and can then be written as

$$u(t) \quad = \quad \sum_{j=1}^{N} F(j,t)u(j) \quad = \quad F^N(t)u^N,\ \forall t \in T, \tag{2.27}$$

where
$$\begin{aligned}
F(j,t) &= \left(f_{1,j}(t)\ \dots\ f_{p,j}(t)\right) \in M(R^p \to R^m),\ \forall j \in I(N),\\
F(j,t) &= O(m,p),\ \forall t \notin T^j,\ \forall j \in I(N),\\
u(j) &\in R^p,\ \forall j \in I(N),\\
F^N(t) &= \left(F(1,t)\ \dots\ F(N,t)\right) \in M(R^{pN} \to R^m),\\
u^N &= \left(u(1)^T\ \dots\ u(N)^T\right)^T \in R^{pN}.
\end{aligned}$$

The following notations will also be used

$$\begin{aligned}
u_k^N &= \left(u(k)^T\ \dots\ u(N)^T\right)^T \in R^{p(N-k+1)},\ \forall k \in I(N-1),\\
u^k &= \left(u(1)^T\ \dots\ u(k)^T\right)^T \in R^{pk},\ \forall k \in I(N),\\
u_N^N &= u(N).
\end{aligned}$$

∇ *Definition 2.4.2* We refer to the control function u of (2.27) as a control function belonging to $L(pN)$ which is exactly characterised by the components u^N. Similarly, we say that such a control function is exactly characterised on T^j by the components $u(j)$, $\forall j \in I(N)$. Δ

∇ *Comment 2.4.1* The important property of $L(pN)$ from our point of view is that control functions belonging to it can be varied on each sub-interval T^j independently of the control function on the other sub-intervals of T. This enables optimisation of the control on the sub-intervals T^j to be carried out in a sequential manner using a Dynamic

Δ Programming based approach.

2.4.2 An Optimisation Problem Defined on $L(pN)$

For the linear dynamical system with the convolution-description

$$y(t) = \psi(t,t_s)x_s + \int_{t_s}^{t} W(t,\tau)u(\tau)d\tau + D(t)u(t), \ \forall t \in T, \qquad (2.28)$$

choose the control function $u = F^N u^N$, belonging to $L(pN)$ and exactly characterised by the components u^N, which minimises the scalar performance index

$$V\left(pN;x_s,u^N\right) = \int_T F\left(y(t),u(t),t\right)dt + G\left(y(t_f)\right), \qquad (2.29)$$

where $x_s = \tilde{x}_s + X^q\delta x^q \in X(q)$,

$y(t) \in R^r, \ u(t) \in R^m, \ \forall t \in T = \{t_s,t_f\}$,

$H_T = t_f - t_s < \infty$,

ψ, W, D and u are continuous almost everywhere and are bounded, and, for all bounded initial conditions and all considered control function

F, F_y, F_{yy}, F_u, F_{uu} and F_{uy} are continuous almost everywhere on T and are bounded, and all higher-order derivatives of F are zero,

G, G_y and G_{yy} are bounded and all higher-order derivatives of G are zero.

For convenience, we rewrite performance index (2.29) as

$$V\left(pN;x_s,u^N\right) = \sum_{k=1}^{N} F\left(y,u(k),k\right) + G\left(y(t_f)\right), \qquad (2.30)$$

where

$$F\left(y,u(k),k\right) = \int_{T^k} F\left(y(t),u(t),t\right)dt, \ \forall k \in I(N).$$

2.4.3 Algorithm Development

The optimisation problem of 2.4.2 is considered.

Suppose that a nominal control function $\tilde{u}$, which belongs to $L(\mathrm{pN})$ and is exactly characterised by the components $\tilde{u}^{\mathrm{N}}$, is applied to the dynamical system with the convolution-description of (2.28) for our nominal initial condition $\tilde{x}_s$, of 1.3. Denote the resulting output function by $\tilde{y}$ and the resulting effective convolution state, of Definition 2.2.2, at time t_k by $\tilde{S}_k$, $\forall k \in I(\mathrm{N}+1)$.

The optimal components $u^{\mathrm{N}}* = \left(u(1)*^{\mathrm{T}} \;\ldots\; u(\mathrm{N})*^{\mathrm{T}}\right)^{\mathrm{T}}$ which exactly characterise the optimal control function belonging to $L(\mathrm{pN})$ for any particular initial condition $x_s = \tilde{x}_s + X^q\delta x^q \in X(q)$ can then be determined by optimising a perturbation $\delta u^{\mathrm{N}} = \left(\delta u(1)^{\mathrm{T}} \;\ldots\; \delta u(\mathrm{N})^{\mathrm{T}}\right)^{\mathrm{T}}$ from $\tilde{u}^{\mathrm{N}}$, when

$$u^{\mathrm{N}}* = \tilde{u}^{\mathrm{N}} + \delta u^{\mathrm{N}}*. \tag{2.31}$$

The optimisation of δu^{N} to give $\delta u^{\mathrm{N}}*$ is next considered.

∇ <u>*Definition 2.4.3*</u> Consider the dynamical system with the convolution-description of (2.28) and the control function $\tilde{u} = F^{\mathrm{N}}\tilde{u}^{\mathrm{N}}$ which belongs to $L(\mathrm{pN})$. From Definition 2.2.2, we see that if the initial condition is changed from $\tilde{x}_s$ to $\tilde{x}_s + X^q\delta x^q$ and the components $\tilde{u}^{\mathrm{N}}$ are changed to $\tilde{u}^{\mathrm{N}} + \delta u^{\mathrm{N}}$, the resulting effective convolution state at time t_k is changed from $\tilde{S}_k$ to $\tilde{S}_k + \delta S_k$ where δS_k is exactly characterised by δs_k - defined as follows:

$$\delta s_1 = \delta x^q \in R^q, \quad \delta s_k = \begin{pmatrix} \delta x^q \\ \delta u^{k-1} \end{pmatrix} \in R^{q+p(k-1)}, \; \forall k \in I(2,\mathrm{N}+1). \tag{2.32}$$
Δ

For all $k \in I(\mathrm{N})$, define $V\left(k;S_k,u_k^{\mathrm{N}}\right)$ to be the contribution following time t_k to performance index (2.30) given the effective convolution state S_k at time t_k and the components $u_k^N = \left(u(k)^T \; \ldots \; u(N)^T\right)^T$, which exactly characterise control functions which belong to $L(pN)$ on T^k, .., T^N.

Then:

$$V\left(k;S_k,u_k^N\right) = \sum_{j=k}^{N} F\left(y,u(j),j\right) + G\left(y(t_f)\right), \; \forall k \in I(N), \qquad (2.33)$$

and

$$V\left(N+1;S_{N+1}\right) = G\left(y(t_f)\right). \qquad (2.34)$$

For all $k \in I(N)$, define $\delta F\left(\delta y(\delta s_k,\delta u(k)),\delta u(k),k\right)$ to be the change in $F\left(y,u(k),k\right)$ (the contribution to performance index (2.30) on T^k) from $F\left(\tilde{y},\tilde{u}(k),k\right)$ to $F\left(\tilde{y}+\delta y(\delta s_k,\delta u(k)),(\tilde{u}+\delta u)(k),k\right)$ due to:

(a) a change from $\tilde{u}(k)$ to $\tilde{u}(k) + \delta u(k)$ in the components $u(k)$ which exactly characterise control functions belonging to $L(pN)$ on T^k, and

(b) the change $\delta y(\delta s_k,\delta u(k))$ from $\tilde{y}$ in the output function y on T^k due to a change from $\tilde{S}_k$ to $\tilde{S}_k + \delta S_k$ in the effective convolution state at time t_k and the change $\delta u(k)$ which is mentioned in (a) above.

Define $\delta G\left(\delta y(\delta s_{N+1})\right)$ to be the change in $G\left(y(t_f)\right)$ from $G\left(\tilde{y}(t_f)\right)$ due to the change in $y(t_f)$ from $\tilde{y}(t_f)$ to $\tilde{y}(t_f) + \delta y(t_f)$ which is caused by a change from $\tilde{S}_{N+1}$ to $\tilde{S}_{N+1} + \delta S_{N+1}$ in the effective convolution state at time t_{N+1}.

For all $k \in I(N)$, define $\delta V\left(k;\delta s_k\right)^*$ to be the change in the contribution following time t_k to performance index (2.30), relative to $V\left(k;\tilde{S}_k,\tilde{u}_k^N\right)$, which is caused by a change in the effective convolution state at time t_k from $\tilde{S}_k$ to $\tilde{S}_k + \delta S_k$ when the components $u_k^N = \left(u(k)^T \;\; \ldots \;\; u(N)^T\right)^T$ are optimised. Then

$$\delta V\left(k;\delta s_k\right)* = \min_{\delta u_k^N} V\left(k;(\tilde{S}+\delta S)_k,(\tilde{u}+\delta u)_k^N\right) - V\left(k;\tilde{S}_k,\tilde{u}_k^N\right), \quad \forall k \in I(N). \tag{2.35}$$

Using the above definitions, it can be seen that

$$\delta V\left(k;\delta s_k\right)* = \min_{\delta u(k)} \{ \delta F\left(\delta y(\delta s_k, \delta u(k)), \delta u(k), k\right) + \delta V\left(k+1;\delta s_{k+1}\right)* \}, \quad \forall k \in I(N), \tag{2.36}$$

where $$\delta V\left(N+1;\delta s_{N+1}\right)* = \delta G\left(\delta y(\delta s_{N+1})\right). \tag{2.37}$$

∇ *Remark 2.4.1* For the optimisation problem of 2.4.2, there exists a unique x_s-optimal control function belonging to $L(pN)$ for each $x_s \in X(q)$ if, for k = N, N-1, .., 1 there exists a unique $\delta u(k)$ which minimises the RHS of (2.36) for all δs_k. From (2.31), if a unique $(\tilde{x}_s + X^q \delta x^q)$-optimal control function belonging to $L(pN)$ exists, it is exactly characterised by the components $u^N* = \left(\{\tilde{u}(1)+\delta u(1)*\}^T \;\; \ldots \;\; \{\tilde{u}(N)+\delta u(N)*\}^T\right)^T$, where $\delta u(k)*$ is the change $\delta u(k)$ which minimises the RHS of (2.36) when $\delta s_k = \delta x^q$ if
Δ k = 1 and $\delta s_k = \left(\{\delta x^q\}^T \;\; \delta u(1)*^T \;\; \ldots \;\; \delta u(k-1)*^T\right)^T$ if k > 1.

Expression (2.36) is a realisation of the Principle of Optimality, the key concept of Dynamic Programming {15}. We refer to (2.36) as a Perturbational Equation of Optimality, since it is concerned with optimal control changes.

We next use the above definitions and Perturbational Equation of Optimality (2.36) to obtain results which enable the optimal component changes $\delta u(k)$ to be determined for all $k \in I(N)$, if they exist.

The unminimised RHS of Perturbational Equation of Optimality (2.36) can be viewed as the change in the contribution following time t_k

to performance index (2.30) due to changes δS_k and $\delta u(k)$ when all the following components u_{k+1}^N (if there are any, i.e. if $k < N$) are chosen optimally given δS_k and $\delta u(k)$, i.e. given δs_{k+1} of Definition 2.4.3. Perturbational Equation of Optimality (2.36) can therefore be rewritten as:

$$\delta V(k;\delta s_k)^* = \min_{\delta u(k)} \delta \mathcal{V}(k;\delta s_{k+1})^*, \ \forall k \in I(N), \tag{2.38}$$

where

$$\delta \mathcal{V}(k;\delta s_{k+1})^* = \delta F(\delta y(\delta s_k, \delta u(k)), \delta u(k), k) + \delta V(k+1;\delta s_{k+1})^*, \quad \forall k \in I(N) \tag{2.39}$$

and

$$\delta V(N+1;\delta s_{N+1})^* = \delta G(\delta y(\delta s_{N+1})). \tag{2.40}$$

For some $k \in I(N)$, assume that a second-order expansion of $\delta V(k+1;\delta s_{k+1})^*$ with respect to $\delta s_{k+1} \in R^{q+pk}$ is exact, so that:

$$\delta V(k+1;\delta s_{k+1})^* = \delta V(k+1)^* + <(\partial V(k+1)^*/\partial s_{k+1}),(\delta s_{k+1})> + \tfrac{1}{2}<(\delta s_{k+1}),(\partial^2 V(k+1)^*/\partial s_{k+1}\partial s_{k+1})(\delta s_{k+1})>. \tag{2.41}$$

Numerical minimisation procedures could be used to minimise $\delta \mathcal{V}(k;\delta s_{k+1})^*$ of (2.38) with respect to $\delta u(k)$ when the second-order expansion for $\delta V(k+1;\delta s_{k+1})^*$ of (2.41) is used in the RHS of $\delta \mathcal{V}(k;\delta s_{k+1})^*$ of (2.39). The optimal value of $\delta u(k)$, denoted by $\delta u(k)^*$, will in general depend on δs_k, i.e. on δx^q and $\delta u^{k-1} = (\delta u(1)^T \ \ldots \ \delta u(k-1)^T)^T$. Since the previous component changes δu^{k-1} have not yet been chosen and any change from $\tilde{x}_s$ to $\tilde{x}_s + X^q \delta x^q \in X(q)$ in the initial condition is to be considered, the optimisation of $\delta \mathcal{V}(k;\delta s_{k+1})^*$ with respect to $\delta u(k)$ would have to be carried out for a range of values of δs_k to give $\delta u(k)^*$, at least approximately, as a function of δs_k of the following type:

$$\delta u(k)^* = \alpha(k) + \beta(\delta s_k, k).$$

Numerical minimisation using search procedures has been explored in {21} for differentially-described dynamical systems. For the problem of 2.4.2, however, a second-order expansion for $\delta F\left(\delta y(\delta s_k, \delta u(k)), \delta u(k), k\right)$ of (2.39) with respect to δs_k and $\delta u(k)$ is exact so that minimisation of the RHS of (2.38) with respect to $\delta u(k)$ can be achieved analytically, This is next considered.

Recall that the output function change $\delta y(\delta s_k, \delta u(k))$ on T^k is the change in y on T^k from $\bar{y}$ due to a change in the effective convolution state at time t_k of $\delta s_k = \left(\{\delta x^q\}^T \ \{\delta u^{k-1}\}^T\right)^T$ and a change of $\delta u(k)$ in the components exactly characterising considered control functions on T^k. On using the notation of 2.4.1 with convolution-description (2.28), we see that $\delta y(\delta s_k, \delta u(k))$ at time $t \in T^k$ is given by

$$\begin{aligned}\delta y(\delta s_k, \delta u(k), t) &= \psi(t, t_s)X^q\delta x^q + \int_{t_s}^{t_2} W(t,\tau)F(1,\tau)\delta u(1)d\tau + \dots \\ &\quad + \int_{t_k}^{t} W(t,\tau)F(k,\tau)\delta u(k)d\tau + D(t)F(k,t)\delta u(k) \\ &= Y_k(t)\delta s_{k+1}, \quad \forall t \in T^k, \end{aligned} \tag{2.42}$$

where

$$\begin{aligned} Y_k(t) &= \Big(\ \psi(t,t_s)X^q \quad \int_{t_1}^{t_2} W(t,\tau)F(1,\tau)d\tau \quad \dots \quad \int_{t_{k-1}}^{t_k} W(t,\tau)F(k-1,\tau)d\tau \\ &\qquad \int_{t_k}^{t} W(t,\tau)F(k,\tau)d\tau + D(t)F(k,t) \ \Big) \\ &\in M(R^{q+pk} \to R^r), \quad \forall t \in T^k. \end{aligned} \tag{2.43}$$

For the optimisation problem of 2.4.2, a second-order expansion for $\delta F\left(\delta y(\delta s_k, \delta u(k)), \delta u(k), k\right)$ in terms of $\delta y(\delta s_k, \delta u(k))$ and the control function change

$$\delta u(t) = F(k,t)\delta u(k), \ \forall t \in T^k, \tag{2.44}$$

which is caused by a change of $\delta u(k)$ in the components $u(k)$ exactly characterising the considered control function on T^k, is exact and can be written as

$$\delta F\big(\delta y(\delta s_k,\delta u(k)),\delta u(k),k\big) = \int_{T^k}\{ <\big(F_{\tilde y}(t)\big),\big(\delta y(\delta s_k,\delta u(k),t)\big)> +$$
$$\tfrac{1}{2}<\big(\delta y(\delta s_k,\delta u(k),t)\big),F_{\tilde y\tilde y}(t)\big(\delta y(\delta s_k,\delta u(k),t)\big)> + <\big(F_{\tilde u}(t)\big),\big(\delta u(t)\big)> +$$
$$\tfrac{1}{2}<\big(\delta u(t)\big),F_{\tilde u\tilde u}(t)\big(\delta u(t)\big)> +$$
$$<\big(\delta u(t)\big),F_{\tilde u\tilde y}(t)\big(\delta y(\delta s_k,\delta u(k),t)\big)> \}dt, \tag{2.45}$$

where all derivatives of F are evaluated for the control function $\tilde u$ and the associated response $\tilde y$, i.e. where $F_{\tilde y}(t) = F\big(\tilde y(t),\tilde u(t),t\big)_y$, etc.

On using (2.42) and (2.44) in (2.45), we see that $\delta F\big(\delta y(\delta s_k,\delta u(k)),\delta u(k),k\big)$ depends only on $\delta s_{k+1} = \big(\delta s_k^T \;\; \delta u(k)^T\big)^T$ and can be expanded exactly to second-order in δs_{k+1} as

$$\delta F\big(\delta y(\delta s_k,\delta u(k)),\delta u(k),k\big) = <\big(\partial F(k)/\partial s_{k+1}\big),\big(\delta s_{k+1}\big)> +$$
$$\tfrac{1}{2}<\big(\delta s_{k+1}\big),\big(\partial^2 F(k)/\partial s_{k+1}\partial s_{k+1}\big)\big(\delta s_{k+1}\big)>, \tag{2.46}$$

where

$$\big(\partial F(k)/\partial s_{k+1}\big) = \int_{T^k}\big(Y_k(t)\big)^T F_{\tilde y}(t)dt + \begin{bmatrix} O(q+p(k-1),1) \\ \int_{T^k}\big(F(k,t)\big)^T F_{\tilde u}(t)dt \end{bmatrix}$$
$$\in R^{q+pk}, \tag{2.47}$$

$$\big(\partial^2 F(k)/\partial s_{k+1}\partial s_{k+1}\big) = \int_{T^k}\big(Y_k(t)\big)^T F_{\tilde y\tilde y}(t)\big(Y_k(t)\big)dt +$$
$$\begin{bmatrix} O(q+p(k-1),q+p(k-1)) & O(q+p(k-1),p) \\ O(p,q+p(k-1)) & \int_{T^k}\big(F(k,t)\big)^T F_{\tilde u\tilde u}(t)\big(F(k,t)\big)dt \end{bmatrix} +$$
$$\begin{bmatrix} O(q+p(k-1),q+pk) \\ \int_{T^k}\big(F(k,t)\big)^T F_{\tilde u\tilde y}(t)\big(Y_k(t)\big)dt \end{bmatrix}^{\dagger} \in M(R^{q+pk}\to R^{q+pk}). \tag{2.48}$$

Then, if expansion (2.41) for $\delta V\big(k+1;\delta s_{k+1}\big)^*$ is exact, $\delta V\big(k;\delta s_{k+1}\big)^*$ of (2.39) can be expanded exactly as:

$$\delta V\big(k;\delta s_{k+1}\big)^* = \delta V(k+1)^* + <\big(\partial V(k)^*/\partial s_{k+1}\big),\big(\delta s_{k+1}\big)> +$$
$$\tfrac{1}{2}<\big(\delta s_{k+1}\big),\big(\partial^2 V(k)^*/\partial s_{k+1}\partial s_{k+1}\big)\big(\delta s_{k+1}\big)>, \tag{2.49}$$

where

$$\left(\partial V(k)^*/\partial s_{k+1}\right) = \left(\partial F(k)/\partial s_{k+1}\right) + \left(\partial V(k+1)^*/\partial s_{k+1}\right), \tag{2.50}$$

$$\left(\partial^2 V(k)^*/\partial s_{k+1}\partial s_{k+1}\right) = \left(\partial^2 F(k)/\partial s_{k+1}\partial s_{k+1}\right) + \left(\partial^2 V(k+1)^*/\partial s_{k+1}\partial s_{k+1}\right). \tag{2.51}$$

Perturbational Equation of Optimality (2.38) then becomes

$$\begin{aligned}\delta V\left(k;\delta s_k\right)^* = \min_{\delta u(k)} \{ \; & \delta V(k+1)^* + <\left(\partial V(k)^*/\partial s_k\right),\left(\delta s_k\right)> + \\ & \tfrac{1}{2}<\left(\delta s_k\right),\left(\partial^2 V(k)^*/\partial s_k\partial s_k\right)\left(\delta s_k\right)> + \\ <\left(\delta u(k)\right),\{ & \left(\partial V(k)^*/\partial u(k)\right) + \left(\partial^2 V(k)^*/\partial u(k)\partial s_k\right)\left(\delta s_k\right) \}> + \\ \tfrac{1}{2}<\left(\delta u(k)\right), & \left(\partial^2 V(k)^*/\partial u(k)\partial u(k)\right)\left(\delta u(k)\right)> \}, \end{aligned} \tag{2.52}$$

where the following have been used:

(a) the symmetry of the second-derivative matrices,

(b) the fact that $\delta s_{k+1} = \begin{pmatrix}\delta s_k \\ \delta u(k)\end{pmatrix}$, and

(c) the partitions

$$\left(\partial V(k)^*/\partial s_{k+1}\right) = \begin{bmatrix}\left(\partial V(k)^*/\partial s_k\right) \\ \left(\partial V(k)^*/\partial u(k)\right)\end{bmatrix}, \tag{2.53}$$

$$\left(\partial^2 V(k)^*/\partial s_{k+1}\partial s_{k+1}\right) = \begin{bmatrix}\left(\partial^2 V(k)^*/\partial s_k\partial s_k\right) & \left(\partial^2 V(k)^*/\partial s_k\partial u(k)\right) \\ \left(\partial^2 V(k)^*/\partial u(k)\partial s_k\right) & \left(\partial^2 V(k)^*/\partial u(k)\partial u(k)\right)\end{bmatrix}. \tag{2.54}$$

Then, if $\Gamma(k)$ of (2.58) is p.d., the minimising component change $\delta u(k)$ of (2.52) is:

$$\delta u(k)^* = \alpha(k) + \left(\beta(k)\right)\left(\delta s_k\right), \tag{2.55}$$

where

$$\alpha(k) = -\Gamma(k)^{-1}\left(\partial V(k)^*/\partial u(k)\right) \in R^p, \tag{2.56}$$

$$\beta(k) = -\Gamma(k)^{-1}\left(\partial^2 V(k)^*/\partial u(k)\partial s_k\right) \in M(R^{q+p(k-1)} \to R^p), \tag{2.57}$$

$$\Gamma(k) = \left(\partial^2 V(k)^*/\partial u(k)\partial u(k)\right) \in M(R^p \to R^p). \tag{2.58}$$

If $\Gamma(k)$ is not p.d., however, a minimising component change $\delta u(k)$ does not exist and no optimal performance index exists on $L(pN)$. For the purposes of this development we assume that $\Gamma(k)$ is p.d.

From (2.55), $\delta u(k)^*$ depends linearly on $\delta s_k \in R^{q+p(k-1)}$, i.e. on the initial condition change from $\tilde{x}_s$ to $\tilde{x}_s + X^q \delta x^q$ which is exactly characterised by δx^q and, if $k > 1$, on the unoptimised component changes δu^{k-1}.

If $k > 1$, $\delta u(k-1)$ needs to be optimised. To optimise $\delta u(k-1)$ we need the expansion terms for $\delta V\left(k;\delta s_k\right)^*$ for insertion in the RHS of (2.50) and (2.51) with k replaced by k-1, so that $\delta u(k-1)^*$ can be determined from (2.52) with k replaced by k-1 in the same way that we have just determined $\delta u(k)^*$.

Inserting $\delta u(k)^*$ of (2.55) into (2.52) leads to the following exact second-order expansion for $\delta V\left(k;\delta s_k\right)^*$:

$$\delta V\left(k;\delta s_k\right)^* = \delta V(k)^* + <\left(\partial V(k)^*/\partial s_k\right),\left(\delta s_k\right)> + \tfrac{1}{2}<\left(\delta s_k\right),\left(\partial^2 V(k)^*/\partial s_k \partial s_k\right)\left(\delta s_k\right)>, \tag{2.59}$$

where

$$\delta V(k)^* = \delta V(k+1)^* - \tfrac{1}{2}<\left(\alpha(k)\right),\Gamma(k)\left(\alpha(k)\right)> \in R, \tag{2.60}$$

$$\left(\partial V(k)^*/\partial s_k\right) = \left(\partial \mathcal{V}(k)^*/\partial s_k\right) - \left(\beta(k)\right)^T \Gamma(k)\alpha(k) \in R^{q+p(k-1)}, \tag{2.61}$$

$$\left(\partial^2 V(k)^*/\partial s_k \partial s_k\right) = \left(\partial^2 \mathcal{V}(k)^*/\partial s_k \partial s_k\right) - \left(\beta(k)\right)^T \Gamma(k)\left(\beta(k)\right) \in M\left(R^{q+p(k-1)} \to R^{q+p(k-1)}\right). \tag{2.62}$$

Equations (2.60), (2.61) and (2.62) are reverse-time recurrence-relations giving the expansion terms for $\delta V\left(k;\delta s_k\right)^*$ from those for $\delta V\left(k+1;\delta s_{k+1}\right)^*$ through those for $\delta \mathcal{V}\left(k;\delta s_{k+1}\right)^*$. The expansion terms $\delta V(k)^*$, $\left(\partial V(k)^*/\partial s_k\right)$ and $\left(\partial^2 V(k)^*/\partial s_k \partial s_k\right)$ are clearly bounded if

$\delta V(k+1)^*$, $\left(\partial V(k+1)^*/\partial s_{k+1}\right)$ and $\left(\partial^2 V(k+1)^*/\partial s_{k+1}\partial s_{k+1}\right)$ are bounded and if $\Gamma(k)$ is p.d. since, under the assumptions of 2.4.2, $\left(\partial F(k)/\partial s_{k+1}\right)$ and $\left(\partial^2 F(k)/\partial s_{k+1}\partial s_{k+1}\right)$ are bounded.

We now need starting conditions for the above reverse-time relations.

Recall from (2.37) that

$$\delta V\left(N+1;\delta s_{N+1}\right)^* = \delta G\left(\delta y(\delta s_{N+1})\right), \tag{2.63}$$

where $\delta G\left(\delta y(\delta s_{N+1})\right)$ is the change in $G\left(y(t_f)\right)$ from $G\left(\tilde{y}(t_f)\right)$ due to the change in $y(t_f)$ from $\tilde{y}(t_f)$ which is caused by a change characterised by $\delta s_{N+1} = \left(\{\delta x^q\}^T \; \{\delta u^N\}^T\right)^T$ in the effective convolution state at time t_{N+1}.

The change in $y(t_f)$ from $\tilde{y}(t_f)$ due to δs_{N+1} is clearly

$$\delta y(\delta s_{N+1}, t_f) = \left(Y_{N+1}\right)\left(\delta s_{N+1}\right), \tag{2.64}$$

where

$$Y_{N+1} = \Big(\psi(t_f,t_s)X^q \quad \int_{t_1}^{t_2} W(t,\tau)F(1,\tau)d\tau \quad \dots \quad \int_{t_{N-1}}^{t_N} W(t_f,\tau)F(N-1,\tau)d\tau \quad \int_{t_N}^{t_{N+1}} W(t_f,\tau)F(N,\tau)d\tau + D(t_f)F(N,t_f) \Big) \in M(R^{q+pN} \to R^r). \tag{2.65}$$

For the optimisation problem of 2.4.2;

$$\delta G\left(\delta y(\delta s_{N+1})\right) = <\left(G_{\tilde{y}}\right),\left(\delta y(\delta s_{N+1},t_f)\right)> + \tfrac{1}{2}<\left(\delta y(\delta s_{N+1},t_f)\right),\left(G_{\tilde{y}\tilde{y}}\right)\left(\delta y(\delta s_{N+1},t_f)\right)>, \tag{2.66}$$

where $G_{\tilde{y}} = G\left(\tilde{y}(t_f)\right)_y$ and $G_{\tilde{y}\tilde{y}} = G\left(\tilde{y}(t_f)\right)_{yy}$.

On using (2.64) in (2.65) and using the result with (2.63), we see that $\delta V\left(N+1;\delta s_{N+1}\right)^*$ can be expanded exactly to second-order in δs_{N+1} as

$$\delta V\left(N+1;\delta s_{N+1}\right)^* = \delta V(N+1)^* + <\left(\partial V(N+1)^*/\partial s_{N+1}\right),\left(\delta s_{N+1}\right)> + \tfrac{1}{2}<\left(\delta s_{N+1}\right),\left(\partial^2 V(N+1)^*/\partial s_{N+1}\partial s_{N+1}\right)\left(\delta s_{N+1}\right)>, \tag{2.67}$$

where

$$\delta V(N+1)^* = 0, \tag{2.68}$$

$$\left(\partial V(N+1)^*/\partial s_{N+1}\right) = \left(Y_{N+1}\right)^T G_{\mathcal{Y}} \in R^{q+pN}, \tag{2.69}$$

$$\left(\partial^2 V(N+1)^*/\partial s_{N+1}\partial s_{N+1}\right) = \left(Y_{N+1}\right)^T G_{\mathcal{YY}}\left(Y_{N+1}\right) \in M(R^{q+pN} \to R^{q+pN}). \tag{2.70}$$

The required starting conditions are therefore provided by (2.68), (2.69) and (2.70).

Having developed the results needed for the (sequential) optimisation of $\delta u(k)$ as k decreases from N to 1, we next state the resulting optimisation algorithm.

2.4.4 Statement of the Second-Order Optimisation Algorithm

The following algorithm is designed to choose the optimal control function belonging to $L(pN)$ for the optimisation problem of 2.4.2 as a linear function of initial conditions $x_s = \tilde{x}_s + X^q \delta x^q \in X(q)$.

1) Choose nominal components $\tilde{u}^N$ which exactly characterise a nominal control function $\tilde{u} = F^N \tilde{u}^N$ which belongs to $L(pN)$. It is desirable (but not at all essential) that $\tilde{u}$ should be a sensible guess at the $\tilde{x}_s$-optimal control function belonging to $L(pN)$, in order to try to limit the size of the control function change relative to $\tilde{u}$ which has to be made to optimise the control function - since large control function changes might lead to significant inaccuracies when digital computers, with their finite word length, are used.

Go to 2)

2) Calculate the output function $\tilde{y}$ associated with the nominal initial condition $\tilde{x}_s$ and the nominal control function $\tilde{u} = F^N \tilde{u}^N$, using convolution-description (2.28). Evaluate and store the following derivatives of

$F\{\mathcal{y}(t),\mathcal{u}(t),t\}$: $F_{\mathcal{y}}(t)$, $F_{\mathcal{yy}}(t)$, $F_{\mathcal{u}}(t)$, $F_{\mathcal{uu}}(t)$, $F_{\mathcal{uy}}(t)$, $\forall t \in T$. Also evaluate and store the following derivatives of $G\{\mathcal{y}(t_f)\}$: $G_{\mathcal{y}}$, $G_{\mathcal{yy}}$. Go to 3).

3) Evaluate the terms $\delta V(N+1)^*$, $\{\partial V(N+1)^*/\partial s_{N+1}\}$ and $\{\partial^2 V(N+1)^*/\partial s_{N+1}\partial s_{N+1}\}$ of an exact second-order expansion for $\delta V\{N+1;\delta s_{N+1}\}^*$ using (2.68), (2.69) and (2.70). Go to 4).

4) Set $k = N$. Go to 5).

5) Calculate the terms $\{\partial F(k)/\partial s_{k+1}\}$ and $\{\partial^2 F(k)/\partial s_{k+1}\partial s_{k+1}\}$ of an exact second-order expansion for $\delta F\{\delta y(\delta s_k,\delta u(k)),\delta u(k),k\}$ using (2.47) and (2.48). Calculate the terms $\{\partial V(k)^*/\partial s_{k+1}\}$ and $\{\partial^2 V(k)^*/\partial s_{k+1}\partial s_{k+1}\}$ of an exact second-order expansion for $\delta V\{k;\delta s_{k+1}\}^*$ using (2.50) and (2.51).

If $\Gamma(k)$ of (2.58) is not p.d., stop - since $\delta u(k)$ cannot be optimised and there exists no optimal control function belonging to $L(pN)$ for any $x_s \in X(q)$.

If $\Gamma(k)$ is p.d., calculate the parameters $\alpha(k)$ and $\{\beta(k)\}$ which determine the optimal component change $\delta u(k)^*$, using (2.56) and (2.57) and recalling partitions (2.53) and (2.54). It will be apparent from Lemma 2.5.4 of 2.5 that a sufficient condition for the existence of an x_s-optimal control function belonging to $L(pN)$ for all $x_s \in X(q)$, and thus for $\Gamma(k)$ to be p.d., is that Assumption 2.5.1 of 2.5 hold. Since $\Gamma(k)$ is p.d., $\delta V\{k;\delta s_k\}^*$ exists, so calculate the terms $\delta V(k)^*$, $\{\partial V(k)^*/\partial s_k\}$ and $\{\partial^2 V(k)^*/\partial s_k\partial s_k\}$ of an exact second-order expansion for $\delta V\{k;\delta s_k\}^*$ using (2.60), (2.61) and 2.62).

If $k \neq 1$, further component changes have yet to be optimised, so set $k = k - 1$ and go to 5).

If k = 1, the parameters α and β which determine all optimal component changes have been determined, so go to 6).

6) The x_s-optimal control function belonging to $L(pN)$ can now be determined for any initial condition $x_s = \tilde{x}_s + X^q \delta x^q \in X(q)$, and is

$$u(t)^* = F^N(t) u^{N*}, \quad \forall t \in T,$$

where, from (2.31), (2.32) and (2.55):

$$u^{N*} = \tilde{u}^N + \delta u^{N*} = \left(\{\tilde{u}(1)+\delta u(1)^*\}^T \quad \dots \quad \{\tilde{u}(N)+\delta u(N)^*\}^T\right)^T,$$

$$\delta u(1)^* = \alpha(1) + \left(\beta(1)\right)\left(\delta x^q\right),$$

$$\delta u(2)^* = \alpha(2) + \left(\beta(2)\right)\begin{pmatrix}\delta x^q \\ \delta u(1)^*\end{pmatrix},$$

....

$$\delta u(j)^* = \alpha(j) + \left(\beta(j)\right)\begin{pmatrix}\delta x^q \\ \delta u^{j-1*}\end{pmatrix},$$

....

$$\delta u(N)^* = \alpha(N) + \left(\beta(N)\right)\begin{pmatrix}\delta x^q \\ \delta u^{N-1*}\end{pmatrix},$$

and where

$$\delta u^{j-1*} = \left(\delta u(1)^{*T} \quad \dots \quad \delta u(j-1)^{*T}\right)^T, \quad \forall j \in I(2,N).$$

2.4.5 The Optimal Performance Index on $L(pN)$

For the optimisation problem of 2.4.2, denote by $V(pN;x_s)^*$ the x_s-minimal performance index on $L(pN)$. Denote the performance index for the nominal initial condition $\tilde{x}_s$ and the nominal control function $\tilde{u} = F^N \tilde{u}^N$ of 2.4.4 by $V(pN;\tilde{x}_s,\tilde{u}^N)$. Then, from the definition of $\delta V(1;\delta s_1)^*$ of (2.35) and the fact that $V(1;S_1,u_1^N)$ of (2.33) is equal to performance index (2.29), we see that:

$$V\left(pN;\tilde{x}_s+X^q\delta x^q\right)* = \min_{\delta u^N} V\left(1;(\tilde{S}+\delta S)_1,(\tilde{u}+\delta u)_1^N\right)$$
$$= V\left(pN;\tilde{x}_s,\tilde{u}^N\right) + \delta V\left(1;\delta s_1\right)*, \qquad (2.71)$$

where, from (2.32): $\delta s_1 = \delta x^q$. (2.72)

If optimisation on $L(pN)$ is possible (i.e. if $\Gamma(k)$ is p.d., $\forall k \in I(N)$), the terms $\delta V(1)*$, $\left(\partial V(1)*/\partial s_1\right)$ and $\left(\partial^2 V(1)*/\partial s_1 \partial s_1\right)$ of an exact second-order expansion for $\delta V\left(1;\delta s_1\right)*$ are available after stage 5) of the algorithm of 2.4.4 has been implemented with k = 1. Then we see from (2.71) and (2.72) that:

$$V\left(pN;\tilde{x}_s+X^q\delta x^q\right)* = V\left(pN;\tilde{x}_s,\tilde{u}^N\right) + \delta V(1)* +$$
$$<\left(\partial V(1)*/\partial s_1\right),\left(\delta x^q\right)> + \tfrac{1}{2}<\left(\delta x^q\right),\left(\partial^2 V(1)*/\partial s_1\partial s_1\right)\left(\delta x^q\right)>, \qquad (2.73)$$

where, from (2.68) and (2.60):

$$\delta V(1)* = -\tfrac{1}{2}\sum_{k=1}^{N} <\left(\alpha(k)\right),\Gamma(k)\left(\alpha(k)\right)>. \qquad (2.74)$$

The dependence of the x_s-minimal performance index on $L(pN)$ on initial conditions $x_s = \tilde{x}_s + X^q\delta x^q \in X(q)$ might be useful information for control system design purposes, and is clear from (2.73).

The condition for $\delta u(k)*$ to exist for all $k \in I(N)$ is that $\Gamma(k)$ should be p.d. for all $k \in I(N)$. If $\Gamma(k)$ is p.d. for all $k \in I(N)$, it is clear from (2.73) and (2.74) that

$$V\left(pN;\tilde{x}_s\right)* \leqq V\left(pN;\tilde{x}_s,\tilde{u}^N\right),$$

i.e. that the x_s-minimal performance index on $L(pN)$ is certainly not greater than the nominal performance index $V\left(pN;\tilde{x}_s,\tilde{u}^N\right)$, as would be expected.

2.4.6 Comments Concerning Computation

∇ *Comment 2.4.2* Since all the second-derivative matrices used by the algorithm of 2.4.4 are symmetric, only the elements belonging to the upper triangle of each such matrix need be evaluated and stored. This reduces the computational effort required and, perhaps more important, Δ reduces the number of computer storage locations needed.

∇ *Comment 2.4.3* The number of elements associated with the expansion terms $\delta V(k)^*$, $(\partial V(k)^*/\partial s_k)$ and $(\partial^2 V(k)^*/\partial s_k \partial s_k)$ of an exact second-order expansion for $\delta V(k;\delta s_k)^*$ decreases as k decreases from N+1 to 1 during the operation of the algorithm of 2.4.4. In fact, for all $k \in I(N)$, the expansion terms for $\delta V(k;\delta s_k)^*$ which are computed using (2.60), (2.61) and (2.62) and the terms $\alpha(k)$ and $(\beta(k))$ of (2.56) and (2.57) can all be stored in the block of storage needed to store the expansion terms for $\delta V(k+1;\delta s_{k+1})^*$, which are no longer needed once the previously mentioned terms have been determined. This arises because:

(a) $\delta V(k)^*$ can be written over $\delta V(k+1)^*$,

(b) $\begin{bmatrix} (\partial V(k)^*/\partial s_k) \\ \alpha(k) \end{bmatrix}$ can be written over $(\partial V(k+1)^*/\partial s_{k+1})$, and

(c) $\left((\partial^2 V(k)^*/\partial s_k \partial s_k) \quad (\beta(k))^T \right)$ can be written over $(\partial^2 V(k+1)^*/\partial s_{k+1} \partial s_{k+1})$ – even when the elements belonging to the upper triangle only of each second-derivative matrix are stored.

The expansion terms for $\delta V(k;\delta s_k)^*$ together with the terms $\alpha(j)$, $(\beta(j))$, $\forall j \in I(k,N)$ which are calculated by the algorithm of 2.4.4 as k decreases from N to 1 can therefore all be stored in the block of storage needed to store $\delta V(N+1)^*$, $(\partial V(N+1)^*/\partial s_{N+1})$ and Δ $(\partial^2 V(N+1)^*/\partial s_{N+1} \partial s_{N+1})$, the starting conditions of (2.68), (2.69) and (2.70

∇ *Comment 2.4.4* When the x_s-optimal control function belonging to

$\bar{L}(pN)$ has been calculated and applied to the dynamical system considered in 2.4.2, the resulting performance index should be $V(pN;x_s)^*$ of (2.73). This can be used as a check on the x_s-optimality of the calculated x_s-optimal control function belonging to $L(pN)$. Δ

∇ *Comment 2.4.5* The behaviour of the x_s-optimal performance index on $L(pN)$ as a function of initial conditions $x_s \in X(q)$ may, however, be of no interest. In this case a reduction in the computational effort required may be achieved by dropping all terms of the type $(\partial-/\partial x^q)$ and $(\partial^2-/\partial x^q \partial x^q)$ from all the expansions associated with the algorithm of 2.4.4. This does not affect the determination of the x_s-optimal control function belonging to $L(pN)$ as a linear function of $x_s \in X(q)$. Δ

∇ *Comment 2.4.6* If the optimal control function belonging to $L(pN)$ is not required as a function of $x_s \in X(q)$ but is only required for the nominal initial condition $\hat{x}_s$, we can re-define δs_k of (2.32) as δu^{k-1}, instead of $\begin{bmatrix} \delta x^q \\ \delta u^{k-1} \end{bmatrix}$. The algorithm of 2.4.4 is then essentially unchanged save in that all expansion terms such as $(\partial-/\partial x^q)$, $(\partial^2-/\partial x^q \partial x^q)$, $(\partial^2-/\partial u(k) \partial x^q)$, etc., are dropped, which clearly reduces the computational effort required. Δ

2.4.7 Extensions

The range of optimisation problems to which the algorithm of 2.4.4 can be applied is here extended by introducing additional variables which can be treated in the same way as initial conditions.

∇ *Comment 2.4.7* The initial condition at time t_s for the system with the convolution-description of (2.28) can be considered to be established by the system input before time t_s if the system exists before t_s (for an arbitrarily long time, say) and if the system is controllable in the sense that any initial condition $x(t_s) = x_s$ can be established by suitable choice of the input function u before t_s (on $(-\infty,t_s)$, say). The system may then be considered to have the following convolution-description:

$$y(t) = \int_{-\infty}^{t} W(t,\tau)u(\tau)d\tau + D(t)u(t), \ \forall t \in T.$$

For many such systems, the history of the entire input function u before time t_s (i.e. on $(-\infty,t_s)$) may be unnecessary from the practical point of view, as far as the output y on T is concerned, if the input before some time $t_I < t_s$ has negligible effect on y on T. The input function on $\{t_I,t_s)$ may then be characterised by components of basis-functions which can be non-zero only on $\{t_I,t_s)$ and are zero elsewhere. The components can then be considered as initial conditions for the system at t_s and can be assembled in a vector x_s. The output y on T can then be written as

$$y(t) = \Psi(t,t_s)x_s + \int_{t_s}^{t} W(t,\tau)u(\tau)d\tau + D(t)u(t), \ \forall t \in T, \qquad (2.75)$$

where each column of $\Psi(t,t_s)$ gives the effect on $y(t)$ of a unit component of a basis-function present in u on $\{t_I,t_s)$. Since (2.75) has the same form as convolution-description (2.28), the second-order optimisation algorithm which we have developed in 2.4.3 and stated in 2.4.4 can be used to optimise performance index (2.29) on $L(pN)$ when convolution-description (2.28) is replaced by convolution-description (2.75). Δ

∇ *Comment 2.4.8* It is sometimes desired to cause the output function y of a dynamical system such as that considered in 2.4.2 to be as close as possible to a specified output function y_d in that

$$\int_T \{ <\left(y_d(t)-y(t)\right),Q(t)\left(y_d(t)-y(t)\right)> \; + \; <\left(u(t)\right),R(t)\left(u(t)\right)> \} \, dt$$

is smallest, where Q is n.n.d. on T and R is p.d. on T.

If y_d is exactly characterised by components (which are assembled into a vector b) of some basis-functions and can be written as

$$y_d(t) = \Xi(t)b, \; \forall t \in T,$$

the error between y and y_d can be written as

$$\varepsilon(t) = \left(\psi(t,t_s) \quad -\Xi(t)\right)\begin{pmatrix} x_s \\ b \end{pmatrix} + \int_{t_s}^{t} W(t,\tau)u(\tau)d\tau + D(t)u(t), \quad \forall t \in T. \qquad (2.76)$$

Then, if performance index (2.29) is replaced by

$$\int_T F\left(\varepsilon(t),u(t),t\right)dt$$

where

$$F\left(\varepsilon(t),u(t),t\right) = <\left(\varepsilon(t)\right),Q(t)\left(\varepsilon(t)\right)> + <\left(u(t)\right),R(t)\left(u(t)\right)>, \; \forall t \in T,$$

the second-order optimisation algorithm stated in 2.4.4 can be used to determine the optimal control function belonging to $L(pN)$ for this 'follower' problem when convolution-description (2.28) is replaced by convolution-description (2.76) and $\begin{pmatrix} x_s \\ b \end{pmatrix}$ is treated as the initial

Δ condition vector.

∇ *Comment 2.4.9* Clearly any combination of initial conditions of the types considered in Comment 2.4.7 and Comment 2.4.8 can be

Δ considered.

2.4.8 Discrete-Time Performance Indices

An algorithm essentially the same as that developed in 2.4.3 and stated in 2.4.4 can be developed for problems of the following type:

for the dynamical system with the convolution-description

$$y_{k+1} = \psi(k+1,1)x_s + \sum_{j=1}^{k} W(k+1,j)u(j), \; \forall k \in I(N), \qquad (2.77)$$

minimise with respect to $u(k)$, $\forall k \in I(N)$, the scalar performance index

$$V = \sum_{k=1}^{N} F\left(y_{k+1},u(k),k\right) + G\left(y_{N+1}\right),$$

where

$F\left(y_{k+1},u(k),k\right)$ is twice differentiable with respect to y_{k+1} and $u(k)$ and has zero higher-order derivatives with respect to these, $\forall k \in I(N)$,

$G\left(y_{N+1}\right)$ is twice differentiable with respect to y_{N+1} and has zero higher-order derivatives with respect to y_{N+1}.

Convolution-description (2.77) is, for example, an exact convolution-description for the system

$$x_{k+1} = A_k x_k + B_k u(k) \;:\; x_1 = x_s,$$
$$y_{k+1} = C_{k+1} x_{k+1} + D_k u(k),$$
$$\forall k \in I(N),$$

when the terms ψ and W of convolution-description (2.77) are given by

$$\psi(k+1,1) = C_{k+1}\Phi(k+1,1), \; \forall k \in I(N),$$
$$W(k+1,j) = C_{k+1}\Phi(k+1,j+1)B_j + \delta(k,j)D_j, \; \forall k \geq j, \; j \in I(N),$$

where

$$\Phi(k+1,j) = A_k\Phi(k,j) \;:\; \Phi(j,j) = I, \; \forall k \geq j, \; j \in I(N).$$

2.4.9 Concluding Comments

We have considered the optimisation problem of 2.4.2 and have developed an optimisation algorithm, summarised in 2.4.4, for choosing the optimal control function belonging to $L(pN)$ as a linear function of initial conditions $x_s = \tilde{x}_s + X^q\delta x^q \in X(q)$. The algorithm is based on

Dynamic Programming and is believed to be novel.

Balakrishnan {25} and Hsieh {26} have also considered control function optimisation for linear convolution-described dynamical systems. They also calculate second-derivative information but they only use it to determine the gradient function (and the optimal step in the associated search direction) for each control function which arises during the implementation of the first-order gradient algorithms which they use. Their algorithms do not necessarily yield convergence to the optimal control function in one (or many) iterations and do not enable optimal control functions to be determined as a function of initial conditions. Our algorithm of 2.4.4, however, makes full use of calculated second-derivative terms to determine the optimal control function belonging to our linear manifold $L(\mathrm{pN})$ as a linear function of initial conditions $x_s \in X(q)$.

In 2.5 we shall consider theoretically optimisation on $L(\mathrm{pN})$ using the inverse of a pN x pN second-derivative matrix, $\left(\partial^2 V/\partial u^N \partial u^N\right)$. The Dynamic Programming based algorithm of 2.4.4 determines the x_s-optimal control function belonging to $L(\mathrm{pN})$ as a function of initial conditions $x_s \in X(q)$ <u>without</u> the explicit evaluation and inversion of $\left(\partial^2 V/\partial u^N \partial u^N\right)$, which is advantageous. Another feature of the Dynamic Programming based algorithm is that the resulting control law (of stage 6) of 2.4.4) determines the optimal control function on each interval T^j as a function of the change δS_j, which is exactly characterised by $\delta s_j = \left(\{\delta x^q\}^T \{\delta u^{j-1}\}^T\right)^T$, in the effective convolution state at the start of T^j from its nominal value of $\tilde{S}_j$. If δs_j were to be estimated from the response of the system up to time T^j in such a way as to cause the response of convolution-description (2.28) to be as close as possible to that of the system being controlled, a potentially useful control law would result which would automatically tend to compensate for any differences there might be between convolution-description (2.28), used for determining the optimal control law, and the actual convolution-description of the system being controlled.

2.5 ε-Approximations to Optimal Control Functions and Optimal Control Laws

The optimisation problem of 2.4.2 is considered. In 2.5.1 is developed a simple and computationally inexpensive means for determining a lower-bound for the x_s-minimal performance index on $L(\mathrm{pN})$ when the x_s-optimal control function belonging to $L(\mathrm{pN})$ is not known. A similar result is also obtained for all initial conditions belonging to $\overline{X}(\mathrm{q})$ when there is available only a non-optimal control law yielding control functions belonging to $L(\mathrm{pN})$ as a function of initial conditions belonging to $X(\mathrm{q})$. We explain in 2.5.2 how the results can be used to decide in a simple and computationally inexpensive way whether the computational expense involved in optimising on a linear manfold $L(pN)$ of larger dimension than that of $L(\mathrm{pN})$ could be profitable, performance-index wise, before actually optimising on $L(pN)$. Some concluding comments are contained in 2.5.3.

2.5.1 Lower-Bounds for the Minimal Performance Index on $L(\mathrm{pN})$

∇ *Assumption 2.5.1* Suppose that a nominal control function $\tilde{u} = F^N\tilde{u}^N$, which belongs to $L(\mathrm{pN})$, is applied to the dynamical system considered in 2.4.2 for the initial condition $\tilde{x}_s$ and yields an output function $\tilde{y}$ and a performance index value of $V(\mathrm{pN};\tilde{x}_s,\tilde{u}^N)$. Suppose also that for all bounded δu and δy:

$$F\big((\tilde{y}+\delta y)(t),(\tilde{u}+\delta u)(t),t\big) = F\big(\tilde{y}(t),\tilde{u}(t),t\big) + <\big(F_{\tilde{y}}(t)\big),\big(\delta y(t)\big)> + \tfrac{1}{2}<\big(\delta y(t)\big),\big(F_{\tilde{y}\tilde{y}}(t)\big)\big(\delta y(t)\big)> + <\big(F_{\tilde{u}}(t)\big),\big(\delta u(t)\big)> + \tfrac{1}{2}<\big(\delta u(t)\big),\big(F_{\tilde{u}\tilde{u}}(t)\big)\big(\delta u(t)\big)>, \ \forall t \in T,$$

$$G\big((\tilde{y}+\delta y)(t_f)\big) = G\big(\tilde{y}(t_f)\big) + <\big(G_{\tilde{y}}\big),\big(\delta y(t_f)\big)> + \tfrac{1}{2}<\big(\delta y(t_f)\big),\big(G_{\tilde{y}\tilde{y}}\big)\big(\delta y(t_f)\big)>,$$

where

$$F_{\tilde{y}}(t) = F\big(\tilde{y}(t),\tilde{u}(t),t\big)_y, \text{ etc.,}$$

F_y, F_{yy}, F_u and F_{uu} are continuous almost everywhere on T and are bounded on T,

$F_{\tilde{u}\tilde{u}}$ is p.d. on T, $F_{\tilde{y}\tilde{y}}$ is n.n.d. on T,

$G_{\tilde{y}}$ exists and $G_{\tilde{y}\tilde{y}}$ is bounded and n.n.d.

Assume finally that the linear independence of the basis-functions f of 2.4.1 is such that, for all $j \in I(N)$, there do not exist scalars α_i, which are not all zero, such that $\sum_{i=1}^{p} \alpha_i f_{i,j}(t) = 0$ for all $t \in T$ except for t belonging to a set of measure zero. Δ

The main result of this sub-section is contained in the following Remark.

∇ *Remark 2.5.1* Consider the optimisation problem of 2.4.2 when Assumption 2.5.1 holds. Then a lower-bound for the $\tilde{x}_s$-minimal performance index on $L(pN)$, evaluated for the (potentially non-optimal) control function $\tilde{u} = F^N\tilde{u}^N \in L(pN)$, is:

$$\tilde{V}\big(pN;\tilde{x}_s,\tilde{u}^N\big)* = V\big(pN;\tilde{x}_s,\tilde{u}^N\big) - \tfrac{1}{2}\sum_{k=1}^{N} <\big(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u(k)\big),\big(E_k\big)^{-1}\big(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u(k)\big)>,$$

where

$$E_k = \int_{T^k}\big(F(k,t)\big)^T F_{\tilde{u}\tilde{u}}(t)\big(F(k,t)\big)dt \quad \in \quad M(R^p \to R^p),\ \forall k \in I(N),$$

$$\big(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N\big) = \int_T\{ \ \big(F^N(t)\big)^T F_{\tilde{u}}(t) + \big(D(t)F^N(t)\big)^T F_{\tilde{y}}(t) + \int_{t_s}^{t}\big(W(t,\tau)F^N(\tau)\big)^T F_{\tilde{y}}(t)d\tau + \big(W(t_f,t)F^N(t)\big)^T G_{\tilde{y}}\ \}dt$$
$$+ \big(D(t_f)F^N(t_f)\big)^T G_{\tilde{y}} \quad \in \quad R^{pN}, \tag{2.78}$$

$\big(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N\big)$ is partitioned as

$\left(\left(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u(1)\right)^T \;\ldots\; \left(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u(N)\right)^T \right)^T$

and where

Δ $\left(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u(k)\right) \in R^p$, $\forall k \in I(N)$.

∇ *Comment 2.5.1* Since $\tilde{x}_s$ may be any initial condition belonging to R^n, the lower-bound result of Remark 2.5.1 may be used to determine a lower-bound for the x_s-minimal performance index on $L(pN)$ for any initial

Δ condition $x_s \in R^n$.

We next prove Remark 2.5.1, for which the following lemmas are helpful.

∇ *Lemma 2.5.1* When Assumption 2.5.1 holds, the performance index for the optimisation problem of 2.4.2 for an initial condition $x_s = \tilde{x}_s + X^q\delta x^q \in X(q)$ and a control function $u = F^N(\tilde{u}+\delta u)^N \in L(pN)$ is:

$$V\left(pN;\tilde{x}_s+X^q\delta x^q,\tilde{u}^N+\delta u^N\right) = V\left(pN;\tilde{x}_s,\tilde{u}^N\right) + <\left(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial x^q\right),\left(\delta x^q\right)> +$$
$$\tfrac{1}{2}<\left(\delta x^q\right),\left(\partial^2 V/\partial x^q\partial x^q\right)\left(\delta x^q\right)> + <\left(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N\right),\left(\delta u^N\right)> +$$
$$<\left(\delta u^N\right),\left(\partial^2 V/\partial u^N\partial x^q\right)\left(\delta x^q\right)> + \tfrac{1}{2}<\left(\delta u^N\right),\left(\partial^2 V/\partial u^N\partial u^N\right)\left(\delta u^N\right)>,$$

where

$\left(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N\right)$ is as defined in Remark 2.5.1,

$$\left(\partial^2 V/\partial u^N\partial u^N\right) = \int_T\{ \left(F^N(t)\right)^T F_{\tilde{u}\tilde{u}}(t)\left(F^N(t)\right) +$$
$$\left(D(t)F^N(t)\right)^T F_{\tilde{y}\tilde{y}}(t)\left(D(t)F^N(t)\right) +$$
$$\int_{t_s}^{t} d\tau_1 \int_{t_s}^{t} d\tau_2 \left(W(t,\tau_1)F^N(\tau_1)\right)^T F_{\tilde{y}\tilde{y}}(t)\left(W(t,\tau_2)F^N(\tau_2)\right) +$$
$$\left(\int_{t_s}^{t} \left(W(t,\tau)F^N(\tau)\right)^T F_{\tilde{y}\tilde{y}}(t)\left(D(t)F^N(t)\right)d\tau \right)^\dagger \}dt +$$
$$\left(D(t_f)F^N(t_f)\right)^T G_{\tilde{y}\tilde{y}}\left(D(t_f)F^N(t_f)\right) +$$
$$\left(\int_T \left(W(t_f,\tau)F^N(\tau)\right)^T G_{\tilde{y}\tilde{y}}\left(D(t_f)F^N(t_f)\right)d\tau \right)^\dagger +$$
$$\int_T d\tau_1 \int_T d\tau_2 \left(W(t_f,\tau_1)F^N(\tau_1)\right)^T G_{\tilde{y}\tilde{y}}\left(W(t_f,\tau_2)F^N(\tau_2)\right) \in M(R^{pN} \to R^{pN}),$$

and where all expansion terms $\left(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N\right)$, $\left(\partial^2 V/\partial u^N\partial u^N\right)$, etc.,

Δ are bounded.

∇ *Proof of Lemma 2.5.1* The control function change from $\tilde{u} = F^N\tilde{u}^N$ to $u = F^N(\tilde{u}+\delta u)^N$ is, from 2.4.1, $\delta u(t) = F^N(t)\delta u^N$, $\forall t \in T$. The output function change due to an initial condition change of $X^q\delta x^q$ from $\tilde{x}_s$ and the control function change $F^N\delta u^N$ is therefore, from convolution-description (2.28):

$$\delta y(t) = \psi(t,t_s)X^q\delta x^q + \int_{t_s}^{t} W(t,\tau)F^N(\tau)\delta u^N d\tau + D(t)F^N(t)\delta u^N, \ \forall t \in T.$$

On using these changes with the expansions of Assumption 2.5.1, the

Δ results of Lemma 2.5.1 emerge.

∇ *Lemma 2.5.2* When Assumption 2.5.1 holds, $\left(\partial^2 V/\partial u^N \partial u^N\right)$ of Lemma 2.5.1 is symmetric, bounded, p.d. and can be written as

$$\left(\partial^2 V/\partial u^N \partial u^N\right) = E + H$$

where

$$E = \begin{pmatrix} E_1 & & & 0 \\ & E_2 & & \\ & & \ddots & \\ 0 & & & E_N \end{pmatrix} \in M(R^{pN} \to R^{pN}),$$

$E_k \in M(R^p \to R^p)$ is as defined in Remark 2.5.1, $\forall k \in I(N)$,

and where

E is symmetric, bounded and p.d., $H \in M(R^{pN} \to R^{pN})$ and is symmetric,

Δ bounded and n.n.d., E_k is bounded and p.d., $\forall k \in I(N)$.

∇ *Proof of Lemma 2.5.2* Suppose Assumption 2.5.1 holds.

Write the RHS of $\left(\partial^2 V/\partial u^N \partial u^N\right)$ of Lemma 2.5.1 as $E + H$ where

$$E = \int_T \left(F^N(t)\right)^T F_{\tilde{u}\tilde{u}}(t)\left(F^N(t)\right)dt. \qquad (2.79)$$

E is therefore symmetric. E is bounded since F^N (of 2.4.1) is bounded, $F_{\tilde{u}\tilde{u}}$ is bounded (Assumption 2.5.1) and $t_f - t_s < \infty$ (from 2.4.2). Also:

$<(\delta u^N),E(\delta u^N)> \; > \; 0$ if $\delta u^N \neq 0$ because

(a) $<(\delta u^N),E(\delta u^N)> \; = \; \int_T <(F^N(t)\delta u^N),F_{\tilde{u}\tilde{u}}(t)(F^N(t)\delta u^N)>dt,$

(b) $F^N\delta u^N$ is not zero everywhere on T if $\delta u^N \neq 0$ since the basis-functions f which constitue F^N (defined in 2.4.1) are linearly-independent,

(c) $F_{\tilde{u}\tilde{u}}$ is p.d. on T (Assumption 2.5.1).

E is therefore p.d.

From 2.4.1, $F^N(t) = (F(1,t) \;\; \ldots \;\; F(N,t)),\; \forall t \in T,$

where $F(k,t) = O(m,p)$ if $t \notin T^k$, $\forall k \in I(N)$. Hence E of (2.79) can be written as

$$E = \begin{bmatrix} \int_{T^1}(F(1,t))^T F_{\tilde{u}\tilde{u}}(t)(F(1,t))dt & & O \\ & \ddots & \\ O & & \int_{T^N}(F(N,t))^T F_{\tilde{u}\tilde{u}}(t)(F(N,t))dt \end{bmatrix}.$$

Clearly E can only be bounded, symmetric and p.d. if

$\int_{T^k}(F(k,t))^T F_{\tilde{u}\tilde{u}}(t)(F(k,t))dt$ is bounded, symmetric and p.d., $\forall k \in I(N)$.

Now $<(\delta u^N),H(\delta u^N)> \; \geq \; 0$ because

(a) $<(\delta u^N),H(\delta u^N)> = \int_T <(\delta y(t)),F_{\tilde{y}\tilde{y}}(t)(\delta y(t))>dt + <(\delta y(t_f)),G_{\tilde{y}\tilde{y}}(\delta y(t_f))>$

when $\delta y(t) \; = \; \int_{t_s}^{t} W(t,\tau)F^N(\tau)\delta u^N d\tau \; + \; D(t)F^N(t)\delta u^N$, $\forall t \in T$, and

(b) $\int_T <(\delta y(t)),F_{\tilde{y}\tilde{y}}(t)(\delta y(t))>dt + <(\delta y(t_f)),G_{\tilde{y}\tilde{y}}(\delta y(t_f))> \; \geq \; 0$ since $F_{\tilde{y}\tilde{y}}$ is n.n.d. on T and $G_{\tilde{y}\tilde{y}}$ is n.n.d. (Assumption 2.5.1).

H is therefore n.n.d. H is bounded since W and D are bounded (from 2.4.2), F^N is bounded (from 2.4.1), $F_{\tilde{y}\tilde{y}}$ and $G_{\tilde{y}\tilde{y}}$ are bounded (Assumption 2.5.1) and $t_f - t_s < \infty$ (from 2.4.2).

We see that $(\partial^2 V/\partial u^N \partial u^N)$ is p.d. since $(\partial^2 V/\partial u^N \partial u^N) = E + H$ where E is p.d. and H is n.n.d. $(\partial^2 V/\partial u^N \partial u^N)$ is symmetric from its definition in Lemma 2.5.1 and H is symmetric since $H = (\partial^2 V/\partial u^N \partial u^N) - E$.

Δ This concludes the proof of Lemma 2.5.2.

∇ *Lemma 2.5.3* Suppose $Z = X + Y$

where

$Z, X, Y \in M(R^i \to R^i)$ and are bounded and symmetric,

X is p.d. and Y is n.n.d.

Then $Z^{-1} = X^{-1} - K$

Δ where K is bounded, symmetric and n.n.d.

∇ *Proof of Lemma 2.5.3* Since Y is real, bounded, symmetric and n.n.d., it has the spectral representation

$$Y = V\Lambda\Lambda^T V^T$$

where

V is a matrix with columns which are the real, orthonormal eigenvectors of Y, and

Λ is a diagonal matrix which has diagonal elements which are the bounded, real positive (or zero) square roots of the eigenvalues of Y.

Then

$$Z = X + V\Lambda\Lambda^T V^T. \tag{2.80}$$

Pre-multiply (2.80) by Z^{-1} and post-multiply the result by X^{-1}, noting that both Z^{-1} and X^{-1} exist since both are p.d., to give

$$X^{-1} = Z^{-1} + Z^{-1}V\Lambda\Lambda^T V^T X^{-1}. \tag{2.81}$$

Post-multiply (2.81) by $V\Lambda$, to give

$$X^{-1}V\Lambda = Z^{-1}V\Lambda\left(I + (V\Lambda)^T X^{-1}(V\Lambda)\right).$$

Since X is p.d., $\left(I + (V\Lambda)^T X^{-1}(V\Lambda)\right)$ is p.d. and therefore invertible, so that

$$Z^{-1}V\Lambda = X^{-1}V\Lambda\left(I + (V\Lambda)^T X^{-1}(V\Lambda)\right)^{-1}. \tag{2.82}$$

On using (2.82) with (2.81) it can be seen that

$$Z^{-1} = X^{-1} - K$$

where $K = \left(\Lambda^T V^T X^{-1}\right)^T \left(I + (V\Lambda)^T X^{-1} (V\Lambda)\right)^{-1} \left(\Lambda^T V^T X^{-1}\right)$.

K is clearly symmetric and is n.n.d. since $\left(I + (V\Lambda)^T X^{-1}(V\Lambda)\right)$ is p.d. That K is bounded follows from the fact that $\left(I + (V\Lambda)^T X^{-1}(V\Lambda)\right)$ and X are p.d., the boundedness of the eigenvalues of Y and the fact that the

Δ columns of V are orthonormal. This concludes the proof of Lemma 2.5.3.

∇ *Lemma 2.5.4* Suppose Assumption 2.5.1 holds. Then we see from Lemmas 2.5.1 and 2.5.2 that the $(\tilde{x}_s + X^q \delta x^q)$-minimal performance index on $L(\mathrm{pN})$ exists for any bounded initial condition $\tilde{x}_s + X^q \delta x^q \in X(q)$ and is

$$V\left(\mathrm{pN};\tilde{x}_s + X^q\delta x^q\right)^* = \min_{\delta u^N} V\left(\mathrm{pN};\tilde{x}_s + X^q\delta x^q, \tilde{u}^N + \delta u^N\right)$$

$$= V\left(\mathrm{pN};\tilde{x}_s,\tilde{u}^N\right) + \left<\left(\partial V(\mathrm{pN};\tilde{x}_s,\tilde{u}^N)/\partial x^q\right), \left(\delta x^q\right)\right>$$

$$+ \tfrac{1}{2}\left<\left(\delta x^q\right), \left(\partial^2 V/\partial x^q \partial x^q\right)\left(\delta x^q\right)\right> - \tfrac{1}{2}\left<\left(g(\delta x^q)\right), \left(\partial^2 V/\partial u^N \partial u^N\right)^{-1}\left(g(\delta x^q)\right)\right>,$$

where $g(\delta x^q) = \left(\partial V(\mathrm{pN};\tilde{x}_s,\tilde{u}^N)/\partial u^N\right) + \left(\partial^2 V/\partial u^N \partial x^q\right)\left(\delta x^q\right)$.

The existence of a unique minimising δu^N confirms the existence of a

Δ unique $(\tilde{x}_s + X^q\delta x^q)$-optimal control function belonging to $L(\mathrm{pN})$.

∇ *Proof of Remark 2.5.1* Suppose Assumption 2.5.1 holds.

From Lemma 2.5.4, the $\tilde{x}_s$-minimal performance index on $L(\mathrm{pN})$ is

$$V\left(\mathrm{pN};\tilde{x}_s\right)^* = V\left(\mathrm{pN};\tilde{x}_s,\tilde{u}^N\right)$$

$$- \tfrac{1}{2}\left<\left(\partial V(\mathrm{pN};\tilde{x}_s,\tilde{u}^N)/\partial u^N\right), \left(\partial^2 V/\partial u^N \partial u^N\right)^{-1}\left(\partial V(\mathrm{pN};\tilde{x}_s,\tilde{u}^N)/\partial u^N\right)\right>. \quad (2.83)$$

From Lemma 2.5.2

$$\left(\partial^2 V/\partial u^N \partial u^N\right) = E + H$$

where E is p.d. and H is n.n.d. On using Lemma 2.5.3 we see that

$$\left(\partial^2 V/\partial u^N \partial u^N\right)^{-1} = E^{-1} - K''$$

where K'' is n.n.d. Then, from (2.83):

$$V\left(\mathrm{pN};\tilde{x}_s\right)^* = V\left(\mathrm{pN};\tilde{x}_s,\tilde{u}^N\right) - \tfrac{1}{2}\left<\left(\partial V(\mathrm{pN};\tilde{x}_s,\tilde{u}^N)/\partial u^N\right), E^{-1}\left(\partial V(\mathrm{pN};\tilde{x}_s,\tilde{u}^N)/\partial u^N\right)\right>$$

$$+ \tfrac{1}{2}\left<\left(\partial V(\mathrm{pN};\tilde{x}_s,\tilde{u}^N)/\partial u^N\right), K''\left(\partial V(\mathrm{pN};\tilde{x}_s,\tilde{u}^N)/\partial u^N\right)\right>$$

$$\geqq V\left(pN;\tilde{x}_s,\tilde{u}^N\right) - \tfrac{1}{2}<\left(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N\right),E^{-1}\left(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N\right)>$$
$$\triangleq \tilde{V}\left(pN;\tilde{x}_s,\tilde{u}^N\right)*. \qquad (2.84)$$

On using the block-diagonal structure for E of Lemma 2.5.2 and the partition for $\left(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N\right)$ of Remark 2.5.1, $\tilde{V}\left(pN;\tilde{x}_s,\tilde{u}^N\right)*$ of (2.84) becomes $\tilde{V}\left(pN;\tilde{x}_s,\tilde{u}^N\right)*$ of Remark 2.5.1.

$\tilde{V}\left(pN;\tilde{x}_s,\tilde{u}^N\right)*$ is, from (2.84) a lower-bound for the $\tilde{x}_s$-minimal performance index on $L(pN)$, evaluated for the (potentially non-optimal) control function $\tilde{u} = F^N\tilde{u}^N \in L(pN)$.

Δ This concludes the proof of Remark 2.5.1.

∇ *Comment 2.5.2* The calculation of the lower-bound for the $\tilde{x}_s$-minimal performance index on $L(pN)$ using Remark 2.5.1 is (potentially) considerably less expensive computationally than the calculation of the $\tilde{x}_s$-minimal performance index on $L(pN)$ using (2.83) because

(a) the determination of the minimal performance index $V\left(pN;\tilde{x}_s\right)*$ using (2.83) requires the evaluation and inversion of the pN x pN matrix $\left(\partial^2 V/\partial u^N \partial u^N\right)$,

(b) the determination of the lower-bound $\tilde{V}\left(pN;\tilde{x}_s,\tilde{u}^N\right)*$ using Remark 2.5.1 requires the evaluation and inversion of N matrices which are each p x p,

(c) the inversion of N matrices which are each p x p is potentially considerably less expensive computationally than the inversion of one pN x pN matrix,

(d) the calculation of the p x p matrix E_k of Remark 2.5.1 for all $k \in I(N)$ requires (potentially) considerably less computational effort than the evaluation of the pN x pN matrix $\left(\partial^2 V/\partial u^N \partial u^N\right)$ of Lemma 2.5.1. Δ

∇ *Comment 2.5.3* We see from (2.83) and (2.84) that

$V(pN;\tilde{x}_s,\tilde{u}^N) - V(pN;\tilde{x}_s)^* =$

$\frac{1}{2}<(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N),(\partial^2 V/\partial u^N \partial u^N)^{-1}(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N)>,$

$V(pN;\tilde{x}_s)^* - \tilde{V}(pN;\tilde{x}_s,\tilde{u}^N)^* =$

$\frac{1}{2}<(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N),K''(\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N)>.$

Hence:

$$\lambda_{\min}(K'')\lambda_{\min}((\partial^2 V/\partial u^N \partial u^N)) \mid V(pN;\tilde{x}_s,\tilde{u}^N) - V(pN;\tilde{x}_s)^* \mid \leq \mid V(pN;\tilde{x}_s)^* - \tilde{V}(pN;\tilde{x}_s,\tilde{u}^N)^* \mid \leq \lambda_{\max}(K'')\lambda_{\max}((\partial^2 V/\partial u^N \partial u^N)) \mid V(pN;\tilde{x}_s,\tilde{u}^N) - V(pN;\tilde{x}_s)^* \mid. \quad (2.85)$$

Relation (2.85) bounds the way in which the lower-bound $\tilde{V}(pN;\tilde{x}_s,\tilde{u}^N)^*$ approaches the $\tilde{x}_s$-minimal performance index on $L(pN)$, $V(pN;\tilde{x}_s)^*$, as the nominal control function $\tilde{u} = F^N\tilde{u}^N \in L(pN)$ approaches the $\tilde{x}_s$-optimal control function belonging to $L(pN)$, i.e. as $V(pN;\tilde{x}_s,\tilde{u}^N)$ approaches $V(pN;\tilde{x}_s)^*$. When $\tilde{u}$ is the $\tilde{x}_s$-optimal control function belonging to $L(pN)$, $V(pN;\tilde{x}_s,\tilde{u}^N) = V(pN;\tilde{x}_s)^*$ so that, from (2.85), the lower-bound $\tilde{V}(pN;\tilde{x}_s,\tilde{u}^N)^*$ for the $\tilde{x}_s$-minimal performance index on $L(pN)$ is equal
Δ to the $\tilde{x}_s$-minimal performance index on $L(pN)$, $V(pN;\tilde{x}_s)^*$.

We next prove

∇ *Remark 2.5.2* Consider the control law

$$u(t) = F^N(t)u^N(\tilde{x}_s+X^q\delta x^q), \ \forall t \in T, \quad (2.86)$$

which determines control functions belonging to $L(pN)$ as a function of initial conditions $x_s = \tilde{x}_s + X^q\delta x^q \in X(q)$ for the optimisation problem of 2.4.2 where

$$u^N(\tilde{x}_s+X^q\delta x^q) = \tilde{u}^N + \alpha^N + (\beta^N)(\delta x^q) \quad (2.87)$$

and

$$u^N, \alpha^N \in R^{pN}, \quad (\beta^N) \in M(R^q \to R^{pN}).$$

Suppose that

(1) Assumption 2.5.1 holds,

(2) for the initial condition $\tilde{x}_s$, the control function $u = F^N u^N(\tilde{x}_s)$ yielded by control law (2.86) is applied to the dynamical system considered in 2.4.2 and yields a gradient $\left(\partial V(pN;\tilde{x}_s,u^N(\tilde{x}_s))/\partial u^N\right) \in R^{pN}$ which is partitioned as $\left(g(\tilde{x}_s,1)^T \;\ldots\; g(\tilde{x}_s,N)^T \right)^T$,

(3) for each $i \in I(q)$, the initial condition $\tilde{x}_s$ is replaced by $x_i = \tilde{x}_s + X_i|\delta\hat{x}_i^q|$ (on the boundary of $\overline{X}(q)$ of Definition 1.3.2) and the corresponding control function $u = F^N u^N(x_i)$ yielded by control law (2.86) is applied to the dynamical system considered in 2.4.2 and yields a gradient $\left(\partial V(pN;x_i,u^N(x_i))/\partial u^N\right) \in R^{pN}$ which is partitioned as $\left(g(x_i,1)^T \;\ldots\; g(x_i,N)^T \right)^T$,

where

$g(x_s,k),\ g(x_i,k) \in R^p,\ \forall k \in I(N),\ \forall i \in I(q),$

For $x_s = \tilde{x}_s$ and $x_s = x_i$, $\forall i \in I(q)$, $\left(\partial V(pN;x_s,u^N(x_s))/\partial u^N\right)$ can be determined using (2.78) with $\tilde{x}_s$ replaced by x_s and $\tilde{u}^N$ replaced by $u^N(x_s)$ and $F_{\tilde{y}}$, $F_{\tilde{u}}$ and $G_{\tilde{y}}$ evaluated for the control function $u = F^N u^N(x_s)$ and the associated response y of (2.28) for the initial condition x_s.

Then:

$V\left(pN;\tilde{x}_s+X^q\delta x^q\right)^* \geq V\left(pN;\tilde{x}_s+X^q\delta x^q,u^N(\tilde{x}_s+X^q\delta x^q)\right) - \varepsilon^*$

for all $\tilde{x}_s + X^q\delta x^q \in \overline{X}(q)$,

where

$V\left(pN;\tilde{x}_s+X^q\delta x^q\right)^*$ is the $(\tilde{x}_s+X^q\delta x^q)$-minimal performance index on on $L(pN)$,

$V\left(pN;\tilde{x}_s+X^q\delta x^q,u^N(\tilde{x}_s+X^q\delta x^q)\right)$ is the performance index for the initial condition $\tilde{x}_s + X^q\delta x^q$ resulting from control law (2.86),

$$\varepsilon^* = \tfrac{1}{2}\sum_{k=1}^{N} <(g(\tilde{x}_s,k)),(E_k)^{-1}(g(\tilde{x}_s,k))>$$

$$+ \sum_{i=1}^{q} \Big| \sum_{k=1}^{N} <(g(\tilde{x}_s,k)),(E_k)^{-1}(g(x_i,k)-g(\tilde{x}_s,k))> \Big|$$

$$+ \tfrac{1}{2}\sum_{j=1}^{q}\sum_{i=1}^{q} \Big| \sum_{k=1}^{N} <(g(x_j,k)-g(\tilde{x}_s,k)),(E_k)^{-1}(g(x_i,k)-g(\tilde{x}_s,k))> \Big|,$$

Δ $$E_k = \int_{T^k}(F(k,t))^T F_{\tilde{u}\tilde{u}}(t)(F(k,t))dt, \ \forall k \in I(N).$$

∇ <u>*Comment 2.5.4*</u> It will be observed that the result of Remark 2.5.2 gives a lower-bound for the x_s-minimal performance index on L(pN) for all $x_s \in \overline{X}(q)$ in terms of the performance index yielded by the

Δ (potentially non-optimal) control law of (2.86).

∇ <u>*Proof of Remark 2.5.2*</u> Suppose Assumption 2.5.1 holds. Then, from Lemma 2.5.1 and (2.87):

$$V(pN;\tilde{x}_s+X^q\delta x^q,u^N(\tilde{x}_s+X^q\delta x^q)+\Delta u^N) = V(pN;\tilde{x}_s+X^q\delta x^q,u^N(\tilde{x}_s+X^q\delta x^q))$$

$$+ <\{(\partial V(pN;\tilde{x}_s,u^N)/\partial u^N) + (\partial^2 V/\partial u^N\partial u^N)(\alpha^N)\},(\Delta u^N)>$$

$$+ <(\Delta u^N),\{(\partial^2 V/\partial u^N\partial x^q) + (\partial^2 V/\partial u^N\partial u^N)(\beta^N)\}(\delta x^q)>$$

$$+ \tfrac{1}{2}<(\Delta u^N),(\partial^2 V/\partial u^N\partial u^N)(\Delta u^N)>.$$

Hence

$$(\partial V(pN;\tilde{x}_s+X^q\delta x^q,u^N(\tilde{x}_s+X^q\delta x^q))/\partial u^N) = g^0 + \sum_{i=1}^{q} g^i\delta x_i^q$$

where

$$g^0 = (\partial V(pN;\tilde{x}_s,\tilde{u}^N)/\partial u^N) + (\partial^2 V/\partial u^N\partial u^N)(\alpha^N),$$

$$g^i = \text{column } i \text{ of } \{(\partial^2 V/\partial u^N\partial x^q) + (\partial^2 V/\partial u^N\partial u^N)(\beta^N)\}, \ \forall i \in I(q),$$

$$\delta x^q = (\delta x_1^q \ \ldots \ \delta x_q^q)^T \text{ (recall Definition 1.3.1).}$$

Therefore, on using the partitions of Remark 2.5.2:

$$(\partial V(pN;\tilde{x}_s,\tilde{u}^N(\tilde{x}_s))/\partial u^N) = (g(\tilde{x}_s,1)^T \ \ldots \ g(\tilde{x}_s,N)^T)^T = g^0, \quad (2.88)$$

$$\left(\partial V(pN;x_i,u^N(x_i))/\partial u^N\right) = \left(g(x_i,1)^T \;\ldots\; g(x_i,N)^T\right)^T$$
$$= g^0 + g^i|\delta\hat{x}_i^q|, \;\forall i \in I(q). \qquad (2.89)$$

Thus

$$g^i|\delta\hat{x}_i^q| = \left(\{g(x_i,1)-g(\tilde{x}_s,1)\}^T \;\ldots\; \{g(x_i,N)-g(\tilde{x}_s,N)\}^T\right)^T,$$
$$\forall i \in I(q). \qquad (2.90)$$

The $(\tilde{x}_s+X^q\delta x^q)$-minimal performance index on $L(pN)$ is clearly given by

$$V\left(pN;\tilde{x}_s+X^q\delta x^q\right)* = \min_{\Delta u^N} V\left(pN;\tilde{x}_s+X^q\delta x^q,u^N(\tilde{x}_s+X^q\delta x^q)+\Delta u^N\right)$$
$$= V\left(pN;\tilde{x}_s+X^q\delta x^q,u^N(\tilde{x}_s+X^q\delta x^q)\right)$$
$$- \tfrac{1}{2}<\left(g^0 + \sum_{j=1}^{q} g^j\delta x_j^q\right),\left(\partial^2 V/\partial u^N\partial u^N\right)^{-1}\left(g^0 + \sum_{i=1}^{q} g^i\delta x_i^q\right)>.$$

On using the arguments involved in the Proof of Remark 2.5.1, it can be seen that

$$V\left(pN;\tilde{x}_s+X^q\delta x^q\right)* \geqq V\left(pN;\tilde{x}_s+X^q\delta x^q,u^N(\tilde{x}_s+X^q\delta x^q)\right)$$
$$- \tfrac{1}{2}<\left(g^0 + \sum_{j=1}^{q} g^j\delta x_j^q\right),E^{-1}\left(g^0 + \sum_{i=1}^{q} g^i\delta x_i^q\right)>$$
$$= V\left(pN;\tilde{x}_s+X^q\delta x^q,u^N(\tilde{x}_s+X^q\delta x^q)\right) - \tfrac{1}{2}<\left(g^0\right),E^{-1}\left(g^0\right)>$$
$$- \sum_{i=1}^{q} <\left(g^0\right),E^{-1}\left(g^i\delta x_i^q\right)> - \tfrac{1}{2}\sum_{j=1}^{q}\sum_{i=1}^{q} <\left(g^j\delta x_j^q\right),E^{-1}\left(g^i\delta x_i^q\right)>$$

where E is that of Lemma 2.5.2.

For all initial conditions $x_s = \tilde{x}_s + X^q\delta x^q \in \bar{X}(q)$, $|\delta x_i^q| \leqq |\delta\hat{x}_i^q|$, $\forall i \in I(q)$ (recall Definition 1.3.2), so that:

$$V\left(pN;\tilde{x}_s+X^q\delta x^q\right)* \geqq V\left(pN;\tilde{x}_s+X^q\delta x^q,u^N(\tilde{x}_s+X^q\delta x^q)\right) - \tfrac{1}{2}<\left(g^0\right),E^{-1}\left(g^0\right)>$$
$$- \sum_{i=1}^{q} \left|<\left(g^0\right),E^{-1}\left(g^i|\delta\hat{x}_i^q|\right)>\right| - \tfrac{1}{2}\sum_{j=1}^{q}\sum_{i=1}^{q} \left|<\left(g^j|\delta\hat{x}_j^q|\right),E^{-1}\left(g^i|\delta\hat{x}_i^q|\right)>\right|,$$
$$\forall\tilde{x}_s+X^q\delta x^q \in \bar{X}(q). \qquad (2.91)$$

On using the block-diagonal form of E of Lemma 2.5.2 with (2.88) and (2.90), the result of (2.91) becomes that of Remark 2.5.2.

Δ This concludes the proof of Remark 2.5.2.

2.5.2 ε-Approximations

Consider a linear manifold $L(pN)$ defined in the same way that $L(\mathrm{pN})$ was defined in 2.4.1. Suppose $pN > \mathrm{pN}$ and $L(\mathrm{pN}) \subset L(pN)$, so that $L(pN)$ is a larger linear manifold than $L(\mathrm{pN})$. Denote the x_s-minimal performance index on $L(\mathrm{pN})$ for the optimisation problem of 2.4.2 by $V(\mathrm{pN};x_s)^*$, and that on $L(pN)$ by $V(pN;x_s)^*$.

∇ *Definition 2.5.1* For the optimisation problem of 2.4.2, the x_s-optimal control function belonging to $L(\mathrm{pN})$ will be referred to as an $\varepsilon(x_s)$-approximation to the x_s-optimal control function belonging to $L(pN)$ if, for $\varepsilon > 0$:

Δ $$\left| V(\mathrm{pN};x_s)^* - V(pN;x_s)^* \right| \leqq \varepsilon.$$

∇ *Definition 2.5.2* The control law which determines x_s-optimal control functions belonging to $L(\mathrm{pN})$ as a function of $x_s \in X(q)$ for the optimisation problem of 2.4.2 will be referred to as an $\varepsilon(\overline{X}(q))$-approximation to the corresponding control law for $L(pN)$ if, for $\varepsilon > 0$:

Δ $$\left| V(\mathrm{pN};x_s)^* - V(pN;x_s)^* \right| \leqq \varepsilon, \ \forall x_s \in \overline{X}(q).$$

Definitions 2.5.1 and 2.5.2 provide a measure of the closeness, performance-index wise, of optimal control functions and optimal control laws for $L(\mathrm{pN})$ to those for $L(pN)$.

If the x_s-optimal control function belonging to the smaller linear manifold $L(\mathrm{pN})$ is available (perhaps computed using the algorithm of 2.4) and Assumption 2.5.1 holds, we can calculate a lower-bound for the x_s-minimal performance index on $L(pN)$ using Remark 2.5.1 by replacing $\tilde{u}$ of Remark 2.5.1 by the x_s-optimal control function belonging to $L(\mathrm{pN})$, by replacing $\tilde{x}_s$ of Remark 2.5.1 by x_s and by replacing $L(\mathrm{pN})$ of Remark

2.5.1 by $L(pN)$. Denote the resulting lower-bound for the x_s-minimal performance index on $L(pN)$ by $\tilde{V}(pN;x_s)^*_{pN}$, where the subscript pN denotes that the lower-bound for the x_s-minimal performance index on $L(pN)$ is evaluated for the x_s-optimal control function belonging to $L(pN)$. Since $\tilde{V}(pN;x_s)^*_{pN} \leq V(pN;x_s)^*$, we have

∇ *Remark 2.5.3* The x_s-optimal control function belonging to $L(pN)$ is an $\varepsilon(x_s)$-approximation to the x_s-optimal control function belonging to $L(pN)$, for $\varepsilon > 0$, if:

Δ $$| V(pN;x_s)^* - \tilde{V}(pN;x_s)^*_{pN}| \leq \varepsilon.$$

Since $\tilde{V}(pN;x_s)^*_{pN}$ is relatively easy to compute, we can now determine with relatively little computational effort whether, for some pre-chosen $\varepsilon > 0$, the x_s-optimal control function belonging to $L(pN)$ is an $\varepsilon(x_s)$-approximation to the x_s-optimal control function belonging to $L(pN)$, and thus whether the former can be considered to be an adequate approximation to the latter, without the computational expense involved in determining the x_s-optimal control function belonging to $L(pN)$ and the associated optimal performance index on $L(pN)$. This information can be used to decide whether, when the x_s-optimal control function belonging to $L(pN)$ has been determined, the extra computational effort which would be required to optimise on $L(pN)$ would be likely to lead to a worthwhile performance improvement (i.e. a worthwhile performance index decrease). An upper-bound for the performance index decrease is clearly

$$\varepsilon(pN;x_s)^*_{pN} = | V(pN;x_s)^* - \tilde{V}(pN;x_s)^*_{pN} |.$$

This information is of considerable use computationally and would not be available with so little computational expense without our lower-bound result of Remark 2.5.1.

∇ *Remark 2.5.4* It can be seen that the result of Remark 2.5.2 can be used when

(a) $L(\mathrm{pN})$ is replaced by $L(pN)$ in Remark 2.5.2,

(b) control law (2.86) is the control law which determines optimal control functions belonging to $L(\mathrm{pN})$ as a function of initial conditions $x_s \in X(\mathrm{q})$ (it is clear that the control law used in stage 6) of the algorithm of 2.5.4 could be written in the form of (2.86)).

Then $V\left(pN;\tilde{x}_s+X^q\delta x^q,u^N(\tilde{x}_s+X^q\delta x^q)\right)$ of Remark 2.5.2 is the $(\tilde{x}_s+X^q\delta x^q)$-minimal performance index on $L(\mathrm{pN})$, $V\left(\mathrm{pN};\tilde{x}_s+X^q\delta x^q\right)^*$, and the result of Remark 2.5.2 states that:

$$V\left(pN;\tilde{x}_s+X^q\delta x^q\right)^* \geqq V\left(\mathrm{pN};\tilde{x}_s+X^q\delta x^q\right)^* - \varepsilon^*$$

for all initial conditions $x_s = \tilde{x}_s + X^q\delta x^q \in \bar{X}(\mathrm{q})$,

i.e. that the optimal control law which determines optimal control functions belonging to $L(\mathrm{pN})$ as a function of initial conditions $x_s \in X(\mathrm{q})$ is an $\varepsilon\left(\bar{X}(\mathrm{q})\right)$-approximation to the optimal control law which determines

Δ optimal control functions belonging to $L(pN)$ for all $\varepsilon \geqq \varepsilon^*$.

We can thus determine whether, for any pre-chosen $\varepsilon > 0$, the optimal control law determining optimal control functions belonging to $L(\mathrm{pN})$ is an $\varepsilon\left(\bar{X}(\mathrm{q})\right)$-approximation to the optimal control law determining optimal control functions belonging to $L(pN)$ <u>without</u> having to determine the optimal control law for $L(pN)$.

This enables us to decide whether the computational expense involved in determining the optimal control law for the larger linear manifold $L(pN)$ would be worthwhile, performance-index wise, when the optimal control law for the smaller linear manifold, $L(\mathrm{pN})$, is already available and optimal control functions are of interest for all initial conditions belonging to $\overline{X}(q)$.

2.5.3 Concluding Comments

We have developed in 2.5.1 results which yield, with relatively little computational expense, lower-bounds for the x_s-minimal performance index on $L(\mathrm{pN})$ for any particular initial condition x_s (Remark 2.5.1) and for all initial conditions belonging to $\overline{X}(q)$ (Remark 2.5.2). These results are believed to be novel. We have explained in 2.5.2 how the results can be used to decide whether, when optimisation on $L(\mathrm{pN})$ has been achieved, optimisation on a linear manifold $L(pN)$ of larger dimension than $L(\mathrm{pN})$ could be profitable, performance-index wise, before actually undertaking the computational expense involved in optimisation on the larger linear manifold. This is believed to be novel and to be of considerable value from the computational point of view.

2.6 A Computed Example

The linear dynamical system which is considered is an axially-symmetric idealisation (sketched in Fig. 2.1, below) of an existing tube and shell counter-flow heat-exchanger.

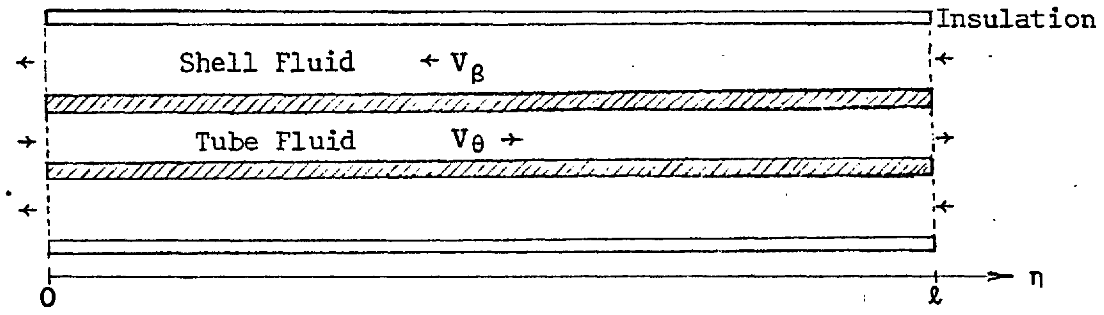

Fig 2.1 The Idealised Heat-exchanger

The shell is insulated on the outside. The temperature (^{o}F) and speed of the shell fluid are denoted by β and V_β, and those of the tube fluid by θ and V_θ. The temperature of the inter-fluid wall is denoted by κ. Under obvious assumptions, the idealisation is described by the following partial differential equations:

$$\begin{aligned}(\partial/\partial t + V_\theta\partial/\partial\eta + m_1)\theta(\eta,t) &= m_1\kappa(\eta,t),\\ (\partial/\partial t - V_\beta\partial/\partial\eta + m_4)\beta(\eta,t) &= m_4\kappa(\eta,t),\\ (\partial/\partial t + m_2 + m_3)\kappa(\eta,t) &= m_2\theta(\eta,t) + m_3\beta(\eta,t),\\ \forall\eta &\in (0,\ell),\end{aligned} \tag{2.92}$$

with boundary conditions

$\theta(0,t) = \theta_{in}(t)$, an uncontrolled input,

$\beta(\ell,t) = u(t)$, a controlled input (note that $m = 1$ here).

The output of interest is

$y(t) = \theta(\ell,t)$ (note that $r = 1$ here).

The parameter values used were:

$V_\theta = 0.356$ ft sec^{-1}		$m_1 = 0.712$ sec^{-1}	
$V_\beta = 0.0150$ ft sec^{-1}		$m_2 = 1.24$ sec^{-1}	
$\ell = 1.59$ ft		$m_3 = 0.404$ sec^{-1}	
		$m_4 = 0.00648$ sec^{-1}.	

The performance index considered was

$$V = \int_T \{ < (y(t)-y_d(t)), Q(y(t)-y_d(t)) > + < (u(t)), R(u(t)) > \} dt \qquad (2.93)$$

where $Q = 10$, $R = 0.5$,

$y_d(t) = (\Xi_1(t) \ \Xi_2(t))(b_1 \ b_2)^T$, $\forall t \in T$,

$\Xi_1 = 1$ on $\{0,100)$, $= 0$ on $\{100,200\}$,

$\Xi_2 = 0$ on $\{0,100)$, $= 1$ on $\{100,200\}$,

$T = \{0,200\}$,

all times are in seconds.

The uncontrolled input $\theta_{in}(t)$ had relatively small effect on the output y on T for all $t < -100$, and was therefore ignored. θ_{in} on $\{-100,200\}$ was assumed to be given exactly by

$$\theta_{in}(t) = (f_1(t) \ \ldots \ f_6(t))(x_1 \ \ldots \ x_6)^T,$$

where

$f_1(t) = 1$ on $\{-100,0)$, $=0$ on $\{0,200\}$,

$f_2(t) = 0.01(t+100)$ on $\{-100,0)$, $= 0$ on $\{0,200\}$,

$f_3(t) = 1$ on $\{0,100)$, $= 0$ on $\{-100,0)$ and $\{100,200\}$,

$f_4(t) = 0.01t$ on $\{0,100)$, $= 0$ on $\{-100,0)$ and $\{100,200\}$,

$f_5(t) = 1$ on $\{100,200\}$, $= 0$ on $\{-100,100)$,

$f_6(t) = 0.01(t-100)$ on $\{100,200\}$, $= 0$ on $\{-100,100)$.

Following the comments of 2.4.7, the costed output $(y-y_d)$ on T

was considered to depend on the control function $\boldsymbol{u}$ on T and on the following 'initial condition' vector:

$$x_s = (b_1\ b_2\ x_1\ x_2\ x_3\ x_4\ x_5\ x_6)^T. \tag{2.94}$$

The linear manifolds $L(pN)$, of 2.4.1, on which optimisation was considered were each spanned by the following N basis-functions:

$$\begin{aligned} f_{1,j}(t) &= 1, \ \forall t \in T^j, \\ &= 0, \ \forall t \notin T^j, \\ \forall j &\in I(N), \end{aligned}$$

where $T^1 = \{0,200/N\}$, $T^i = (200(i-1)/N, 200i/N\}$, $\forall i \in I(2,N)$. Since one basis-function was non-zero on each interval T^j, p of 2.4.1 was one.

A convolution-description for the heat-exchanger was obtained for the initial condition x_s of (2.94) and control functions belonging to $L(pN)$ by

(a) determining the frequency-domain transfer functions $y(j\omega)/u(j\omega)$ and $y(j\omega)/\theta_{in}(j\omega)$ from the partial differential equation description of (2.92), and

(b) calculating, for each case, the required time-domain results by evaluating numerically the inversion integral which maps from the frequency domain to the time-domain for that case.

The second-order, Dynamic Programming based, optimisation algorithm of 2.4 was then used to determine the optimal control function belonging to $L(pN)$ as a function of initial conditions $x_s \in R^8$ by setting (in 2.4) $\tilde{x}_s = O(8,1)$, q = 8 and $X^q = I(8,8)$, for N = 1, 2, 4, 10 and 20. Trapezoidal integration was used with a step-length of one second.

For pN = 1, 2, 4, 10, 20 and each of the following initial conditions

$$x_s^1 = \left(40 \quad 40 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0\right)^T$$
$$x_s^2 = \left(20 \quad 40 \; -20 \; -20 \; -20 \; -20 \; -10 \; -30\right)^T$$
$$x_s^3 = \left(20 \quad 10 \; -10 \; -10 \; -20 \; -20 \quad 40 \quad 20\right)^T$$
$$x_s^4 = \left(0 \quad 0 \; -10 \; -10 \; -20 \; -20 \quad 40 \quad 20\right)^T,$$

the optimal control function belonging to $L(\mathrm{pN})$ was then calculated and applied to the convolution-description of the (idealised) heat-exchanger. For each case the results were used to determine:

(a) $V\left(\mathrm{pN};x_s^i\right)^*$, the x_s^i-minimal performance index on $L(\mathrm{pN})$, which was sufficiently close to the value which was predicted using the results of 2.4.5 as to be indistinguishable on Fig. 2.2 (below), and

(b) $\tilde{V}\left(\mathrm{p}N;x_s^i\right)^*_{\mathrm{pN}}$, a lower-bound for the x_s^i-minimal performance index on $L(\mathrm{p}N)$ for $N = 100$, evaluated for the calculated x_s^i-optimal control function belonging to $L(\mathrm{pN})$ using the results of 2.5.

The results are shown in Fig. 2.2 and reveal that the x_s-optimal control function belonging to $L(20)$ is, for each of the above initial conditions, an excellent approximation, performance-index wise, to that which would be obtained after x_s-optimising on $L(100)$. The results thus reveal that, for each of the above initial conditions, there would be negligible return, performance-index wise, for the computational effort which would be required to optimise on $L(100)$ when the optimal control function belonging to $L(20)$ for each initial condition is already available. Since the computational effort required to achieve optimisation on a linear manifold increases with the dimension of the linear manifold, this information is valuable from the computational point of view and demonstrates the usefulness of the lower-bound result of Remark 2.5.1.

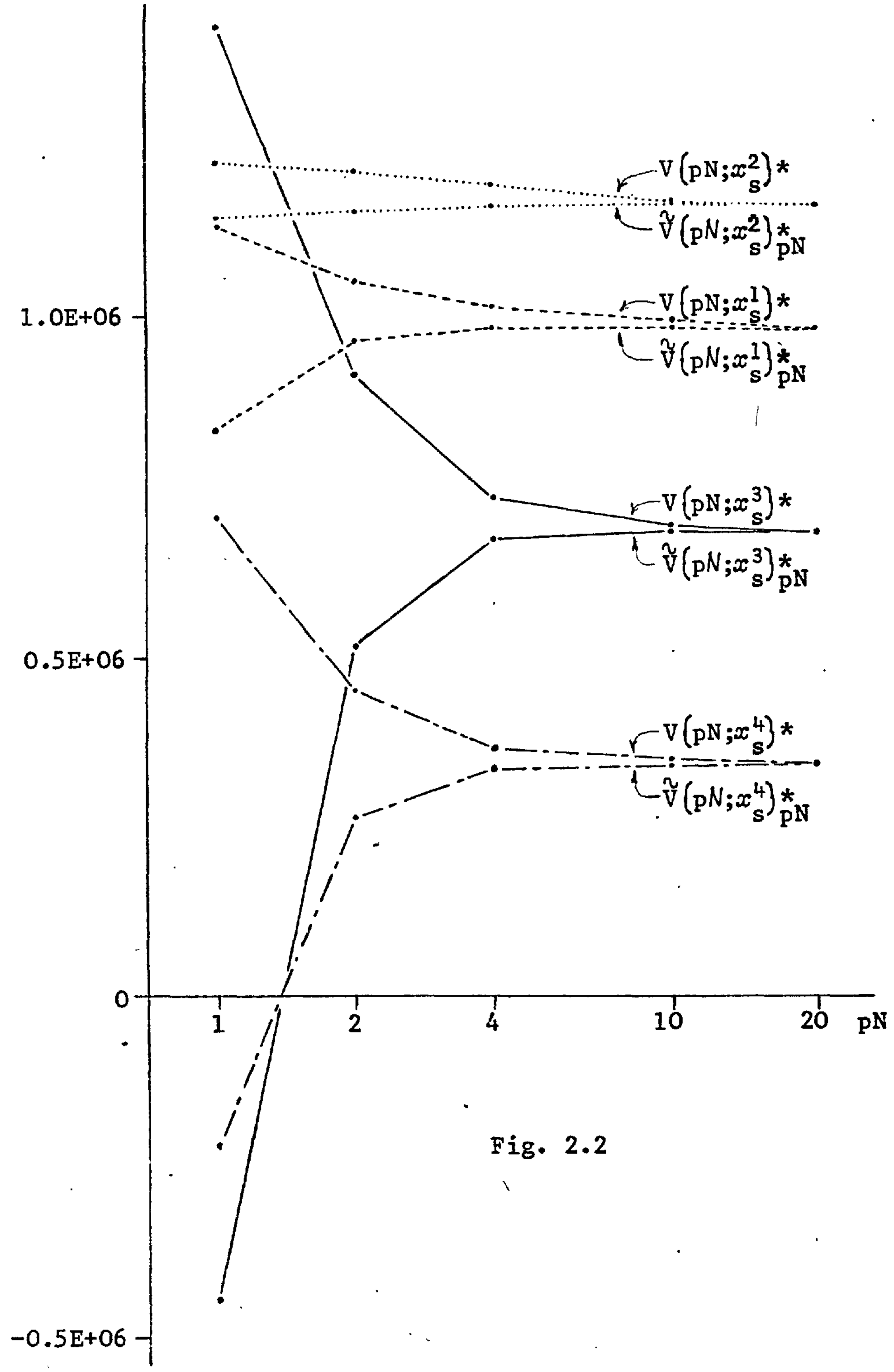

Fig. 2.2

Chapter 3 : Optimal Control Function Synthesis using a Gradient-Decomposition Approach

3.1 Summary

The main optimisation problem considered is stated in 3.2. Some preliminary definitions and discussion are contained in 3.3. In 3.4 are considered lower-bounds for the x_s-minimal performance index on the control space in the gradient-decomposition context of this chapter, as well as $\varepsilon(x_s)$-approximations to the x_s-optimal control function belonging to the control space. Gradient function decomposition is considered further in 3.5. An iterative algorithm for determining an $\varepsilon(x_s)$-approximation to the x_s-optimal control function belonging to the control space is developed in 3.6 and its effectiveness at control function optimisation is compared with that of the steepest-descent algorithm in 3.7. The results of the application of the algorithm and lower-bound results to numerical examples are presented in 3.8. In 3.9 we consider optimal control function determination as a function of initial conditions and develop a simple procedure for determining an $\varepsilon(\overline{X}(q))$-approximation to the optimal control law which determines optimal control functions belonging to the control space as a function of initial conditions belonging to $X(q)$. Some related computational results are presented in 3.10. The approach of this chapter is applied in 3.11 to x_s-optimal control function determin ation for nonlinear systems with (potentially) non-quadratic and nonlinear performance index terms. Computational results which demonstrate the superiority of the resulting algorithm over the steepest-descent and conjugate-gradient algorithms are presented in 3.12. Some concluding comments are contained in 3.13.

3.2 An Optimisation Problem

The optimisation problem considered throughout most of Chapter 3 is: minimise with respect to the control function u on T the scalar performance index

$$V(x_s,u) = \int_T F\big(y(t),u(t),t\big)dt + G\big(y(t_f)\big) \tag{3.1}$$

for a linear dynamical system which can be described by

$$y(t) = \psi(t,t_s)x_s + \int_{t_s}^{t} W(t,\tau)u(\tau)d\tau, \quad \forall t \in T, \tag{3.2}$$

where ψ, W and u are continuous and bounded,

$x_s = \tilde{x}_s + X^q\delta x^q \in X(q)$, the initial condition

$y(t) \in R^r$, $\forall t \in T$, the costed output,

$u(t) \in R^m$, $\forall t \in T$, the control,

$T = \{t_s,t_f\}$, $t_f - t_s < \infty$,

and, for any bounded initial condition x_s and considered control function u

F has first- and second-derivatives with respect to y and u which are continuous on T and are bounded, and has zero higher-order derivatives,

G has bounded first- and second-derivatives with respect to $y(t_f)$ and has zero higher-order derivatives.

Under the above conditions, we see from 2.3 that the gradient function $\big(\partial V(x_s,u)/\partial u\big)$ can be computed using Remark 2.3.1 for any bounded initial condition x_s and any bounded and continuous control function u, and is bounded and continuous on T.

It will become apparent that the results which will be obtained will be applicable to optimisation problems similar to the above problem save in that they are defined when ψ, W and u are continuous almost everywhere and are bounded or are defined for discrete-time or distributed parameter-systems.

3.3 Optimisation on Translated Linear Manifolds

In this section the k-dimensional translated linear manifold $U(1,k)''$ of the control space for the optimisation problem of 3.2 is defined and the x_s-optimal control function belonging to it is determined, as well as the x_s-minimal performance index on $U(1,k)''$.

∇ *Definition 3.3.1* Define $F(k) = \{\delta_1, .., \delta_k\}$ to be a set of k bounded and m-vector valued basis-functions δ, each with domain T and each continuous on that domain, which are orthonormal in that

Δ $$\int_T <\delta_i(t), \delta_j(t)> dt = \delta(i,j), \ \forall i,j \in I(k).$$

∇ *Definition 3.3.2* Denote by U the linear space of all bounded control functions $u: T \rightarrow R^m$ which are continuous on their domain T. Define $U(1,j)$ to be that linear manifold of U which is spanned by $\delta_1, .., \delta_j$, $\forall j \in I(k)$. Also define $U(j)$ to be that linear manifold

Δ of U which is spanned by δ_j, $\forall j \in I(k)$.

∇ *Definition 3.3.3* Define $U(1,j)''$ to be the translate of $U(1,j)$ along an initial control function $u_1 \in U$, so that

$$U(1,j)'' = \{u : u = u_1 + s; \ \forall s \in U(1,j)\}, \ \forall j \in I(k).$$

Δ Define $U(j)''$ similarly, $\forall j \in I(k)$.

∇ *Definition 3.3.4* For all $j \in I(k)$, a control function change δu belonging to $U(1,j)$ will be said to be exactly characterised by the components δu^j (of basis-functions $\delta_1, .., \delta_j$) if

$$\delta u = F_j \delta u^j \quad (\text{i.e. } \delta u(t) = F_j(t)\delta u^j,\ \forall t \in T),$$

where $F_j = \left(f_1 \ \ldots \ f_j\right)$ (i.e. $F_j(t) = \left(f_1(t) \ \ldots \ f_j(t)\right)$, $\forall t \in T$),

Δ $$\delta u^j = \left(\delta u_1 \ \ldots \ \delta u_j\right)^T \in R^j,$$

∇ *Definition 3.3.5* For all $j \in I(k)$, define $G(1,j)$ to be that linear manifold of the linear space of all gradient functions $\left(\partial V(x_s,u)/\partial u\right)$ which is spanned by $f_1, .., f_j$. Also, define $G(1)$ to be that linear

Δ manifold which is spanned by f_1.

∇ *Definition 3.3.6* For all $j \in I(k)$, a gradient function $\left(\partial V(x_s,u)/\partial u\right)$ belonging to $G(1,j)$ will be said to be exactly characterised by the components $g^j(x_s,u)$ (of basis-functions $f_1, .., f_j$) if

$$\left(\partial V(x_s,u)/\partial u(t)\right) = F_j(t) g^j(x_s,u),\ \forall t \in T,$$

Δ where $g^j(x_s,u) = \left(g_1(x_s,u) \ \ldots \ g_j(x_s,u)\right)^T \in R^j.$

∇ *Definition 3.3.7* For all $j \in I(k)$, a gradient function $\left(\partial V(x_s,u)/\partial u\right)$ will be said to contain components $g^j(x_s,u)$ (of basis-functions $f_1, .., f_j$), or it will be said that components $g^j(x_s,u)$ are present in $\left(\partial V(x_s,u)/\partial u\right)$, if $g^j(x_s,u)$ minimises

$$\int_T \langle\{\left(\partial V(x_s,u)/\partial u(t)\right) - F_j(t) g^j\}, \{\left(\partial V(x_s,u)/\partial u(t)\right) - F_j(t) g^j\}\rangle dt$$

with respect to $g^j \in R^j$, i.e. if

Δ $$g^j(x_s,u) = \int_T \left(F_j(t)\right)^T \left(\partial V(x_s,u)/\partial u(t)\right) dt. \tag{3.3}$$

∇ *Comment 3.3.1* If $\left(\partial V(x_s,u)/\partial u\right) \in G(1,j)$, the components $g^j(x_s,u)$ of (3.3) exactly characterise it, so that

Δ $$\left(\partial V(x_s,u)/\partial u\right) = F_j g^j(x_s,u).$$

For the optimisation problem of 3.2, it can be seen that the performance index $V(x_s,u)$ can be expanded about $V(\tilde{x}_s,u_1)$ in terms of $\delta x_s = x_s - \tilde{x}_s \in R^n$ and $\delta u = u - u_1 \in U$ to give

$$\begin{aligned}
V(\tilde{x}_s+\delta x_s,u_1+\delta u) &= V(\tilde{x}_s,u_1) + <(\partial V(\tilde{x}_s,u_1)/\partial x_s),(\delta x_s)> \\
&+ \tfrac{1}{2}<(\delta x_s),(\partial^2 V/\partial x_s \partial x_s)(\delta x_s)> + \int_T <(\partial V(\tilde{x}_s,u_1)/\partial u(t)),(\delta u(t))>dt \\
&+ \tfrac{1}{2}\int_T d\tau_1 \int_T d\tau_2 <(\delta u(\tau_1)),(\partial^2 V/\partial u(\tau_1)\partial u(\tau_2))(\delta u(\tau_2))> \\
&+ \int_T <(\delta u(\tau)),(\partial^2 V/\partial u(\tau)\partial x_s)(\delta x_s)>d\tau \qquad (3.4)
\end{aligned}$$

where $x_s \in R^n$, $u_1 \in U$ and every contribution to the RHS of (3.4) is bounded for bounded δx_s and δu.

For an initial condition change, from $\tilde{x}_s$, of $\delta x_s = X^q \delta x^q \in X(q)$ and a control function change, from u_1, of $\delta u = F_j \delta u^j \in U(1,j)$, $\forall j \in I(k)$, we see from (3.4) that

$$\begin{aligned}
V(\tilde{x}_s+X^q\delta x^q,u_1+F_j\delta u^j) &= V(\tilde{x}_s,u_1) + <(\partial V(\tilde{x}_s,u_1)/\partial x^q),(\delta x^q)> \\
&+ \tfrac{1}{2}<(\delta x^q),(\partial^2 V/\partial x^q \partial x^q)(\delta x^q)> + <(\partial V(\tilde{x}_s,u_1)/\partial u^j),(\delta u^j)> \\
&+ \tfrac{1}{2}<(\delta u^j),T(1,j\to 1,j)(\delta u^j)> + <(\delta u^j),P(1,q\to 1,j)(\delta x^q)>, \qquad (3.5)
\end{aligned}$$

where

$$\begin{aligned}
(\partial V(\tilde{x}_s,u_1)/\partial x^q) &= (X^q)^T(\partial V(\tilde{x}_s,u_1)/\partial x_s) \in R^q, \\
(\partial^2 V/\partial x^q \partial x^q) &= (X^q)^T(\partial^2 V/\partial x_s \partial x_s)(X^q) \in M(R^q \to R^q), \\
(\partial V(\tilde{x}_s,u_1)/\partial u^j) &= \int_T (F_j(t))^T(\partial V(\tilde{x}_s,u_1)/\partial u(t))dt \in R^j, \qquad (3.6) \\
T(1,j\to 1,j) &= \int_T d\tau_1 \int_T d\tau_2 (F_j(\tau_1))^T(\partial^2 V/\partial u(\tau_1)\partial u(\tau_2))(F_j(\tau_2)) \\
&\in M(R^j \to R^j), \qquad (3.7) \\
P(1,q\to 1,j) &= \int_T (F_j(\tau))^T(\partial^2 V/\partial u(\tau)\partial x_s)(X^q)d\tau \\
&\in M(R^q \to R^j), \qquad (3.8)
\end{aligned}$$

and all expansion terms such as $(\partial V(\tilde{x}_s,u_1)/\partial x^q)$, etc., are bounded.

∇ <u>*Comment 3.3.2*</u> We see from (3.7) that $T(1,j\to 1,j)$ is symmetric, Δ $\forall j \in I(k)$.

∇ <u>*Comment 3.3.3*</u> For all $j \in I(k)$, we see from Definition 3.3.7 that $(\partial V(\tilde{x}_s,u_1)/\partial u^j)$ of (3.6) is equal to the vector $g^j(\tilde{x}_s,u_1)$ of the components of basis-functions $\delta_1, .., \delta_j$ present in $\{\partial V(\tilde{x}_s,u_1)/\partial u\}$, i.e.

Δ $$\{\partial V(\tilde{x}_s,u_1)/\partial u^j\} = g^j(\tilde{x}_s,u_1), \ \forall j \in I(k). \tag{3.9}$$

∇ *Comment 3.3.4* From (3.9) and (3.5):

$$g^j(\tilde{x}_s+X^q\delta x^q,u_1+F_j\delta u^j) = \{\partial V(\tilde{x}_s+X^q\delta x^q,u_1+F_j\delta u^j)/\partial u^j\}$$
$$= g^j(\tilde{x}_s,u_1) + P(1,q\to 1,j)\delta x^q + T(1,j\to 1,j)\delta u^j, \ \forall j \in I(k). \tag{3.10}$$

Since $g^j(\tilde{x}_s+X^q\delta x^q,u_1+F_j\delta u^j)$ is the vector of the components of basis-functions $\delta_1, .., \delta_j$ present in the gradient function $\{\partial V(\tilde{x}_s+X^q\delta x^q,u_1+F_j\delta u^j)/\partial u\}$, we see from (3.10) that $P(1,q\to 1,j)$ maps initial condition changes characterised by δx^q to changes in the components of basis-functions $\delta_1, .., \delta_j$ present in the gradient function. For this reason $P(1,q\to 1,j)$ will be referred to as an $X\to G$ map **matrix**. Similarly, $T(1,j\to 1,j)$ maps changes δu^j in the components of basis-functions $\delta_1, .., \delta_j$ present in the control function u to changes in the components of basis-functions $\delta_1, .., \delta_j$ present in the gradient function, and is

Δ referred to as a $U\to G$ map matrix.

∇ *Definition 3.3.8* In general we denote the $U\to G$ map matrix which maps changes $\delta u_a^b = (\delta u_a \ \dots \ \delta u_b)^T$ in the components of basis-functions $\delta_a, .., \delta_b$ present in the control function to changes $\delta g_c^d = (\delta g_c \ .. \ \delta g_d)^T$ in the components of basis-functions $\delta_c, .., \delta_d$ present in the gradient function by $T(a,b\to c,d) \in M(R^{b-a+1} \to R^{d-c+1})$. Then $\delta g_c^d = T(a,b\to c,d)\delta u_a^b$. Similarly, we denote by $T(a\to c,d) \in M(R^1 \to R^{d-c+1})$ the $U\to G$ map matrix which maps changes in the component of δ_a present in the control function to changes in the components of $\delta_c, .., \delta_d$ present in the gradient function, and we denote by $T(a,b\to c) \in M(R^{b-a+1} \to R^1)$ the $U\to G$ map matrix which maps changes in the components of basis-functions $\delta_a, .., \delta_b$ present in the control function to changes in the component of basis-function δ_c present in the gradient function. We interpret $T(a,b\to c,c)$ as the $U\to G$ map matrix which maps changes δu_a^b in the components of basis-functions $\delta_a, .., \delta_b$ present in the control function to changes in the component of basis-function δ_c present in the gradient function, i.e. we interpret $T(a,b\to c,c)$ as $T(a,b\to c)$. Similarly, we interpret $T(a,a\to c,d)$ as $T(a\to c,d)$. We sometimes denote by $T(a\to c)$ the (scalar) $U\to G$ map matrix element which maps changes δu_a in the component of basis-function δ_a present in the control function to changes δg_c in the component of δ_c present in the gradient

Δ function (so that $\delta g_c = T(a\to c)\delta u_a$), and we interpret $T(a,a\to c,c)$ as $T(a\to c)$.

∇ <u>*Remark 3.3.1*</u> For all $j \in I(k)$, the $(\tilde{x}_s+X^q\delta x^q)$-optimal control function belonging to $U(1,j)$" for the optimisation problem of 3.2 is $u_1 + F_j\delta u^j(\delta x^q)^*$ if $\delta u^j(\delta x^q)^*$ minimises $V\big(\tilde{x}_s+X^q\delta x^q, u_1+F_j\delta u^j\big)$ of (3.5) with respect to δu^j. Therefore if $T(1,j\to 1,j)$ of (3.7) is p.d., the $(\tilde{x}_s+X^q\delta x^q)$-optimal control function belonging to $U(1,j)$" exists (since all the expansion terms of (3.5) are bounded) and is

$$u\big(j;\tilde{x}_s+X^q\delta x^q\big)^* \;=\; u_1 \;+\; F_j\delta u^j(\delta x^q)^*$$

where $\delta u^j(\delta x^q)^* \;=\; -\big(T(1,j\to 1,j)\big)^{-1}\{\, g^j(\tilde{x}_s,u_1) \;+\; P(1,q\to 1,j)\delta x^q \,\}$.

The associated $(\tilde{x}_s+X^q\delta x^q)$-minimal performance index on $U(1,j)$" is then

$$V\big(j;\tilde{x}_s+X^q\delta x^q\big)^* \;=\; V\big(\tilde{x}_s+X^q\delta x^q, u_1+F_j\delta u^j(\delta x^q)^*\big)$$

$$= V\big(\tilde{x}_s,u_1\big) + <\big(\partial V(\tilde{x}_s,u_1)/\partial x^q\big),\big(\delta x^q\big)> + \tfrac{1}{2}<\big(\delta x^q\big),\big(\partial^2 V/\partial x^q\partial x^q\big)\big(\delta x^q\big)>$$

Δ $$- \tfrac{1}{2}<\big(g^j(\tilde{x}_s,u_1) + P(1,q\to 1,j)\delta x^q\big),T(1,j\to 1,j)^{-1}\big(g^j(\tilde{x}_s,u_1) + P(1,q\to 1,j)\delta x^q\big)>.$$

∇ <u>*Comment 3.3.5*</u> The control function $u\big(j;\tilde{x}_s+X^q\delta x^q\big)^*$ of Remark 3.3.1 ensures that the components of basis-functions $\delta_1, .., \delta_j$ present in the gradient function $\big(\partial V(\tilde{x}_s+X^q\delta x^q, u_1+F_j\delta u^j(\delta x^q)^*)/\partial u\big)$ are all zero. This is clearly a necessary condition for $(\tilde{x}_s+X^q\delta x^q)$-optimality on $U(1,j)$" since it ensures that a small arbitrary control function change belonging to $U(1,j)$, relative to $u\left(j;\tilde{x}_s+X^q\delta x^q\right)^*$,

Δ can cause no first-order performance index change.

3.4 $\varepsilon(x_s)$-Approximations to the x_s-Optimal Control Function

We consider in 3.4.1 the determination of lower-bounds for the x_s-minimal performance index on the control space U for the optimisation problem of 3.2 when only a non-x_s-optimal control function belonging to U is available. The lower-bounds can be computed with little computational effort. An $\varepsilon(x_s)$-approximation to the x_s-optimal control function belonging to U is defined in 3.4.2 and is there discussed. The definitions and notations of 3.3 are used throughout.

3.4.1 Lower-bounds for the x_s-Minimal Performance Index on U

∇ *Assumption 3.4.1* Consider the optimisation problem of 3.2 and suppose that for a bounded initial condition x_s and some control function $u_1 \in U$, which together determine an output function y through convolution-description (3.2), the following expansions hold for any bounded δy and $\delta u \in U$:

$$\begin{aligned} F\big((y+\delta y)(t),(u_1+\delta u)(t),t\big) &= F\big(y(t),u_1(t),t\big) \\ &+ <\big(F_y(t)\big),\big(\delta y(t)\big)> + \tfrac{1}{2}<\big(\delta y(t)\big),F_{yy}(t)\big(\delta y(t)\big)> \\ &+ <\big(F_u(t)\big),\big(\delta u(t)\big)> + \tfrac{1}{2}<\big(\delta u(t)\big),F_{uu}(t)\big(\delta u(t)\big)>, \ \forall t \in T, \\ G\big((y+\delta y)(t_f)\big) &= G\big(y(t_f)\big) + <\big(G_y\big),\big(\delta y(t_f)\big)> \\ &+ \tfrac{1}{2}<\big(\delta y(t_f)\big),G_{yy}\big(\delta y(t_f)\big)>, \end{aligned}$$

where F_y and F_u are continuous on T and are bounded,
F_{yy} is continuous on T and is bounded and n.n.d. on T,
F_{uu} is continuous, bounded and p.d. on T,
Δ G_y exists and G_{yy} is bounded and n.n.d.

∇ *Assumption 3.4.2* Suppose that $\left(\partial V(m;x_s)^*/\partial u\right)$ is the gradient function $\left(\partial V(x_s,u)/\partial u\right)$ for the optimisation problem of 3.2 with the initial condition x_s of Assumption 3.4.1 and a control function u which is the x_s-optimal control function belonging to $U(1,m)''$, $0 < m \leq j \leq k$, and that $V\left(m;x_s\right)^*$ is the x_s-minimal performance index on $U(1,m)''$. Suppose also that $\left(\partial V(m;x_s)^*/\partial u\right)$ is exactly characterised by the components $g^j(m;x_s)^*$ of basis-functions $\delta_1, .., \delta_j$, so that

Δ
$$\left(\partial V(m;x_s)^*/\partial u\right) = F_j g^j(m;x_s)^*. \tag{3.11}$$

The main results of this section are contained in

∇ *Remark 3.4.1* Consider the optimisation problem of 3.2 and assume that Assumptions 3.4.1 and 3.4.2 hold. Then, lower-bounds for the x_s-minimal performance index on the control space U - evaluated for the x_s-optimal control function belonging to $U(1,m)''$ - are

(a)
$$\begin{aligned} V\left(j;x_s\right)^*_m &= V\left(m;x_s\right)^* - \tfrac{1}{2}<\left(g^j(m;x_s)^*\right),\left(g^j(m;x_s)^*\right)>/\lambda^* \\ &= V\left(m;x_s\right)^* - \tfrac{1}{2}\int_T<\left(\partial V(m;x_s)^*/\partial u(t)\right),\left(\partial V(m;x_s)^*/\partial u(t)\right)>dt/\lambda^* \end{aligned}$$

where $\lambda^* = \min_{t\in T} \lambda_{\min}\left(F_{uu}(t)\right)$, and

(b)
$$\begin{aligned} \hat{V}\left(j;x_s\right)^*_m &= V\left(m;x_s\right)^* \\ &- \tfrac{1}{2}\int_T<\left(\partial V(m;x_s)^*/\partial u(t)\right),\left(F_{uu}(t)\right)^{-1}\left(\partial V(m;x_s)^*/\partial u(t)\right)>dt, \end{aligned}$$

where the lower-bound $\hat{V}\left(j;x_s\right)^*_m$ is, potentially, a better (more positive)

Δ lower-bound than is $V\left(j;x_s\right)^*_m$.

∇ *Comment 3.4.1* By considering $U(1,0)''$ to be the sub-set of U which contains only the initial control function u_1 of Assumption 3.4.1, it can be seen that the lower-bound results of Remark 3.4.1 are valid when m is set equal to zero, $V\{0;x_s\}^*$ is replaced by $V\{x_s,u_1\}$, $\{\partial V(0;x_s)^*/\partial u\}$ is replaced by $\{\partial V(x_s,u_1)/\partial u\}$ and when $g^j(0;x_s)^*$ exactly characterises $\{\partial V(x_s,u_1)/\partial u\}$, i.e. when $\{\partial V(x_s,u_1)/\partial u\} = F_j g^j(0;x_s)^*$. The lower-bounds $V\{j;x_s\}_0^*$ and $\tilde{V}\{j;x_s\}_0^*$ are then lower-bounds for the x_s-minimal performance index on the control space U - evaluated for the

Δ control function u_1.

Although statement (a) of Remark 3.4.1 follows directly from statement (b), it is illuminating to derive it an a somewhat different way. The proof of Remark 3.4.1 is facilitated by three lemmas, which are next presented and proved.

∇ *Lemma 3.4.1* When Assumption 3.4.1 holds and $j \in I(k)$: $T(1,j\to 1,j)$ of 3.3 is p.d. and can be written as

$$T(1,j\to 1,j) = E(1,j\to 1,j) + D(1,j\to 1,j)$$

where $E(1,j\to 1,j) = \int_T \{F_j(t)\}^T F_{uu}(t)\{F_j(t)\}dt$ and is p.d.,

$D(1,j\to 1,j)$ is n.n.d.,

$T(1,j\to 1,j)$, $E(1,j\to 1,j)$, $D(1,j\to 1,j) \in M(R^j \to R^j)$ and are

Δ bounded and symmetric.

∇ *Proof of Lemma 3.4.1* It can be seen that when Assumption 3.4.1 holds, $\{\partial^2 V/\partial u(\tau_1)\partial u(\tau_2)\}$ of (3.4) satisfies

$$\int_T d\tau_1 \int_T d\tau_2 \langle\{\delta u(\tau_1)\},\{\partial^2 V/\partial u(\tau_1)\partial u(\tau_2)\}\{\delta u(\tau_2)\}\rangle = \langle\{\delta y(t_f)\},G_{yy}\{\delta y(t_f)\}\rangle$$
$$+ \int_T \langle\{\delta u(t)\},F_{uu}(t)\{\delta u(t)\}\rangle dt + \int_T \langle\{\delta y(t)\},F_{yy}(t)\{\delta y(t)\}\rangle dt$$

<u>when</u> $\delta y(t) = \int_{t_s}^{t} W(t,\tau)\delta u(\tau)d\tau$, $\forall t \in T$ (from (3.2)).

From (3.7):

$$\langle\{\delta u^j\},T(1,j\to 1,j)\{\delta u^j\}\rangle = \int_T d\tau_1 \int_T d\tau_2 \langle\{\delta u(\tau_1)\},\{\partial^2 V/\partial u(\tau_1)\partial u(\tau_2)\}\{\delta u(\tau_2)\}\rangle$$

<u>when</u> $\delta u(t) = F_j(t)\delta u^j$, $\forall t \in T$.

Thus:

$$<(\delta u^j),T(1,j\to 1,j)(\delta u^j)> \;=\; <(\delta u^j),E(1,j\to 1,j)(\delta u^j)> + <(\delta u^j),D(1,j\to 1,j)(\delta u^j)> \qquad (3.12)$$

where $$<(\delta u^j),E(1,j\to 1,j)(\delta u^j)> \;=\; \int_T <(\delta u(t)),F_{uu}(t)(\delta u(t))>dt, \qquad (3.13)$$

$$<(\delta u^j),D(1,j\to 1,j)(\delta u^j)> \;=\; \int_T <(\delta y(t)),F_{yy}(t)(\delta y(t))>dt + <(\delta y(t_f)),G_{yy}(\delta y(t_f))>, \qquad (3.14)$$

<u>when</u> $$\delta u(t) \;=\; F_j(t)\delta u^j,\; \forall t \in T, \qquad (3.15)$$

$$\delta y(t) \;=\; \int_{t_s}^{t} W(t,\tau)F_j(\tau)\delta u^j d\tau,\; \forall t \in T.$$

From (3.12), $T(1,j\to 1,j)$ can be written as

$$T(1,j\to 1,j) \;=\; E(1,j\to 1,j) \;+\; D(1,j\to 1,j),$$

where, from (3.13) and (3.15):

$$E(1,j\to 1,j) \;=\; \int_T (F_j(t))^T F_{uu}(t)(F_j(t))dt. \qquad (3.16)$$

Since δu of (3.15) is not non-zero only on a set of measure zero when $\delta u^j \neq O(j,1)$ (because $\int_T <\delta u(t),\delta u(t)>dt = <\delta u^j,\delta u^j>$, due to the orthonormality of the bounded basis-functions f which constitute F_j) and since F_{uu} is p.d. on T (Assumption 3.4.1), we see from (3.13) that $<(\delta u^j),E(1,j\to 1,j)(\delta u^j)> > 0$ whenever $\delta u^j \neq O(j,1)$. Hence $E(1,j\to 1,j)$ is p.d. when Assumption 3.4.1 holds. $E(1,j\to 1,j)$ is bounded since F_j and F_{uu} are bounded (from 3.3 and Assumption 3.4.1) and since $t_f-t_s< \infty$ (from 3.2).

Since F_{yy} is n.n.d. on T and G_{yy} is n.n.d. (Assumption 3.4.1), we see from (3.14) that $<(\delta u^j),D(1,j\to 1,j)(\delta u^j)> \geqq 0$. Hence $D(1,j\to 1,j)$ is n.n.d. when Assumption 3.4.1 holds. It is clear from (3.14) that $D(1,j\to 1,j)$ is then bounded since W, F_j, G_{yy} and F_{yy} are all bounded and $t_f-t_s < \infty$ (from 3.2, 3.3 and Assumption 3.4.1).

$D(1,j\to 1,j)$ is symmetric since $D(1,j\to 1,j) = T(1,j\to 1,j) - E(1,j\to 1,j)$

and $T(1,j\to 1,j)$ is symmetric (Comment 3.3.2) and, from (3.16), $E(1,j\to 1,j)$ is symmetric.

$T(1,j\to 1,j)$ is p.d. since it is the sum of a p.d. matrix, $E(1,j\to 1,j)$, and a n.n.d. matrix, $D(1,j\to 1,j)$.

Δ This concludes the proof of Lemma 3.4.1.

∇ *Lemma 3.4.2* When Assumption 3.4.1 holds for the optimisation problem of 3.2:

$$\lambda_{\min}\big(T(1,j\to 1,j)\big) \geqq \lambda^* > 0, \; \forall j \in I(k),$$

Δ where $\lambda^* = \min\limits_{t \in T} \lambda_{\min}\big(F_{uu}(t)\big)$ and is independent of j and k.

∇ *Proof of Lemma 3.4.2* From (3.13), (3.15) and the fact that $\int_T <\big(\delta u(t)\big),\big(\delta u(t)\big)>dt = <\big(\delta u^j\big),\big(\delta u^j\big)>$ when $\delta u = F_j \delta u^j$ (due to the orthonormality of the basis-functions f which constitute F_j), we see that <u>when</u> $\delta u^j \neq O(j,1)$ and $\delta u(t) = F_j(t)\delta u^j$, $\forall t \in T$:

$$<\big(\delta u^j\big),E(1,j\to 1,j)\big(\delta u^j\big)> \;/\; <\big(\delta u^j\big),\big(\delta u^j\big)>$$
$$= \int_T <\big(\delta u(t)\big),F_{uu}(t)\big(\delta u(t)\big)>dt \;/\; \int_T <\big(\delta u(t)\big),\big(\delta u(t)\big)>dt$$
$$\geqq \int_T \lambda_{\min}\big(F_{uu}(t)\big)<\big(\delta u(t)\big),\big(\delta u(t)\big)>dt \;/\; \int_T <\big(\delta u(t)\big),\big(\delta u(t)\big)>dt$$
$$\geqq \min_{t \in T} \lambda_{\min}\big(F_{uu}(t)\big) \triangleq \lambda^*.$$

Since F_{uu} is p.d. on T (Assumption 3.4.1), $\lambda^* > 0$.

From Lemma 3.4.1 and the above we see that when $\delta u^j \neq O(j,1)$:

$$<\big(\delta u^j\big),T(1,j\to 1,j)\big(\delta u^j\big)> \;/\; <\big(\delta u^j\big),\big(\delta u^j\big)>$$
$$\geqq <\big(\delta u^j\big),E(1,j\to 1,j)\big(\delta u^j\big)> \;/\; <\big(\delta u^j\big),\big(\delta u^j\big)> \geqq \lambda^* > 0.$$

Hence $\lambda_{\min}\big(T(1,j\to 1,j)\big) \geqq \lambda^* > 0$, a result which does not depend on j or k.

Δ This concludes the proof of Lemma 3.4.2.

∇ *Lemma 3.4.3* When Assumption 3.4.1 holds for the optimisation problem of 3.2:

$$<(g^j),E(1,j\to 1,j)^*(g^j)> \;\geqq\; <(g^j),(E(1,j\to 1,j))^{-1}(g^j)>,\ \forall g^j \in R^j,\ \forall j \in I(k),$$

where

$$E(1,j\to 1,j)^* = \int_T (F_j(t))^T (F_{uu}(t))^{-1}(F_j(t))dt \;\in\; M(R^j \to R^j),$$

Δ $$E(1,j\to 1,j) = \int_T (F_j(t))^T\,(F_{uu}(t))\,(F_j(t))dt \;\in\; M(R^j \to R^j).$$

∇ *Proof of Lemma 3.4.3* Suppose Assumption 3.4.1 holds and, for some $j \in I(k)$, $g^j \in R^j$, $w \in U$, consider the scalar-valued function

$$\nu(g^j,w) = \int_T\{<(F_j(t)g^j),(w(t))> + \tfrac{1}{2}<(w(t)),F_{uu}(t)(w(t))>\}dt.$$

Functions w belonging to $U(1,j)$ can be written as $w = F_j w^j$ for some $w^j \in R^j$, and for such functions

$$\begin{aligned}\nu(g^j,w) = \nu(g^j,F_j w^j) &= \int_T <(F_j(t)g^j),(F_j(t)w^j)>dt \\ &\quad + \tfrac{1}{2}\int_T <(F_j(t)w^j),F_{uu}(t)(F_j(t)w^j)>dt \\ &= <(g^j),(w^j)> + \tfrac{1}{2}<(w^j),E(1,j\to 1,j)(w^j)>.\end{aligned}$$

The minimal value of $\nu(g^j,w)$ with respect to $w \in U(1,j)$ is thus

$$\nu(j;g^j)^* = \min_{w^j} \nu(g^j,F_j w^j) = -\tfrac{1}{2}<(g^j),(E(1,j\to 1,j))^{-1}(g^j)>.$$

The function $w \in U$ which minimises $\nu(g^j,w)$ with respect to w is

$$w(t)^* = -(F_{uu}(t))^{-1}F_j(t)g^j,\ \forall t \in T,$$

so that the minimal value of $\nu(g^j,w)$ with respect to $w \in U$ is

$$\begin{aligned}\nu(g^j)^* = \nu(g^j,w^*) &= -\tfrac{1}{2}\int_T <(F_j(t)g^j),(F_{uu}(t))^{-1}(F_j(t)g^j)>dt \\ &= -\tfrac{1}{2}<(g^j),E(1,j\to 1,j)^*(g^j)>.\end{aligned}$$

Now $\nu(g^j)^* \leq \nu(j;g^j)^*$, $\forall g^j \in R^j$, $\forall j \in I(k)$, since $U(1,j)$ is a linear manifold of U. Using this with the above expressions for $\nu(g^j)^*$ and $\nu(j;g^j)^*$ yields the result of Lemma 3.4.3. Δ

∇ *Proof of Remark 3.4.1* By applying the arguments of 3.3 to the optimisation problem of 3.2 when Assumptions 3.4.1 and 3.4.2 hold, it can be seen that the x_s-minimal performance index on $\mathcal{U}(1,j)''$, $0 < m \leqq j \leqq k$, is related to that on $\mathcal{U}(1,m)''$ by

$$V(j;x_s)^* = V(m;x_s)^* - \tfrac{1}{2}\langle (g^j(m;x_s)^*), (\mathcal{T}(1,j{\to}1,j))^{-1}(g^j(m;x_s)^*)\rangle \quad (3.17)$$

where $g^j(m;x_s)^*$ exactly characterises $(\partial V(m;x_s)^*/\partial u)$.

From Lemma 3.4.2: $\lambda_{\min}(\mathcal{T}(1,j{\to}1,j)) \geqq \lambda^* > 0$.

Hence, from (3.17):

$$\begin{aligned} V(j;x_s)^* &\geqq V(m;x_s)^* - \tfrac{1}{2}\langle (g^j(m;x_s)^*), (g^j(m;x_s)^*)\rangle / \lambda^* \\ &\triangleq V(j;x_s)^*_m. \end{aligned} \quad (3.18)$$

Clearly $V(j;x_s)^*_m$ of (3.18) is a lower-bound for the x_s-minimal performance index on $\mathcal{U}(1,j)''$, $V(j;x_s)^*$. A stronger statement can in fact be made, as we next show. Consider optimisation on $\mathcal{U}(1,k)''$, where $k > j$. The x_s-minimal performance index on $\mathcal{U}(1,k)''$ is then given by (3.17) with j replaced by k when $g^k(m;x_s)^*$ is the vector of the components of basis-functions $\delta_1, .., \delta_k$ present in $(\partial V(m;x_s)^*/\partial u)$. Because we have assumed that $(\partial V(m;x_s)^*/\partial u)$ is exactly characterised by the components $g^j(m;x_s)^*$ of basis-functions $\delta_1, .., \delta_j$, $(\partial V(m;x_s)^*/\partial u) = F_j g^j(m;x_s)^*$ and contains no components of basis-functions $\delta_{j+1}, .., \delta_k$. Hence $g^k(m;x_s)^*$ is given by

$$g^k(m;x_s)^* = (g^j(m;x_s)^{*T} \;\; \mathcal{O}(k-j,1)^T)^T.$$

Since the lower-bound of (3.18) for j replaced by k depends only on the non-zero components of basis-functions $\delta_1, .., \delta_k$ present in the gradient function $(\partial V(m;x_s)^*/\partial u)$, the lower-bound $V(k;x_s)^*_m$ for the x_s-minimal performance index on $\mathcal{U}(1,k)''$ is equal to the lower-bound $V(j;x_s)^*_m$ for the x_s-minimal performance index on $\mathcal{U}(1,j)''$. Clearly k can be increased until the orthonormal basis-functions $\delta_1, .., \delta_k$ span the control space with no change in this result. Thus $V(j;x_s)^*_m$ of (3.18) is a lower-bound for the x_s-minimal performance index on the control space U - evaluated for the x_s-optimal control function belonging to $U(1,m)$".

Because $(\partial V(m;x_s)^*/\partial u) = F_j g^j(m;x_s)^*$ (Assumption 3.4.2) and because $\int_T (F_j(t))^T(F_j(t))dt = I(j,j)$ (due to the orthonormality of the basis-functions δ which constitute F_j):

$$\begin{aligned}<(g^j(m;x_s)^*),(g^j(m;x_s)^*)> &= \int_T <(F_j(t)g^j(m;x_s)^*),(F_j(t)g^j(m;x_s)^*)>dt \\ &= \int_T <(\partial V(m;x_s)^*/\partial u(t)),(\partial V(m;x_s)^*/\partial u(t))>dt.\end{aligned}$$

The lower-bound $V(j;x_s)^*_m$ of (3.18) can therefore be written as

$$V(j;x_s)^*_m = V(m;x_s)^* - \tfrac{1}{2}\int_T <(\partial V(m;x_s)^*/\partial u(t)),(\partial V(m;x_s)^*/\partial u(t))>dt / \lambda^*.$$

This concludes the proof of statement (a) of Remark 3.4.1. We next prove statement (b).

We see from Lemma 3.4.1 that the x_s-minimal performance index on $U(1,j)$", of (3.17), can be written as

$$\begin{aligned}V(j;x_s)^* = {} & V(m;x_s)^* \\ & - \tfrac{1}{2}<(g^j(m;x_s)^*),(E(1,j\to 1,j) + D(1,j\to 1,j))^{-1}(g^j(m;x_s)^*)>.\end{aligned}$$

On using Lemma 2.5.3 we see that

$$\begin{aligned}V(j;x_s)^* = V(m;x_s)^* & - \tfrac{1}{2}<(g^j(m;x_s)^*),(E(1,j\to 1,j))^{-1}(g^j(m;x_s)^*)> \\ & + \tfrac{1}{2}<(g^j(m;x_s)^*),K(1,j\to 1,j)(g^j(m;x_s)^*)>,\end{aligned}$$

where $K(1,j\to 1,j) \in M(R^j \to R^j)$ is n.n.d.

Hence:

$$\begin{aligned}V(j;x_s)^* & \geq V(m;x_s)^* - \tfrac{1}{2}<(g^j(m;x_s)^*),(E(1,j\to 1,j))^{-1}(g^j(m;x_s)^*)> \\ & \triangleq \hat{V}(j;x_s)^*_m. \qquad (3.19)\end{aligned}$$

Clearly $\hat{V}(j;x_s)^*_m$ of (3.19) is a lower-bound for the x_s-minimal performance index on $\mathcal{U}(1,j)$", but it is not necessarily a lower-bound for the x_s-minimal performance index on the control space $\mathcal{U}$ since the approach used to explain why the lower-bound $V(j;x_s)^*_m$ of (3.18) is such a lower-bound cannot be used to show that $\hat{V}(j;x_s)^*_m$ is a lower-bound for the x_s-minimal performance index on $\mathcal{U}$ due to the potentially complicated way in which $(E(1,j\to1,j))^{-1}$ depends on j.

However, we see from Lemma 3.4.3 and (3.19) that

$$V(j;x_s)^* \geqq V(m;x_s)^* - \tfrac{1}{2}<(g^j(m;x_s)^*),E(1,j\to1,j)^*(g^j(m;x_s)^*)> \triangleq \tilde{V}(j;x_s)^*_m. \tag{3.20}$$

On using the definition of $E(1,j\to1,j)^*$ of Lemma 3.4.3 and (3.11), it can be seen that

$$\begin{aligned}\tilde{V}(j;x_s)^*_m &= V(m;x_s)^* - \tfrac{1}{2}\int_T<(F_j(t)g^j(m;x_s)^*),(F_{uu}(t))^{-1}(F_j(t)g^j(m;x_s)^*)>dt\\ &= V(m;x_s)^* - \tfrac{1}{2}\int_T<(\partial V(m;x_s)^*/\partial u(t)),(F_{uu}(t))^{-1}(\partial V(m;x_s)^*/\partial u(t))>dt.\end{aligned}$$

The approach used to show that $V(j;x_s)^*_m$ of (3.18) is a lower-bound for the x_s-minimal performance index on the control space $\mathcal{U}$ can now be used to show that $\tilde{V}(j;x_s)^*_m$ is also such a lower-bound.

This concludes the proof of statement (b) of Remark 3.4.1.

It may be seen that $\tilde{V}(j;x_s)^*_m$ of Remark 3.4.1 is more positive than $V(j;x_s)^*_m$ if, say, $\lambda_{\min}(F_{uu}(t))$ is not constant on T and $(\partial V(m;x_s)/\partial u)$ is not zero on T. Thus $\tilde{V}(j;x_s)^*_m$ is a better (more positive) lower-bound, potentially, than is $V(j;x_s)^*_m$.

Δ This concludes the proof of Remark 3.4.1.

3.4.2 $\varepsilon(x_s)$-Approximations

For the optimisation problem of 3.2 when Assumption 3.4.1 holds, the existence of an x_s-optimal control function (unique in the Hilbert space sense) belonging to $H^m(T)$ (the Hilbert space of m-vector valued functions with domain T) for the initial condition x_s of Assumption 3.4.1 can be seen from the studies of Hsieh {26}. It can also be seen that, under the continuity assumptions of 3.2, there exists a control function u belonging to U such that $V(x_s,u)$ is equal to the x_s-minimal performance index on $H^m(T)$ and such that the gradient function $(\partial V(x_s,u)/\partial u)$ is zero on T. Such a control function is referred to as the x_s-optimal control function belonging to U and the associated performance index $V(x_s,u)$ is referred to as the x_s-minimal performance index on U.

∇ *Definition 3.4.1* For the initial condition x_s of Assumption 3.4.1, the x_s-optimal control function belonging to $U(1,m)$" will be said to be an $\varepsilon(x_s)$-approximation ($\varepsilon > 0$) to the x_s-optimal control function belonging to U if

$$| V(m;x_s)* - V(x_s)* | \leqq \varepsilon,$$

where $V(m;x_s)*$ is the x_s-minimal performance index on $U(1,m)$" and $V(x_s)*$

Δ is the x_s-minimal performance index on U.

When Assumptions 3.4.1 and 3.4.2 hold, lower-bounds for the x_s-minimal performance index on U can easily be calculated using the results of Remark 3.4.1. Then:

∇ *Remark 3.4.2* For the optimisation problem of 3.2 when Assumptions 3.4.1 and 3.4.2 hold, the x_s-optimal control function belonging to $U(1,m)$" is an $\varepsilon(x_s)$-approximation to the x_s-optimal control function belonging to $\mathcal{U}$ if either

$$|\ V(m;x_s)^* \ - \ V(j;x_s)^*_m\ | \ \leq \ \varepsilon$$

or $$|\ V(m;x_s)^* \ - \ \tilde{V}(j;x_s)^*_m\ | \ \leq \ \varepsilon,$$

where $V(j;x_s)^*_m$ and $\tilde{V}(j;x_s)^*_m$ are those of Remark 3.4.1.

Also, the minimal ε for which the x_s-optimal control function belonging to $\mathcal{U}(1,m)$" is an $\varepsilon(x_s)$-approximation to the x_s-optimal control function belonging to $\mathcal{U}$ has as upper-bounds both

(a) $$\varepsilon(m;x_s)^* = \tfrac{1}{2}<(g^j(m;x_s)^*),(g^j(m;x_s)^*)> / \lambda^*$$
$$= \tfrac{1}{2}\int_T<(\partial V(m;x_s)^*/\partial u(t)),(\partial V(m;x_s)^*/\partial u(t))>dt / \lambda^*$$

where $\lambda^* = \min_{t\in T} \lambda_{\min}(F_{uu}(t))$, and

Δ (b) $$\tilde{\varepsilon}(m;x_s)^* = \tfrac{1}{2}\int_T<(\partial V(m;x_s)^*/\partial u(t)),(F_{uu}(t))^{-1}(\partial V(m;x_s)^*/\partial u(t))>dt.$$

Since the lower-bounds of Remark 3.4.1 or the upper-bounds of Remark 3.4.2 can be calculated with relatively little computational effort (compared with that which would be needed to optimise on $\mathcal{U}(1,m+1)$" or $\mathcal{U}$, say), we can easily determine whether the x_s-optimal control function belonging to $\mathcal{U}(1,m)$" is an $\varepsilon(x_s)$-approximation to the x_s-optimal control function belonging to the control space $\mathcal{U}$ with little computational expense <u>without</u> knowing the x_s-optimal control function belonging to $\mathcal{U}$ or the associated x_s-minimal performance index on $\mathcal{U}$. We can then ascertain whether the x_s-optimal control function belonging to $\mathcal{U}(1,m)$" can be considered to be an adequate approximation, performance-index wise, to that belonging to the control space $\mathcal{U}$ by checking whether it is an $\varepsilon(x_s)$-approximation for a suitably small $\varepsilon > 0$. If it is not an adequate approximation in the above sense, optimisation on a linear manifold which includes $\mathcal{U}(1,m)$" but is of larger dimension should be attempted.

∇ <u>*Comment 3.4.2*</u> $\varepsilon(m;x_s)^*$ and $\tilde{\varepsilon}(m;x_s)^*$ of Remark 3.4.2 are both upper-bounds for the further performance index decrease, relative to the x_s-minimal performance index on $U(1,m)''$, which can be achieved by optimising on U itself, but $\tilde{\varepsilon}(m;x_s)^*$ is a potentially better (i.e.
Δ less positive) upper-bound than is $\varepsilon(m;x_s)^*$.

The practical usefulness of the results we have obtained depends, of course, on $\tilde{\varepsilon}(m;x_s)^*$ and $\varepsilon(m;x_s)^*$ approaching zero as $V(m;x_s)^*$ approaches $V(x_s)^*$ with increasing m. Since $\tilde{\varepsilon}(m;x_s)^* \leqq \varepsilon(m;x_s)^*$, we need only consider the behaviour of $\varepsilon(m;x_s)^*$ - the subject of

∇ <u>*Remark 3.4.3*</u> Consider the optimisation problem of 3.2 when Assumption 3.4.1 holds. Then

$$(b^2/B\lambda^*)\{V(m;x_s)^* - V(x_s)^*\} \leqq \varepsilon(m;x_s)^* \leqq (B^2/b\lambda^*)\{V(m;x_s)^* - V(x_s)^*\}$$

Δ for some scalars b and B such that $0 < b \leqq B < \infty$.

∇ <u>*Comment 3.4.3*</u> We see from Remark 3.4.3 that, when Assumption 3.4.1 holds for the optimisation problem of 3.2, $\varepsilon(m;x_s)^*$ approaches
Δ zero as $V(m;x_s)^*$ approaches $V(x_s)^*$.

∇ <u>*Proof of Remark 3.4.3*</u> Denote the x_s-optimal control function belonging to $U(1,m)''$ by u_m^*, and that belonging to U by u^*. Let $\delta u_m^* = u_m^* - u^*$. Then, it may be seen from (3.4) with u_1 replaced by u_m^* and δu replaced by δu_m^* that:

$$V(m;x_s)^* = V(x_s,u_m^*) = V(x_s)^* + \tfrac{1}{2}\int_T d\tau_1 \int_T d\tau_2 \langle (\delta u_m^*(\tau_1)), (\partial^2 V/\partial u(\tau_1)\partial u(\tau_2)) (\delta u_m^*(\tau_2))\rangle.$$

Also: $$(\partial V(m;x_s)^*/\partial u(t)) = (\partial V(x_s,u_m^*)/\partial u(t)) = \int_T (\partial^2 V/\partial u(t)\partial u(\tau)) (\delta u_m^*(\tau)) d\tau, \ \forall t \in T.$$

Suppose that Assumption 3.4.1 holds for the optimisation problem of 3.2. Then, as stated in the proof of Lemma 3.4.1:

$$\int_T d\tau_1 \int_T d\tau_2 <\left(\delta u(\tau_1)\right), \left(\partial^2 V/\partial u(\tau_1)\partial u(\tau_2)\right)\left(\delta u(\tau_1)\right)> \;=\; <\left(\delta y(t_f)\right), G_{yy}\left(\delta y(t_f)\right)>$$
$$+ \int_T <\left(\delta u(t)\right), F_{uu}(t)\left(\delta u(t)\right)> dt \;+\; \int_T <\left(\delta y(t)\right), F_{yy}(t)\left(\delta y(t)\right)> dt$$

when $\delta y(t) = \int_{t_s}^{t} W(t,\tau)\delta u(\tau)d\tau$, $\forall t \in T$.

Since then F_{uu} is bounded and p.d. on T, F_{yy} is bounded and n.n.d. on T, G_{yy} is bounded and n.n.d. and W is bounded, it can be seen that scalars b and B exist, $0 < b \leqq B < \infty$, such that

$$b\int_T \|\delta u(t)\|^2 dt \;\leqq\; \int_T d\tau_1 \int_T d\tau_2 <\left(\delta u(\tau_1)\right), \left(\partial^2 V/\partial u(\tau_1)\partial u(\tau_2)\right)\left(\delta u(\tau_2)\right)>$$
$$\leqq\; B\int_T \|\delta u(t)\|^2 dt$$

for all δu, where $\|\delta u(t)\| = \sqrt{<\left(\delta u(t)\right),\left(\delta u(t)\right)>}$.

Let

$$a(\kappa,t) = \kappa\delta u_m^*(t) + (1/\kappa)\left(\partial V(m;x_s)^*/\partial u(t)\right), \;\forall t \in T,$$
$$b(\kappa,t) = \kappa\delta u_m^*(t) - (1/\kappa)\left(\partial V(m;x_s)^*/\partial u(t)\right), \;\forall t \in T,$$

where κ is a scalar. Then

$$\int_T \|\left(\partial V(m;x_s)^*/\partial u(t)\right)\|^2 dt$$
$$= \int_T d\tau_1 <\left(\partial V(m;x_s)^*/\partial u(\tau_1)\right), \int_T \left(\partial^2 V/\partial u(\tau_1)\partial u(\tau_2)\right)\delta u_m^*(\tau_2)d\tau_2>$$
$$= \tfrac{1}{4}\int_T d\tau_1 \int_T d\tau_2 <\left(a(\kappa,\tau_1)\right), \left(\partial^2 V/\partial u(\tau_1)\partial u(\tau_2)\right)\left(a(\kappa,\tau_2)\right)>$$
$$- \tfrac{1}{4}\int_T d\tau_1 \int_T d\tau_2 <\left(b(\kappa,\tau_1)\right), \left(\partial^2 V/\partial u(\tau_1)\partial u(\tau_2)\right)\left(b(\kappa,\tau_2)\right)>$$
$$\leqq \tfrac{1}{4}B\int_T \|a(\kappa,t)\|^2 dt + \tfrac{1}{4}B\int_T \|b(\kappa,t)\|^2 dt$$
$$= \ell(\kappa^2),$$

where $\ell(\kappa^2) = (B\kappa^2/2)\int_T \|\delta u_m^*(t)\|^2 dt + (B/2\kappa^2)\int_T \|\left(\partial V(m;x_s)/\partial u(t)\right)\|^2 d$

Since this holds whatever κ^2 is, it holds when κ^2 is chosen to minimise $\ell(\kappa^2)$ with respect to κ^2,

i.e. when $\kappa^2 = \sqrt{\{\int_T \|\left(\partial V(m;x_s)^*/\partial u(t)\right)\|^2 dt / \int_T \|\delta u_m^*(t)\|^2 dt\}}$.

Hence $\int_T \|\left(\partial V(m;x_s)^*/\partial u(t)\right)\|^2 dt \leqq B\sqrt{\{\int_T \|\delta u_m^*(t)\|^2 dt \int_T \|\left(\partial V(m;x_s)^*/\partial u(t)\right)\|^2 dt\}}$

so that $\int_T \|\left(\partial V(m;x_s)^*/\partial u(t)\right)\|^2 dt \;\leqq\; B^2\int_T \|\delta u_m^*(t)\|^2 dt.$

Since

$$b\int_T \|\delta u^*_m(t)\|^2 dt \leqq \int_T dt \int_T d\tau \langle (\delta u^*_m(t)), (\partial^2 V/\partial u(t)\partial u(\tau)) (\delta u^*_m(\tau)) \rangle$$
$$= \int_T \langle (\delta u^*_m(t)), (\partial V(m;x_s)^*/\partial u(t)) \rangle dt,$$

we see that $(\partial V(m;x_s)^*/\partial u)$ must have the form

$$(\partial V(m;x_s)^*/\partial u(t)) = (b+\eta)\delta u^*_m(t) + k(t), \ \forall t \in T,$$

where $\eta \geqq 0$ and $\int_T \langle (k(t)), (\delta u^*_m(t)) \rangle dt = 0$.

Hence $\int_T \|(\partial V(m;x_s)^*/\partial u(t))\|^2 dt$

$$\geqq (b+\eta)^2 \int_T \|\delta u^*_m(t)\|^2 dt + \int_T \|k(t)\|^2 dt$$
$$\geqq b^2 \int_T \|\delta u^*_m(t)\|^2 dt.$$

Therefore

$$b^2 \int_T \|\delta u^*_m(t)\|^2 dt \leqq \int_T \|(\partial V(m;x_s)^*/\partial u(t))\|^2 dt \leqq B^2 \int_T \|\delta u^*_m(t)\|^2 dt.$$

Also, from the result for $V(m;x_s)^*$ of page 97 and the above:

$$\tfrac{1}{2}b\int_T \|\delta u^*_m(t)\|^2 dt \leqq \{V(m;x_s)^* - V(x_s)^*\} \leqq \tfrac{1}{2}B\int_T \|\delta u^*_m(t)\|^2 dt.$$

Hence

$$2(b^2/B)\{V(m;x_s)^* - V(x_s)^*\} \leqq \int_T \langle (\partial V(m;x_s)^*/\partial u(t)), (\partial V(m;x_s)^*/\partial u(t)) \rangle dt$$
$$\leqq 2(B^2/b)\{V(m;x_s)^* - V(x_s)^*\}.$$

On using this result with the definition of $\varepsilon(m;x_s)^*$ of

Δ Remark 3.4.2, the result of Remark 3.4.3 emerges.

3.5 Gradient Decomposition

The optimisation algorithms which are developed in the remainder of this chapter define (using Gram-Schmidt orthonormalisation) the basis-functions $\mathfrak{f}$ which are considered in 3.3 and 3.4 so that each calculated gradient function can be exactly characterised by components of the defined basis-functions. We shall use the following definitions:

∇ *Definition 3.5.1* By decompose the gradient function $\left(\partial V(x_s,u)/\partial u\right)$ as $F_1 g_1(x_s,u)$ we mean define a normalised basis-function $\mathfrak{f}_1$ so that $\left(\partial V(x_s,u)/\partial u\right) \in G(1)$ and can be written as $\left(\partial V(x_s,u)/\partial u(t)\right) = F_1(t) g_1(x_s,u)$ for all $t \in T$. This requires that

$$g_1(x_s,u) = \left|\surd\int_T <\left(\partial V(x_s,u)/\partial u(t)\right), \left(\partial V(x_s,u)/\partial u(t)\right)> dt\right| \in R,$$

$$\mathfrak{f}_1(t) = \left(\partial V(x_s,u)/\partial u(t)\right)/g_1(x_s,u), \; \forall t \in T, \text{ if } g_1(x_s,u) \neq 0.$$

If, however, $g_1(x_s,u) = 0$, we define $\mathfrak{f}_1$ to be zero on T even though $\mathfrak{f}_1$ is

Δ then not normalised.

∇ *Definition 3.5.2* By decompose the gradient function $\left(\partial V(x_s,u)/\partial u\right)$ as $F_{j+1} g^{j+1}(x_s,u)$ we mean

(a) determine the components $g^j(x_s,u) \in R^j$ of the already defined orthonormal basis-functions $\mathfrak{f}_1, \ldots, \mathfrak{f}_j$ present in $\left(\partial V(x_s,u)/\partial u\right)$, and

(b) define a new basis-function $\mathfrak{f}_{j+1}$ orthonormal to $\mathfrak{f}_1, \ldots, \mathfrak{f}_j$ so that $\left(\partial V(x_s,u)/\partial u\right) \in G(1,j+1)$ and can be written as $\left(\partial V(x_s,u)/\partial u(t)\right) = F_{j+1}(t) g^{j+1}(x_s,u)$, $\forall t \in T$. This requires that:

$$g^j(x_s,u) = \int_T \left(F_j(t)\right)^T \left(\partial V(x_s,u)/\partial u(t)\right) dt \in R^j,$$

$$g_{j+1}(x_s,u) = \left|\surd\int_T <\left(z(t)\right), \left(z(t)\right)> dt\right| \in R,$$

$$\mathfrak{f}_{j+1}(t) = z(t)/g_{j+1}(x_s,u), \; \forall t \in T, \text{ if } g_{j+1}(x_s,u) \neq 0,$$

$$g^{j+1}(x_s,u) = \left(g^j(x_s,u)^T \;\; g_{j+1}(x_s,u)\right)^T \in R^{j+1},$$

where $\quad z(t) = \left(\partial V(x_s,u)/\partial u(t)\right) - F_j(t)g^j(x_s,u), \ \forall t \in T.$

If, however, $g_{j+1}(x_s,u) = 0$, we define $\mathfrak{f}_{j+1}$ to be zero on T even though

Δ it is then not normalised.

Since the algorithms we develop later depend heavily on gradient-decomposition (in the above sense), we refer to them as being gradient-decomposition based. We shall be able to see, later, that all the contro functions u considered in the algorithms for the optimisation problem of 3.2 (with bounded x_s) are bounded and continuous. From 3.2, all the associated gradient functions $\left(\partial V(x_s,u)/\partial u\right)$ are also bounded and continuous. It can therefore be seen that all the orthonormal basis-functions $\mathfrak{f}$ which will be defined in the algorithms using Definitions 3.5.1 and 3.5.2 will be bounded and continuous, i.e. will be of the type considered so far in this chapter.

Once basis-functions $\mathfrak{f}_1, .., \mathfrak{f}_j$ have been defined, we see from 3.3 how, for the optimisation problem 3.2, optimisation on $U(1,j)$" can be achieved for any bounded initial condition $\tilde{x}_s + X^q\delta x^q \in X(q)$ when $T(1,j\to 1,j)$ and $P(1,q\to 1,j)$ are known. $T(1,j\to 1,j)$ and $P(1,q\to 1,j)$ could be determined from (3.7) and (3.8), but that would require the determination of $\left(\partial^2 V/\partial u(\tau_1)\partial u(\tau_2)\right)$ and $\left(\partial^2 V/\partial u(\tau_1)\partial x_s\right)$ for all τ_1, $\tau_2 \in T$, which is not desirable from the computational point of view. We therefore determine $T(1,j\to 1,j)$ and $P(1,q\to 1,j)$ in our algorithms from those changes in the components of considered basis-functions present in the gradient function which are caused by changes in the components of considered basis-functions present in the control function and by initial condition changes (recall from Comment 3.3.4 the interpretation of $T(1,j\to 1,j)$ and $P(1,q\to 1,j)$ as $U\to G$ and $X\to G$ map matrices, respectively).

We see that minimisation on $U(1,j)''$ using Remark 3.3.1 involves the inverse of $T(1,j\to 1,j)$. Since the algorithms which we develop minimise on $U(1,m)''$ as m increases from 1 in unit steps, it is desirable to be able to compute $T(1,m+1\to 1,m+1)^{-1}$ in terms of $T(1,m\to 1,m)^{-1}$ and to be able to check whether $T(1,m+1\to 1,m+1)$ is p.d. This is the subject of

∇ *Lemma 3.5.1* Suppose

$$T(1,m+1\to 1,m+1) = \begin{pmatrix} T(1,m\to 1,m) & T(m+1\to 1,m) \\ T(1,m\to m+1) & T(m+1\to m+1)\end{pmatrix} \in M(R^{m+1}\to R^{m+1})$$

where $T(1,m\to 1,m) \in M(R^m \to R^m)$ and is symmetric and p.d.,

$T(1,m\to m+1) = T(m+1\to 1,m)^T \in M(R^m \to R)$,

$T(m+1\to m+1) \in R$.

Let $\Gamma_{m+1} = T(m+1\to m+1) - <\left(T(m+1\to 1,m)\right), T(1,m\to 1,m)^{-1}\left(T(m+1\to 1,m)\right)> \in R$,

$$n^{m+1} = \begin{pmatrix} -T(1,m\to 1,m)^{-1}T(m+1\to 1,m) \\ 1 \end{pmatrix} \in R^{m+1}.$$

Then $T(1,m+1\to 1,m+1)$ is p.d. if and only if $\Gamma_{m+1} > 0$, and if $\Gamma_{m+1} > 0$:

$$T(1,m+1\to 1,m+1)^{-1} = \begin{pmatrix} T(1,m\to 1,m)^{-1} & O(m,1) \\ O(1,m) & 0 \end{pmatrix} + (1/\Gamma_{m+1})\left(n^{m+1}\right)><\left(n^{m+1}\right).$$

Δ

∇ *Proof of Lemma 3.5.1* Consider the positive-definiteness of $T(1,m+1\to 1,m+1)$. Since n^{m+1} belongs to the null-space of $\left(T(1,m\to 1,m)\ T(m+1\to 1,m)\right)$, we see that

$$<\left(s^{m+1} + \alpha n^{m+1}\right), T(1,m+1\to 1,m+1)\left(s^{m+1} + \alpha n^{m+1}\right)>$$

$$= <\left(s^m\right), T(1,m\to 1,m)\left(s^m\right)> + \alpha^2\Gamma_{m+1},\ \forall s^{m+1} = \begin{pmatrix} s^m \\ 0\end{pmatrix} \in R^{m+1},\ \forall\alpha \in R.$$

Now any $(m+1)$-vector $g^{m+1} = \left(\{g^m\}^T\ g_{m+1}\right)^T$ with $g^m \in R^m$ can be uniquely written as $s^{m+1} + \alpha n^{m+1}$ when $s^m \in R^m$ and $\alpha \in R$ (in fact, $\alpha = g_{m+1}$ and $s^m = g^m + g_{m+1}T(1,m\to 1,m)^{-1}T(m+1\to 1,m)$). Therefore, from the above, we see that

$$<(g^{m+1}), T(1,m+1\to 1,m+1)(g^{m+1})> \;>\; 0, \; \forall g^{m+1} \in R^{m+1},$$

iff $T(1,m\to 1,m)$ is p.d. and $\Gamma_{m+1} > 0$.

That the stated inverse of $T(1,m+1\to 1,m+1)$ is correct when $\Gamma_{m+1} > 0$ can be seen by pre- and post-multiplying it by $T(1,m+1\to 1,m+1)$ when the latter is partitioned as in Lemma 3.5.1.

Δ

3.6 An Iterative, Gradient-Decomposition Based, Algorithm for Control Function Optimisation

The optimisation problem of 3.2 is considered when an $\varepsilon(x_s)$-approximation to the x_s-optimal control function belonging to the control space U is required and when Assumption 3.4.1 holds. The initial condition x_s considered may, of course, be the initial condition $\mathfrak{X}_s$ of 1.3. The algorithm may be used when Assumption 3.4.1 does not hold provided that $T(1,m+1,m)$ is p.d. for all considered m.

The optimisation algorithm iteratively defines basis-functions δ and deduces $U\to G$ map matrix elements which enable optimisation on $U(1,m)''$ to be achieved as m is (iteratively) increased until the x_s-optimal control function belonging to $U(1,\dot{m})''$ is the desired $\varepsilon(x_s)$-approximation. The structure of the algorithm is such that checks can easily be made on the validity of the deduced $U\to G$ map matrix elements without the computational expense of applying to the dynamical system of 3.2 the calculated optimal control function belonging to each translated linear manifold $U(1,m)''$ and checking whether the resulting performance index and gradient function are those predicted using the deduced $U\to G$ map matrix elements. For clarity, and because the checks are easy to implement, we do not include the checks in the algorithm statement but outline them in Comment 3.6.2, which follows the algorithm statement. The definitions and notations introduced so far in Chapter 3 are used throughout. The algorithm is next stated, with sufficient discussion to explain its operation.

1) Choose an initial control function $u_1 \in \mathcal{U}$ which is a guess at the x_s-optimal control function belonging to $\mathcal{U}$. Go to 2).

2) Calculate the gradient function $\left(\partial V(x_s,u_1)/\partial u\right)$ and decompose it as $F_1 g_1(x_s,u_1)$, which involves the definition of a basis-function $\mathfrak{f}_1$ (recall Definition 3.5.1). If $g_1(x_s,u_1) = 0$, stop since u_1 is the desired x_s-optimal control function belonging to $\mathcal{U}$. If $g_1(x_s,u_1) \neq 0$, continue by choosing a variation $\Delta u_1(1) \in R$ in the component of $\mathfrak{f}_1$ present in the control function and by perturbing u_1 to

$$u_2 = u_1 + F_1 \Delta u_1(1).$$

Set the iteration index, j, equal to 2. Go to 3) to calculate the resulting performance index and gradient function.

3) Calculate the performance index $V\left(x_s,u_j\right)$ and the gradient function $\left(\partial V(x_s,u_j)/\partial u\right)$. Decompose the latter as $F_j g^j(x_s,u_j)$, which involves the definition of a basis-function $\mathfrak{f}_j$ (recall Definition 3.5.2). Go to 4) if j = 2 and go to 5) if j > 2 – to deduce $\mathcal{U}\rightarrow G$ map matrix elements.

4) <u>Deduce $\mathcal{U}\rightarrow G$ map matrix elements for j = 2.</u>

The gradient function $\left(\partial V(x_s,u_1)/\partial u\right)$ is exactly characterised by the components

$$g^k(x_s,u_1) = \begin{bmatrix} g_1(x_s,u_1) \\ O(k-1,1) \end{bmatrix} \in R^k, \ \forall k > 1, \tag{3.21}$$

of basis-functions $\mathfrak{f}_1, .., \mathfrak{f}_k$ (i.e. $\left(\partial V(x_s,u_1)/\partial u\right) = F_k g^k(x_s,u_1)$) even though $\mathfrak{f}_2, .., \mathfrak{f}_k$ have not yet been defined. This arises because $g_1(x_s,u_1)$ and $\mathfrak{f}_1$ were chosen so that $\left(\partial V(x_s,u_1)/\partial u\right) = F_1 g_1(x_s,u_1)$.

Similarly, the gradient function $\left(\partial V(x_s,u_2)/\partial u\right)$ is exactly characterised by the components

$$g^k(x_s,u_2) = \begin{bmatrix} g^2(x_s,u_2) \\ O(k-2,1) \end{bmatrix} \in R^k, \ \forall k > 2,$$

of basis-functions $\delta_1, \ldots, \delta_k$, even though $\delta_3, \ldots, \delta_k$ have not yet been defined.

The change in the components of basis-functions $\delta_1, \ldots, \delta_k$ present in the gradient function due to the change $\Delta u_1(1)$ in the component of δ_1 present in the control function is therefore

$$\Delta g^k\big(\Delta u_1(1)\big) = g^k(x_s,u_2) - g^k(x_s,u_1), \ \forall k > 2.$$

The $U \to G$ map matrix $T(1\to 1,k)$ must therefore satisfy

$$\Delta g^k\big(\Delta u_1(1)\big) = T(1\to 1,k)\Delta u_1(1), \ \forall k > 2,$$

so that

$$T(1\to 1,k) = \Delta g^k\big(\Delta u_1(1)\big)/\Delta u_1(1) \in R^k, \ \forall k > 2. \tag{3.22}$$

Clearly $T(1\to 1,k)$ of (3.22) can be partitioned as

$$T(1\to 1,k) = \begin{bmatrix} T(1,1\to 1,1) \\ T(1\to 2) \\ O(k-2,1) \end{bmatrix}, \ \forall k > 2, \tag{3.23}$$

where $T(1,1\to 1,1) = T(1\to 1) \in R$.

From Lemma 3.4.1, $T(1,1\to 1,1)$ is p.d. so that its inverse exists. Compute $T(1,1\to 1,1)^{-1}$. Since optimisation on $U(1)''$ is now possible, go to 6) to determine whether the x_s-optimal control function belonging to $U(1)''$ is the desired $\varepsilon\big(x_s\big)$-approximation.

5) <u>Deduce $U\to G$ map matrix elements for j > 2.</u>

Employing the approach used in 4), we see that the gradient function change caused by the control function change from u_{j-1} to u_j of $F_{j-1}\Delta u^{j-1}(j-1)$ is exactly characterised by the change

$$\Delta g^k\left(\Delta u^{j-1}(j-1)\right) = \begin{pmatrix} g^j(x_s,u_j) \\ O(k-j,1) \end{pmatrix} - \begin{pmatrix} g^{j-1}(x_s,u_{j-1}) \\ O(k-j+1,1) \end{pmatrix} \in R^k, \ \forall k > j,$$

in the components of $f_1, .., f_k$ present in the gradient function, i.e. $\left(\partial V(x_s,u_j)/\partial u\right) - \left(\partial V(x_s,u_{j-1})/\partial u\right) = F_k \Delta g^k\left(\Delta u^{j-1}(j-1)\right)$, $\forall k > j$.

Partition $\Delta g^k\left(\Delta u^{j-1}(j-1)\right)$ and $\Delta u^{j-1}(j-1)$ as

$$\Delta g^k\left(\Delta u^{j-1}(j-1)\right) = \begin{pmatrix} \Delta g^{j-2}\left(\Delta u^{j-1}(j-1)\right) \\ \Delta g_{j-1}\left(\Delta u^{j-1}(j-1)\right) \\ \Delta g_j\left(\Delta u^{j-1}(j-1)\right) \\ O(k-j,1) \end{pmatrix}, \quad \Delta u^{j-1}(j-1) = \begin{pmatrix} \Delta u^{j-2}(j-1) \\ \Delta u_{j-1}(j-1) \end{pmatrix}, \quad \forall k > j,$$

where $\Delta g^{j-2}\left(\Delta u^{j-1}(j-1)\right), \Delta u^{j-2}(j-1) \in R^{j-2}$,

$\Delta g_{j-1}\left(\Delta u^{j-1}(j-1)\right), \Delta g_j\left(\Delta u^{j-1}(j-1)\right), \Delta u_{j-1}(j-1) \in R$.

Then, since $T(1,j-2\to 1,j-2)$ is already available from the last iteration of the algorithm , it can be seen (from the Proof at the end of 5), below) that

$$T(j-1\to 1,j-2) = \{\Delta g^{j-2}\left(\Delta u^{j-1}(j-1)\right) - T(1,j-2\to 1,j-2)\Delta u^{j-2}(j-1)\}/\Delta u_{j-1}(j-1) \in R^{j-2}, \tag{3.24}$$

$$T(j-1\to j-1) = \{\Delta g_{j-1}\left(\Delta u^{j-1}(j-1)\right) - T(j-1\to 1,j-2)^T \Delta u^{j-2}(j-1)\}/\Delta u_{j-1}(j-1) \in R, \tag{3.25}$$

$$T(j-1\to j) = \Delta g_j\left(\Delta u^{j-1}(j-1)\right)/\Delta u_{j-1}(j-1) \in R, \tag{3.26}$$

$$T(j\to 1,j-1) = \begin{pmatrix} O(j-2,1) \\ T(j-1\to j) \end{pmatrix} \in R^{j-1}. \tag{3.27}$$

Since $T(1,j-1\to 1,j-1)$ is symmetric (Comment 3.3.2), it can now be constructed by bordering $T(1,j-2\to 1,j-2)$, which is already available, giving: $T(1,j-1\to 1,j-1) = \begin{pmatrix} T(1,j-2\to 1,j-2) & T(j-1\to 1,j-2) \\ T(j-1\to 1,j-2)^T & T(j-1\to j-1) \end{pmatrix}$.

Since $T(1,j-2\to 1,j-2)$ and $T(1,j-1\to 1,j-1)$ are p.d. when Assumption 3.4.1 holds (Lemma 3.4.1) and are both symmetric, $T(1,j-1\to 1,j-1)^{-1}$ exists and can be computed using Lemma 3.5.1. Since x_s-optimisation on $U(1,j-1)$" is now possible, go to 6) to determine whether the x_s-optimal control function belonging to $U(1,j-1)$" is the required $\varepsilon(x_s)$-approximation

∇ *Proof of (3.24-27)* $T(1,j-1\to 1,k) \in M(R^{j-1}\to R^k)$ can be partitioned as

$$T(1,j-1\to 1,k) = \begin{pmatrix} T(1,j-2\to 1,j-2) & T(j-1\to 1,j-2) \\ T(1,j-2\to j-1) & T(j-1\to j-1) \\ T(j,j-2\to j) & T(j-1\to j) \\ T(1,j-2\to j+1,k) & T(j-1\to j+1,k) \end{pmatrix}, \ \forall k \geqq j+1,$$

where $T(1,j-2\to 1,j-2) \in M(R^{j-2}\to R^{j-2})$, $T(j-1\to 1,j-2) \in R^{j-2}$,

$T(1,j-2\to j-1)$, $T(1,j-2\to j) \in M(R^{j-2}\to R^1)$,

$T(j-1\to j-1)$, $T(j-1\to j) \in R$,

$T(1,j-2\to j+1,k) \in M(R^{j-2}\to R^{k-j})$, $T(j-1\to j+1,k) \in R^{k-j}$.

Since $T(1,j-1\to 1,j-1)$ is symmetric (Comment 3.3.2) and can be partitioned as $T(1,j-1\to 1,j-1) = \begin{pmatrix} T(1,j-2\to 1,j-2) & T(j-1\to 1,j-2) \\ T(1,j-2\to j-1) & T(j-1\to j-1) \end{pmatrix}$, we see that

$$T(1,j-2\to j-1) = T(j-1\to 1,j-2)^T. \tag{3.28}$$

Now $T(i\to 1,k)$ can be partitioned as

$$T(i\to 1,k) = \begin{pmatrix} T(i\to 1,i+1) \\ T(i\to i+2,k) \end{pmatrix} \in R^k, \ \forall i \in I(j-2), \ \forall k \geqq i+2, \tag{3.29}$$

where $T(i\to 1,i+1) \in R^{i+1}$ and where, as we shall see later:

$$T(i\to i+2,k) = O(k-i-1,1), \ \forall i \in I(j-2), \ \forall k \geqq i+2. \tag{3.30}$$

Also, $T(1,j-2\to 1,k)$ may be partitioned in the following two ways:

$$T(1,j-2\to 1,k) = \begin{pmatrix} T(1\to 1,k) & \dots & T(j-2\to 1,k) \end{pmatrix} = \begin{pmatrix} T(1,j-2\to 1,j-2) \\ T(1,j-2\to j-1) \\ T(1,j-2\to j) \\ T(1,j-2\to j+1,k) \end{pmatrix}. \tag{3.31}$$

On using (3.29) and (3.30) in (3.31), it may be seen that

$$\mathcal{T}(1,j-2\to j) = \mathcal{O}(1,j-2), \tag{3.32}$$

$$\mathcal{T}(1,j-2\to j+1,k) = \mathcal{O}(k-j,j-2),\ \forall k > j. \tag{3.33}$$

Now $\mathcal{T}(1,j-1\to 1,k)$ must satisfy

$$\Delta g^k\left(\Delta u^{j-1}(j-1)\right) = \mathcal{T}(1,j-1\to 1,k)\Delta u^{j-1}(j-1),\ \forall k > j. \tag{3.34}$$

Since $\mathcal{T}(1,j-2\to 1,j-2)$ is known at this stage of the algorithm, we can use our partitions for $\Delta g^k\left(\Delta u^{j-1}(j-1)\right)$ and $\Delta u^{j-1}(j-1)$ of stage 5) and our partition for $\mathcal{T}(1,j-1\to 1,k)$ of the previous page in (3.34), together with (3.28), (3.32) and (3.33), to give (3.24), (3.25), (3.26) and

$$\mathcal{T}(j-1\to j+1,k) = \mathcal{O}(k-j,1),\ \forall k \geq j+1. \tag{3.35}$$

We can now justify (3.30). It can be seen from (3.23) that $\mathcal{T}(i\to i+2,k)$ is indeed given by (3.30) when $i = 1$. We can then see that (3.35) is valid for $j = 3$, so that (3.30) holds when $i = 2$. Using this argument iteratively enables (3.30) to be justified.

Since $\mathcal{T}(1,j\to 1,j)$ can be partitioned as

$$\mathcal{T}(1,j\to 1,j) = \begin{bmatrix} \mathcal{T}(1,j-1\to 1,j-1) & \mathcal{T}(j\to 1,j-1) \\ \mathcal{T}(1,j-1\to j) & \mathcal{T}(j\to j) \end{bmatrix}$$

and $\mathcal{T}(1,j\to 1,j)$ is symmetric (Comment 3.3.2), we see that

$$\mathcal{T}(j\to 1,j-1) = \mathcal{T}(1,j-1\to j)^T. \tag{3.36}$$

On partitioning $\mathcal{T}(1,j-1\to j)$ as $\left(\mathcal{T}(1,j-2\to j)\ \mathcal{T}(j-1\to j)\right)$ and on using (3.32) and (3.36), (3.27) results.

Δ This concludes the proof of (3.24-27).

6) <u>Consider x_s-Optimisation on $U(1,j-1)$"</u>

Since $T(1,j-1\rightarrow 1,j-1)$ is p.d. when Assumption 3.4.1 holds (from Lemma 3.4.1), it can be predicted by using the approach of 3.3 that:

(a) the x_s-optimal control function belonging to $U(1,j-1)$" is

$$u(j-1;x_s)^* = u_{j-1} + F_{j-1}\delta u^{j-1}(j-1;x_s)^*$$

where $\delta u^{j-1}(j-1;x_s)^* = -T(1,j-1\rightarrow 1,j-1)^{-1}g^{j-1}(x_s,u_{j-1})$,

(b) the x_s-minimal performance index on $U(1,j-1)$" is

$$V(j-1;x_s)^* = V(x_s,u_{j-1}) - \tfrac{1}{2}\langle (g^{j-1}(x_s,u_{j-1})), T(1,j-1\rightarrow 1,j-1)^{-1}(g^{j-1}(x_s,u_{j-1}))\rangle,$$

(c) the gradient function following x_s-optimisation on $U(1,j-1)$" is

$$(\partial V(j-1;x_s)^*/\partial u) = F_k g^k(j-1;x_s)^*, \ \forall k > j,$$

where
$$g^k(j-1;x_s)^* = \begin{pmatrix} g^{j-1}(x_s,u_{j-1}) \\ O(k-j+1,1) \end{pmatrix} + T(1,j-1\rightarrow 1,k)\delta u^{j-1}(j-1;x_s)^*$$

$$= \begin{pmatrix} g^{j-1}(x_s,u_{j-1}) \\ O(k-j+1,1) \end{pmatrix} - \begin{pmatrix} T(1,j-1\rightarrow 1,j-1) & \\ T(1,j-2\rightarrow j) & T(j-1\rightarrow j) \\ T(1,j-2\rightarrow j+1,k) & T(j-1\rightarrow j+1,k) \end{pmatrix} T(1,j-1\rightarrow 1,j-1)^{-1}g^{j-1}(x_s,u_{j-1})$$

$$= \begin{pmatrix} O(j-1,1) \\ g_j(j-1;x_s)^* \\ O(k-j,1) \end{pmatrix}$$

where $g_j(j-1;x_s)^* = \big(O(1,j-2)\ T(j-1\rightarrow j)\big)T(1,j-1\rightarrow 1,j-1)^{-1}g^{j-1}(x_s,u_{j-1})$ and $T(j-1\rightarrow j)$, etc., are those just determined in stage 4) if $j = 2$ or in stage 5) if $j > 2$. Note that if $j = 2$, $T(1,j-2\rightarrow j)$ and $T(1,j-2\rightarrow j+1,k)$ can be omitted from the above partition of $T(1,j-1\rightarrow 1,k)$ since $T(1,1\rightarrow 1,k) = T(1\rightarrow 1,k) = \big(T(1,1\rightarrow 1,1)\ T(1\rightarrow 2)\ T(1\rightarrow 3,k)\big)^T$. This does not, however, affect the final result for $g^k(j-1;x_s)^*$.

The gradient function following x_s-optimisation on $\mathcal{U}(1,j-1)''$ is therefore predicted to be exactly characterised by the components $g^j(j-1;x_s)* = \left(\mathcal{O}(1,j-1)\ g_j(j-1;x_s)*\right)^T$ of basis-functions $\delta_1, .., \delta_j$.

Lower-bounds $V\left(j;x_s\right)^*_{j-1}$ or $\tilde{V}\left(j;x_s\right)^*_{j-1}$ for the x_s-minimal performance index on the control space $\mathcal{U}$ can now be evaluated using Remark 3.4.1. We suppose here that $\tilde{V}\left(j;x_s\right)^*_{j-1}$ is actually calculated, although the following holds when $\tilde{V}\left(j;x_s\right)^*_{j-1}$ is replaced by $V\left(j;x_s\right)^*_{j-1}$.

If $\left|\ V\left(j-1;x_s\right)* - \tilde{V}\left(j;x_s\right)^*_{j-1}\ \right| \leqq \varepsilon$, the x_s-optimal control function belonging to $\mathcal{U}(1,j-1)''$, $u\left(j-1;x_s\right)*$, is the desired $\varepsilon\left(x_s\right)$-approximation, so stop.

If $\left|\ V\left(j-1;x_s\right)* - \tilde{V}\left(j;x_s\right)^*_{j-1}\ \right| \not\leqq \varepsilon$, the x_s-optimal control function belonging to $\mathcal{U}(1,j-1)''$ is not the desired $\varepsilon\left(x_s\right)$-approximation, so go to 7) to cause a control function change which will enable those $\mathcal{U}\to\mathcal{G}$ map matrix elements to be deduced which are required for x_s-optimisation on $\mathcal{U}(1,j)''$.

7) <u>Choose next control function</u>

We desire to cause a change in the component of basis-function δ_j present in the control function to enable sufficient $\mathcal{U}\to\mathcal{G}$ map matrix elements to be deduced to enable x_s-optimisation on $\mathcal{U}(1,j)''$ to be achievable. To test the accuracy of the $\mathcal{U}\to\mathcal{G}$ map matrix elements which have already been deduced (see Comment 3.6.2), and to optimise as far as possible, we attempt to reduce to zero the components of basis-functions $\delta_1, .., \delta_{j-1}$ present in the gradient function at the same time as making a change of $\Delta u_j(j)$ in the component of δ_j which is present in the control function. This is possible because $\mathcal{T}(j\to1,j-1)$ is available at this stage of the algorithm, even though no change in the component of δ_j present in the control function has yet been made. The procedure for choosing a suitable control function u_{j+1} is next explained.

Choose $\Delta u_j(j) \in R$.

Partition $g^j(x_s,u_j) \in R^j$ as $\left(g^{j-1}(x_s,u_j)^T \; g_j(x_s,u_j)\right)^T$, where $g^{j-1}(x_s,u_j) \in R^{j-1}$. Then the components of basis-functions $\delta_1, .., \delta_{j-1}$ present in the gradient function $\left(\partial V(x_s,u_{j+1})/\partial u\right)$ when $u_{j+1} = u_j + F_j\Delta u^j(j)$ are given as

$g^{j-1}(x_s,u_{j+1}) = g^{j-1}(x_s,u_j) + T(1,j-1\to 1,j-1)\Delta u^{j-1}(j) + T(j\to 1,j-1)\Delta u_j(j)$

when $\Delta u^j(j) \in R^j$ is partitioned as $\left(\Delta u^{j-1}(j)^T \; \Delta u_j(j)\right)^T$ and $\Delta u^{j-1}(j) \in R^{j-1}$.

If j = 2, $T(j\to 1,j-1) = T(2\to 1) = T(1\to 2)$ -of (3.23)- since $T(1,2\to 1,2)$ can be partitioned as $\begin{pmatrix} T(1\to 1) & T(2\to 1) \\ T(1\to 2) & T(2\to 2) \end{pmatrix}$ and $T(1,2\to 1,2)$ is symmetric (Comment 3.3.2).

If j > 2, $T(j\to 1,j-1)$ is available from (3.27).

The components of basis-functions $\delta_1, .., \delta_{j-1}$ present in the gradient function $\left(\partial V(x_s,u_{j+1})/\partial u\right)$ should therefore all be zero (if the deduced $U\to G$ map matrix terms $T(1,j-1\to 1,j-1)$ and $T(j\to 1,j-1)$ are correct) in spite of the chosen change $\Delta u_j(j)$ in the component of δ_j present in the control function if

$$\Delta u^j(j) = \begin{pmatrix} -T(1,j-1\to 1,j-1)^{-1}\{g^{j-1}(x_s,u_j) + T(j\to 1,j-1)\Delta u_j(j)\} \\ \Delta u_j(j) \end{pmatrix}. \qquad (3.37)$$

Calculate $\Delta u^j(j)$ of (3.37). Set

$$u_{j+1} = u_j + F_j\Delta u^j(j)$$
$$j = j+1$$

and go to 3) to determine the gradient function for the new control function u_j.

This concludes the statement of the gradient-decomposition based optimisation algorithm.

∇ *Comment 3.6.1* Each change $\Delta u_j(j)$ of the above algorithm should be chosen to be large enough to cause a gradient function change sufficiently large to enable *U→G* map matrix elements to be deduced with adequate accuracy. Since the *U→G* map is linear for the optimisation problem of 3.2, each change $\Delta u_j(j)$ may be made as large as is desired. There should therefore be no difficulty in choosing suitable changes

Δ (our computational experience confirms this).

∇ *Comment 3.6.2* For all j > 2, the control function u_j generated by the above algorithm is designed (using the deduced matrices $T(1,j-2\to 1,j-2)$ and $T(j-1\to 1,j-2)$) to have a gradient function $\{\partial V(x_s,u_j)/\partial u\}$ which contains zero components of basis-functions δ_1, .., δ_{j-2}. Thus if the actual components of δ_1, .., δ_{j-2} present in $\{\partial V(x_s,u_j)/\partial u\}$ are relatively small compared to, say, the components of δ_1, .., δ_{j-2} present in $\{\partial V(x_s,u_{j-1})/\partial u\}$, we can deduce that the *U→G* map matrices used to construct u_j were of satisfactory accuracy and that it is worthwhile to continue to determine $T(1,j-1\to 1,j-1)$ by bordering $T(1,j-2\to 1,j-2)$. If, however, the components of δ_1, .., δ_{j-2} contained in $\{\partial V(x_s,u_j)/\partial u\}$ are not relatively small, we can deduce that the elements of the deduced matrices $T(1,j-2\to 1,j-2)$ and $T(j-1\to 1,j-2)$ are significantly in error and that there can be little justification for continuing to determine $T(1,j-1\to 1,j-1)$ by bordering $T(1,j-2\to 1,j-2)$. If $T(1,j-2\to 1,j-2)$ and $T(j-1\to 1,j-2)$ contain unacceptable errors (in the above sense) but $T(1,i\to 1,i)$ and $T(i+1\to 1,i)$ do not, for some $i \in I(j-2)$, a sensible plan would be to restart optimisation using the above algorithm with the predicted (using the deduced matrix $T(1,i\to1,i)$) x_s-optimal control function belonging to $U(1,i)$" (for the largest $i \in I(j-2)$ with acceptable $T(1,i\to1,i)$ and $T(i+1\to1,i)$, preferably) as the initial control function u_1.

For each $j \geqq 2$, it is desirable to check that each deduced $U\to G$ map matrix $T(1,j-1\to1,j-1)$ is p.d., as it should be when Assumption 3.4.1 holds (by Lemma 3.4.1). This is trivial when $j = 2$ and can be done when evaluating $T(1,j-1\to1,j-1)^{-1}$ using Lemma 3.5.1 if $j > 2$ since we shall only construct $T(1,j-1\to1,j-1)$ if $T(1,j-2\to1,j-2)$ is p.d. If $T(1,j-1\to1,j-1)$ is not p.d., its elements must be significantly in error and the same action should be taken as that which would be taken were it found that $\left(\partial V(x_s,u_{j+1})/\partial u\right)$ contains non-negligible components of basis-functions $\delta_1, .., \delta_{j-1}$.

Checks on the accuracy of the deduced $U\to G$ map matrices are desirable because without the checks there would be no way of telling whether or not the deduced $U\to G$ map matrix elements contain numerical errors which are large enough to render valueless all further calculations based on them. The ability to check the accuracy of each deduced $U\to G$ map matrix $T(1,j-1\to1,j-1)$ by calculating the gradient function $\left(\partial V(x_s,u_{j+1})/\partial u\right)$ <u>without</u> having to compute and apply the predicted (using the deduced matrix $T(1,j-1\to1,j-1)$) x_s-optimal control function belonging to $U(1,j-1)$" to the considered dynamical system equations and calculate the resulting gradient function is a valuable feature of our algorithm because $\left(\partial V(x_s,u_{j+1})/\partial u\right)$ contains information needed for optimisation on $U(1,j)$" while the gradient function for the x_s-optimal control function belonging to $U(1,j-1)$" does not, since the latter gradient function can be predicted in stage 6) of

Δ the above algorithm <u>before</u> x_s-optimisation on $U(1,j)$" is considered.

The following lemma will be of use later.

∇ <u>*Lemma 3.6.1*</u> Suppose k orthonormal basis-functions $\delta_1, .., \delta_k$ have been defined by the above algorithm and that there are no numerical errors. Then, if Assumption 3.4.1 holds, $T(1,k\to 1,k)$ is symmetric and tri-diagonal with all tri-diagonal elements non-zero and all diagonal

Δ elements p.d.

∇ <u>*Proof of Lemma 3.6.1*</u> Denote the scalar elements of $T(1,k\to 1,k)$ by t_{ip}, $\forall i,p \in I(k)$. Since $T(1,k\to 1,k)$ is symmetric (Comment 3.3.2):

$$t_{ip} = t_{pi}, \forall i,p \in I(k). \tag{3.38}$$

$T(1,k\to 1,k)$ can be partitioned as follows:

$$T(1,k\to 1,k) = \big(T(1\to 1,k) \quad \ldots \quad T(k\to 1,k)\big) \tag{3.39}$$

where $T(i\to 1,k) \in R^k$, $\forall i \in I(k)$.

From (3.23):

$$T(1\to 1,k) = \big(t_{11} \quad t_{21} \quad O(1,k-2)\big)^T, \forall k > 2. \tag{3.40}$$

From (3.29) and (3.30):

$$T(i\to 1,k) = \big(t_{1i} \quad t_{2i} \quad .. \quad t_{i+1,1} \quad O(1,k-i-1)\big)^T, \quad \forall k > i+1 \tag{3.41}$$

On using (3.38), (3.39), (3.40) and (3.41) together, it can be seen that $T(1,k\to 1,k)$ must have the following tri-diagonal form if $k = 6$, say:

$$T(1,k\to 1,k) = \begin{pmatrix} t_{11} & t_{21} & 0 & 0 & 0 & 0 \\ t_{21} & t_{22} & t_{32} & 0 & 0 & 0 \\ 0 & t_{32} & t_{33} & t_{43} & 0 & 0 \\ 0 & 0 & t_{43} & t_{44} & t_{54} & 0 \\ 0 & 0 & 0 & t_{54} & t_{55} & t_{65} \\ 0 & 0 & 0 & 0 & t_{65} & t_{66} \end{pmatrix}.$$

The above tri-diagonal structure of $T(1,k\to 1,k)$ is clearly independent of k.

It may be seen from Lemma 3.4.1 that $T(1,k\to 1,k)$ is p.d. when Assumption 3.4.1 holds, which requires that:

$$t_{ii} > 0, \forall i \in I(k).$$

It can be seen from (3.22) and (3.23) that $T(1\to 2) \neq 0$ only if $g_2(x_s,u_2) \neq 0$, i.e. if an orthonormal basis-function δ_2 needs to be defined to enable the gradient function $\left(\partial V(x_s,u_2)/\partial u\right)$ to be exactly characterised by components of defined basis-functions. If $g_2(x_s,u_2) = 0$, we see from Definition 3.5.2 that δ_2 will be zero on T, and so will not be orthonormal to δ_1. Similarly, it may be seen that $T(i\to i+1) \neq 0$ only if $g_{i+1}(x_s,u_{i+1}) \neq 0$, i.e. if an orthonormal basis-function δ_{i+1} has to be defined to enable $\left(\partial V(x_s,u_{i+1})/\partial u\right)$ to be exactly characterised by components of defined basis-functions. Since we have assumed in Lemma 3.6.1 that orthonormal basis-functions δ_1, .., δ_k have been defined, $T(i\to i+1) \neq 0$, $\forall i \in I(k-1)$. On using (3.38) and Definition 3.3.8, we therefore see that

$$t_{i+1,i} = t_{i,i+1} \neq 0, \forall i \in I(k-1).$$

Δ This concludes the proof of Lemma 3.6.1.

3.7 Comparative Effectiveness of the Gradient-Decomposition Based and Steepest-Descent Optimisation Algorithms

The object of this section is to compare the effectiveness at control function optimisation of the gradient-decomposition based optimisation algorithm of 3.6 with that of the well-known (piece-wise) steepest-descent algorithm. The optimisation problem of 3.2 is considered when Assumption 3.4.1 holds for some particular (but arbitrary) initial condition $x_s \in X(q)$, perhaps $\tilde{x}_s$ of 1.3. The results obtained are valid, however, when Assumption 3.4.1 does not hold provided that $T(1,m+1,m)$ is p.d. for all considered m. For comparison purposes it is assumed that the steepest-descent algorithm and gradient-decomposition based algorithm both start with the same initial control function $u_1 \in U$, that each algorithm is used independently and that no numerical errors occur. As in 3.4, we denote the x_s-minimal performance index on $U(1,m)$" by $V\big(m;x_s\big)^*$ and the x_s-minimal performance index on U by $V\big(x_s\big)^*$.

We next state the considered steepest-descent algorithm, designed to perform at most k iterations (i.e. k optimal control function changes in steepest-descent directions):

1) Set $\tilde{u}_0 = u_1$ and set $m = 0$. Go to 2).

2) Calculate the gradient function $\big(\partial V(x_s,\tilde{u}_m)/\partial u\big)$.

Stop if $\big(\partial V(x_s,\tilde{u}_m)/\partial u\big)$ is zero almost everywhere on T, since no further performance index improvement is then possible.

If $\big(\partial V(x_s,\tilde{u}_m)/\partial u\big)$ is not zero-almost-everywhere-on-T, minimise with respect to $\Omega(m) \in R$ the performance index $V\big(x_s,\tilde{u}_m-\Omega(m)\big(\partial V(x_s,\tilde{u}_m)/\partial u\big)\big)$. Denote the minimising value of $\Omega(m)$ by $\Omega(m)^*$.

Set $\tilde{u}_{m+1} = \tilde{u}_m - \Omega(m)*\left(\partial V(x_s,\tilde{u}_m)/\partial u\right)$ and set $m = m+1$.

Stop if $m = k$, since the desired k iterations have been performed. Otherwise, go to 2).

The main results of this section are contained in

∇ *Remark 3.7.1* Suppose that Assumption 3.4.1 holds and that $V(x_s,u_1)$ is not the x_s-minimal performance index on U, so that it is not true that $\left(\partial V(x_s,u_1)/\partial u\right)$ is zero almost everywhere on T.

(I) Suppose also that the gradient-decomposition based optimisation algorithm of 3.6 is used to achieve x_s-optimisation on $U(1)$", which involves the definition of a normalised basis-function δ_1, and that $V(1;x_s)* = V(x_s)*$. Then $V(x_s,\tilde{u}_1) = V(x_s)*$.

(II) Alternatively, suppose in addition that $V(1;x_s)* \neq V(x_s)*$ and that the gradient-decomposition based optimisation algorithm of 3.6 is used to optimise on $U(1,k)$" for $k > 1$ (which involves the sequential definition of k orthonormal basis-functions $\delta_1, .., \delta_k$). Then k iterations of the steepest-descent algorithm can be performed and

(i) $V(1;x_s)* = V(x_s,\tilde{u}_1)$,

(ii) $V(m;x_s)* < V(x_s,\tilde{u}_m)$, $\forall m \in I(2,k)$,

Δ (iii) $V(k;x_s)* < V(k-1;x_s)* < \ldots < V(1;x_s)* < V(x_s,\tilde{u}_0)$.

∇ *Comment 3.7.1* The gradient-decomposition based optimisation algorithm of 3.6 has to perform two gradient function evaluations before it can achieve x_s-optimisation on $U(1)$", and the gradient function evaluations required are the most expensive feature, computationally, associated with x_s-optimisation on $U(1)$" using the algorithm of 3.6. The steepest-descent algorithm considered requires one gradient function evaluation and one optimisation in a steepest-descent direction to determine $\tilde{u}_1$. Hence x_s-optimisation on $U(1)''$ and the determination of $\tilde{u}_1$ require about the same amount of computational effort. We therefore see from Remark 3.7.1(I) that the gradient-decomposition based optimisation algorithm of 3.6 and the steepest-descent algorithm considered are equally effective at control function optimisation if $V(1;x_s)^* = V(x_s)^*$. No further iterations of either algorithm are needed in this case since the x_s-minimal performance index on U is achieved in the first iteration.

If, however, $V(1;x_s)^* \neq V(x_s)^*$, the gradient-decomposition based algorithm can be used to optimise on $U(1,2)''$, etc. Probably the most expensive feature, computationally, associated with x_s-optimisation on $U(1,m)''$ using the algorithm of 3.6 after x_s-optimisation on $U(1)''$ is possible is the calculation of the further m-1 gradient functions which are required. The determination of $\tilde{u}_m$ given $\tilde{u}_1$ using the steepest-descent algorithm also requires m-1 further gradient function evaluations. We see from Remark 3.7.1(II) that $V(1;x_s)^* = V(x_s,\tilde{u}_1)$, $V(m;x_s)^* < V(x_s,\tilde{u}_m)$, $\forall m \in I(2,k)$. Hence $\tilde{u}_m$ is not the x_s-optimal control function belonging to $U(1,m)''$ if $m > 1$, and the further gradient function evaluations performed by the algorithm of 3.6 to enable x_s-optimisation on $U(1,m)''$ to be possible (after x_s-optimisation on $U(1)''$ is possible) can be considered to be used more effectively by that algorithm than the further gradient function evaluations (which are used by the steepest-descent algorithm to obtain $\tilde{u}_m$ given $\tilde{u}_1$) are used by the steepest-descent algorithm. Thus if $V(1;x_s)^* \neq V(x_s)^*$, the gradient-decomposition based optimisation algorithm of 3.6 can be considered to be more effective at control function

Δ optimisation than the steepest-descent algorithm.

To prove Remark 3.7.1 we first consider the case of Remark 3.7.1(II). We relate the orthonormal basis-functions $\delta_1, .., \delta_k$ defined by the gradient-decomposition based optimisation algorithm of 3.6 to the first k iterations of the steepest-descent algorithm, as follows.

Since $\tilde{u}_0 = u_1$, we see from (3.21) that

$$\left(\partial V(x_s,\tilde{u}_0)/\partial u\right) = F_k g^k(x_s,\tilde{u}_0) \;\; \varepsilon \;\; G(1)$$

where $$g^k(x_s,\tilde{u}_0) = \begin{bmatrix} g_1(x_s,u_1) \\ O(k-1,1) \end{bmatrix} \;\; \varepsilon \;\; R^k. \tag{3.42}$$

We see from (3.42), (3.48) and (3.50), by using an iterative argument, that

$$\left(\partial V(x_s,\tilde{u}_m)/\partial u\right) = F_k g^k(x_s,\tilde{u}_m) \;\varepsilon\; G(1,m+1),\; \forall m \;\varepsilon\; I(k-1) \tag{3.43}$$

where $$g^k(x_s,\tilde{u}_m) = \begin{bmatrix} g^m(x_s,\tilde{u}_m) \\ g_{m+1}(x_s,\tilde{u}_m) \\ O(k-m-1,1) \end{bmatrix} \;\varepsilon\; R^k,\; g^m(x_s,\tilde{u}_m) \;\varepsilon\; R^m, \quad \forall m \;\varepsilon\; I(k-1). \tag{3.44}$$

By using the results of 3.3 it can be seen that

$$\begin{aligned} V\left(x_s,\tilde{u}_m-\Omega(m)\left(\partial V(x_s,\tilde{u}_m)/\partial u\right)\right) &= V\left(x_s,\tilde{u}_m-\Omega(m)F_k g^k(x_s,\tilde{u}_m)\right) \\ = V\left(x_s,\tilde{u}_m\right) &- \Omega(m)<\left(g^k(x_s,\tilde{u}_m)\right),\left(g^k(x_s,\tilde{u}_m)\right)> \\ &+ \tfrac{1}{2}\Omega(m)^2<\left(g^k(x_s,\tilde{u}_m)\right),T(1,k\to1,k)\left(g^k(x_s,\tilde{u}_m)\right)>, \\ &\forall m \;\varepsilon\; I(0,k-1). \end{aligned} \tag{3.45}$$

Hence $\Omega(m)^*$, which minimises $V\left(x_s,\tilde{u}_m-\Omega(m)\left(\partial V(x_s,\tilde{u}_m)/\partial u\right)\right)$ with respect to $\Omega(m)$, is given by

$$\Omega(m)^* = <\left(g^k(x_s,\tilde{u}_m)\right),\left(g^k(x_s,\tilde{u}_m)\right)>/<\left(g^k(x_s,\tilde{u}_m)\right),T(1,k\to1,k)\left(g^k(x_s,\tilde{u}_m)\right)>, \quad \forall m \;\varepsilon\; I(0,k-1). \tag{3.46}$$

Then:

$$\tilde{u}_{m+1} = \tilde{u}_m - \Omega(m)^* F_k g^k(x_s,\tilde{u}_m) \in U(1,m+1)'', \forall m \in I(0,k-1).$$

Hence, from (3.45) and (3.46):

$$V(x_s,\tilde{u}_{m+1}) = V(x_s,\tilde{u}_m) - \tfrac{1}{2}\frac{<(g^k(x_s,\tilde{u}_m)),(g^k(x_s,\tilde{u}_m))>^2}{<(g^k(x_s,\tilde{u}_m)),T(1,k\to 1,k)(g^k(x_s,\tilde{u}_m))>}, \forall m \in I(0,k-1). \quad (3.47)$$

On using the $U\to G$ map property (Comment 3.3.4) of $T(1,k\to 1,k)$, we see that

$$\begin{aligned}(\partial V(x_s,\tilde{u}_{m+1})/\partial u) &= (\partial V(x_s,\tilde{u}_m - \Omega(m)^* F_k g^k(x_s,\tilde{u}_m))/\partial u) \\ &= F_k g^k(x_s,\tilde{u}_{m+1}) \in G(1,m+2), \forall m \in I(0,k-2),\end{aligned} \quad (3.48)$$

where $$g^k(x_s,\tilde{u}_{m+1}) = g^k(x_s,\tilde{u}_m) - \Omega(m)^* T(1,k\to 1,k) g^k(x_s,\tilde{u}_m). \quad (3.49)$$

Because $g^k(x_s,\tilde{u}_m)$ has the form of (3.42) if $m = 0$ and the form of (3.44) if $m > 0$, and because $T(1,k\to 1,k)$ is tridiagonal for our basis-functions $\delta_1, .., \delta_k$ (by Lemma 3.6.1), we see that $g^k(x_s,\tilde{u}_{m+1})$ of (3.49) has the form

$$g^k(x_s,\tilde{u}_{m+1}) = \begin{bmatrix} g^{m+1}(x_s,\tilde{u}_{m+1}) \\ g_{m+2}(x_s,\tilde{u}_{m+1}) \\ O(k-m-2,1) \end{bmatrix} \in R^k, \forall m \in I(0,k-2), \quad (3.50)$$

where $g^{m+1}(x_s,\tilde{u}_{m+1}) \in R^{m+1}$.

It may be seen that although $(\partial V(x_s,\tilde{u}_k)/\partial u)$ does not necessarily belong to $G(1,k)$, the components of basis-functions $\delta_1, ..., \delta_k$ present in it are the elements of $g^k(x_s,\tilde{u}_{m+1})$ of (3.49) when $m+1 = k$.

The x_s-minimal performance index on $U(1,m)''$, $V(m;x_s)^*$, can be seen to be related to the performance index $V(x_s,\tilde{u}_p)$ in the following way:

$$V(m;x_s)^* = V(x_s,\tilde{u}_p) - \tfrac{1}{2}<(g^m(x_s,\tilde{u}_p)),T(1,m\to 1,m)^{-1}(g^m(x_s,\tilde{u}_p))>,$$
$$\forall m \in I(k), \forall p \in I(0,m), \quad (3.51)$$

where $g^m(x_s,\tilde{u}_p)$ is the m-vector consisting of the components of basis-functions $\delta_1, .., \delta_m$ which are present in $(\partial V(x_s,\tilde{u}_p)/\partial u)$.

The remainder of the proof of Remark 3.7.1 is facilitated by the following simple lemmas.

∇ *Lemma 3.7.1* For the optimisation algorithm of 3.6, the

Δ x_s-optimal control function belonging to $U(1)$" is equal to $\tilde{u}_1$.

∇ *Proof of Lemma 3.7.1* We see from (3.42) that $(\partial V(x_s,\tilde{u}_0)/\partial u) = g_1(x_s,u_1)\delta_1$. The first iteration of the steepest-descent algorithm therefore optimises the component $-\Omega(0)g_1(x_s,u_1)$ of basis-function δ_1 present in the control function $u_1 - \Omega(0)g_1(x_s,u_1)\delta_1$. The resulting control function, $\tilde{u}_1$, is

Δ thus the x_s-optimal control function belonging to $U(1)$".

∇ *Lemma 3.7.2* Suppose $T(1,k{\to}1,k)$ is p.d. and $m{+}1 \in I(k)$. If $T(1,k{\to}1,k)$ has no distinct eigenvalues, $g^k(x_s,\tilde{u}_1) = O(k,1)$. Suppose next that $T(1,k{\to}1,k)$ does not have eigenvalues which are all equal. Define K to be the set of all non-zero k-vectors which cannot be written as a weighted sum of eigenvectors h_i of $T(1,k{\to}1,k)$ with associated eigenvalues λ_i which are equal. Define W to be that set which consists of all those k-vectors which contain no non-zero components of eigenvectors h_i with associated eigenvalues $\lambda_i \notin \{\lambda_b,\lambda_B\}$ and contain at least one non-zero component of an eigenvector with eigenvalue λ_b and at least one non-zero component of an eigenvector with eigenvalue equal to λ_B, where λ_b and λ_B are distinct. Clearly $W \subset K$. Then $g^k(x_s,\tilde{u}_{m+1}) \neq O(k,1)$ if and only if $g^k(x_s,\tilde{u}_m) \in K$. Also, $g^k(x_s,\tilde{u}_{m+1}) \in K$ if $g^k(x_s,\tilde{u}_m) \in K$. Further,

Δ $g^k(x_s,\tilde{u}_{m+1}) \in W$ if $g^k(x_s,\tilde{u}_m) \in W$.

∇ *Proof of Lemma 3.7.2* Since $T(1,k{\to}1,k)$ is assumed to be p.d. and is symmetric, it has the spectral representation

$$T(1,k\to 1,k) \;=\; \sum_{i=1}^{k} h_i\lambda_i h_i^{\mathrm{T}} \tag{3.52}$$

where $h_i \in R^k$, $\forall i \in I(k)$, are the orthonormal eigenvectors of $T(1,k\to 1,k)$

$\lambda_i > 0$, $\forall i \in I(k)$, are the associated eigenvalues (all real).

Now $g^k(x_s,\tilde{u}_{m+1})$ and $g^k(x_s,\tilde{u}_m)$ can be uniquely decomposed into components $\sigma_i(m+1)$ and $\sigma_i(m)$ of the eigenvectors h_i of $T(1,k\to 1,k)$, so that

$$g^k(x_s,\tilde{u}_{m+1}) \;=\; \sum_{i=1}^{k}\sigma_i(m+1)h_i, \quad g^k(x_s,\tilde{u}_m) \;=\; \sum_{i=1}^{k}\sigma_i(m)h_i. \tag{3.53}$$

On using (3.52) and (3.53) in (3.49) and (3.46), we see that the components $\sigma_i(m+1)$ of the eigenvectors h_i contained in $g^k(x_s,\tilde{u}_{m+1})$ are related to those in $g^k(x_s,\tilde{u}_m)$ by

$$\sigma_i(m+1) \;=\; \big(1-\Omega(m)^*\lambda_i\big)\sigma_i(m), \;\forall i \in I(k), \tag{3.54}$$

where $$\Omega(m)^* \;=\; \Big(\sum_{s=1}^{k}\sigma_s(m)^2\Big)\Big/\Big(\sum_{s=1}^{k}\sigma_s(m)^2\lambda_s\Big). \tag{3.55}$$

Suppose first that the only non-zero components $\sigma_i(m)$ of the eigenvectors h_i present in $g^k(x_s,\tilde{u}_m)$ are components of those eigenvectors associated with eigenvalues equal to λ_c, say, so that $\sigma_i(m) = 0$ if $\lambda_i \neq \lambda_c$, where λ_c is an eigenvalue. Then, $g^k(x_s,\tilde{u}_m) \notin K$ and, from (3.55)

$$\Omega(m)^* \;=\; 1/\lambda_c$$

so that, from (3.54): $\sigma_i(m+1) \;=\; 0$, $\forall i \in I(k)$.

Thus $g^k(x_s,\tilde{u}_{m+1}) = O(k,1)$ and $g^k(x_s,\tilde{u}_{m+1}) \notin K$ if $g^k(x_s,\tilde{u}_m) \notin K$.

Suppose next that $g^k(x_s,\tilde{u}_m) \in W$. Then $\sigma_i(m) = 0$ if $\lambda_i \notin \{\lambda_b,\lambda_B\}$ and $$\sum_{s=1}^{k}\sigma_s(m)^2\lambda_s/\lambda_B \;<\; \sum_{s=1}^{k}\sigma_s(m)^2,$$

so that, from (3.55): $\Omega(m)^* > 1/\lambda_B$.

Similarly: $\Omega(m)^* < 1/\lambda_b$.

We see from (3.54) that $\sigma_i(m+1) = 0$ only if either $\sigma_i(m) = 0$ or $\Omega(m)^* = 1/\lambda_i$. Since we have just shown that $1/\lambda_B < \Omega(m)^* < 1/\lambda_b$ when $g^k(x_s,\tilde{u}_m) \in W$, $g^k(x_s,\tilde{u}_{m+1})$ contains at least one non-zero component of an eigenvector of $T(1,k\to 1,k)$ which has λ_b as its associated eigenvalue and at least one non-zero component of an eigenvector which has λ_B as its associated eigenvalue. Also, $\sigma_i(m+1) = 0$ if $\lambda_i \notin \{\lambda_b, \lambda_B\}$.

Hence $g^k(x_s, \tilde{u}_{m+1}) \in W$ if $g^k(x_s, \tilde{u}_m) \in W$. Since this result is independent of λ_b and λ_B provided that $\lambda_b \neq \lambda_B$, it can be seen that $g^k(x_s, \tilde{u}_{m+1}) \in K$ if $g^k(x_s, \tilde{u}_m) \in K$.

Δ This concludes the proof of Lemma 3.7.2.

In the following we sometimes refer to element (i,j) of a matrix T, say, when we mean the scalar element t_{ij} of T.

∇ <u>*Lemma 3.7.3*</u> Consider k orthonormal basis-functions $\delta_1, .., \delta_k$ defined by the gradient-decomposition based optimisation algorithm of 3.6 when Assumption 3.4.1 holds. Denote element (1,1) of $T(1,m\to 1,m)^{-1}$ by t_{11}^m. Then, for the $U\to G$ map matrices $T(1,m\to 1,m)$ defined for the considered basis-functions $\delta_1, .., \delta_k$:

Δ $$t_{11}^k > t_{11}^{k-1} > \dots > t_{11}^2 > t_{11}^1 > 0.$$

∇ <u>*Proof of Lemma 3.7.3*</u> It can be seen (by Lemma 3.4.1) that $T(1,m\to 1,m)$ is bounded and p.d. when Assumption 3.4.1 holds, $\forall m \in I(k)$. We see from Definition 3.3.8 that $T(1,1\to 1,1)$ is a scalar. Since $T(1,1\to 1,1)$ is bounded and p.d.: $t_{11}^1 > 0$.

By Lemma 3.6.1, $T(1,k\to 1,k)$ is symmetric and tridiagonal with all tridiagonal elements non-zero. For all $m \in I(k-1)$, $T(1,k\to 1,k)$ can be partitioned as

$$T(1,k\to 1,k) = \begin{bmatrix} T(1,m\to 1,m) & T(m+1\to 1,m) & \dots \\ T(1,m\to m+1) & T(m+1\to m+1) & \dots \\ \dots\dots & \dots\dots & \dots \end{bmatrix}.$$

Hence, for all $m \in I(k-1)$, $T(1,m\to1,m)$ is tridiagonal with all tridiagonal elements non-zero. Also, $T(m+1\to1,m) = \left(O(1,m-1) \quad t_{m,m+1}\right)^{T}$, where $t_{m,m+1}$ denotes element $(m,m+1)$ of $T(1,k\to1,k)$ and $t_{m,m+1} \neq 0$ since element $(m,m+1)$ of $T(1,k\to1,k)$ is a tridiagonal element.

Now $T(1,m\to1,m)^{-1} = \text{adj}\left(T(1,m\to1,m)\right)/\Delta$ where Δ is the determinant of $T(1,m\to1,m)$. Since $T(1,m\to1,m)$ is p.d., $\Delta \neq 0$. Thus element $(1,m)$ of $T(1,m\to1,m)^{-1}$, denoted here by $t^{m}_{1,m}$, is equal to $t_{12} \times t_{23} \times .. \times t_{m-1,m}/\Delta$, where t_{12} denotes element $(1,2)$ of $T(1,m\to1,m)$, etc. Since all the tridiagonal elements of $T(1,m\to1,m)$ are non-zero, $t^{m}_{1,m} \neq 0$.

We see from Lemma 3.5.1 that

$$t^{m+1}_{11} = t^{m}_{11} + \left(1/\Gamma_{m+1}\right)(n_1)^2$$

where

n_1 denotes the first element of the $(m+1)$-vector

$$n^{m+1} = \begin{pmatrix} -T(1,m\to1,m)^{-1}T(m+1\to1,m) \\ 1 \end{pmatrix},$$

Γ_{m+1} is positive and bounded since $T(1,m+1\to1,m+1)$ is p.d. and bounded when Assumption 3.4.1 holds (by Lemma 3.4.1).

On using the above results we see that $n_1 = -t^{m}_{1,m} \times t_{m,m+1} \neq 0$. Hence $t^{m+1}_{11} > t^{m}_{11}$. Since this holds for all $m \in I(k-1)$, Lemma 3.7.3 is Δ now proved.

We can now come to the

∇ *Proof of Remark 3.7.1* Remark 3.7.1(I) follows from the fact that $V(1;x_s)^* = V(x_s,\tilde{u}_1)$, by Lemma 3.7.1. Result (i) of Remark 3.7.1(II) also comes from Lemma 3.7.1. We next consider the proof of the remainder of Remark 3.7.1(II).

Since we assume in Remark 3.7.1(II) that $V(1;x_s)^* \neq V(x_s)^*$, and since (by Lemma 3.7.1) $\tilde{u}_1$ is equal to the x_s-optimal control function belonging to $U(1)$", $\left(\partial V(x_s,\tilde{u}_1)/\partial u\right)$ is non-zero. Hence the components $g^k(x_s,\tilde{u}_1)$, of (3.44), which exactly characterise $\left(\partial V(x_s,\tilde{u}_1)/\partial u\right)$ are not all zero, i.e. $g^k(x_s,\tilde{u}_1) \neq O(k,1)$. We see from Lemma 3.7.2 that this can only occur if $g^k(x_s,\tilde{u}_0) \neq O(k,1)$ and contains non-zero components of eigenvectors of $T(1,k\to 1,k)$ which have distinct eigenvalues associated with them. On using the result of Lemma 3.7.2 iteratively, we see that $g^k(x_s,\tilde{u}_m) \neq O(k,1)$, $\forall m \in I(0,k)$. Hence, from (3.43), $\left(\partial V(x_s,\tilde{u}_m)/\partial u\right)$ is non-zero for all $m \in I(0,k)$ and k iterations of the steepest-descent considered can actually be performed. Since $T(1,k\to 1,k)$ is p.d. when Assumption 3.4.1 holds, we can also see, from (3.51), that $V(k;x_s)^* < V(x_s,\tilde{u}_k)$. This result is independent of k provided that $k > 1$. Hence $V(m;x_s)^* < V(x_s,\tilde{u}_m)$, $\forall m \in I(2,k)$, which proves result (ii) of Remark 3.7.1(II).

It can be seen from (3.51) and (3.42) that

$$V(m;x_s)^* = V(x_s,\tilde{u}_0) - \tfrac{1}{2}\left\langle \begin{pmatrix} g_1(x_s,u_1) \\ O(m-1,1) \end{pmatrix}, T(1,m\to 1,m)^{-1} \begin{pmatrix} g_1(x_s,u_1) \\ O(m-1,1) \end{pmatrix} \right\rangle$$

$$= V(x_s,\tilde{u}_0) - \tfrac{1}{2} g_1(x_s,u_1)^2 t_{11}^m, \ \forall m \in I(k),$$

where t_{11}^m denotes element (1,1) of $T(1,m\to 1,m)^{-1}$.

We have said above that, under the assumptions involved in Remark 3.7.1(II), $g^k(x_s,\tilde{u}_0) \neq O(k,1)$. This can only occur if $g_1(x_s,u_1) \neq 0$ since, from (3.42), $g^k(x_s,\tilde{u}_0) = \left(g_1(x_s,u_1) \ \ O(1,k-1)\right)^T$. We see from Lemma 3.7.3 that $t_{11}^k > t_{11}^{k-1} > \ldots > t_{11}^2 > t_{11}^1 > 0$. Hence $V(k;x_s)^* < V(k-1;x_s)^* < \ldots < V(2;x_s)^* < V(1;x_s)^* < V(x_s,\tilde{u}_0)$, which proves result (iii) of Remark 3.7.1(II) and completes the proof

Δ of Remark 3.7.1.

Further information concerning the comparative behaviour of the gradient-decomposition based optimisation algorithm of 3.6 and the steepest-descent algorithm considered is yielded by

∇ *Remark 3.7.2* Suppose that the assumptions of Remark 3.7.1(II) hold and that $g^k(x_s,\tilde{u}_0)$ contains non-zero components of eigenvectors of $T(1,k{\to}1,k)$ with associated eigenvalues equal to λ_b and λ_B, where λ_b and λ_B are eigenvalues of $T(1,k{\to}1,k)$ and $\lambda_b < \lambda_B$. Assume also that $g^k(x_s,\tilde{u}_0)$ contains no non-zero components of eigenvectors of $T(1,k{\to}1,k)$ with associated eigenvectors smaller than λ_b or larger than λ_B. Then:

$$\{V(x_s,\tilde{u}_m) - V(k;x_s)^*\} \leqq ((\lambda_B-\lambda_b)/(\lambda_B+\lambda_b))^{2m}\{V(x_s,\tilde{u}_0) - V(k;x_s)^*\},$$

Δ $\forall m \in I(k)$.

∇ *Comment 3.7.2* The result of Remark 3.7.2 provides an upper-bound for the rate at which the performance index $V(x_s,\tilde{u}_m)$ achieved by m steepest-descent iterations approaches from above the performance index $V(k;x_s)^*$, which can be achieved after x_s-optimisation on $U(1,k)$" using the gradient-decomposition based optimisation algorithm of 3.6. This is of conceptual interest but is not of direct computational significance since λ_b and λ_B will not usually be known. The result reveals, however, that $V(x_s,\tilde{u}_k) \nless V(k;x_s)^*$, which is consistent with the previous results

Δ of this section.

The proof of Remark 3.7.2 is facilitated by

∇ *Lemma 3.7.4* Suppose:

(a) $T(1,k{\to}1,k)$ is bounded and p.d.,

(b) Z denotes that linear manifold of R^k which is spanned by all those eigenvectors of $T(1,k{\to}1,k)$ which have associated eigenvalues λ_i such that

$\lambda_b \leqq \lambda_i \leqq \lambda_B$, where λ_b and λ_B are eigenvalues of $T(1,k\to 1,k)$,

(c) $z(g^k) = <(g^k), T(1,k\to 1,k)(g^k)><(g^k), T(1,k\to 1,k)^{-1}(g^k)>$,

(d) $\hat{z}(g^k) = [(\lambda_b+\lambda_B)^2/4\lambda_b\lambda_B]<(g^k),(g^k)>^2$.

Δ Then: $z(g^k) \leqq \hat{z}(g^k),\ \forall g^k \in Z$.

The proof of Lemma 3.7.4 is facilitated by

∇ <u>*Lemma 3.7.5*</u> If $0 < \lambda_b \leqq \lambda_i \leqq \lambda_B < \infty$:

(a) $(\lambda_b+\lambda_B)((1/\lambda_i)-(1/\lambda_B)) + ((1/\lambda_b)+(1/\lambda_B))(\lambda_i-\lambda_B) \leqq 0$

Δ (b) $(\lambda_b+\lambda_B)((1/\lambda_i)-(1/\lambda_b)) + ((1/\lambda_b)+(1/\lambda_B))(\lambda_i-\lambda_b) \leqq 0$.

∇ <u>*Proof of Lemma 3.7.5*</u> The results may perhaps best be proved by multiplying (a) throughout by $\lambda_i\lambda_b\lambda_B^2$ and by multiplying (b) throughout

Δ by $\lambda_i\lambda_b^2\lambda_B$. On cancelling the common terms, the required results emerge.

∇ <u>*Proof of Lemma 3.7.4*</u> We consider first the proof for the case when $g^k \in Q$, where $Q = \{g^k : <(g^k),(g^k)> = 1 \text{ and } g^k \in Z\}$.

Recall from (3.52) that $T(1,k\to 1,k)$ can be written as

$$T(1,k\to 1,k) = \sum_{i=1}^{k} h_i\lambda_i h_i^T.$$

For notational convenience, suppose that the orthonormal eigenvectors h_i are ordered so that $\lambda_1 = \lambda_b$ and $\lambda_k = \lambda_B$.

Let $\hat{g}^k = (h_1+h_k)/\sqrt{2}$. Then $\hat{g}^k \in Q$ and $z(\hat{g}^k) = \hat{z}(\hat{g}^k)$.

We have thus shown that the upper-bound $\hat{z}(g^k)$ of Lemma 3.7.4 is actually attained for $g^k = \hat{g}^k$. We complete the proof by first showing that $z(g^k) \leqq \hat{z}(g^k)$ for all $g^k = \hat{g}^k + \delta g^k \in Q$.

Now $\delta g^k \in Q$ can be uniquely decomposed into components σ_i of the orthonormal eigenvectors h_i of $T(1,k\to 1,k)$, and can then be written as

$$\delta g^k = \sum_{i=1}^{k} \sigma_i h_i \qquad (3.56)$$

where $\sigma_i = 0$ if $\lambda_i \notin \{\lambda_b,\lambda_B\},\ \forall i \in I(k)$.

For $\hat{g}^k + \delta g^k$ to belong to $\mathcal{Q}$, we require that

$$2<(\delta g^k),(\hat{g}^k)> \;=\; -<(\delta g^k),(\delta g^k)>. \tag{3.57}$$

On using (3.56), we see that (3.57) requires that

$$\sigma_1 + \sigma_k \;=\; -\left(\sum_{i=1}^{k} \sigma_i^2\right)/\sqrt{2}. \tag{3.58}$$

It can be seen that

$$z(\hat{g}^k+\delta g^k) \;=\; x(\hat{g}^k+\delta g^k)\,y(\hat{g}^k+\delta g^k) \tag{3.59}$$

where $x(\hat{g}^k+\delta g^k) = <(\hat{g}^k),T(1,k{+}1,k)(\hat{g}^k)> + 2<(\delta g^k),T(1,k{+}1,k)(\hat{g}^k)>$

$$+ <(\delta g^k),T(1,k{+}1,k)(\delta g^k)>$$

$$= (\lambda_b+\lambda_B)/2 + \sqrt{2}(\sigma_1\lambda_b+\sigma_k\lambda_B) + \sum_{i=1}^{k}\sigma_i^2\lambda_i,$$

$$y(\hat{g}^k+\delta g^k) = <(\hat{g}^k),T(1,k{+}1,k)^{-1}(\hat{g}^k)>$$

$$+ <(\delta g^k),T(1,k{+}1,k)^{-1}(\hat{g}^k)> + <(\delta g^k),T(1,k{+}1,k)^{-1}(\delta g^k)>$$

$$= ((1/\lambda_b)+(1/\lambda_B))/2 + \sqrt{2}((\sigma_1/\lambda_b)+(\sigma_k/\lambda_B)) + \sum_{i=1}^{k}\sigma_i^2/\lambda_i.$$

Two cases can occur: (i) $\sigma_1 = 0$; (ii) $\sigma_k = 0$. We treat these separately, for convenience.

Suppose first that $\sigma_1 = 0$. Then, we see from (3.58) and (3.59) that

$$x(\hat{g}^k+\delta g^k) = (\lambda_b+\lambda_B)/2 + \sum_{i=1}^{k}\sigma_i^2(\lambda_i-\lambda_B),$$

$$y(\hat{g}^k+\delta g^k) = ((1/\lambda_b)+(1/\lambda_B))/2 + \sum_{i=1}^{k}\sigma_i^2((1/\lambda_i)-(1/\lambda_B)).$$

Hence

$$z(\hat{g}^k+\delta g^k) = \hat{z}(\hat{g}^k+\delta g^k)$$

$$+ \{\sum_{i=1}^{k}\sigma_i^2(\lambda_i-\lambda_B)\}\{\sum_{i=1}^{k}\sigma_i^2((1/\lambda_i)-(1/\lambda_B))\}$$

$$+ \tfrac{1}{2}\sum_{i=1}^{k}\sigma_i^2\{(\lambda_b+\lambda_B)((1/\lambda_i)-(1/\lambda_B)) + ((1/\lambda_b)+(1/\lambda_B))(\lambda_i-\lambda_B)\}.$$

Since $\sigma_i = 0$ if $\lambda_i \notin \{\lambda_b,\lambda_B\}$ whenever $\hat{g}^k + \delta g^k \in \mathcal{Q}$, we see from Lemma 3.7.5(a) and the above that $z(\hat{g}^k+\delta g^k) \leqq \hat{z}(\hat{g}^k+\delta g^k)$ for all $\hat{g}^k + \delta g^k$ belonging to $\mathcal{Q}$ such that $\sigma_1 = 0$.

The case with $\sigma_k = 0$ can be treated in a similar way. We then see that when $\sigma_k = 0$:

$$z(\hat{g}^k+\delta g^k)_k = \hat{z}(\hat{g}^k+\delta g^k) + \{\sum_{i=1}^{k}\sigma_i^2(\lambda_i-\lambda_b)\}\{\sum_{i=1}^{k}\sigma_i^2((1/\lambda_i)-(1/\lambda_b))\}$$
$$+ \tfrac{1}{2}\sum_{i=1}^{k}\sigma_i^2\{(\lambda_b+\lambda_B)((1/\lambda_i)-(1/\lambda_b)) + ((1/\lambda_b)+(1/\lambda_B))(\lambda_i-\lambda_b)\}$$

Since $\sigma_i = 0$ if $\lambda_i \notin \{\lambda_b,\lambda_B\}$ whenever $\hat{g}^k + \delta g^k \in \mathcal{Q}$, we therefore see from Lemma 3.7.5(b) and the above that $z(\hat{g}^k+\delta g^k) \leq \hat{z}(\hat{g}^k+\delta g^k)$ for all $\hat{g}^k + \delta g^k$ belonging to $\mathcal{Q}$ such that $\sigma_k = 0$.

We have thus shown that

$$z(g^k) \leqq \hat{z}(g^k),\ \forall g^k \in \mathcal{Q}.$$

Since $z(\alpha g^k) = \alpha^4 z(g^k)$ and $\hat{z}(\alpha g^k) = \alpha^4\hat{z}(g^k)$ for any scalar α, we see that

$$z(g^k) \leqq \hat{z}(g^k),\ \forall g^k \in Z,$$

Δ which proves Lemma 3.7.4.

We can now come to the

∇ *Proof of Remark 3.7.2* Clearly

$$\{V(x_s,\tilde{u}_{m+1}) - V(k;x_s)^*\} = r(m)\{V(x_s,\tilde{u}_m) - V(k;x_s)^*\},\ \forall m+1 \in I(k), \quad (3.60)$$

where $r(m) = 1 - \{V(x_s,\tilde{u}_m) - V(x_s,\tilde{u}_{m+1})\}/\{V(x_s,\tilde{u}_m) - V(k;x_s)^*\}$.

From (3.47) and (3.51):

$$r(m) = (a(m) - d(m))/a(m),\ \forall m \in I(0,k-1), \quad (3.61)$$

where

$$a(m) = <(g^k(x_s,\tilde{u}_m)),T(1,k+1,k)(g^k(x_s,\tilde{u}_m))> \times$$
$$<(g^k(x_s,\tilde{u}_m)),T(1,k+1,k)^{-1}(g^k(x_s,\tilde{u}_m))>,$$
$$d(m) = <(g^k(x_s,\tilde{u}_m)),(g^k(x_s,\tilde{u}_m))>^2.$$

Under the assumptions of Remark 3.7.2, $g^k(x_s,\tilde{u}_0) \in W$, where W is as defined in Lemma 3.7.2. On using Lemma 3.7.2, we see that $g^k(x_s,\tilde{u}_{m+1}) \in W$, $\forall m \in I(0,k-1)$. Since $W \subset Z$, where Z is as defined in Lemma 3.7.4, we see from Lemma 3.7.4 and (3.61) that:

$$r(m) \leqq \left((\lambda_B-\lambda_b)/(\lambda_B+\lambda_b)\right)^2, \ \forall m \in I(0,k-1). \tag{3.62}$$

On using (3.62) in (3.61) as m is increased from one in unit Δ steps, the result of Remark 3.7.2 can be obtained.

3.8 Computed Examples

To demonstrate the usefulness of the developments which have been presented so far in this chapter, we present here the results of computations performed for two optimisation problems which are essentially the same as the optimisation problem of 3.2. For computational purposes, however, the linear space (for each problem) to which considered control functions belong is not infinite dimensional but is of finite (100) dimension. Since the modifications to enable our results and algorithm to be used for such problems consist mainly of obvious changes in notation, they are not discussed.

3.8.1 A Boiler

The boiler considered here is that of Nicholson {37} and is assumed to be described by the following difference equation:

$$x_{i+1} = Ax_i + Bu(i) + Cu_d(i)$$

where

$$x_i = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}_i = \begin{pmatrix} \text{steam density} \\ \text{superheated steam temperature} \\ \text{steam quality} \\ \text{water level displacement in drum} \end{pmatrix}_i$$

$$u(i) = \begin{pmatrix} u_1(i) \\ u_2(i) \end{pmatrix} = \begin{pmatrix} \text{water mass flow rate} \\ \text{fuel oil mass flow rate} \end{pmatrix}(i)$$

$$u_d(i) = \big(\text{steam (demand) mass flow rate}\big)(i)$$

$$A = \begin{pmatrix} 0.912E+00 & 0.367E-04 & 0.192E+01 & 0. \\ 0.631E+00 & 0.921E+00 & -0.909E+01 & 0. \\ 0.187E-02 & 0.168E-06 & 0.381E+00 & 0. \\ 0.120E+00 & 0.224E-04 & -0.217E+02 & 0.100E+01 \end{pmatrix}$$

$$B = \begin{bmatrix} -0.368E-03 & 0.499E-01 \\ 0.384E-02 & 0.131E+02 \\ -0.138E-03 & -0.619E-02 \\ -0.219E-02 & -0.244E+00 \end{bmatrix}, \quad C = \begin{bmatrix} -0.183E-02 \\ -0.729E+00 \\ 0.253E-03 \\ 0.731E-02 \end{bmatrix}.$$

The units associated with each element above are given in the paper by Nicholson, but do not concern us directly here.

A problem considered by Nicholson was the regulation of the boiler response to minimise the effect on x of a change in $u_d(i)$ from $u_d(i) = 0$ for $i < 1$ to $u_d(i) = 2.778$ for $i \geq 1$. We also considered this problem and attempted to minimise with respect to $u(i)$, $\forall i \in I(50)$, the performance index

$$V = \sum_{i=1}^{51} <(x_i), Q(x_i)>$$

where $Q = \text{diag}\{0.1E+06 \quad 0.1E+01 \quad 0.0 \quad 0.1E+05\}$.

The discrete-time convolution-description for the boiler was determined from the difference-equation and was used as the system description in our computations.

Because the discrete-time version of Assumption 3.4.1 is not satisfied in this case, lower-bounds for the x_s-minimal performance index on the control space cannot be calculated for the above problem. The nominal initial condition considered was $\tilde{x}_s = x_1 = O(4,1)$. The steepest-descent algorithm of 3.7 and the gradient-decomposition based optimisation algorithm of 3.6 were both used (with obvious modifications to allow for the discrete-nature of the considered control functions and the fact that lower-bounds could not be calculated), both starting with the same initial control function - namely $u(i) = O(2,1)$, $\forall i \in I(50)$. The performance index $V(\tilde{x}_s, \tilde{u}_m)$ obtained after m iterations of the steepest-descent algorithm is shown in Fig. 3.1 for each of the 17 iterations performed. The performance index predicted to be achievable after m iterations (control function changes) of the gradient-decomposition based optimisatio algorithm (calculated using the approach of stage 6) of the algorithm statement), $V(m;\tilde{x}_s)^*$, is also shown in Fig. 3.1 for each of the 17 iterations of that algorithm which were performed. The performance index for the initial control function, denoted by $V(\tilde{x}_s,\tilde{u}_0)$ and by $V(0;\tilde{x}_s)^*$ in Fig. 3.1, was actually 0.351E+05 (too large to be plotted in Fig. 3.1).

The predicted (using the $U\rightarrow G$ map matrix elements deduced using the gradient-decomposition based algorithm) $\tilde{x}_s$-optimal control function belonging to $U(1,17)''$ was applied to the convolution-description of the boiler and the associated gradient function was calculated. The gradient was found to be exactly characterised by the following components of basis-functions $\delta_1, .., \delta_{18}$:

$g^{18}(17;\tilde{x}_s)^*$ = (-0.1E+00 0.3E-01 -0.5E-01 0.3E-01 -0.1E-01 -0.6E-02 -0.2E-02 0.2E-02 -0.3E-01 -0.1E-01 -0.9E-01 -0.9E-01 -0.1E+00 -0.7E-01 -0.6E-01 -0.3E-01 -0.8E-02 0.3E+03)^T.

The components of basis-functions $\delta_1, .., \delta_{17}$ would all have been zero had the predicted $\tilde{x}_s$-optimal control function belonging to $U(1,17)''$ been precisely optimal. Although the components were not actually zero, they were very small compared with the component of δ_{18} present in the gradient function, which component could not have been reduced to zero after 17 iterations since optimisation on $U(1,18)''$ would only be possible after 18 iterations. The relatively small size of the components of $\delta_1, .., \delta_{17}$ present in the gradient function, compared to the component of δ_{18}, was considered to be very satisfactory for an IBM 7090 computer operating in the single precision mode (i.e. using eight significant figure, floating point, arithmetic). The performance index change, relative to the performance index for the initial control function, which occurred on applying the predicted $\tilde{x}_s$-optimal control function belonging to $\mathcal{U}(1,17)$" was within 0.15% of that change predicted using the deduced $\mathcal{U}\to\mathcal{G}$ map matrix elements, which was also considered to be rather satisfactory.

We see from Fig. 3.1 that the gradient-decomposition based optimisation algorithm of 3.6 reduced the performance index more rapidly, as a function of iterations, than did the steepest-descent algorithm: this result would have been expected from the discussion of 3.7 since the first iteration of the steepest-descent algorithm did not achieve the $\tilde{x}_s$-minimal performance index on the considered control space and $k > 1$ orthonormal basis-functions $\mathfrak{f}$ were defined by the gradient-decomposition based optimisation algorithm. Because of this and because 17 iterations of each algorithm required about the same amount of computing time (1.0 minutes), the gradient-decomposition based algorithm is far superior to the steepest-descent algorithm for this problem.

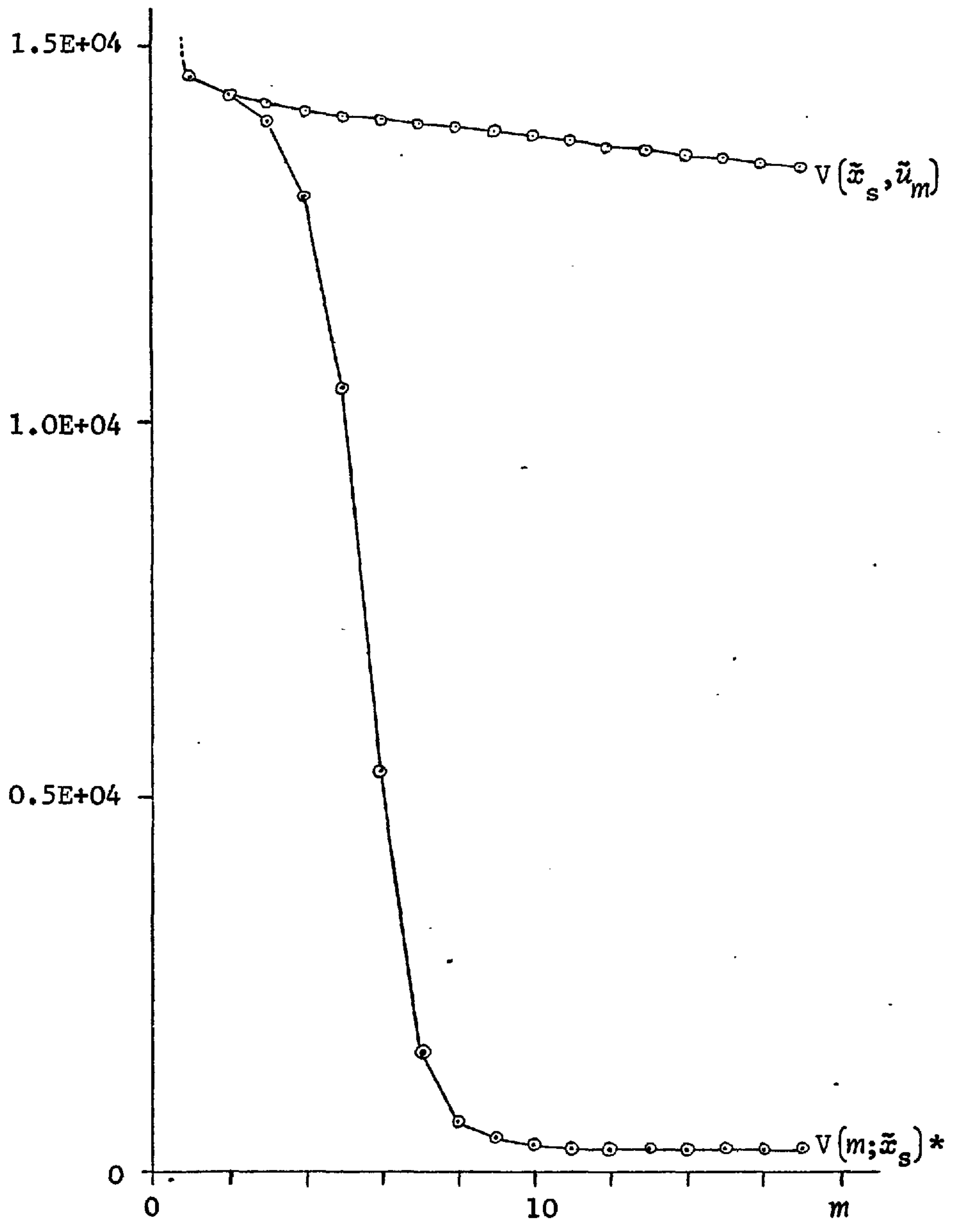

Fig. 3.1

3.8.2 A Heat-Exchanger

The optimisation problem considered here is defined for the heat-exchanger of 2.6 and has the same performance function as that of 2.6 but with $Q = 10$ and $R = 0.0025$. Trapezoidal integration was used with a step-length of 1 second. The initial control function considered was zero on $\{0,200\}$ and the initial condition considered was

$$\tilde{x}_s = (40 \quad 40 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0)^T.$$

Fourteen iterations of the gradient-decomposition based optimisation algorithm of 3.6 were performed. The predicted (using stage 6) of the algorithm) $\tilde{x}_s$-minimal performance index on $U(1,m)''$, $V(m;\tilde{x}_s)^*$, and the associated lower-bound $\tilde{V}(m+1;\tilde{x}_s)^*_m$ for the $\tilde{x}_s$-minimal performance index on the control space are plotted in Fig. 3.2 for $m = 9, \ldots, 14$.

The predicted (using the $U \mapsto G$ map matrix elements deduced by the gradient-decomposition based algorithm) $\tilde{x}_s$-optimal control function belonging to $U(1,14)''$ was applied to the heat-exchanger description. The resulting performance index and lower-bound for the $\tilde{x}_s$-minimal performance index on the control space were 0.45173336E+05 and 0.45171316E+05, respectively. These compare favourably with the values predicted using stage 6) of the gradient-decomposition based algorithm, which were 0.45173349E+05 and 0.45171340E+05 (respectively) - bearing in mind that an IBM 7090 computer operating in the single precision mode was used for the computations.

From the above results we see that the predicted $\tilde{x}_s$-optimal control function belonging to $U(1,14)''$ has an associated performance index value which is negligibly different from the $\tilde{x}_s$-minimal performance index on the control space considered, which was of dimension 100 because the control functions considered were constrained by the integration algorithm which was used to be constant over each of the 100 integration steps which were employed. Optimisation on a translated linear manifold containing $U(1,14)$" but of larger dimension would therefore be relatively unprofitable, performance-index wise. This is a result which is useful from the computational point of view, and which would not be available without our lower-bound results of 3.4.

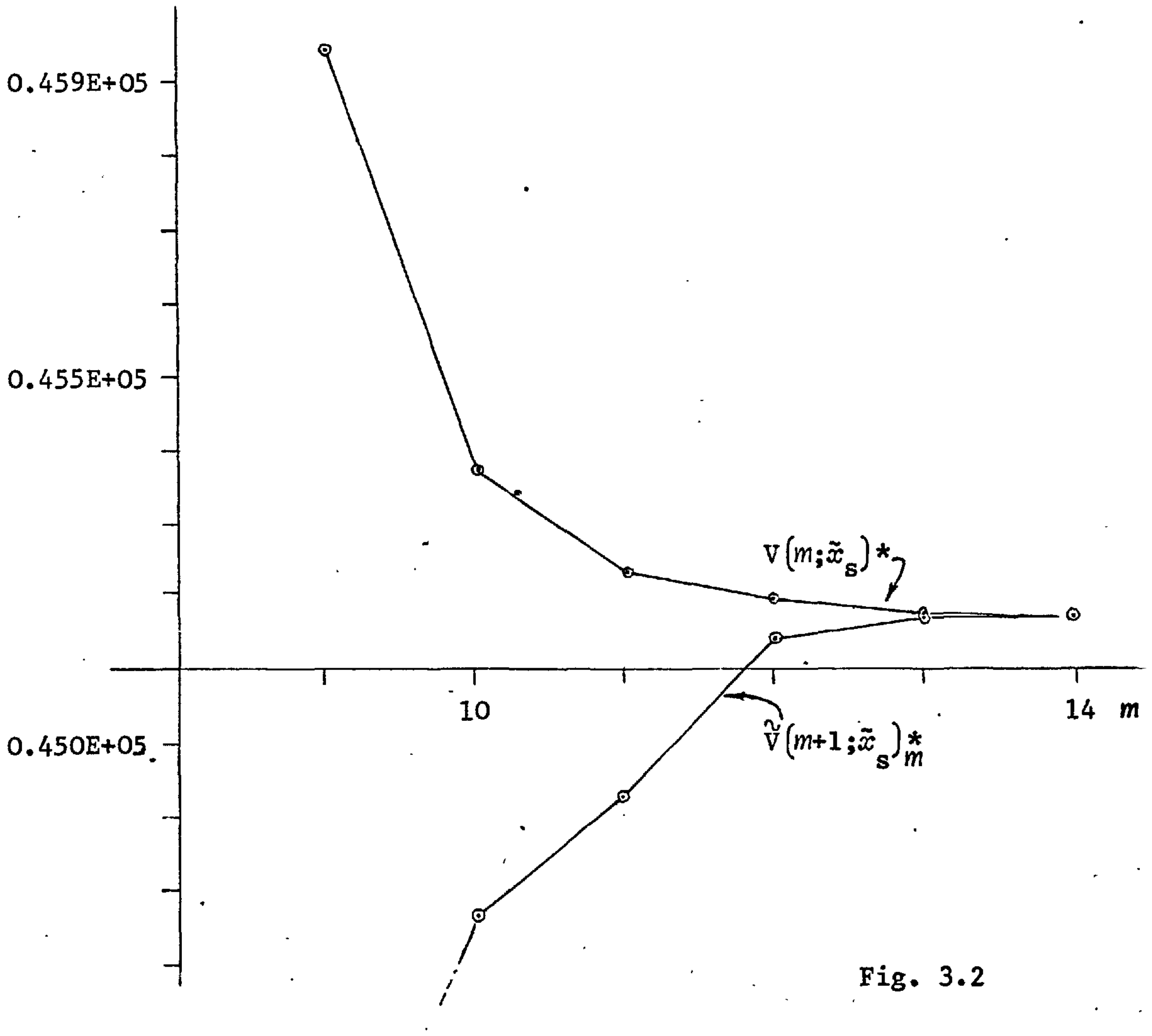

Fig. 3.2

3.9 Optimal Control Function Determination as a Function of Initial Conditions

The optimisation problem of 3.2 is considered when Assumption 3.4.1 holds for $x_s = \tilde{x}_s$, of 1.3. A simple procedure is stated for defining a translated linear manifold and determining the x_s-optimal control function belonging to it as a function of initial conditions $x_s \in X(q)$ such that a certain approximation condition holds for all $x_s \in \bar{X}(q)$.

Recall the definition, in 1.3, of the closed and bounded neighbourhood $\bar{X}(q)$ of $\tilde{x}_s$. From 3.3, the x_s-optimal control function belonging to $U(1,j)''$ is, for all initial conditions $x_s = \tilde{x}_s + X^q\delta x^q \in X(q)$:

$$u\left(j;\tilde{x}_s+X^q\delta x^q\right)* \;=\; u_1 - F_j T(1,j\to 1,j)^{-1}\left(g^j(\tilde{x}_s,u_1) + P(1,q\to 1,j)\delta x^q\right) \quad (3.63)$$

where $\quad g^j(\tilde{x}_s,u_1) \;=\; \int_T F_j(t)^T\left(\partial V(\tilde{x}_s,u_1)/\partial u(t)\right)dt$

when $T(1,j\to 1,j)$ is p.d., which it is when Assumption 3.4.1 holds. Expression (3.63) is an optimal control law which determines the x_s-optimal control function belonging to $U(1,j)''$ as a function of initial conditions $x_s = \tilde{x}_s + X^q\delta x^q \in X(q)$. Recall the definition of an $\varepsilon\left(x_s\right)$-approximation, in 3.4.2.

∇ *Definition 3.9.1* Control law (3.63) will be referred to as an $\varepsilon\left(\bar{X}(q)\right)$-approximation to the optimal control law which determines the optimal control function belonging to the control space U as a function of initial conditions if $u\left(j;\tilde{x}_s+X^q\delta x^q\right)*$ is an $\varepsilon\left(\tilde{x}_s+X^q\delta x^q\right)$-approximation to the $(\tilde{x}_s+X^q\delta x^q)$-optimal control function belonging to U for all initial

Δ conditions $x_s = \tilde{x}_s + X^q\delta x^q \in \bar{X}(q)$.

Definition 3.9.1 provides a useful means for characterising the effectiveness of control law (3.63). If an $\varepsilon\left(\bar{X}(q)\right)$-approximation is required for a specified ε, it is desirable to use the smallest integer j for which control law (3.63) is the desired $\varepsilon(\bar{X}(q))$-approximation, to reduce as far as possible the computational effort required to determine the control law. A simple algorithm is next outlined for determining a suitable small j, j(q), and the associated control law, although j(q) is not necessarily the smallest j for which control law (3.63) is an $\varepsilon(\bar{X}(q))$-approximation. Some of the quantities used in the algorithm are defined in Remark 3.9.1, which follows the statement of the algorithm.

1) Choose an initial control function $u_1 \in U$, where u_1 is a guess at the $\tilde{x}_s$-optimal control function belonging to U.

Set the initial condition to $\tilde{x}_s$. Define orthonormal basis-functions $\mathfrak{f}$ and choose the smallest integer j(0) such that the gradient function $(\partial V(j(0);\tilde{x}_s)^{*}/\partial u)$ (defined to be equal to $(\partial V(\tilde{x}_s,u_0^*)/\partial u)$, where u_0^* is the $\tilde{x}_s$-optimal control function belonging to $U(1,j(0))$") satisfies

$$\beta(j(0)) \leqq \delta(0), \tag{3.64}$$

where

$$\beta(j(0)) = \tfrac{1}{2}\sum_{i=1}^{m}\int_T\{<(h_i(t)),(\partial V(j(0);\tilde{x}_s)^{*}/\partial u(t))>^2/\lambda_i(t)\}dt. \tag{3.65}$$

The gradient-decomposition based optimisation algorithm of 3.6 can be simply modified to define, and $\tilde{x}_s$-optimise on, $U(1,m)$" as m is iteratively increased from one in unit steps until condition (3.64), with j(0) replaced by m, is satisfied. The required value of j(0) is then the first value of m which causes condition (3.64), with j(0) replaced by m, to be satisfied. Clearly $T(1,j(0)+1,j(0))^{-1}$ will be available when j(0) has been determined in this way. Denote the total number of orthonormal basis-functions $\mathfrak{f}$ defined at this stage by $n(0)$, and denote the $\tilde{x}_s$-optimal control function belonging to $U(1,j(0))$" by u_0^*. Go to 2).

2) Set $r = 1$. Go to 3).

3) Equate the control function to u_0^*. Perturb the initial condition from $\tilde{x}_s$ to $x_r = \tilde{x}_s + X_r|\delta\hat{x}_r^q|$, on the boundary of $\bar{X}(q)$. Calculate the resulting gradient function, $\left(\partial V(x_r,u_0^*)/\partial u\right)$. The gradient function change caused by the initial condition change $X_r|\delta\hat{x}_r^q|$ from $\tilde{x}_s$ when the control function remains unchanged at u_0^* is therefore

$$\delta g\left(|\delta\hat{x}_r^q|\right) = \left(\partial V(x_r,u_0^*)/\partial u\right) - \left(\partial V(j(0);\tilde{x}_s)^*/\partial u\right). \qquad (3.66)$$

Decompose the gradient function change $\delta g\left(|\delta\hat{x}_r^q|\right)$ as $F_{i(r)}\delta g_r^{i(r)}$, in exactly the same way as we would decompose a gradient function, where $i(r) = n(r-1) + 1$. If the last element of $\delta g_r^{i(r)} \in R^{i(r)}$ is zero, set $i(r) = n(r-1)$ since $\delta g\left(|\delta\hat{x}_r^q|\right)$ is then exactly characterised by components of basis-functions $f_1, .., f_{n(r-1)}$ alone, and no new orthonormal basis-function $f_{n(r-1)+1}$ has to be defined to enable $\delta g\left(|\delta\hat{x}_r^q|\right)$ to be exactly characterised by components of defined orthonormal basis-functions. The total number of orthonormal basis-functions defined at this stage is $i(r)$. Then:

$$\delta g\left(|\delta\hat{x}_r^q|\right) = F_{i(r)}\delta g_r^{i(r)}.$$

Therefore

$$\delta g\left(|\delta\hat{x}_r^q|\right) = F_k\delta g_r^k, \ \forall k \geq i(r), \qquad (3.67)$$

where

$$\delta g_r^k = \begin{pmatrix} \delta g_r^{i(r)} \\ O(k-i(r),1) \end{pmatrix} \in R^k, \ \forall k \geq i(r), \qquad (3.68)$$

no matter how $f_{i(r)+1}, .., f_k$ are defined (if $f_1, .., f_k$ are orthonormal).

On recalling the $X\to G$ map property of $P(1,q\to 1,k)$, of Comment 3.3.4, we see that column r of $P(1,q\to 1,k)$ is given by

$$P(r\to 1,k) = \delta g_r^k/|\delta\hat{x}_r^q| = \begin{pmatrix} \delta g_r^{i(r)}/|\delta\hat{x}_r^q| \\ O(k-i(r),1) \end{pmatrix}, \ \forall k \geq i(r). \qquad (3.69)$$

For any integer $j(r) > 0$, the $j(r)$-vector $\delta g_r^{j(r)}$ of the components of basis-functions $\delta_1, .., \delta_{j(r)}$ (which have not all been defined yet if $j(r) > i(r)$, since only basis-functions $\delta_1, .., \delta_{i(r)}$ have been defined at this stage of the algorithm) present in the gradient function change $\delta g(|\delta\hat{x}_r^q|)$ is equal to $P(r+1,j(r))|\delta\hat{x}_r^q|$, where $P(r+1,j(r))$ is the $j(r)$-vector consisting of the first $j(r)$ elements of $P(r+1,k)$ of (3.69) - for any $k \geqq \max(i(r),j(r))$. On using the $U \to G$ map property of $T(1,j(r)+1,j(r))$, it can be seen that the components of basis-functions $\delta_1, .., \delta_{j(r)}$ present in the gradient function can be changed by $-\delta g_r^{j(r)}$ by making a change of $-T(1,j(r)+1,j(r))^{-1}\delta g_r^{j(r)}$ in the components of basis-functions $\delta_1, .., \delta_{j(r)}$ present in the control function. Thus the components of basis-functions $\delta_1, .., \delta_{j(r)}$ present in the gradient function change $\delta g(|\delta\hat{x}_r^q|)$ from $(\partial V(j(0);\tilde{x}_s)^*/\partial u)$ which is caused by an initial condition change of $X_r|\delta\hat{x}_r^q|$ from $\tilde{x}_s$ can be reduced to, or maintained at, zero by changing the control function from u_0^* to

$$u_r^* = u_0^* - F_{j(r)}\big(T(1,j(r)+1,j(r))\big)^{-1}P(r+1,j(r))|\delta\hat{x}_r^q| \tag{3.70}$$

when the initial condition is changed from $\tilde{x}_s$ to x_r.

The gradient function change from $(\partial V(j(0);\tilde{x}_s)^*/\partial u)$ which is caused by the initial condition change from $\tilde{x}_s$ to x_r when the control function is changed from u_0^* to u_r^* is then

$$\delta\hat{g}\big(j(r);|\delta\hat{x}_r^q|\big) = \big(\partial V(x_r,u_r^*)/\partial u\big) - \big(\partial V(j(0);\tilde{x}_s)^*/\partial u\big). \tag{3.71}$$

Choose $j(r)$ so that

$$\beta\big(j(r)\big) \leqq \delta(r) \tag{3.72}$$

where

$$\beta\big(j(r)\big) = \tfrac{1}{2}\sum_{i=1}^{m}\int_T\{<(h_i(t)),(\delta\hat{g}(j(r);|\delta\hat{x}_r^q|)(t))>^2/\lambda_i(t)\}dt, \tag{3.73}$$

and where $\delta\hat{g}(j(r);|\delta\hat{x}^q_r|)(t)$ is the value of $\delta\hat{g}(j(r);|\delta\hat{x}^q_r|)$ at time t.

Since $T(1,j(r)+1,j(r))^{-1}$ is available at the start of stage 3), we can easily check whether $j(r) = j(r-1)$ satisfies condition (3.72). If it does, set $j(r) = j(r-1)$ and go to 4). If it does not, an algorithm similar to that of 3.6 can be simply constructed to choose the smallest integer $j(r)$ greater than $j(r-1)$ such that condition (3.72) is satisfied (and to define further basis-functions f if necessary, i.e. if $j(r) = i(r)$ does not cause condition (3.72) to be satisfied). Note, however, that the $U \to G$ map matrices T will not necessarily be tridiagonal in this case since basis-functions f may have to be defined by the current algorithm to enable gradient function changes which have been caused by initial condition changes to be exactly characterised in terms of components of defined basis-functions, which breaks the chain of reasoning which lead us to deduce that T is tridiagonal for the basis-functions defined by the algorithm of 3.6. The procedure used for calculating T in the algorithm of 3.6 is still valid, however, since it does not depend on T being tridiagonal. When a suitable $j(r)$ has been found in this way, $T(1,j(r)+1,j(r))^{-1}$ will be available. Denote the total number of orthonormal basis-functions f which have been defined at this stage of the algorithm by $n(r)$. Go to 4).

4) If the initial condition changes which have been made so far do not span $X(q)$, i.e. if $r \neq q$, set $r = r + 1$ and go to 3). Otherwise, if $r = q$, stop.

This concludes the statement of the algorithm.

∇ *Remark 3.9.1* Consider the optimisation problem of 3.2 when Assumption 3.4.1 holds for $x_s = \tilde{x}_s$. Since $F_{uu}(t)$ is then symmetric and p.d. for all $t \in T$, it can be written as

$$F_{uu}(t) = \sum_{i=1}^{m} h_i(t)\lambda_i(t)h_i(t)^T \tag{3.74}$$

where, for all $t \in T$,

$h_i(t)$, $\forall i \in I(m)$, are the orthonormal eigenvectors of $F_{uu}(t)$,

$\lambda_i(t)$, $\forall i \in I(m)$, are the associated eigenvalues of $F_{uu}(t)$ (all > 0).

Suppose that in the above algorithm

$$\delta(0) = \varepsilon/(q+1)^2$$

$$\delta(r) = \left(\varepsilon + \alpha(r)^2 - 2\alpha(r)|\sqrt{\varepsilon}|\right)/(q-r+1)^2, \ \forall r \in I(q)$$

where $\alpha(r) = \sum_{a=1}^{r-1} |\sqrt{\beta(j(a))}|$.

Then control law (3.63) with $j = j(q)$ is an $\varepsilon(\bar{X}(q))$-approximation to the optimal control law which determines optimal control functions belonging to the control space U as a function of initial conditions Δ $x_s \in X(q)$.

Before proving Remark 3.9.1 we make the following comments.

∇ *Comment 3.9.1* Using Remark 3.9.1 with the above algorithm enables the j (actually $j(q)$) to be determined in a simple way for which control law (3.63) is an $\varepsilon(\bar{X}(q))$-approximation for any desired $\varepsilon > 0$. The $X{\to}G$ map matrix $P(1,q{\to}1,j(q))$ and the inverse $U{\to}G$ map matrix $T(1,j(q){\to}1,j(q))^{-1}$ will be available when $j(q)$ has been determined, so all the terms needed to implement control law (3.63) with $j = j(q)$ are available once $j(q)$ has been determined in the above way. Note that

$$P(1,q{\to}1,j(q)) = \left(P(1{\to}1,j(q)) \ \ldots \ P(q{\to}1,j(q))\right)$$

where $P(r{\to}1,j(q))$, $\forall r \in I(q)$, is the $j(q)$-vector consisting of the first

Δ j(q) elements of $P(r+1,k)$ of (3.69) for any $k \geq \max(j(q),i(r))$.

∇ *Comment 3.9.2* Remark 3.9.1 is especially easy to implement if, as is common, F is chosen so that F_{uu} is diagonal, since then $h_i(t)$ is that m-vector which has element i equal to unity and all other elements

Δ equal to zero, $\forall t \in T$.

∇ *Proof of Remark 3.9.1* For the initial condition $x_s = \tilde{x}_s + X^q\delta x^q$, consider the control function

$$u(\tilde{x}_s+X^q\delta x^q)'' = u_1 - F_{j(0)}\big(T(1,j(0)+1,j(0))\big)^{-1}g^{j(0)}(\tilde{x}_s,u_1) - \sum_{r=1}^{q} F_{j(r)}\big(T(1,j(r)+1,j(r))\big)^{-1}P(r+1,j(r))\delta x^q_r$$

where $g^{j(0)}(\tilde{x}_s,u_1) = \int_T \big(F_{j(0)}(t)\big)^T\big(\partial V(\tilde{x}_s,u_1)/\partial u(t)\big)dt$,

$\delta x^q = \big(\delta x^q_1 \;\ldots\; \delta x^q_q\big)^T$,

$P(r+1,j(r))$ is the $j(r)$-vector consisting of the first $j(r)$ elements of $P(r+1,j(q))$ of Comment 3.9.1.

On using (3.70) and the fact that $u_1 - F_{j(0)}\big(T(1,j(0)+1,j(0))\big)^{-1}\times g^{j(0)}(\tilde{x}_s,u_1)$ is the $\tilde{x}_s$-optimal control function belonging to $U(1,j(0))''$, it can be seen that

$$u(\tilde{x}_s+X^q\delta x^q)'' = u^*_0 + \sum_{r=1}^{q}\{u^*_r - u^*_0\}\delta x^q_r/|\delta\hat{x}^q_r|. \tag{3.75}$$

We can consider $u(\tilde{x}_s+X^q\delta x^q)''$ to be an $\varepsilon(\tilde{x}_s+X^q\delta x^q)$-approximation to the $(\tilde{x}_s+X^q\delta x^q)$-optimal control function belonging to U if

$$\big|\, V\big(\tilde{x}_s+X^q\delta x^q, u(\tilde{x}_s+X^q\delta x^q)''\big) - V\big(\tilde{x}_s+X^q\delta x^q\big)^* \,\big| \leq \varepsilon \tag{3.76}$$

where $V(\tilde{x}_s+X^q\delta x^q)^*$ is the $(\tilde{x}_s+X^q\delta x^q)$-minimal performance index on U, in exactly the same way that we consider the $(\tilde{x}_s+X^q\delta x^q)$-optimal control function belonging to $U(1,j(q))''$ to be an $\varepsilon(\tilde{x}_s+X^q\delta x^q)$-approximation if

$$\big|\, V\big(j(q);\tilde{x}_s+X^q\delta x^q\big)^* - V\big(\tilde{x}_s+X^q\delta x^q\big)^* \,\big| \leq \varepsilon \tag{3.77}$$

where $V(j(q);\tilde{x}_s+X^q\delta x^q)^*$ is the $(\tilde{x}_s+X^q\delta x^q)$-minimal performance index on $U(1,j(q))''$.

Recall from 3.4.2 that the minimal ε for which the $(\tilde{x}_s+X^q\delta x^q)$-optimal control function belonging to $U(1,j(q))''$ is an $\varepsilon(\tilde{x}_s+X^q\delta x^q)$-approximation has as an upper-bound

$$\tilde{\varepsilon}(j(q);\tilde{x}_s+X^q\delta x^q)^* \;=\; \tfrac{1}{2}\int_T\langle(g(t)),(F_{uu}(t))^{-1}(g(t))\rangle dt$$

where g is equal to the gradient function $(\partial V(\tilde{x}_s+X^q\delta x^q,u)/\partial u)$ when u is the $(\tilde{x}_s+X^q\delta x^q)$-optimal control function belonging to $U(1,j(q))''$. In an exactly similar way, it can be shown that the minimal ε for which $u(\tilde{x}_s+X^q\delta x^q)''$ is an $\varepsilon(\tilde{x}_s+X^q\delta x^q)$-approximation to the $(\tilde{x}_s+X^q\delta x^q)$-optimal control function belonging to U has as an upper-bound

$$\tilde{\varepsilon}(\tilde{x}_s+X^q\delta x^q)'' \;=\; \tfrac{1}{2}\int_T\langle(g''(t)),(F_{uu}(t))^{-1}(g''(t))\rangle dt \tag{3.78}$$

where $$g'' \;=\; (\partial V(\tilde{x}_s+X^q\delta x^q,u(\tilde{x}_s+X^q\delta x^q)'')/\partial u). \tag{3.79}$$

From the structure of the algorithm considered, $j(r) \leqq j(q)$, $\forall r \in I(0,q)$. Therefore

$$V(j(q);\tilde{x}_s+X^q\delta x^q)^* \;\leqq\; V(\tilde{x}_s+X^q\delta x^q,u(\tilde{x}_s+X^q\delta x^q)''), \tag{3.80}$$

since (a) $u(\tilde{x}_s+X^q\delta x^q)''$ is the $(\tilde{x}_s+X^q\delta x^q)$-optimal control function belonging to $U(1,j(q))''$ if $j(r) = j(q)$ for all $r \in I(0,q-1)$, and

(b) $u(\tilde{x}_s+X^q\delta x^q)''$ is a potentially 'less-optimal' control function than the $(\tilde{x}_s+X^q\delta x^q)$-optimal control function belonging to $U(1,j(q))''$ if $j(r) < j(q)$ for any $r \in I(0,q-1)$.

Thus to prove that control law (3.63) with $j = j(q)$ is an $\varepsilon(\bar{X}(q))$-approximation to the optimal control law for U we need only show that $u(\tilde{x}_s+X^q\delta x^q)''$ is an $\varepsilon(\tilde{x}_s+X^q\delta x^q)$-approximation to the $(\tilde{x}_s+X^q\delta x^q)$-optimal control function belonging to U for all initial conditions $x_s = \tilde{x}_s + X^q\delta x^q \in \bar{X}(q)$. This arises because (3.80) ensures that (3.77) is satisfied for a given ε if (3.76) is satisfied for the same ε.

We therefore prove Remark 3.9.1 by showing that the choice of $\delta(h)$, $\forall h \in I(0,q)$, used there ensures that $\mathcal{E}(\tilde{x}_s+X^q\delta x^q)" \leqq \varepsilon$ for all initial conditions $\tilde{x}_s + X^q\delta x^q \in \bar{X}(q)$, which ensures that $u(\tilde{x}_s+X^q\delta x^q)"$ is an $\varepsilon(\tilde{x}_s+X^q\delta x^q)$-approximation for all $\tilde{x}_s+X^q\delta x^q \in \bar{X}(q)$.

Because the gradient function for the optimisation problem of 3.2 varies linearly with the initial condition x_s and with the control function, and because $u(\tilde{x}_s+X^q\delta x^q)"$ varies linearly with δx^q, it can be shown that

$$\left(\partial V(\tilde{x}_s+X^q\delta x^q, u(\tilde{x}_s+X^q\delta x^q)")/\partial u\right) = g_0 + \sum_{h=1}^{q} g_h \delta x_h^q / |\delta\hat{x}_h^q| \tag{3.81}$$

for some time-functions $g_0, \ldots, g_q$.

We can in fact identify the functions $g_0, \ldots, g_q$. Consider first the case when $\delta x^q = 0$. Then we see from (3.75) that $u(\tilde{x}_s+X^q\delta x^q)" = u_0^*$, the $\tilde{x}_s$-optimal control function belonging to $U(1,j(0))"$. Hence $\left(\partial V(\tilde{x}_s, u(\tilde{x}_s)")/\partial u\right) = \left(\partial V(\tilde{x}_s, u_0^*)/\partial u\right) = \left(\partial V(j(0);\tilde{x}_s)^*/\partial u\right)$ and, from (3.81):

$$g_0 = \left(\partial V(j(0);\tilde{x}_s)^*/\partial u\right). \tag{3.82}$$

Next consider the case when $\delta x^q = \left(0(1,h-1)\ |\delta\hat{x}_h^q|\ 0(1,q-h)\right)^T$, $h \in I(q)$. Then, from (3.75), $u(\tilde{x}_s+X^q\delta x^q)" = u_h^*$ and, from (3.71):

$$\left(\partial V(\tilde{x}_s+X^q\delta x^q, u(\tilde{x}_s+X^q\delta x^q)")/\partial u\right) = \left(\partial V(x_h, u_h^*)/\partial u\right)$$
$$= \left(\partial V(j(0);\tilde{x}_s)^*/\partial u\right) + \delta\hat{g}\left(j(h);|\delta\hat{x}_h^q|\right).$$

On comparing this with (3.81), it can be seen that

$$g_h = \delta\hat{g}\left(j(h);|\delta\hat{x}_h^q|\right). \tag{3.83}$$

On using the spectral representation of F_{uu} of (3.74) in (3.78), we see that $\mathcal{E}(\tilde{x}_s+X^q\delta x^q)"$

$$= \tfrac{1}{2}\sum_{i=1}^{m}\int_T \{<(h_i(t)), \left(\partial V(\tilde{x}_s+X^q\delta x^q, u(\tilde{x}_s+X^q\delta x^q)")/\partial u(t)\right)>^2/\lambda_i(t)\}dt.$$

Using (3.81) in the above yields:

$$\tilde{\varepsilon}(\tilde{x}_s+X^q\delta x^q)''$$

$$\leqq \tfrac{1}{2}\sum_{a=0}^{q}\sum_{b=0}^{q}\sum_{i=1}^{m}\left|\int_T\{<(g_a(t)),(h_i(t))><(h_i(t)),(g_b(t))>/\lambda_i(t)\}dt\right|,$$

$$\forall\tilde{x}_s+X^q\delta x^q \in \bar{X}(q), \tag{3.84}$$

since $\delta x^q_\hbar \leqq |\delta\hat{x}^q_\hbar|$, $\forall\hbar \in I(q)$, if $\tilde{x}_s + X^q\delta x^q \in \bar{X}(q)$.

On using Hölder's inequality for integrals, (3.84) yields

$$\tilde{\varepsilon}(\tilde{x}_s+X^q\delta x^q)'' \leqq \sum_{a=0}^{q}\sum_{b=0}^{q}\sum_{i=0}^{m}|\surd\beta_i(j(a))|\ \ |\surd\beta_i(j(b))|,$$

$$\forall\tilde{x}_s+X^q\delta x^q \in \bar{X}(q), \tag{3.85}$$

where $\beta_i(j(a)) = \tfrac{1}{2}\int_T\{<(h_i(t)),(g_a(t))>^2/\lambda_i(t)\}dt$, $\forall a \in I(0,q)$. (3.86)

Using the Cauchy-Schwarz inequality with (3.85) yields

$$\tilde{\varepsilon}(\tilde{x}_s+X^q\delta x^q)'' \leqq \sum_{a=0}^{q}\sum_{b=0}^{q}|\surd\beta(j(a))|\ \ |\surd\beta(j(b))|,$$

$$\forall\tilde{x}_s+X^q\delta x^q \in \bar{X}(q), \tag{3.87}$$

where $\beta(j(a)) = \sum_{i=1}^{m}\beta_i(j(a))$, $\forall a \in I(0,q)$. (3.88)

By using (3.82), (3.83) and (3.86) with (3.88), we see that $\beta(j(a))$ of (3.87) is defined as in the algorithm (i.e., $\beta(j(0))$ of (3.87) is equal to $\beta(j(0))$ of (3.65) and $\beta(j(\hbar))$ of (3.87) is equal to $\beta(j(\hbar))$ of (3.73), $\forall\hbar \in I(q)$).

Define $\alpha(\hbar)$ as follows:

$$\alpha(\hbar) = 0 \text{ if } \hbar = 0$$
$$= \sum_{a=0}^{\hbar-1}|\surd\beta(j(a))| \text{ if } \hbar \in I(q+1). \tag{3.89}$$

Then:

$$\alpha(q+1)^2 \geqq \alpha(q)^2 \geqq \dots \geqq \alpha(1)^2 \geqq \alpha(0)^2. \tag{3.90}$$

It is convenient to next state and prove two simple lemmas.

∇ *Lemma 3.9.1* Suppose that for some $\hbar \in I(0,q)$, $\alpha(\hbar)^2 < \varepsilon$. Then $\alpha(q+1)^2 \leqq \varepsilon$ if $j(a)$ is chosen so that $\beta(j(a)) \leqq \delta(\hbar)$, $\forall a \in I(\hbar,q)$, Δ where $\delta(h) = (\alpha(h)^2 + \varepsilon - 2\alpha(h)|\sqrt{\varepsilon}|)/(q-h+1)^2$.

∇ *Proof of Lemma 3.9.1* We see from (3.89) that

$$\alpha(q+1)^2 = \alpha(h)^2 + 2\alpha(h)\sum_{a=h}^{q}|\sqrt{\beta(j(a))}| + (\sum_{a=h}^{q}|\sqrt{\beta(j(a))}|)^2.$$

Suppose $j(a)$ is chosen so that

$$\beta(j(a)) \leqq \delta(h),\ \forall a \in I(h,q).$$

Then $\alpha(q+1)^2 \leqq \alpha(h)^2 + 2\alpha(h)(q-h+1)|\sqrt{\delta(h)}| + (q-h+1)^2\delta(h)$

and $\alpha(q+1)^2 \leqq \varepsilon$ if $|\sqrt{\delta(h)}| = (|\sqrt{\varepsilon}| - \alpha(h))/(q-h+1)$,

Δ which completes the proof of Lemma 3.9.1.

∇ *Lemma 3.9.2* Suppose that for some $h \in I(0,q-1)$ such that $\alpha(h)^2 < \varepsilon$, $j(h)$ is chosen so that $\beta(j(h)) < \delta(h)$ where $\delta(h) = (\alpha(h)^2 + \varepsilon - 2\alpha(h)|\sqrt{\varepsilon}|)/(q-h+1)^2$. Then $\delta(h+1) = (\alpha(h+1)^2 + \varepsilon - 2\alpha(h+1)|\sqrt{\varepsilon}|)/(q-h)^2 > \delta(h)$. Also

Δ if $\beta(j(h)) = \delta(h)$, then $\delta(h+1) = \delta(h)$.

Proof of Lemma 3.9.2 Since $\alpha(h+1) = \alpha(h) + |\sqrt{\beta(j(h))}|$:

$$\delta(h+1) = \{(q-h+1)^2\delta(h) + 2(\alpha(h) - |\sqrt{\varepsilon}|)|\sqrt{\beta(j(h))}| + \beta(j(h))\}/(q-h)^2$$

$$= \delta(h) + \{(2(q-h)+1)\delta(h) + 2(\alpha(h) - |\sqrt{\varepsilon}|)|\sqrt{\beta(j(h))}| + \beta(j(h))\}/(q-h)^2.$$

Now if $\beta(j(h)) < \delta(h)$, then $|\sqrt{\beta(j(h))}| = |\sqrt{\delta(h)}| - \kappa$ for some $\kappa > 0$. On using the facts that $(\alpha(h) - |\sqrt{\varepsilon}|)/(q-h+1) = -|\sqrt{\delta(h)}|$ and that $q-h > 0$ if $h \in I(0,q)$, it can be seen from the above that

$$\delta(h+1) = \delta(h) + \{\kappa^2 + 2(q-h)\kappa|\sqrt{\delta(h)}|\}/(q-h)^2 > \delta(h).$$

Also, if $\beta(j(h)) = \delta(h)$, then $\kappa = 0$ and $\delta(h+1) = \delta(h)$, which

Δ concludes the proof of Lemma 3.9.2.

We see from (3.87) and (3.89) that

$$\mathscr{E}(\tilde{x}_s+X^q\delta x^q)'' \leqq \alpha(q+1)^2,\ \forall \tilde{x}_s+X^q\delta x^q \in \bar{X}(q), \tag{3.91}$$

so that $\mathscr{E}(\tilde{x}_s+X^q\delta x^q)'' \leqq \varepsilon$ for all $\tilde{x}_s + X^q\delta x^q \in \bar{X}(q)$ if $j(a)$, $\forall a \in I(0,q)$, is chosen so that

$$\alpha(q+1)^2 \leqq \varepsilon. \tag{3.92}$$

From Lemma 3.9.1 with $\hbar = 0$ and, from (3.89), $\alpha(\hbar) = 0$, condition (3.92) is satisfied if j(a) is chosen so that

$$\beta\bigl(j(a)\bigr) \leqq \delta\bigl(0\bigr) = \varepsilon/(q+1)^2, \ \forall a \in I(0,q). \tag{3.93}$$

Now $\delta\bigl(0\bigr)$ is defined in Remark 3.9.1 in the same way as in (3.93), so the algorithm considered chooses j(0) so that condition (3.93) is satisfied for a = 0, which also ensures that $\alpha(1)^2 < \varepsilon$. Due to the integer nature of j(0), it is unlikely that the number j(0) chosen by the algorithm will be such that $\beta\bigl(j(0)\bigr) = \delta\bigl(0\bigr)$. If $\beta\bigl(j(0)\bigr)$ is actually less than $\delta\bigl(0\bigr)$, which is likely, a less stringent condition on j(a), $\forall a \in I(q)$, than $\beta\bigl(j(a)\bigr) \leqq \delta\bigl(0\bigr)$, need be imposed to ensure that $\alpha(q+1)^2 \leqq \varepsilon$. We see from Lemma 3.9.1 that to ensure that $\alpha(q+1)^2 \leqq \varepsilon$ it is sufficient, when j(0) has been chosen as above, that j(a), $\forall a \in I(q)$, be chosen so that

$$\beta\bigl(j(a)\bigr) \leqq \delta\bigl(1\bigr) = \bigl(\alpha(1)^2 + \varepsilon - 2\alpha(1)|\sqrt{\varepsilon}|\bigr)/q^2. \tag{3.94}$$

This is a less stringent condition on j(a), $\forall a \in I(q)$, than condition (3.93) if $\beta\bigl(j(0)\bigr) < \delta\bigl(0\bigr)$ since then, from Lemma 3.9.2, $\delta\bigl(1\bigr) > \delta\bigl(0\bigr)$. If, however, $\beta\bigl(j(0)\bigr) = \delta\bigl(0\bigr)$, the conditions are equivalent since then, from Lemma 3.9.2, $\delta\bigl(1\bigr) = \delta\bigl(0\bigr)$.

By proceeding in the above way it may be seen that the conditions of Remark 3.9.1 are such that $\alpha(q+1)^2 \leqq \varepsilon$, i.e. are such that $\tilde{\mathcal{E}}(\tilde{x}_s + X^q\delta x^q)" \leqq \varepsilon$ for all initial conditions $\tilde{x}_s + X^q\delta x^q \in \overline{X}(q)$, which ensures that control law (3.63) with j = j(q) is the desired $\varepsilon\bigl(\overline{X}(q)\bigr)$-approximation to the optimal control law which determines optimal control functions belonging to the control space $\mathcal{U}$ as a function of initial conditions. Δ

3.10 Computed Results

The theoretical results and discussion of 3.9 do not need computational verification. In this section, however, computed results are presented which demonstrate the usefulness for control function re-optimisation following an initial condition change of $U \to G$ map matrix data deduced from optimisation iterations performed before the considered initial condition change. We use the theoretical result that the x_s-optimal control function belonging to $U(1,m)$" is equal to $u_1 - F_m\big(T(1,m\to 1,m)\big)^{-1}\int_T \big(F_m(t)\big)^T\big(\partial V(x_s,u_1)/\partial u(t)\big)dt$. Also demonstrated is the usefulness of the results of 3.4 for determining when optimisation on a larger (translated) linear manifold than that on which optimisation has already been obtained would not be profitable, performance-index wise. We do not explain in detail the operations which were carried out since they are fairly obvious and such a description would be tedious and would not enable the results which are presented to be appreciated better. As before, orthonormal basis-functions $\mathfrak{f}$ were defined so that each calculated gradient function could be exactly characterised by components of the defined basis-functions.

The optimisation problem considered is that of 3.8.2 with $Q = 10$ and $R = 0.5$. The initial control function u_1 considered was 20 on $\{0,100\}$ and 100 on $(100,200\}$. The nominal initial condition considered was $x_s = \big(40\ 40\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\big)^T$. Computed results are presented to eight figures since an eight significant figure computer (IBM 7090) was used.

The performance index for the initial control function and the nominal initial condition $\tilde{x}_s$ was calculated and is denoted here by $V(0;\tilde{x}_s)^*$. It is shown in Table 3.10.1. Four iterations of the gradient decomposition based optimisation algorithm of 3.6 were then applied. The resulting $\tilde{x}_s$-minimal performance index on $U(1,m)''$, $V(m;\tilde{x}_s)^*$, and the associated upper-bound $\tilde{\varepsilon}(m;\tilde{x}_s)^*$ for the remaining performance index improvement possible, relative to $V(m;\tilde{x}_s)^*$, on optimising on the control space are shown in the following table for $m = 1, .., 4$. Further iterations were not considered because of the small size of the remaining performance index decrease possible after optimising on $U(1,4)''$ (bounded from above by $\tilde{\varepsilon}(4;\tilde{x}_s)^*$).

<u>Table 3.10.1</u>

m	$V(m;\tilde{x}_s)^*$	$\tilde{\varepsilon}(m;\tilde{x}_s)^*$
0	0.19737759E+07	
1	0.10882303E+07	0.02559285E+07
2	0.10031260E+07	0.00212385E+07
3	0.09867041E+07	0.00001809E+07
4	0.09865632E+07	0.00000051E+07

The control function was then set to the $\tilde{x}_s$-optimal control function belonging to $U(1,4)''$ and the initial condition changed to $x_s^1 = (40\ \ 40\ -20\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0)^T$. The resulting performance index value is denoted here by $V(0;x_s^1)^*$. It is given in table 3.10.2. The x_s^1-optimal control function belonging to $U(1,4)''$ was then determined using $T(1,4\to1,4)^{-1}$, which had already been established by the four iterations of the gradient-decomposition based optimisation algorithm of 3.6 for the nominal initial condition $\tilde{x}_s$. The resulting x_s^1-minimal performance index on $U(1,4)''$, $V(4;x_s^1)^*$, and upper-bound $\tilde{\varepsilon}(4;x_s^1)^*$ for the remaining performance index decrease possible, relative to $V(4;x_s^1)^*$, on optimising on the control space were evaluated. Further iterations of the gradient-decomposition type were used to defined further basis-functions δ and to optimise on $U(1,5)''$, .., $U(1,7)''$, after which $T(1,7\to1,7)^{-1}$ was available. The results are listed in the following table. No further iterations were used because of the negligible size of the upper-bound $\tilde{\varepsilon}(7;x_s^1)^*$ for the remaining performance index improvement possible after optimising on $U(1,7)''$.

Table 3.10.2

m	$V(m;x_s^1)^*$	$\tilde{\varepsilon}(m;x_s^1)^*$
0	0.13697292E+07	
4	0.13137437E+07	0.00038777E+07
5	0.13110189E+07	0.00003546E+07
6	0.13107011E+07	0.00000084E+07
7	0.13106940E+07	0.00000002E+07

Note that most of the performance index decrease obtained after the initial condition change from $\tilde{x}_s$ to x_s^1 was achieved by optimisation on $U(1,4)''$ using the $U\to G$ map matrix inverse $T(1,4\to1,4)^{-1}$ which was determined before the initial condition change (optimisation on $U(1,4)''$ causing a performance index decrease of 0.00559855E+07, from $V(0;x_s^1)^*$ to $V(4;x_s^1)^*$). The further performance index decrease achieved by the three following iterations (which caused a decrease of 0.00030497E+07, from $V(4;x_s^1)^*$ to $V(7;x_s^1)^*$) was only about 6% of that achieved by optimisation on $U(1,4)''$. The usefulness of the $U\to G$ map matrix data determined before the initial condition change from $\tilde{x}_s$ to x_s^1 for optimising the control function for the initial condition x_s^1 is clear from the above results.

The control function was then set to the x_s^1-optimal control function belonging to $U(1,7)''$ and the initial condition changed to $x_s^2 = (40 \quad 60 \; -20 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0)^T$. The resulting performance index value, denoted by $V(0;x_s^2)*$, is shown in Table 3.10.3, below. Optimisation on $U(1,7)''$ was then achieved using the $U\to G$ map matrix inverse $T(1,7\to 1,7)^{-1}$ which was determined before the initial condition change to x_s^2. Two iterations of the gradient-decomposition type were then applied to enable optimisation to be achieved on $U(1,8)''$ and $U(1,9)''$, after which $T(1,9\to 1,9)^{-1}$ was available. The results are shown in Table 3.10.3, below. Optimisation on a larger (translated) linear manifold than $U(1,9)''$ was not considered because of the small size of $\tilde{\varepsilon}(9;x_s^2)*$.

Table 3.10.3

m	$V(m;x_s^2)*$	$\tilde{\varepsilon}(m;x_s^2)*$
0	0.20409221E+07	
7	0.17670290E+07	0.00011335E+07
8	0.17662468E+07	0.00002357E+07
9	0.17660436E+07	0.00000203E+07

Note that most of the performance index decrease achieved was obtained by optimisation on $U(1,7)''$ using the $U\to G$ map matrix inverse $T(1,9\to 1,9)^{-1}$, which had been established before the initial condition change to x_s^2, since the performance index decrease obtained by optimising on $U(1,7)''$ was 0.02738931E+07 (from $V(0;x_s^2)*$ to $V(7;x_s^2)*$) while the further performance index decrease obtained by the two further iterations of the gradient-decomposition type was only 0.00009854E+07 (from $V(7;x_s^2)*$ to $V(9;x_s^2)*$). The usefulness of the $U\to G$ map matrix data determined before the initial condition change to x_s^2 for optimising the control function for the initial condition x_s^2 is clear from the above results.

The control function was then set to the x_s^2-optimal control function belonging to $U(1,9)$" and the initial condition changed to $x_s^3 = (20 \;\; 40 \; -20 \; -10 \; -20 \; -30 \; -20 \; -20)^T$. The resulting performance index value, denoted by $V(0;x_s^3)*$, was calculated and is shown in Table 3.10.4, below. The x_s^3-optimal control function belonging to $U(1,9)$" was determined using $T(1,9\rightarrow 1,9)^{-1}$, which was already available from the optimisation iterations carried out for the previous initial conditions. The resulting x_s^3-minimal performance index on $U(1,9)$", $V(9;x_s^3)*$, was calculated as well as the associated upper-bound $\tilde{\varepsilon}(9;x_s^3)*$ for the remaining performance index decrease, relative to $V(9;x_s^3)*$, which could be obtained by optimising on the control space. Further iterations of the gradient-decomposition type were used to optimise on $U(1,13)$". Optimisation on a larger (translated) linear manifold than $U(1,13)$" was not considered because of the small size of $\tilde{\varepsilon}(13;x_s^3)*$. The results are shown in the following table.

Table 3.10.4

m	$V(m;x_s^3)*$	$\tilde{\varepsilon}(m;x_s^3)*$
0	0.21296120E+07	
9	0.11664665E+07	0.00003334E+07
13	0.11661538E+07	0.00000003E+07

Note that, once again, most of the performance index decrease achieved was obtained by optimising using $U\rightarrow G$ map matrix data determined before the initial condition change to x_s^3. The performance index decrease obtained by optimising on $U(1,9)$" using the $U\rightarrow G$ map matrix inverse $T(1,9\rightarrow 1,9)^{-1}$, which had been established by the previous iterations, was in fact 0.09631455E+07 (from $V(0;x_s^3)^*$ to $V(9;x_s^3)^*$) while the further performance index decrease obtained by the further iterations of the gradient-decomposition type was only 0.00003127E+07 (from $V(9;x_s^3)^*$ to $V(13;x_s^3)^*$).

Now there is no obvious reason why $U \to G$ map data which is insufficient to enable optimisation on the whole control space to be achieved, and which has been determined through control function optimisations for a set of initial conditions, should be of particular help in control function optimisation for another initial condition (unless, as does not occur in the above case, the initial conditions yield linearly dependent contributions to the costed response of the system). The above results, however, show that, for the example considered, the $U \to G$ map data is actually very useful. The results have also demonstrated again the usefulness of our results of 3.4 for deciding when optimisation on a larger (translated) linear manifold of the control space than that on which optimisation has already been achieved would not be particularly profitable, performance-index wise.

3.11 A Gradient-Decomposition Based Algorithm for Optimising in Non-Quadratic Environments

In this section we define non-quadratic to mean (simultaneously) both non-linear and non-quadratic. The gradient-decomposition based optimisation algorithm of 3.6 was designed for use in quadratic environments, for which the second-derivatives of the performance index on the control function - and thus the $U \rightarrow G$ map matrices T - are independent of the control function. In non-quadratic environments the second-derivatives of the performance index on the control function depend on the control function, so that the $U \rightarrow G$ map matrices T also depend on the control function. Thus in non-quadratic environments the constant $U \rightarrow G$ map matrix elements which could be deduced from past control function changes and the resulting gradient function changes are not necessarily correct locally or helpful. In this section we use the gradient-decomposition approach to construct an optimisation algorithm which can decide when the results of past control function changes seem to be no longer helpful and should be discarded. The resulting algorithm can reduce the performance index more rapidly than does the steepest-descent algorithm or the conjugate-gradient algorithm.

The optimisation problem considered is essentially the same as that of 3.2 when the optimal control function is required for some specific initial condition $\tilde{x}_s$ and F and G may be non-quadratic functions of their arguments y and u and the dynamical system is non-linear. We assume that the gradient function $\left(\partial V(\tilde{x}_s, u)/\partial u\right)$ can be calculated for any control function u which is to be applied to the system.

Because non-quadratic environments are considered, it is desirable that all control function variations which are made should lead to immediate improvements in the performance index. A somewhat different algorithm structure to that of 3.6 is therefore needed. At the start of each iteration the algorithm should use the gradient function for the last control function (and perhaps the gradient function changes caused by some or all of the previous control function changes) to compute a new search direction, in which the control function can be optimised using a numerical procedure for minimising with respect to a scalar variable. There are three basic problems associated with each iteration of such an algorithm:

(a) determination of whether the results of past control function changes are likely to be helpful

(b) construction of the $U \to G$ map matrix

(c) choice of the search direction.

The suggested procedures for solving these problems should be clear from stages 3), 4) and 5) of the following algorithm statement.

1) Choose an initial control function u_1 which is a guess at the required $\tilde{x}_s$-optimal control function. Set the iteration index, j, equal to 1. Set k, the number of defined orthonormal basis-functions, equal to 0. Go to 2).

2) Calculate the gradient function $\left(\partial V(\tilde{x}_s, u_j)/\partial u\right)$.

If $\int_T <\left(\partial V(\tilde{x}_s,u_j)/\partial u(t)\right), \left(\partial V(\tilde{x}_s,u_j)/\partial u(t)\right)> dt$ is sufficiently small, consider u_j to be the required $\tilde{x}_s$-optimal control function and stop.

Otherwise:

if k = 0 go to 5) to choose the next search direction,

if k > 0 decompose $\left(\partial V(\tilde{x}_s, u_j)/\partial u\right)$ as $F_{k+1}g^{k+1}(\tilde{x}_s, u_j)$.

Then:

if k = 1 go to 4) to begin constructing a $U \to G$ map matrix,

if k > 1 go to 3) to check the usefulness of the $U \to G$ map matrix elements which have already been established.

3) Check Effectiveness of the Stored $U \to G$ Map Matrix Elements

The structure of this optimisation algorithm is such that basis-function δ_{k+1} can only be defined following a control function change designed to reduce to, or maintain at, zero the components of basis-functions δ_1, .., δ_k present in the gradient function. That control function change was in a search direction chosen by stage 5) of this algorithm with j = j-1 using $U \to G$ map matrix elements deduced from past control function changes and the consequent gradient function changes If that control function change were unsuccessful in that it did not cause the contribution to the gradient function which belongs to $G(1,k)$ (the linear manifold spanned by δ_1, .., δ_k) to be small, it would seem that the $U \to G$ map matrices $T(1,k-1 \to 1,k-1)$ and $T(k \to 1,k-1)$ used to determine the search direction were sufficiently incorrect to be harmful to the convergence of the algorithm. Additional gradient function calculations could be used to determine the locally correct $U \to G$ map matrix for the defined basis-functions but since the gradient calculations would be expensive computationally and the resulting matrix might be of use only once, such an approach would seem to be undesirable.

Also, since optimisation in a specified search direction is expensive computationally, it is undesirable to use $U \rightarrow G$ map matrix elements which are of doubtful validity to choose the next search direction. Perhaps the most suitable measure of the effectiveness of the $U \rightarrow G$ map data used to determine the last search direction is the 'size' $<g^k(\tilde{x}_s,u_j),g^k(\tilde{x}_s,u_j)>$ of the components of basis-functions δ_1, .., δ_k present in the gradient function $\left(\partial V(\tilde{x}_s,u_j)/\partial u\right)$ relative to the 'size' $g_{k+1}(\tilde{x}_s,u_j)^2$ of the component of basis-function δ_{k+1} present in that gradient function.

Thus if $<g^k(\tilde{x}_s,u_j),g^k(\tilde{x}_s,u_j)>/g_{k+1}(\tilde{x}_s,u_j)^2 \leqq \rho$ for some pre-chosen $\rho > 0$, decide that the stored $U \rightarrow G$ map data is satisfactory and go to 4) to deduce new $U \rightarrow G$ map matrix data from the gradient function change caused by the last control function change. Otherwise decide that the stored $U \rightarrow G$ map matrix elements do not describe satisfactorily the $U \rightarrow G$ map in the neighbourhood of the control function u_j and so discard basis-functions δ_1, .., δ_{k+1} and all the stored $U \rightarrow G$ map matrix elements, set k equal to 0 (since there will then be no basis-functions defined) and go to 5) to use the only information which is known to be useful and correct, the current gradient function $\left(\partial V(\tilde{x}_s,u_j)/\partial u\right)$.

4) <u>Deduce $U \rightarrow G$ Map Matrix Elements</u>

The last control function change which was made was exactly characterised by components $\Delta u^k(j-1)^*$ of basis-functions δ_1, .., δ_k and caused a change from $g^k(\tilde{x}_s,u_{j-1})$ to $g^{k+1}(\tilde{x}_s,u_j)$ in the components of the basis-functions which exactly characterise the gradient. By assuming that the environment has been quadratic for the last k control function changes (which seems reasonable in view of the check made in 3), above) and by using the approach of stages 4) and 5) of the algorithm of 3.6, we can now determine the $U{\to}G$ map matrices $T(1,k{\to}1,k)$ and $T(k+1{\to}1,k)$.

Consider first the case when k = 1. On partitioning $g^{k+1}(\tilde{x}_s,u_j) \in R^2$ as $\{g_1(\tilde{x}_s,u_j) \;\; g_2(\tilde{x}_s,u_j)\}^T$, it may be seen that

$$T(1{\to}1) = T(1,1{\to}1,1) = \{g_1(\tilde{x}_s,u_j) - g_1(\tilde{x}_s,u_{j-1})\}/\Delta u^1(j-1)^*,$$
$$T(2{\to}1) = g_2(\tilde{x}_s,u_j)/\Delta u^1(j-1)^*.$$

If $T(1{\to}1) \not> 0$, it would seem to be undesirable to choose a search direction for minimisation purposes using it so discard all the defined basis-functions and $U{\to}G$ map matrix elements, set k equal to zero (since there are then no defined basis-functions) and go to 5) to choose the next search direction using the current gradient function alone.

If $T(1{\to}1) > 0$, compute $T(1{\to}1)^{-1}$ (which is trivial since $T(1{\to}1)$ is a scalar) and go to 5) to choose the next search direction.

Consider next the case when k > 1. We already have $T(1,k-1{\to}1,k-1)$ and its inverse available when we reach this point in the algorithm. Partition $g^k(\tilde{x}_s,u_{j-1}) \in R^k$ as $\{g^{k-1}(\tilde{x}_s,u_{j-1})^T \;\; g_k(\tilde{x}_s,u_{j-1})\}^T$ where $g^{k-1}(\tilde{x}_s,u_{j-1}) \in R^{k-1}$. Partition $g^{k+1}(\tilde{x}_s,u_j) \in R^{k+1}$ as $\{g^{k-1}(\tilde{x}_s,u_j)^T \;\; g_k(\tilde{x}_s,u_j) \;\; g_{k+1}(\tilde{x}_s,u_j)\}^T$ where $g^{k-1}(\tilde{x}_s,u_j) \in R^{k-1}$ and $g_k(\tilde{x}_s,u_j)$, $g_{k+1}(\tilde{x}_s,u_j) \in R$. Partition $\Delta u^k(j-1)^* \in R^k$ as $\{\{\Delta u^{k-1}(j-1)^*\}^T \;\; \Delta u_k(j-1)^*\}^T$, where $\Delta u^{k-1}(j-1)^* \in R^{k-1}$. Then:

$$T(1,k{\to}1,k) = \begin{bmatrix} T(1,k-1{\to}1,k-1) & T(k{\to}1,k-1) \\ T(k{\to}1,k-1)^T & T(k{\to}k) \end{bmatrix}$$

where

$$T(k{\to}1,k-1) = \frac{\{g^{k-1}(\tilde{x}_s,u_j) - g^{k-1}(\tilde{x}_s,u_{j-1}) - T(1,k-1{\to}1,k-1)\Delta u^{k-1}(j-1)^*\}}{\Delta u_k(j-1)^*},$$

$T(\mathrm{k}\to\mathrm{k}) = \left(g_{\mathrm{k}}(\tilde{x}_{\mathrm{s}},u_{\mathrm{j}}) - g_{\mathrm{k}}(\tilde{x}_{\mathrm{s}},u_{\mathrm{j}-1}) - T(\mathrm{k}\to 1,\mathrm{k}-1)^{\mathrm{T}}\Delta u^{\mathrm{k}-1}(\mathrm{j}-1)^*\right)/\Delta u_{\mathrm{k}}(\mathrm{j}-1)^*.$

Also: $T(\mathrm{k}+1\to 1,\mathrm{k}) = \begin{pmatrix} O(\mathrm{k}-1,1) \\ g_{\mathrm{k}+1}(\tilde{x}_{\mathrm{s}},u_{\mathrm{j}})/\Delta u_{\mathrm{k}}(\mathrm{j}-1)^* \end{pmatrix} \in R^{\mathrm{k}}.$

$T(1,\mathrm{k}\to 1,\mathrm{k})^{-1}$ is required for choosing the next search direction in stage 5) of this algorithm and can be constructed from $T(1,\mathrm{k}-1\to 1,\mathrm{k}-1)^{-1}$ (which is p.d. - otherwise this point in the algorithm could not have been reached - and which has already been computed),
$T(\mathrm{k}\to 1,\mathrm{k}-1)$ and $T(\mathrm{k}\to\mathrm{k})$ by using Lemma 3.5.1, if $T(1,\mathrm{k}\to 1,\mathrm{k})$ is p.d. Whether or not $T(1,\mathrm{k}\to 1,\mathrm{k})$ is p.d. will become apparent when Lemma 3.5.1 is used to invert it. If it is p.d., compute $T(1,\mathrm{k}\to 1,\mathrm{k})^{-1}$ and go to 5) to choose the next search direction. If $T(1,\mathrm{k}\to 1,\mathrm{k})$ is not p.d., choice of a search direction for minimisation purposes using it should not be attempted so discard all defined basis-functions and $U\to G$ map matrix elements, set k equal to 0 (since there will then be no defined basis-functions) and go to 5) to use the only remaining useful information - the current gradient function - to define the next search direction.

5) <u>Choice of the Next Search Direction</u>

<u>Consider first the case when k = 0.</u> Since the current gradient function is the only information which is available or which seems to be useful, optimisation in the local steepest-descent direction will be attempted. Decompose the current gradient function $\left(\partial V(\tilde{x}_{\mathrm{s}},u_{\mathrm{j}})/\partial u\right)$ as $F_1 g^1(\tilde{x}_{\mathrm{s}},u_{\mathrm{j}})$. Set $\Delta u^1(\mathrm{j}) = -1$ and go to 6) to optimise in the steepest-descent direction.

<u>Consider next the case when k > 0.</u> To choose a search direction

we assume that the performance index is, locally, a quadratic function of the control function. At this stage the current gradient function, $\{\partial V(\tilde{x}_s,u_j)/\partial u\}$, is exactly characterised by the components $g^{k+1}(\tilde{x}_s,u_j)$ of basis-functions δ_1, .., δ_{k+1} and $T(1,k\to 1,k)$ and $T(k+1\to 1,k)$ are both available, from stage 4). Recall that a necessary condition for a control function to be the $\tilde{x}_s$-optimal control function belonging to $U(1,k+1)$" is that the components of basis-functions δ_1, .., δ_{k+1} present in the gradient function for the control function should all be zero. Due to the check of stage 3) of this algorithm, we can only reach this point in the algorithm if the components of basis-functions δ_1, .., δ_k present in the current gradient function $\{\partial V(\tilde{x}_s,u_j)/\partial u\}$ are all relatively small compared to the component of δ_{k+1} present in the current gradient function. Hence we need only consider the problem of reducing to zero the component of basis-function δ_{k+1} present in the gradient function in order to achieve (at least approximately) the $\tilde{x}_s$-optimal control function belonging to $U(1,k+1)$". Because $T(1,k+1\to 1,k+1)$ is not yet known, we cannot predict a control function change to do this. If the performance index is, locally, a quadratic function of the control function and $T(1,k+1\to 1,k+1)$ is p.d. we can, however, choose a search direction such that a control function variation in the search direction should decrease the component of basis-function δ_{k+1} present in the gradient function while leaving unchanged the components of basis-functions δ_1, .., δ_k present in the gradient function.

We next shown that the search direction which is exactly characterised by the components $\Delta u^{k+1}(j) = \begin{Bmatrix} T(1,k\to 1,k)^{-1}T(k+1\to 1,k) \\ -1 \end{Bmatrix}$ is suitable.

If the performance index function is locally quadratic, we see from the $U \to G$ map property of $T(1,k+1 \to 1,k+1)$ that a change of $\Omega \Delta u^{k+1}(j)$ (Ω being a scalar) in the components of basis-functions $\delta_1, .., \delta_{k+1}$ present in the control function should cause a change of $\Omega \Delta g^{k+1}$ in the components of basis-functions $\delta_1, .., \delta_{k+1}$ present in the gradient function where

$$\begin{aligned}\Delta g^{k+1} &= T(1,k+1 \to 1,k+1)\Delta u^{k+1}(j) \\ &= \begin{pmatrix} T(1,k \to 1,k) & T(k+1 \to 1,k) \\ T(k+1 \to 1,k)^{T} & T(k+1 \to k+1) \end{pmatrix} \begin{pmatrix} T(1,k \to 1,k)^{-1} T(k+1 \to 1,k) \\ -1 \end{pmatrix} \\ &= \begin{pmatrix} O(k,1) \\ T(k+1 \to 1,k)^{T} T(1,k \to 1,k)^{-1} T(k+1 \to 1,k) - T(k+1 \to k+1) \end{pmatrix}.\end{aligned}$$

Since $T(1,k \to 1,k)$ is p.d. (otherwise we would not have been able to reach this point of the algorithm), we see from Lemma 3.5.1 that if $T(1,k+1 \to 1,k+1)$ is p.d. (as it would have to be for minimisation on $U(1,k+1)''$ to be possible):

$$T(k+1 \to 1,k)^{T} T(1,k \to 1,k)^{-1} T(k+1 \to 1,k) - T(k+1 \to k+1) < 0.$$

Hence, under the above assumptions, a change of $\Omega \Delta u^{k+1}(j)$ in the components of basis-functions $\delta_1, .., \delta_{k+1}$ present in the control function should cause no change in the components of basis-functions $\delta_1, .., \delta_k$ present in the gradient function but should cause a reduction (for $\Omega > 0$) in the component of basis-function δ_{k+1} present in the gradient function. Therefore, under the above assumptions, $\Delta u^{k+1}(j)$ does exactly characterise a suitable search direction.

Hence compute $\Delta u^{k+1}(j)$ and go to 6) to optimise in the direction exactly characterised by it (note that if $k = 1$ the search direction is exactly characterised by $\Delta u^{2}(j) = \left(T(1 \to 1)^{-1} T(2 \to 1) \quad -1\right)^{T}$).

6) Optimisation in the Next Search Direction

Optimise the control function in the search direction chosen by stage 5) of this algorithm by minimising with respect to the scalar Ω the performance index $V(\tilde{x}_s, u_j + \Omega F_{k+1} \Delta u^{k+1}(j))$.

Denote the minimising value of Ω by Ω^*.

Then set $u_{j+1} = u_j + \Omega^* F_{k+1} \Delta u^{k+1}(j)$.

The optimal control function change in the search direction was clearly exactly characterised by the components $\Delta u^{k+1}(j)^* = \Omega^* \Delta u^{k+1}(j)$ of basis-functions $\delta_1, .., \delta_{k+1}$.

Since k+1 basis-functions have now been defined, set k = k + 1.

Set j, the iteration index, equal to j + 1. Go to 2).

This concludes the statement of the optimisation algorithm.

∇ *Comment 3.11.1* It will be noted that there are several rather arbitrary decisions taken by the above algorithm which have only been justified intuitively. Nevertheless, it is likely that our algorithm will cause the performance index to decrease more rapidly as a function of iterations than would the steepest-descent algorithm which uses the optimal step in each search direction, because our algorithm always attempts to optimise on as large a linear manifold of the control space as possible and, as may be seen from 3.7, an algorithm which succeeds in doing that will cause faster convergence to the minimal performance index than the steepest-descent algorithm. It is also likely that our algorithm will cause non-quadratic performance indices to be decreased more rapidly as a function of iterations than would the conjugate-gradient

algorithm, since the latter algorithm does not check the validity of the information it uses to construct its search directions, unlike our algorithm. Since the major computational effort associated with each iteration of each algorithm mentioned above is common (the gradient function evaluation and the optimisation in the computed search direction), it therefore seems likely that our algorithm will be more effective at minimising in non-quadratic environments than either the steepest-descent algorithm or the conjugate-gradient algorithm. This statement is, in fact, justified for a particular problem by the computed results which
Δ are presented in the following section.

∇ *Comment 3.11.2* The $U \to G$ map matrices T constructed by the above algorithm will not necessarily be tridiagonal unless the environment is perfectly quadratic and no numerical errors occur. For this reason we have not forced a tridiagonal structure on them but have tried to choose them so as to extract as much information as possible from
Δ the available results.

3.12 A Computed Example

The results of applying the gradient-decomposition based optimisation algorithm of 3.11 to a well known rocket problem are here presented. The problem is one on which the conjugate-gradient algorithm was demonstrated in {27}. The mathematical formulation of the problem is:

$$\min_{u\{0,100\}} V(x_s,u) \text{ for the dynamical system}$$

$$\dot{x}_1(t) = x_3(t)$$

$$\dot{x}_2(t) = 64\cos(u(t))$$

$$\dot{x}_3(t) = 64\sin(u(t)) - 32, \quad \forall t \in \{0,100\},$$

where $x_s = (x_1(0) \;\; x_2(0) \;\; x_3(0))^T = O(3,1)$

$$V(x_s,u) = -x_2(100) + 0.002(x_1(100) - 10^5)^2 + 0.05(x_3(100))^2.$$

The following algorithms were applied, all starting with the initial control function $u(t) = 1.55 - 0.014t$ (rads), $\forall t \in \{0,100\}$:

(a) the steepest-descent algorithm with optimisation in each steepest-descent direction,

(b) the conjugate-gradient algorithm,

(c) the gradient-decomposition based optimisation algorithm of 3.11 with $\rho = 0.16$.

For each algorithm, the performance index value achieved after optimisation in each search direction is plotted versus iterations (i.e. the number of optimisations in chosen search directions) in Fig. 3.3. It can be seen from Fig. 3.3 that our algorithm of 3.11 did tend to choose more effective search directions than did the other algorithms.

The same algorithms were also applied when the initial control function was that control function which resulted from the first iteration of the steepest-descent algorithm for the previous trial. The results are shown in Fig. 3.4. It can be seen from Fig. 3.4 that the gradient-decomposition based algorithm of 3.11 again tended to choose more effective search directions than did the other algorithms.

For each algorithm, the most expensive computational feature associated with each iteration was the optimisation in the chosen search direction. This feature was common, so that each iteration of each algorithm took about the same amount of computation time. Therefore, for this example, the gradient-decomposition based optimisation algorithm of 3.11 has been shown to be more effective at control function optimisation than the steepest-descent and conjugate-gradient algorithms.

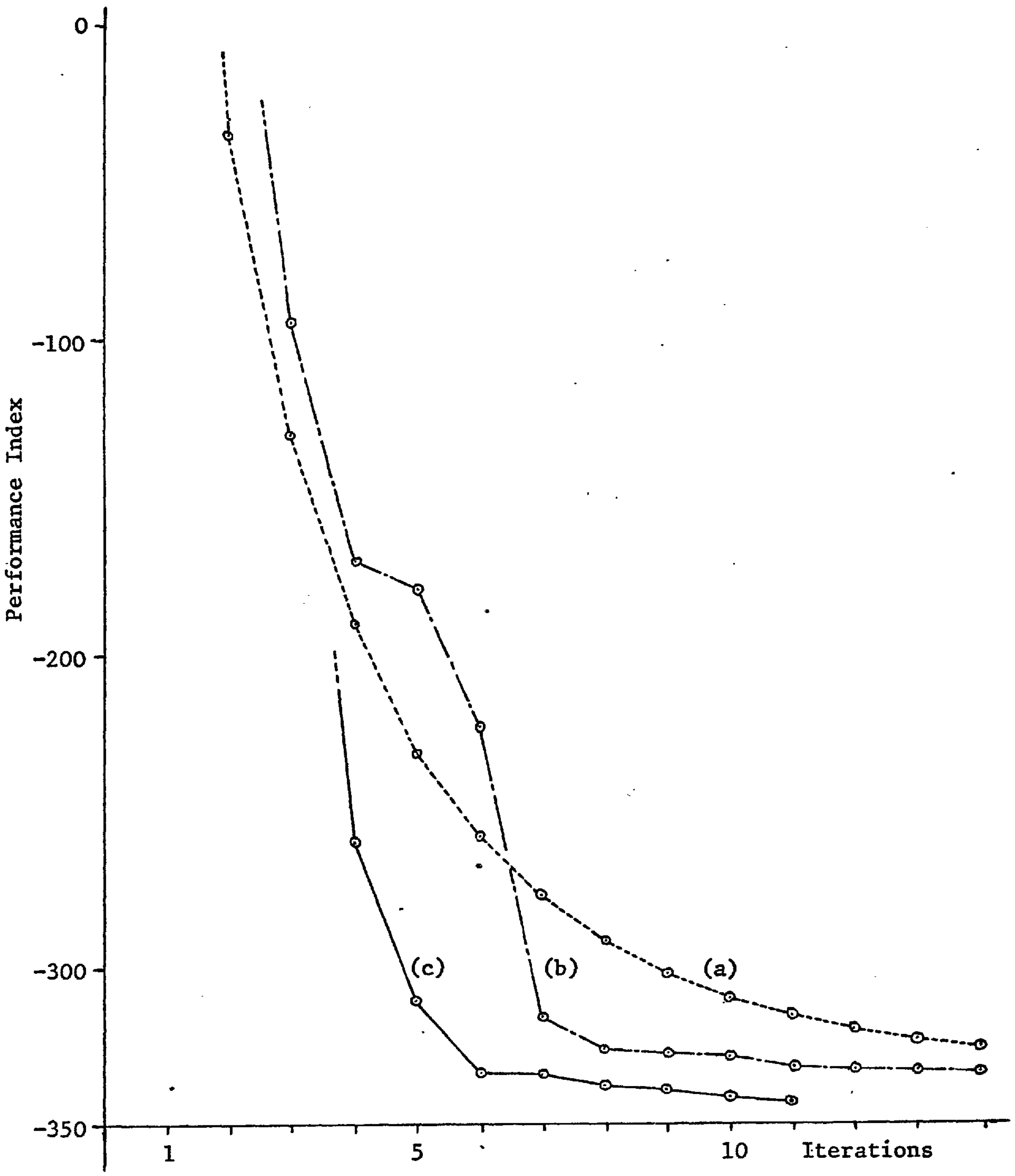

Fig. 3.3

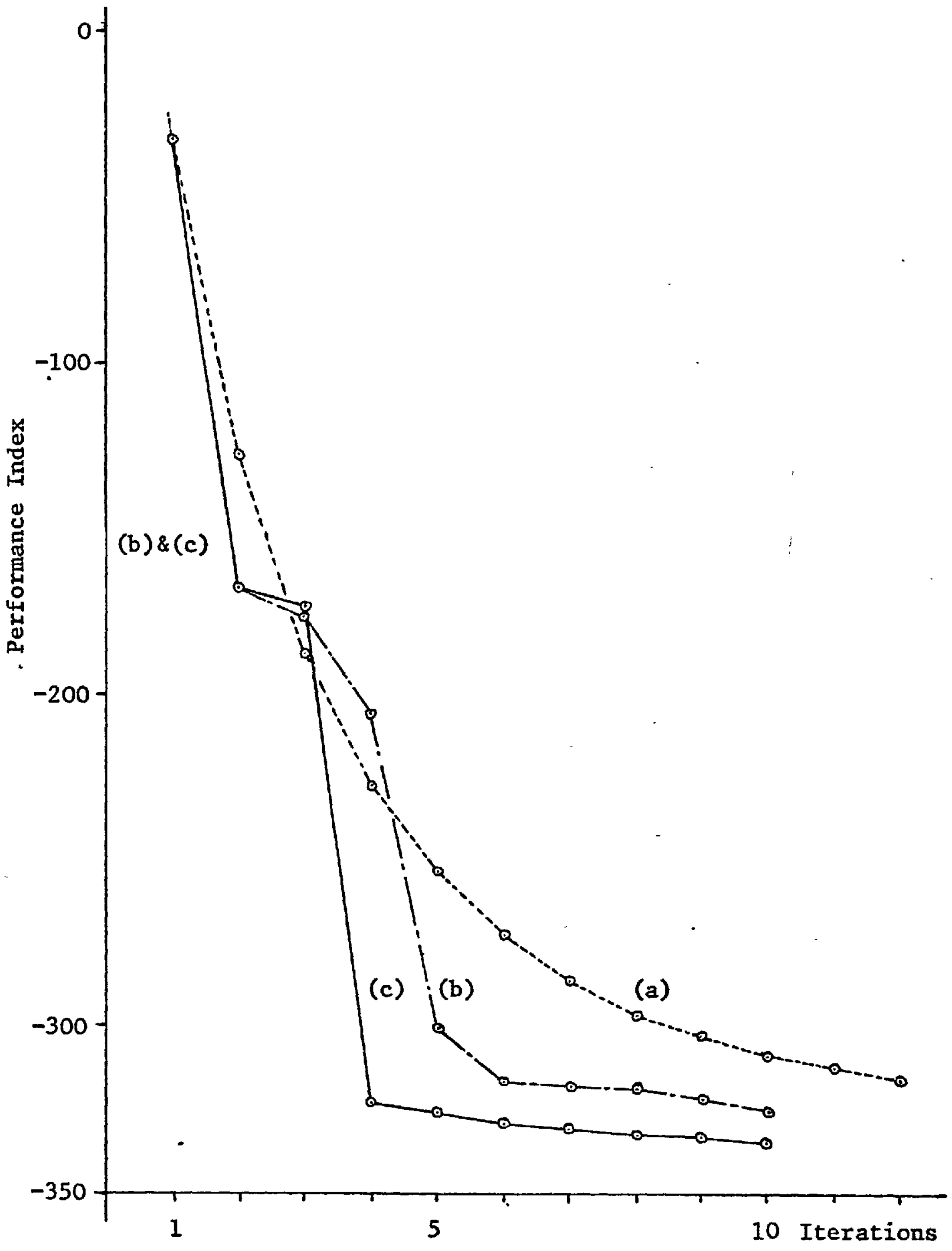

Fig. 3.4

3.13 Concluding Comments

In this chapter, a gradient-decomposition approach to control function optimisation has been introduced. The approach has been fruitful in that it has enabled new optimisation algorithms to be developed. The algorithm of 3.6 is designed for optimisation problems defined for linear dynamical systems with quadratic performance index functions. As well as optimising the control function for a particular initial condition, the algorithm determines a $\mathcal{U}\rightarrow\mathcal{G}$ map matrix which is useful for optimising the control function for other initial conditions. The behaviour of the gradient-decomposition based optimisation algorithm of 3.6 has been compared with that of the steepest-descent algorithm in 3.7, where it is shown that the gradient-decomposition based algorithm is at least as effective as the steepest-descent algorithm, and is usually more effective. It can be shown that our gradient-decomposition based algorithm of 3.6 is related to the conjugate-gradient algorithm, but our algorithm is more versatile since it can check the accuracy of the information it deduces from previous iterations and since it deduces $\mathcal{U}\rightarrow\mathcal{G}$ map data which is helpful for optimising for other initial conditions. The derivation of the lower-bound results of 3.4 depends on the gradient-decomposition approach. The results are used in the algorithm of 3.6 to decide when a required approximation to the optimal control function belonging to the infinite-dimensional control space has been achieved. They are also used in 3.9, where we consider the determination of a control law which is a certain desired $\varepsilon\left(\bar{X}(q)\right)$-approximation to the control law which determines the optimal control function belonging to the infinite-dimensional control space as a function of initial conditions. The lower-bound results, and their applications, are believed to be novel. In 3.11 the gradient-decomposition approach is used in a new algorithm for optimising in non-quadratic environments. The algorithm has proved, for a particular example, to be more effective than either the steepest-descent or the conjugate-gradient algorithms. The algorithm may also be used in quadratic environments.

Some of the results developed in this chapter have been published elsewhere by the author {38}.

An algorithm which has not been mentioned so far is that of Fletcher-Powell (Davidon). The algorithm was originally designed for optimising on a finite-dimensional space. It has recently been extended, in a straightforward way, to enable optimisation on an infinite-dimensional space to be performed {39}. The extended version, however, requires operations to be performed using infinite-dimensional matrices (estimates of the inverse of the second-derivative operator). It is interesting, and computationally significant, that our gradient-decomposition approach leads to algorithms which, while behaving in essentially the same way as the Fletcher-Powell (Davidon) algorithm, use matrices the dimension of which depends only on the number of iterations which have been performed, and does not depend on the dimensionality of the space on which optimisation is desired.

Chapter 4 : Final Remarks

It may be considered that, loosely speaking, there are three necessary conditions for efficient control function optimisation to be possible:

(a) there should exist a mathematical model of the system to be controlled which enables system responses to be evaluated with relatively little computational effort,

(b) there should exist effective optimisation algorithms which can be used with the efficient model of the system, and

(c) since most effective algorithms are (at least in some sense) iterative, there should exist a means of determining when sufficient iterations have been performed.

In this thesis we have discussed the use of the convolution-description of linear dynamical systems since it can sometimes have considerable computational advantages, in spite of the storage requirements, and can thus help to enable condition (a) to be satisfied.

By employing procedures based on synthesising optimal control functions from components of linearly independent basis-functions, we have developed new algorithms for optimising control functions for linear convolution-described dynamical systems. We have thus contributed towards satisfying condition (b).

Our lower-bounds for the minimal performance index have been used for deciding when an adequate approximation to the optimal control function, or optimal control law, has been achieved. We have thus made a contribution towards satisfying condition (c).

A more detailed discussion of our contributions will not be given here, since they have all been discussed in 1.3, 1.4 and in the body of the thesis.

In the future it is intended to apply our results to more examples. In particular, it would be especially interesting to compare the behaviour of the gradient-decomposition based optimisation algorithm of 3.11 with that of the steepest-descent and conjugate-gradient algorithms for a number of large, non-quadratic, examples. The ideas used in this thesis have been applied (by the author) to optimisation problems subject to linear, terminal-equality, constraints. It is hoped to be able to apply them to optimisation problems subject to more general constraints.

References

1 A.R. Forsyth : Calculus of Variations;
Dover Publications Inc, 1960

2 Pontryagin, Boltyanskii, Gamkrelidze, Mischenko:
The Mathematical Theory of Optimal Processes;
Wiley Interscience, 1962

3 L.I. Rozonoer : L.S. Pontryagin Maximum Principle in the Theory of Optimum Systems;
Automation and Remote Control, 20, 1959
I No 10, 1288-1302
II No 11, 1405-1421
III No 12, 1517-1532

4 M. Athans, P.L. Falb : Optimal Control : An Introduction to the Theory and Its Applications;
McGraw Hill, 1966

5 A.G. Butkovskii : The Maximum Principle for Optimum Systems with Distributed Parameters;
Automation and Remote Control, 22, No 10, 1156-1169, 1962

6 A.G. Butkovskii : The Broadened Principle of the Maximum for Optimal Control Problems;
Automation and Remote Control, 24, No 3, 292-304, 1963

7 H. Halkin : Optimal Control for Systems Described by Difference Equations;
Advances in Control Systems, 1, 173-196, 1964

8 H. Halkin : A Maximum Principle of the Pontryagin Type for Systems Described by Non-linear Difference Equations;
J. SIAM Control, 4, No 1, 90-111, 1966

9 H. Halkin : The Principle of Optimal Evolution;
Report on International Symposium on Non-linear Differential Equations and Non-linear Mechanics,
284-302, Academic Press, 1963

10 H. Halkin : Mathematical Foundations of System Optimisation;
in Topics in Optimisation, Ed. G. Leitmann, Academic Press, 1967

11 P. Kenneth, R. McGill : Two Point Boundary-Value-Problem Techniques;
Advances in Control Systems, 3, 69-109, 1966

12 M.D. Levine : Trajectory Optimisation using the Newton-Raphson Method;
Automatica, 3, 203-217, 1966

13 J.V. Breakwell, J.L. Speyer, A.E. Bryson : Optimisation and Control of Nonlinear Systems using the Second Variation;
J. SIAM Control, 1, No 2, 193-223, 1963

14 M.D. Levine : Parameterized Feedback Control of Non-linear Processes;
Int. J. Control, 3, No 1, 39-49, 1966

15 R. Bellman : Adaptive Control Processes, A Guided Tour;
Princeton University Press, 1961

16 R. Bellman : Dynamic Programming;
Princeton University Press, 1957

17 S.E. Dreyfus : Dynamic Programming and the Calculus of Variations;
Academic Press, 1965

18 R.E. Larson : Dynamic Programming with Reduced Computational Requirements;
IEEE Trans. on Automatic Control, AC-10, 135-143, April 1965

19 A.E. Bryson, W.F. Denham : A Steepest-Ascent Method for Solving Optimum Programming Problems;
Report BR-1303, Raytheon Missile and Space Division, 1961

20 D.Q. Mayne : A Second-Order Gradient Method for Determining Optimal Trajectories of Non-linear Discrete-time Systems;
Int. J. Control, 3, No 1, 85-95, 1966

21 D.H. Jacobson : Differential Dynamic Programming Methods for Determining Optimal Control of Non-linear Systems;
Ph.D. thesis, London University, 1967

22 D.H. Jacobson : Second-Order and Second-Variation Methods for Determining Optimal Control: A Comparitive Study using Differential Dynamic Programming;
Int. J. Control, 7, No 2, 175-196, 1968

23 D.H. Jacobson : Differential Dynamic Programming Methods for Solving Bang-Bang Control Problems;
IEEE Trans. on Automatic Control, AC-13, December 1968

24 W.A. Porter : Modern Foundations of Systems Engineering;
Macmillan, 1966

25 A.V. Balakrishnan : An Operator Theoretic Formulation of a Class of Control Problems and a Steepest-Descent Method of Solution;
J. SIAM Control, 1, No 2, 109-127, 1963

26 H.C. Hsieh : Synthesis of Adaptive Control Systems by Function Space Methods;
Advances in Control Systems, 2, 117-208, 1965

27 L.S. Lasdon, S.K. Mitter, A.D. Warren : The Conjugate Gradient Method for Optimal Control Problems;
IEEE Trans. on Automatic Control, AC-12, No 2, April 1967

28 J.D. Pearson : Studies in the Optimal Control of Dynamic Systems;
Ph.D. Thesis, London University, 1963

29 J.D. Pearson : Reciprocity and Duality in Control Programming Problems;
J. Mathematical Analysis and Applications, 10, 388-408, 1965

30 J.D. Pearson : Duality and Decomposition Techniques;
J. SIAM Control, 4, No 1, 164-172, 1966

31 L.A. Zadeh, C.A. Desoer : Linear System Theory: The State Space Approach;
McGraw Hill, 1963

32 K. Petchsuwan : The Optimal Control of Linear Distributed Parameter Systems;
Ph.D. Thesis, London University, 1967

33 R.E. Kalman : Mathematical Description of Linear Dynamical Systems;
J. SIAM Control, 1, No 2, 1963

34 R.E. Kalman : Towards a Theory of Difficulty of Computation in Optimal Control;
Proceedings of IBM Scientific Computing Symposium on Control Theory and Applications at T.J. Watson Research Center, Yorktown Heights, New York, 19-21 October, 1964

35 A.E. Pearson : A Regression Analysis in Function Space for the Identification of Multivariable Linear Systems;
Paper 46B, IFAC Congress, London 1966

36 H.C. Hsieh : An On-Line Identification Scheme for Multivariable Nonlinear Systems;
Computing Methods in Optimisation Problems,
Ed. A.V. Balakrishnan, L.W. Neustadt, Academic Press, 1964

37 H. Nicholson : Dynamic Optimisation of a Boiler;
Proc. IEE, 111, No 8, 1479-1499, August 1964

38 J.C. Allwright : Optimal Control Synthesis using
Function Decomposition Techniques;
Paper 7.3, Preprints of 4th IFAC Congress, Warsaw 1969

39 L.B. Horwitz, P.E. Sarachik : Davidon's Method
in Hilbert Space;
SIAM J. Appl. Math., 16, No 4, 677-695, July 1968