Computational Solution of Dynamic Optimization Problems with General Differential-Algebraic Constraints

by

Vassilios Vassiliadis

July 1993

A thesis submitted for the degree of Doctor of Philosophy of the University of London and for the Diploma of Membership of the Imperial College

Department of Chemical Engineering and Chemical Technology Imperial College of Science, Technology and Medicine
London SW7 2BY, U.K.

ABSTRACT

This thesis considers the optimal control of a class of multistage systems described by mixed sets of differential-algebraic equations. The set of variables characterizing the system behaviour is invariant throughout the time horizon of interest; however the describing equations may change from stage to stage. A general form of initial condition specification is allowed at the start of the first stage while general junction conditions are allowed between stages.

The algorithm allows general interior point constraints to be taken into account. A control parameterization approach is employed with the control profiles being approximated by piecewise continuous Lagrange polynomials. Control parameterization transforms the original optimal control problem to a nonlinear programming problem. The objective function and constraint values are obtained by integrating the differential-algebraic equations, while their gradients are calculated via simultaneous integration of the differential-algebraic equations and their trajectory sensitivity equations. The integration code used in both cases is based on a backward difference formula scheme. Sparse matrix techniques are used to allow the solution of large scale problems. The optimization is carried out using a successive quadratic programming code.

The problem of inequality path constraints is addressed by converting them to end-point ϵ relaxed constraints using a violation norm integral, together with a small number of interior point constraints.

ACKNOWLEDGEMENTS

I wish to express my gratitude to Professor R. W. H. Sargent for his support and motivation during the course of this work. I am also very grateful to Dr. C. C. Pantelides for guiding me throughout this research. Financial support from the Centre for Process Systems Engineering at Imperial College and the Science and Engineering Research Council (SERC) is gratefully acknowledged.

From the many friends I made at the Centre for Process Systems Engineering, I wish to thank Steve P. Walsh for the valuable discussions on dynamic optimization and for sharing his knowledge in systems engineering in general.

My friend Lazaros G. Papageorgiou is a person whose presence in the Centre I shall always remember for his moral support, advice and sincere friendship.

Sofia I. Siachou, I will remember for her sincerity and intuition, and culinary skills!

Michail T. Diamantakis, has been a very good friend and flatmate for many years. His controlled behaviour has been an example to follow.

I am grateful to my friend Merten C. Johnson, who with his friendship and advice had helped me cope with the difficulties of academic research during the first year of my stay in London.

Last I wish to thank the people I consider to be the only family I have in my life at this moment. My father Sotirios with his strong character and personality has definitely influenced every aspect of my life. His knowledge in engineering and foreign languages I shall always strive to match. My mother Nina has been the sentimental part in my upbringing. She helped me through my early years in school and practically taught me English as well. Both my parents have offered me memories which to this date I remember, and as my own experiences grow I am still able to extract knowledge from them. Finally, my godmother Eleni C. Dimitriou, has always been like a second mother to me, and for her devotion and love I shall never forget her.

Dedicated to my Parents

... ...

Πάντα στον νου σου νάχεις την Ιθάκη.
Το φθάσιμο εκεί είν' ο προορισμός σου.
Αλλά μη βιάζεις το ταξείδι διόλου.
Καλλίτερα χρόνια πολλά να διαρκέσει·
και γέρος πιά ν' αράξεις στο νησί,
πλούσιος με όσα κέρδισες στον δρόμο,
μη προσδοκώντας πλούτη να σε δώσει η Ιθάκη.

Η Ιθάκη σ' έδωσε τ' ωραίο ταξείδι.
Χωρίς αυτήν δεν θάβγαινες στον δρόμο.
Αλλα δεν έχει να σε δώσει πια.

Κι αν πτωχική την βρεις, η Ιθάκη δεν σε γέλασε.
Ετσι σοφός που έγινες, με τόση πείρα,
ήδη θα το κατάλαβες οι Ιθάκες τι σημαίνουν.

"Ιθάκη", Κ. Π. Καβάφης.
("Ithaki", C. P. Cavafy)

CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

A = matrix defined by (4.9)

A_i = matrix defined by (2.39)

B = general matrix defined by (4.10)

B_i = general matrix defined by (2.41)

C = constant appearing in (1.61); objective function of general multistage system in (2.11)

C_A = concentration of component A

C_B = concentration of component B

C_C = concentration of component C

C_D = concentration of component D

C_R = concentration of component R

C_S = concentration of component S

C_T = concentration of component T

c = objective function used in (1.35) and (1.56) throughout Chapter 1; general symbol for path constraints from Chapter 2 onwards

$c_i^{[eq]}$ = equality path constraints for stage i

$c_i^{[in]}$ = inequality path constraints for stage i

d_i = injected material amount

E_0 = equality constraints at initial time

E_i = equality constraints at end of stage i

f = differential-algebraic equations

f_i = differential-algebraic equations for stage i

g_j = auxiliary relations in section 3.3, $j = 1..3$

H = Hamiltonian function

h_i = length of element i

h_i^L = lower bound on length of element i

h_i^U = upper bound on length of element i

I = objective function in (1.1); unit matrix symbol from (1.54) to the end of Chapter 1;

INT = number of integrations defined in (5.1)

i = standard index for finite elements or stages depending on the context

J = junction conditions and initial conditions, dimension n in Chapter 1

J_0 = initial conditions

J_i = junction conditions at end of stage i

j = node index in Chapter 1, control variable index from Chapter 2 onwards

K = number of collocation points within a finite element

k = node index

k_j = rate for reaction j

L = lower triangular matrix defined by (4.10)

L = integrated function in (1.1); reactor length in section 3.2; lower triangular matrix defined by (2.41)

l = general index; length of chord in section 3.11; dimension of equality path constraints in Chapter 4

M_0 = initialization Jacobian defined in (1.30), dimension $2n \times 2n$

M_j = polynomial order for the approximation of control variable j

$M_{jk}^{[i]}$ = matrix block element $[j,k]$ of Jacobians resulting from discretization methods in Chapter 1

m = number of algebraic variables

m_i = mass of vehicle in (2.32) for stage $i = 1..3$

ND = number of time invariant parameters defined in (5.1)

NP = number of parameters per element due to the parameterization of the control variables defined in (5.1)

NS = number of finite elements or stages, defined according to context

n = number of differential variables

P = general end-point evaluated function defined by (1.13); utility profit in section 3.2; row permutation matrix defined by (4.10)

$\bar{P}$ = implicit definition of P as a function of v and t_f in (1.18)

P_1 = row permutation matrix defined by (4.11)

P_2 = row permutation matrix defined by (4.11)

P_i = row permutation matrix defined by (2.41)

p = vector of time invariant parameters defined by (2.57)

Q = column permutation matrix defined by (4.10)

Q_1 = column permutation matrix defined by (4.11)

Q_2 = column permutation matrix defined by (4.11)

Q_i = column permutation matrix defined by (2.41)

q = number of design variables

R = radius in section 3.11

$R^{[i]}(\tau)$ = residual of function f evaluated at normalized time τ in element i, dimension n

r_i = norm defined in (4.24)

$r_0^{[eq]}$ = equality point constraints at initial time

$r_0^{[in]}$ = inequality point constraints at initial time

$r_i^{[eq]}$ = equality point constraints at end of stage i

$r_i^{[in]}$ = inequality point constraints at end of stage i

S = length of arc in section 3.11

s_i = fixed distances in (2.33), (2.34), (2.35) for $i = 1..3$

T = stage end-times in Chapter 2; temperature in sections 3.2 and 4.4.4

$\bar{T}$ = normalized temperature in section 3.2

T_P = reactor product temperature in section 3.2

T_R = reactor inlet temperature in section 3.2

T_S = steam temperature in section 3.2

t = time

t_0 = initial time

t_f = final time

t_i = end-point of stage i

U = set of control variables for all stages; upper triangular matrix defined by (4.10)

U_i = upper triangular matrix defined by (2.41)

u = control variables, dimension π

u^L = control variable lower bounds

u_i^L = control variable lower bounds for stage i

u^U = control variable upper bounds

u_i^U = control variable upper bounds for stage i

$u^{[i]}$ = approximation of differential variables in element i defined in (1.49), dimension π

$u_k^{[i]}$ = value of differential variables in element i at node k, dimension π

u_i = control variables for stage i

u_{ijk} = value at node k of control j in stage/element i

V = set of design variables for all stages in Chapter 2; volume in Chapter 3

v = design variables, dimension q

v_i = design variables for stage i

v^L = design variable lower bounds

v_i^L = design variable lower bounds for stage i

v^U = design variable upper bounds

v_i^U = design variable upper bounds for stage i

w = parameters in (1.9)

w^u = parameters in (1.37)

w^x = parameters in (1.38)

X = set differential variables for all stages

$\dot{X}$ = time-derivatives of X

x = differential variables, dimension n

$\dot{x}$ = time-derivatives of differential variables, dimension n

$\ddot{x}$ = second order time-derivatives of differential variables, dimension n

x_i = differential variables for stage i

$\dot{x}_i$ = time-derivatives of differential variables for stage i

$x^{[i]}$ = approximation of differential variables in element i defined in (1.48), dimension n

$x_k^{[i]}$ = value of differential variables in element i at node k, dimension n

Y = set of algebraic variables for all stages

$\mathcal{Y}$ = overall product yield

y = algebraic variables, dimension m; conversion in section 3.4

y_i = algebraic variables for stage i

Greek Letters

α_k = BDF coefficient for step k in (1.69)

α_{jk} = Runge-Kutta coefficient in (1.75)

$\gamma_l^{[i]}$ = vector of the values of differential variable x_l ($l = 1..n$) at collocation points $1..K$ within element i, dimension K

$\dot{\gamma}_l^{[i]}$ = time-derivatives of $\gamma_l^{[i]}$

δ = discretization error measure used in section 1.4.2

ϵ = higher order terms in (1.14); tolerance in (1.62); tolerance used from Chapter 3 onwards

ϵ_0 = higher order terms in (1.15)

ϵ_i = tolerance for point constraints at time t_i

θ = control variable approximating function in (1.9), dimension π; dimensionless temperature in section 3.4; angle in section 3.11

θ^u = control variable approximating function in (1.37), dimension π

θ^x = differential variable approximating function in (1.38), dimension n

λ = adjoint multipliers in (1.3), dimension n; variables in the adjoint system (1.21), dimension n

μ = variables in the adjoint system (1.21), dimension n

$\dot{\mu}$ = time-derivatives of μ, dimension n

ρ = multipliers used in (1.24), dimension n

τ = normalized time

Φ = overall fractional yield

$\dot{\Phi}$ = matrix of $\dot{\phi}_{jk}$ for $j, k = 1..K$

ϕ = end-point term in (1.1)

$\phi_j(\tau)$ = Lagrange basis polynomials for differential variables, for normalized time τ_j

ϕ_{kj} = Lagrange basis polynomials for differential variables, for normalized time τ_j, evaluated at normalized time τ_k

$\dot{\phi}_{jk}$ = time-derivative of Lagrange basis polynomials for differential variables at normalized time τ_k evaluated at normalized time τ_j

$\phi_j^{[i]}(t)$ = Lagrange basis polynomials for differential variables, in element i for time t_j

$\phi_{kj}^{[i]}$ = Lagrange basis polynomials for differential variables, in element i for time t_j, evaluated at time t_k

$\psi_j(\tau)$ = Lagrange basis polynomials for control variables, for normalized time τ_j

ψ_{kj} = Lagrange basis polynomials for control variables, for normalized time τ_j, evaluated at normalized time τ_k

$\psi_j^{[i]}(t)$ = Lagrange basis polynomials for control variables, in element i for time t_j

$\psi_{kj}^{[i]}$ = Lagrange basis polynomials for control variables, in element i for time t_j, evaluated at time t_k

ω_0 = multipliers used in (1.24), dimension n

ω_f = multipliers used in (1.24), dimension n

CHAPTER 1

Introduction and Literature Review

This chapter reviews recent advances in numerical algorithms for the solution of general optimal control problems. The first section presents briefly some definitions. Subsequent sections provide a general overview of relevant literature, and a critical comparison of the various methods presented.

1.1 Preliminaries

Optimal control may be defined as the determination of time-dependent *controls* and time invariant *parameters* that optimize some performance index of a dynamic system. This problem plays an important rôle in chemical engineering in the area of dynamic optimization, examining the performance of a system operating under transient conditions. Two main reasons for this interest are modelling complex process plants coupled with tight operating targets, and modelling processes that are inherently dynamic, such as batch processes. Some examples of process application areas that may benefit from optimal control (Edgar and Himmelblau, 1989) are:

- Maximum production of a key component.
- Minimum yield of a contaminant.
- Start-up or shut-down of processes.
- Moving from one steady state to another.

Dynamic modelling of most chemical engineering applications results in a differential-algebraic equation (DAE) formulation, arising usually from dynamic mass and energy balances, which yield ordinary differential equations (ODEs), and thermodynamic relations, which yield the algebraic constraints.

This thesis deals with optimal control problems subject to general types of constraints, addresses an approach for multistage dynamic optimization problems, and presents a hybrid approach for the efficient handling of state constraints.

1.2 Review of Recent Literature on Computational Methods for Optimal Control

The mathematical statement of a simple optimal control problem to motivate further discussion follows:

PROBLEM SOCP

Objective function

$$\min_{u(\cdot)} I = \phi(x(t_f), t_f) + \int_{t_0}^{t_f} L(x(t), u(t), t)\, dt \tag{1.1}$$

subject to:

ODE system

$$\dot{x}(t) = f(x(t), u(t), t); \quad x(t_0) = x_0; \quad t_0 \in [t_0, t_f]; \quad t_f \text{ is fixed} \tag{1.2}$$

---

where $x, \dot{x}, f \in \mathcal{R}^n$, and $u(t) \in \mathcal{R}^\pi$.

A useful quantity is the Hamiltonian which is defined as the following scalar function:

$$H(x(t), u(t), \lambda(t), t) = L(x(t), u(t), t) + \lambda^T(t) f(x(t), u(t), t) \tag{1.3}$$

where $\lambda(t)$ is an n-dimensional vector of multipliers.

Considering first order variations in I due to variations in the control variables $u(t)$ (with t_0 and t_f remaining fixed), the following first order necessary conditions are derived[1] (Bryson and Ho, 1988):

$$\dot{x} = f(x, u, t) \tag{1.4}$$

$$\dot{\lambda}^T = -\frac{\partial H}{\partial x} = -\frac{\partial L}{\partial x} - \lambda^T \frac{\partial f}{\partial x} \tag{1.5}$$

$$\frac{\partial H}{\partial u} = \frac{\partial L}{\partial u} + \lambda^T \frac{\partial f}{\partial u} = 0 \tag{1.6}$$

$$x(t_0) = x_0 \tag{1.7}$$

$$\lambda(t_f) = \left(\frac{\partial \phi}{\partial x}\right)^T \tag{1.8}$$

The equations derived above are known as the *Euler-Lagrange equations* in the calculus of variations. Variables λ and eq. (1.5) are known as *co-state* (or *adjoint*) variables and equations respectively.

It is clear that the necessary conditions define a system with split boundary conditions, *i.e.* comprising a *two-point boundary value problem* (TPBVP) since $x(t)$ and $\lambda(t)$ are specified at the initial and the end times, respectively.

Most of the numerical techniques presented in the literature can be classified under four major categories, all of which are reviewed in the following subsections.

1.2.1 Dynamic Programming

Bellman (1957) (pp. 245–282) used dynamic programming principles to derive conditions of optimality for optimal control problems. These ideas were extended by Jacobson and Mayne (1970) and solution algorithms were proposed.

[1] For simplicity, time is omitted as an argument of state variables and the controls.

Recently, Luus (1990a) proposed another dynamic programming approach. The proposed method uses grids for both the state variables and the controls. The grid of the states defines accessible points in the state trajectory and the grid of the controls defines permissible control values. The grids are refined iteratively until a satisfactory control policy is reached. Applications of this approach were demonstrated in Luus (1990a, 1990b), and the implementation of final state constraints were considered by Luus and Rosen (1991), who used a penalty function approach to incorporate the constraints into the objective function. Originally piecewise constant controls were employed and recently Luus (1993) extended the implementation to piecewise linear continuous control policies.

1.2.2 Complete Discretization

The characteristic of this class of algorithms is the discretization of all equations with respect to all variables. Originally, finite difference approximations were used for derivatives, *e.g.* Canon *et al.* (1970).

Later collocation was introduced as a means of discretizing the equations. A description of the principle is given in Polak (1971) who used finite difference approximations. Tsang *et al.* (1975) proposed the use of collocation for transforming dynamic optimization problems to standard nonlinear programming problems. Biegler (1984) applied global orthogonal collocation using Lagrange polynomials to this end. However, collocation over the entire optimization time-horizon using a single polynomial is not an effective way to approximate the control functions as it results in oscillations and cannot easily guarantee the satisfaction of bounds on the controls.

To remedy this deficiency, the time horizon was divided into finite elements, applying collocation over each element. Renfro (1986) and Renfro *et al.* (1987) used global spline collocation to convert the differential equations into a set of residual algebraic equations, while the control variables were assumed to be piecewise constant.

Vlassenbroeck and Van Dooren (1988) used Chebyshev series to express the

state and control variables for the discretization of the ODEs in their problems. In their examples, they used a discretization over the whole time interval, and noted that their approach can be applied over finite elements in the time-horizon as well.

Cuthrell and Biegler (1987) proposed the use of finite element collocation for both state and control variables using Lagrange polynomials to convert the DAEs involved to algebraic equations with unknown coefficients, evaluated at each collocation point. The choice of Lagrange polynomials has the advantage that their coefficients are the values of the approximated functions at the collocation points, and therefore bounding them is straightforward. Following this initial publication, Biegler and his co-workers produced a substantial amount of work based on this idea.

Collocation on finite elements is equivalent to fully implicit Runge-Kutta schemes, and for this reason Logsdon and Biegler (1989) considered the stability and error properties of fully implicit Runge-Kutta methods to derive and enforce appropriate error constraints directly within the discretized nonlinear programming problem (NLP) model.

Later, Vasantharajan and Biegler (1990) examined different error enforcement strategies, and ways of estimating a sufficient number of elements for the approximation of state variables.

More recently, Logsdon (1990) and Logsdon and Biegler (1992) recognized that the number of algebraic equations produced by collocation for realistic size chemical engineering problems is large enough to cause problems for a direct solution via a standard *sequential quadratic programming* algorithm (SQP) (Fletcher, 1991; Luenberger, 1984). They therefore derived a feasible path approach solving the quadratic programming problems (QPs) in the reduced control space instead of the full variable space. It is indeed interesting to note that Polak (1971, p. 3), observes that the very high dimensionality of the vector in the fully discretized version of the optimal control problem is unreasonably large and consequently it is preferable to work in the reduced space.

1.2.3 Infinite Nonlinear Programming Problem Approximation

Originally a steepest descent method in the "function space" discretized using adjoints was proposed by Kelley (1960, 1962). Lasdon *et al.* (1967) used conjugate gradients instead of steepest descent and their technique was modified by Pagurek and Woodside (1968) to incorporate lower and upper bounds on the controls. Horwitz and Sarachik (1968) employed quasi-Newton methods to evaluate the control profiles.

In essence these techniques are what became known as *control vector iteration* (CVI) a simplified example of which is the first variation based algorithm given in Ray (1981).

The convergence of the first variation based methods slows in the terminal phase of the optimization, which is their main weakness. To remedy this situation second order approaches were considered and an example of such an approach is given in the work of Kelley (1964).

Control parameterization (CP) can be regarded as a development from the techniques above. Pollard and Sargent (1970) suggested piecewise constant controls, using levels and switching times as variables, instead of values of controls at every integration step as with CVI algorithms. Essentially they employed an unconstrained quasi-Newton algorithm for the optimization, using adjoints for gradient derivation.

Sargent and Sullivan (1977) and Sullivan (1977) presented a formulation of optimal control problems as NLPs using control parameterization. They were the first to produce a large scale package for the solution of optimal control problems using the adjoint approach for gradient evaluations and piecewise constant approximations to the controls.

Their approach is a *feasible path approach* because it produces feasible trajectories for any choice of controls due to the integration stage. It was later extended by Morison and Sargent (1984) and Morison (1984) to index-one differential-algebraic equation models and multistage problems, and again a general purpose package was produced based on adjoints for gradient evaluation and piecewise

constant controls.

Later, Gritsis (1990) proposed a formulation for index-two DAE optimal control models, clarifying the implications of the increased index on dynamic optimization by extending the formulation of the adjoint equations to this class of problems.

It is worth mentioning that parameterization based approaches, using both complete discretization and discretization of the control variables only, are considered in a very concise and unified manner in Polak (1971, pp. 1–6). Also Kraft (1985) presents and compares schemes for control parameterization and control and state parameterization.

1.2.4 Solution of Necessary Conditions

Variational formulation (Pontryagin's Principle) gives a two-point boundary value problem with the control given by local optimization at each step.

The original shooting method, according to which the initial values of the adjoint variables are guessed and updated in an iterative manner (*e.g.* Mengel, 1951), was unsuccessful because of sensitivity to initial adjoint values. To deal with this problem, three main approaches have appeared in the literature.

The first approach is quasilinearization proposed by Miele and co-workers (Miele *et al.*, 1970, 1974, 1982a, 1982b, Miele, 1975). They solved linearized necessary conditions by finding n linearly independent solutions and combining these to satisfy boundary conditions. A "restoration phase" was then used in which the controls were corrected to satisfy the nonlinear constraints. In a sense they borrowed the idea of "gradient restoration" or "correction step" from projection methods for NLP. Their method relies on the fact that both the gradient phase and the restoration phase deal with a linear TPBVP and thus special techniques for such problems can be used effectively. In this manner the corrections obtained result in feasible trajectories.

Since direct shooting is difficult to implement successfully in practice, the idea of multiple shooting emerged (*e.g.* Keller, 1968, Roberts and Shipman, 1972).

Using the necessary conditions of optimality, the problem is transformed to a multipoint boundary value problem for the state and the adjoint variables, which can be solved by the multiple shooting method as described in Bulirsch (1971) and Stoer and Bulirsch (1980).

Dixon and Bartholomew-Biggs (1981) used an alternative shooting algorithm that involves guessing the values of the control variables and their time-derivatives at the initial time, thereby avoiding the need for good guesses for the adjoint variables. This may be advantageous since, due to the lack of physical meaning, good initial guesses for the adjoint variables may be difficult to obtain and convergence may not easily be achievable. In their method, the values of the initial controls and their derivatives were obtained via standard optimization techniques. Bartholomew-Biggs *et al.* (1988) used this approach successfully to solve interplanetary trajectory optimization problems.

1.3 Control Parameterization Techniques

As described in subsection 1.2.3, this class of methods relies on the discretization of the control variables using a finite set of parameters w such that the controls can be expressed as:

$$u(t) = \theta(t, w) \tag{1.9}$$

The problem (SOCP) becomes thus a finite dimensional optimization problem with the only difficulty remaining being the calculation of values and gradients of the objective function and the constraints with respect to the parameters w. All practical solution techniques in this category involve direct integration of the describing ODEs (or DAEs) and some variational equations to derive the required values and gradients, respectively. With regard to the variational equations to be used, there is a choice of two methods, as described in the following subsection.

1.3.1 Derivation of Sensitivity Equations and Gradient Evaluation

Adjoint equations and sensitivity equations are the two options for variational equations for gradient evaluation. The equivalence of sensitivity equations and adjoint equations in the sense that they both result in the same value of the reduced gradients for prescribed functions at a given time, will be demonstrated.

We start by considering the following problem. Given the following DAE system:

$$f(\dot{x}, x, v, t) = 0; \quad t \in [t_0, t_f] \tag{1.10}$$

and the set of initial conditions:

$$J(\dot{x}(0), x(0), v) = 0; \quad t = t_0 \tag{1.11}$$

where v is a set of parameters, evaluate the gradient matrix of variables with respect to the parameters:

$$x_v = \frac{\partial x}{\partial v} \qquad \text{at times} \quad t_I \in [t_0, t_f] \tag{1.12}$$

where $x, \dot{x}, f \in \mathcal{R}^n$, and $v \in \mathcal{R}^q$.

Use this gradient matrix to derive the reduced gradient of a function:

$$P(\dot{x}_f, x_f, v, t_f) \tag{1.13}$$

at the final time t_f with respect to the parameters v.

Using first order expansion about a solution of (1.10) and (1.11) the following is obtained:

$$\delta f = f_{\dot{x}} \delta \dot{x} + f_x \delta x + f_v \delta v + \epsilon = 0 \tag{1.14}$$

$$J_{\dot{x}} \delta \dot{x}(0) + J_x \delta x(0) + J_v \delta v + \epsilon_0 = 0 \tag{1.15}$$

where ϵ, ϵ_0 represent higher-order terms.

By dividing eqs. (1.14) and (1.15) component-wise by the elements of δv and taking the limit as $\delta v \to 0$, the trajectory sensitivity equations emerge:

$$f_{\dot{x}} \frac{\partial \dot{x}}{\partial v} + f_x \frac{\partial x}{\partial v} + f_v = 0 \tag{1.16}$$

$$J_{\dot{x}}\frac{\partial \dot{x}}{\partial v}(0) + J_x\frac{\partial x}{\partial v}(0) + J_v = 0 \tag{1.17}$$

Noting that (1.10) and (1.11) implicitly define P as a function of v, t_f:

$$\bar{P}(v, t_f) = P(\dot{x}_f, x_f, v, t_f) \tag{1.18}$$

we obtain:

$$\bar{P}_v(v, t_f) = P_{\dot{x}}(\dot{x}_v)_f + P_x(x_v)_f + P_v \tag{1.19}$$

$$\bar{P}_{t_f}(v, t_f) = P_{\dot{x}}\ddot{x}_f + P_x\dot{x}_f + P_t \tag{1.20}$$

where $(\dot{x}_v)_f$, $(x_v)_f$ are obtained by integrating the linear DAE system (1.16) with initial conditions (1.17).

1.3.2 Gradient Evaluation using Adjoint Equations

To the same problem of the previous section we introduce the adjoint variables:

$$\dot{\mu}^T = \lambda^T f_x; \quad \mu^T = \lambda^T f_{\dot{x}} \tag{1.21}$$

where $\mu, \dot{\mu}, \lambda \in \mathcal{R}^n$

Then using the total differential property along with (1.16) we get:

$$\begin{aligned}\frac{d(\mu^T x_v)}{dt} &= \mu^T\dot{x}_v + \dot{\mu}^T x_v \\ &= \lambda^T f_{\dot{x}}\dot{x}_v + \lambda^T f_x x_v \\ &= -\lambda^T f_v \end{aligned} \tag{1.22}$$

Integrating the above equation from t_0 to t_f we obtain:

$$\mu_f^T(x_v)_f - \mu_0^T(x_v)_0 + \int_{t_0}^{t_f}\lambda^T f_v\, dt = 0 \tag{1.23}$$

To derive the reduced derivative of P with respect to v we write the following expression using appropriate multipliers and eqs. (1.16), (1.17) and (1.23):

$$\begin{aligned}\bar{P}_v(v, t_f) = \quad & P_{\dot{x}}(\dot{x}_v)_f + P_x(x_v)_f + P_v + \\ & \omega_f^T\left\{f_{\dot{x}}^f(\dot{x}_v)_f + f_x^f(x_v)_f + f_v^f\right\} +\end{aligned}$$

$$\rho^T \{J_{\dot{x}}(\dot{x}_v)_0 + J_x(x_v)_0 + J_v\} +$$
$$\omega_0^T \left\{f_{\dot{x}}^0(\dot{x}_v)_0 + f_x^0(x_v)_0 + f_v^0\right\} +$$
$$\mu_f^T(x_v)_f - \mu_0^T(x_v)_0 + \int_{t_0}^{t_f} \lambda^T f_v \, dt \tag{1.24}$$

Now, we choose multipliers ρ, ω_0, ω_f, μ_f so that terms in $(\dot{x}_v)_0$, $(x_v)_0$, $(\dot{x}_v)_f$, $(x_v)_f$ vanish in (1.24):

$$\bar{P}_v(v, t_f) = P_v + \omega_f^T f_v^f + \rho^T J_v + \omega_0^T f_v^0 + \int_{t_0}^{t_f} \lambda f_v \, dt \tag{1.25}$$

and:

$$P_{\dot{x}} + \omega_f^T f_{\dot{x}}^f = 0 \tag{1.26}$$

$$P_x + \omega_f^T f_x^f + \mu_f^T = 0 \tag{1.27}$$

$$\rho^T J_{\dot{x}} + \omega_0^T f_{\dot{x}}^0 = 0 \tag{1.28}$$

$$\rho^T J_x + \omega_0^T f_x^0 - \mu_0^T = 0 \tag{1.29}$$

For the derivative with respect to the final time t_f we have the same expression (1.20) as before.

If the DAE system is of index zero, then $f_{\dot{x}}$ is nonsingular everywhere. From (1.10) and (1.11) the Jacobian matrix is:

$$M_0 = \begin{bmatrix} f_{\dot{x}}^0 & f_x^0 \\ J_{\dot{x}} & J_x \end{bmatrix} \tag{1.30}$$

If there is a solution of eqs. (1.10) and (1.11) at which M_0 is nonsingular, then this solution is *locally* unique for $\dot{x}_0$, x_0. In this case, this is *sufficient* for a *regular* solution of the DAEs through the initial point, thus (1.10) defines $\dot{x}(t)$, $x(t)$, $t \in [t_0, t_f]$, given v.

From (1.26):

$$\omega_f^T = -P_{\dot{x}}(f_{\dot{x}}^f)^{-1} \tag{1.31}$$

From (1.27):

$$\mu_f^T = -(P_x + \omega_f^T f_x^f) \tag{1.32}$$

From (1.21): $\lambda_f^T = \mu_f^T (f_{\dot{x}}^f)^{-1}$ and (1.21) defines $\lambda(t)$, $\mu(t)$, $t \in [t_0, t_f]$.

From (1.28), (1.29):

$$\begin{bmatrix} \omega_0^T & \rho^T \end{bmatrix} \begin{bmatrix} f_{\dot{x}}^0 & f_x^0 \\ J_{\dot{x}} & J_x \end{bmatrix} = \begin{bmatrix} 0 & \mu_0^T \end{bmatrix} \tag{1.33}$$

Since M_0 is nonsingular, (1.33) determines ω_0, ρ. Then $\bar{P}_v$ can be determined from (1.25).

If $P_{\dot{x}} = 0$ (1.20) gives $\bar{P}_{t_f}$. If $P_{\dot{x}} \neq 0$, differentiation of (1.10) yields:

$$f_{\dot{x}} \ddot{x} + f_x \dot{x} + f_t = 0 \tag{1.34}$$

and since $f_{\dot{x}}^f$ is nonsingular, this yields $\ddot{x}_f$, hence $\bar{P}_{t_f}$.

We can also consider a special class of DAE systems of index one such that $x = (\bar{x}, \bar{y})$ with $f_{\dot{\bar{y}}} = 0$, and $[f_{\dot{\bar{x}}}, f_{\bar{y}}]$ nonsingular. In this case, we define new variables $\dot{z} = y$, $z_0 = 0$. We then have an index-zero system in $(\bar{x}, z)$, and the above applies. Of course we do not need to integrate $\dot{z} = y$ to obtain z (note that, since y variables are counted as $\dot{z}$, the initial conditions must supply additional relations only for the *reduced* set $\bar{x}$).

For higher index, the above equations are not enough, and must be supplemented by differentiations of (1.10).

1.3.3 Comparison of Gradient Evaluation Methods in Control Parameterization

It was mentioned in the previous sections that sensitivity equations may be used directly to evaluate the gradients of a set of functions at a given set of times in the integration horizon. Any function gradients may be evaluated at an interior (or the final) point using expressions (1.19) and (1.20).

In general, the use of sensitivity equations involves forward (with respect to time) integrations. Furthermore, the values of the sensitivities can be sampled easily at *any* point in the trajectory and not just the final time t_f, a property which is important for evaluating the gradients of interior point constraints.

In the case of adjoints, for each interior point an adjoint backward integration starting from that point is required to evaluate derivatives of functions at that point.

As far as the cost of integrating the sensitivity equations is concerned, one may consider a simultaneous evaluation technique with the solution of the DAEs involved. In this approach the fact that the original DAE system (1.10) and the sensitivity system (1.16) have the same Jacobian can be exploited. Therefore, an integrator can be tailored to solve for the sensitivities at each state integration step simply by using the already available LU factorization (Golub and Van Loan, 1989) of this matrix. In particular a *backward difference formula* (BDF) integration code can be modified most effectively to this end (Dunker, 1984) (Leis and Kramer, 1985; Caracotsios and Stewart, 1985). The additional cost involved during the integration is that of an extra backsubstitution per parameter per step.

With adjoints one has to solve a DAE system with $2n$ equations, and also evaluate the integrals in eq. (1.23). With sensitivity equations, q DAE systems with n equations each have to be solved, and if $q > n$ the sensitivity approach requires more integrations than the approach based on adjoints. However, this is only a rough comparison as the evaluation of the integrals mentioned is not a trivial task. Moreover, this does not take into account the savings achieved with a simultaneous integration scheme for sensitivities as described above.

Rosen and Luus (1991) compared sensitivity and adjoint implementations as well as finite difference methods for gradient evaluations in optimal control and presented a thorough comparative study of these approaches supported by both theoretical considerations and computational experiments. They concluded that using the trajectory sensitivity and finite difference approaches can be computationally more efficient than the adjoint system approach, especially on computers with non-conventional architecture.

Finally, finite difference schemes might be employed for gradient evaluation, as noted by Kraft (1985). Such an approach is straightforward and conceptually simple. However, in the context of control parameterization methods, this would

require repeated integrations for perturbed parameter values. Furthermore, finite difference calculations are dependent on the selection of the parameter perturbations and a careful balance between truncation and cancellation error is required.

1.4 Collocation Techniques for Optimal Control Problems

All collocation techniques reduce an optimal control problem to a finite dimensional optimization problem by the discretization of both controls and state variables using some approximating functions involving a finite set of parameters. The values of the latter are in turn selected optimally with respect to the objective function. To motivate the discussion in this section, the following optimal control problem is considered:

PROBLEM TOCP

Objective function

$$\min_{u(\cdot),v,t_f} c(\dot{x}(t_f), x(t_f), u(t_f), v, t_f) \tag{1.35}$$

subject to:

ODE dynamic model

$$\dot{x} = f(x(t), u(t), v, t); \quad x(t_0) = \text{given}; \quad t \in [t_0, t_f] \tag{1.36}$$

$x(t)$: Differential state variables

$\dot{x}(t)$: Time derivatives of the differential state variables

$u(t)$: Control variables

v: Time invariant design parameters

t_f: Final time

It is assumed that a set of initial conditions $x(0)$ is given.

1.4.1 Discretization using collocation

Control parameterization restricts the discretization to the control variables, whereas collocation requires the discretization of the state variables as well. With reference to the model (TOCP), given in eqs. (1.35)–(1.36), the controls u, and variables x are approximated by the following functions:

$$u(t) = \theta^u(t, w^u) \tag{1.37}$$

$$x(t) = \theta^x(t, w^x) \tag{1.38}$$

where w^u and w^x are finite vectors of parameters.

In order to derive a discretized version of (TOCP), it is necessary to introduce notation for describing the approximation of variables over a set of collocation points within a number of finite elements.

The number of finite elements is taken to be NS. The number of collocation points within each element is K and is taken to be the same for every element. To simplify notation a composite node index $[i, j]$ is defined as the time corresponding to node j within element i as shown in Figure 1.1. In terms of the global time axis, this index denotes the true time-point given by:

$$[i, j] = (i - 1)(K + 1) + j \tag{1.39}$$

The length of each element will be denoted by h_i. It is convenient to map each finite element into the domain $\tau \in [0, 1]$ using the following transformation:

$$t = t_0^{[i]} + \tau h_i; \quad i = 1, 2, \ldots, NS; \quad \tau \in [0, 1] \tag{1.40}$$

where

$$h_i = t_0^{[i+1]} - t_0^{[i]} \tag{1.41}$$

With the above transformation all time derivatives when mapped to the local time τ are given by:

$$\frac{d(\cdot)}{dt} = \frac{1}{h_i}\frac{d(\cdot)}{d\tau} \tag{1.42}$$

Since the polynomial of degree m that collocates a given set of $(m + 1)$ distinct points is unique, one can conclude that it does not matter what type of

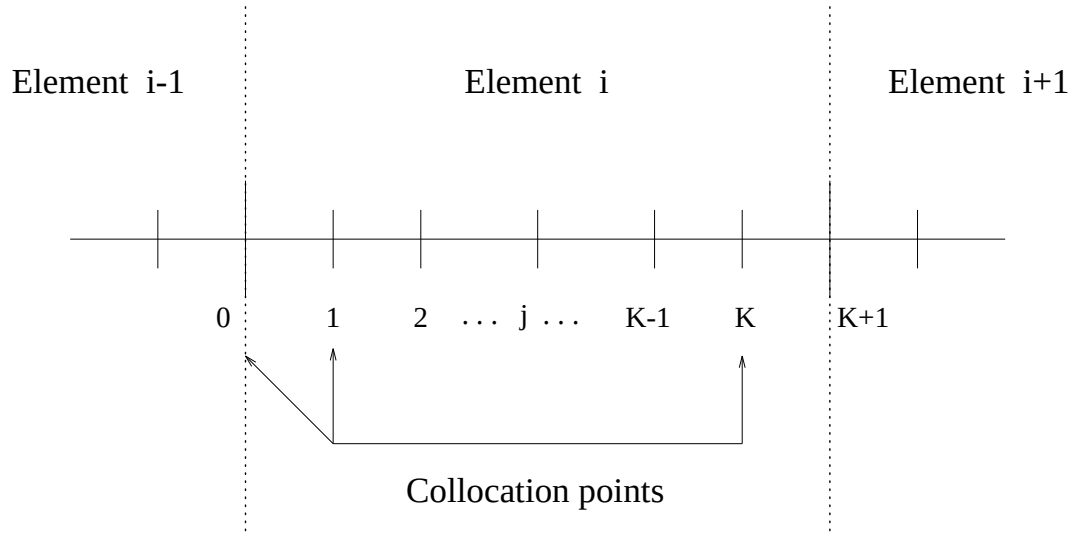

Figure 1.1. Discretization of time in finite elements

polynomial is used to collocate points within each finite element. However, for practical reasons, it is advantageous to choose Lagrange polynomials as these have as parameters the values of the approximated functions at the collocation points (Ferziger, 1981). The definitions of the Lagrange basis polynomials used are:

$$\phi_j^{[i]}(t) = \prod_{\substack{k=0 \\ k\neq j}}^{K} \frac{(t - t_k^{[i]})}{(t_j^{[i]} - t_k^{[i]})}; \quad j = 0, 1, \ldots, K \tag{1.43}$$

$$\psi_j^{[i]}(t) = \prod_{\substack{k=0 \\ k\neq j}}^{K} \frac{(t - t_k^{[i]})}{(t_j^{[i]} - t_k^{[i]})}; \quad j = 1, 2, \ldots, K \tag{1.44}$$

The property of the Lagrange basis polynomials can be expressed as:

$$\phi_j^{[i]}(t_k^{[i]}) = \phi_{kj}^{[i]} = \delta_{kj}; \quad j, k = 0, 1, \ldots, K \tag{1.45}$$

$$\psi_j^{[i]}(t_k^{[i]}) = \psi_{kj}^{[i]} = \delta_{kj}; \quad j, k = 1, 2, \ldots, K \tag{1.46}$$

where δ_{jk} are Kronecker deltas.

If the collocation nodes are located at the same normalized times τ_j, $j = 0, 1, \ldots, K$ within each element, the superscript $[i]$ on the basis functions can be omitted:

$$\left.\begin{array}{ll} \phi_j(\tau), & j = 0, 1, \ldots, K \\ \psi_j(\tau), & j = 1, 2, \ldots, K \end{array}\right\} \quad \tau \in [0, 1] \tag{1.47}$$

The polynomial approximation for the state variables and the control variables can then be expressed as:

$$x^{[i]}(t) = \sum_{k=0}^{K} \phi_k(t) x_k^{[i]}; \quad t \in [t_0^{[i]}, t_0^{[i+1]}) \tag{1.48}$$

$$u^{[i]}(t) = \sum_{k=1}^{K} \psi_k(t) u_k^{[i]}; \quad t \in [t_0^{[i]}, t_0^{[i+1]}) \tag{1.49}$$

where we define $t_0^{[1]} = t_0$ and $t_0^{[NS+1]} = t_f$.

By using eqs. (1.42), (1.43), and (1.48) we obtain the following expression for the time derivatives of $x(t)$ at a collocation point $[i, j]$:

$$\dot{x}_j^{[i]}(t) = \frac{1}{h_i} \sum_{k=0}^{K} \dot{\phi}_{jk} x_k^{[i]}; \quad t \in [t_0^{[i]}, t_0^{[i+1]}) \tag{1.50}$$

where we have defined $\dot{\phi}_{jk} = \frac{d\phi_k}{d\tau}(\tau_j)$.

We now enforce eq. (1.36) at collocation points τ_j, $j = 1, 2, \ldots, K$. By substituting in the polynomial approximations (1.48), (1.49) and making use of (1.45), (1.46), and (1.50) we obtain:

$$f(x_j^{[i]}, u_j^{[i]}, v, t_j^{[i]}) = \frac{1}{h_i} \sum_{k=0}^{K} \dot{\phi}_{jk} x_k^{[i]} \tag{1.51}$$

$$j = 1, 2, \ldots, K; \quad i = 1, 2, \ldots, NS$$

For the first element, given initial conditions $x_0^{[1]}$, the above equations determine $x_j^{[1]}$, $j = 1, 2, \ldots, K$.

If the K-th collocation point coincides with the element end-point (*i.e.* $\tau_k = 1$ $\forall\, i = 1, 2, \ldots, NS$) we could set:

$$x_0^{[i+1]} = x_K^{[i]}; \quad i = 1, 2, \ldots, NS - 1 \tag{1.52}$$

If the K-th collocation point is *not* the element end-point (*e.g.* as shown in Figure 1.1) we could extrapolate for $\tau = 1$ to obtain:

$$x_0^{[i+1]} - \sum_{k=0}^{K} \phi_k(1) x_k^{[i]} = 0; \quad i = 1, 2, \ldots, NS - 1 \tag{1.53}$$

Finally, we note that the block structure of eq. (1.51) is:

$$M_{jk}^{[i]} = \dot{\phi}_{jk} I; \quad k = 1, 2, \ldots, j-1, j+1, \ldots, K; \quad j = 1, 2, \ldots, K \tag{1.54}$$

$$M_{jj}^{[i]} = \dot{\phi}_{jj} I - \frac{\partial f}{\partial x}(x_j^{[i]}, u_j^{[i]}, v, t_j^{[i]}); \quad j = 1, 2, \ldots, K \tag{1.55}$$

Using the derived equations, we may obtain a discretized version of (TOCP):

PROBLEM DTOCP

Objective function

$$\min_{u_j^{[i]}, v, h_i} c(\dot{x}(t_0^{[NS+1]})), x(t_0^{[NS+1]}), u(t_0^{[NS+1]}), v, t_0^{[NS+1]}) \tag{1.56}$$

subject to:

Discretized DAE dynamic model

$$f(x_j^{[i]}, u_j^{[i]}, v, t_j^{[i]}) = \frac{1}{h_i} \sum_{k=0}^{K} \dot{\phi}_{jk} x_k^{[i]} \tag{1.51}$$

$$j = 1, 2, \ldots, K; \quad i = 1, 2, \ldots, NS$$

Profile continuity / element initial conditions

$$x_0^{[1]} = \text{given} \tag{1.57}$$

$$x_0^{[i+1]} - \sum_{k=0}^{K} \phi_k(1) x_k^{[i]} = 0; \quad i = 1, 2, \ldots, NS-1 \tag{1.53}$$

Element node definitions

$$t_j^{[1]} = t_0 + h_1 \tau_j; \quad j = 0, 1, \ldots, K \tag{1.58}$$

$$t_j^{[i]} = t_0 + \sum_{l=1}^{i-1} h_l + h_i \tau_j; \quad j = 0, 1, \ldots, k; \quad i = 2, 3, \ldots, NS \tag{1.59}$$

$$t_0^{[NS+1]} = t_0 + \sum_{l=1}^{NS} h_l \tag{1.60}$$

Extrapolation of states and controls to final time

$$x_0^{[NS+1]} - \sum_{k=0}^{K} \phi_k(1) x_k^{[NS]} = 0$$

$$u_0^{[NS+1]} - \sum_{k=1}^{K} \psi_k(1) u_k^{[NS]} = 0$$

1.4.2 Integration error and element number control

The basic formulation of (DTOCP) presented above did not consider the error incurred by the discretization. It should be noted that the element sizes, h_i, $i = 1, 2, \ldots, NS$ are decision variables within the NLP formulation, and as such can be chosen to satisfy as far as possible the constraints describing the model. Taking advantage of this property, additional constraints may be added to the discretized problem to control the error of integration. In the recent literature Logsdon and Biegler (1989) and Vasantharajan and Biegler (1990) presented two types of schemes for controlling the error.

The first scheme is termed *equidistribution*, as the error estimates over each element are required to be all equal. The optimization algorithm will attempt to find a feasible solution for these additional constraints by adjusting the sizes of a given number of elements. At the optimal solution these errors are examined and, if any are found to exceed the required accuracy, more elements are introduced and the procedure is repeated.

The second scheme is more flexible and lends itself to more practical algorithms. According to this, the derived error estimates are set to be less than the required tolerance, without requiring that all errors be equal for all elements. The constraints are included directly within the optimization: any infeasibility due to the inability of the optimization algorithm to satisfy an error constraint points directly to an insufficient number of elements. The number is then increased and the solution proceeds with a new NLP from the beginning, since the addition of finite elements alters the structure of the discretized problem.

Both of the above approaches require reliable estimates of the approximation errors. It can be proved (Russell and Christiansen, 1978; Ascher, 1986) that the error of collocation can be estimated as:

$$\|e\|^{[i]} = C\left\|R^{[i]}(\tau_{\mathrm{nc}})\right\| + O(\Delta h^{K+2}) \tag{1.61}$$

where $R^{[i]}(\tau_{\mathrm{nc}})$ is the residual of functions f evaluated at some non-collocation point, $\tau_{\mathrm{nc}} \in [0, 1]$ for each element i, and C is a constant depending only on τ_{nc} and the order of the collocating polynomials.

Using an infinity norm in the above estimate, error control may be achieved by introducing the following constraints in the optimization:

$$-\epsilon \leq CR_l^{[i]}(\tau_{\rm nc}) \leq \epsilon, \quad i = 1, 2, \ldots, NS; \quad l = 1, 2, \ldots, n \tag{1.62}$$

where $R_l^{[i]}$ refers to the residual of the l-th system equation for element i.

Vasantharajan and Biegler in a recent paper (1990) have used a linear programming (LP) based algorithm to estimate a sufficient number of elements. The technique is particularly suitable for the solution of the nonlinear programming problem arising from the application of the collocation method using a sequential quadratic programming (SQP) algorithm. It is based on the observation that the quadratic programming (QP) subproblem will be feasible if the linearization of the constraints has a non-empty feasible region. They define and solve the following linear programming (LP) problem:

LP1

$$\min_{\delta, u_j^i, v, h_i} \delta \tag{1.63}$$

subject to:

$$\left\{ \text{Linearized constraints and bounds} \right\}$$

$$-\delta \leq C\bar{R}^{[i]}(\tau_{\rm nc}) \leq \delta, \quad i = 1, 2, \ldots, NS \tag{1.64}$$

$$\delta \geq 0 \tag{1.65}$$

Here (1.64) represents a linearization of the error estimates (1.61). (LP1) minimizes an index δ which becomes equal to the maximum error of the discretization for the linearized problem. If this is greater than the specified tolerance ϵ, then those elements for which the linearized error constraints are binding in (LP1) are further subdivided into smaller elements.

This procedure is invoked initially to find a sufficiently large starting number of elements. Thereafter, the procedure is invoked again in the case of infeasible

QPs. In this way, it is always possible to determine a sufficiently large number of elements so that the error constraints will be satisfied at the solution. This, however, is not always the minimum number of elements required.

We note that these ideas can be extended to ensure feasibility of the NLP. The same error enforcing criterion as that in (1.62) can be adopted for all elements except the last one. The error criterion for this element is modified to:

$$-(\epsilon+\delta) \leq CR^{NS}(\tau_{\mathrm{nc}}) \leq \epsilon+\delta \tag{1.66}$$

also the objective function is augmented by a penalty term:

$$\min_{(\cdots)} \phi(\cdot) + M\delta \tag{1.67}$$

The selection of an appropriate value for the penalty coefficient M may of course be problematic in practice, but setting this aside it can be observed that the proposed scheme is always feasible with respect to the error constraints, as δ will become as small as possible to satisfy the error constraints for the last element. Also, as a consequence of this scheme, all other elements will be chosen so that they satisfy their error constraints exactly, *i.e.* error equidistribution is automatically achieved for them.

Now, δ can serve as a measure of the sufficiency of the number of elements. If, after a number of consecutive QP iterations of the SQP optimization, the optimizer does not reduce the value of δ to within the order of magnitude of ϵ, while equidistribution is achieved for all other elements, it may be advantageous to interrupt the optimization and subdivide the last element into two or more smaller elements. The optimization may then be restarted from that point.

1.5 Comparison of Collocation and Control Parameterization

The main attribute of collocation that makes it appealing for optimal control algorithms is that it reduces the objective function (minimization) and the violations of the constraints simultaneously, thus potentially reducing the total

computational cost. Another advantage is that it poses the problem as an independent NLP model, therefore facilitating the incorporation of general constraints in the formulation. Direct solution of the resulting NLP can exploit the banded structure of its coefficient matrix (Wright, 1991).

CP on the other hand requires the integration of the DAEs for constraint and objective values and the integration of variational equations (adjoint or sensitivity) in order to obtain the corresponding gradients. Integrations can indeed be very expensive, and this is claimed by the supporters of collocation to be the major drawback of CP methods. On the other hand, the advantage of the CP approach is that it decouples the DAEs from the optimization stage. Each iteration is a feasible solution to the DAEs—hence functions are more likely to be well behaved. Moreover, the stiffness of the DAEs is taken care of by the integrator, and no longer implies ill-conditioning in the resulting NLPs. Of course, as noted by Vasantharajan and Biegler (1990), CP is only as reliable as the DAE solver employed, but such an argument would seem to favour the CP approach as currently available state-of-the-art integrators are quite robust for stiff, large and in any way complex DAE models.

From a more fundamental viewpoint, the two methodologies are very similar as both discretize the control variables, and CP solves the DAEs using an integrator which essentially also discretizes the state variables. A closer look at the corresponding discretization schemes is therefore necessary in order to arrive at some conclusions.

We consider the following ODE system:

$$\dot{x}(t) = f(x(t), u(t), v, t); \quad x(0) = x_0 \text{ given} \tag{1.68}$$

A K-th order *linear multistep method* (LMM), such as a BDF method, approximates $\dot{x}(t)$ at time step i (*i.e.* at time $t^{[i]}$) by the following:

$$\dot{x}^{[i]} = \frac{1}{\beta h_i} \sum_{k=0}^{K} \alpha_k x^{[i-k]} \tag{1.69}$$

which for $\alpha_0 = 0$ is an explicit method and for $\alpha_0 \neq 0$ is an implicit method.

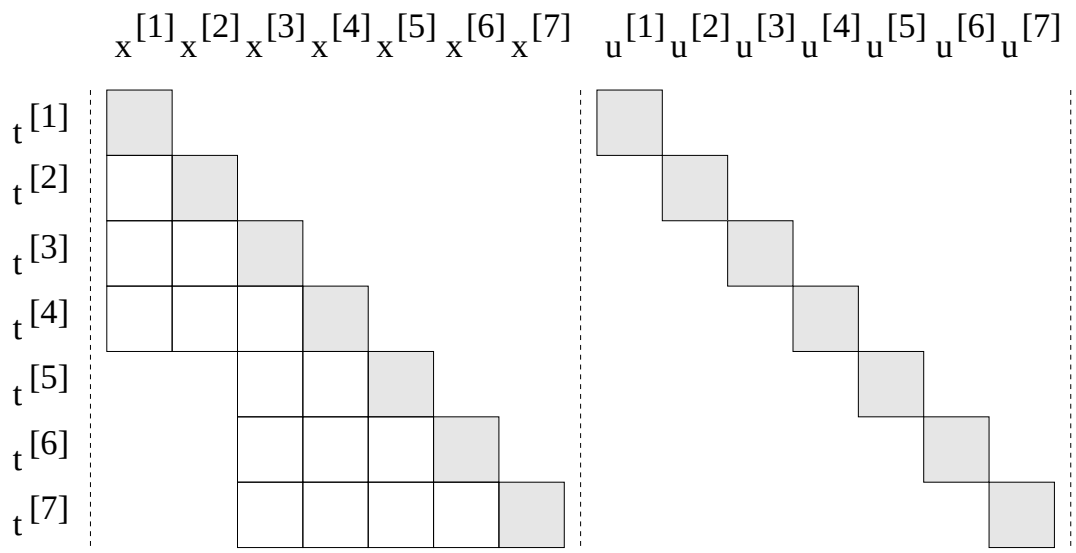

Figure 1.2. Discretization using variable order linear multistep methods

Using (1.68) and (1.69) we obtain:

$$\frac{1}{\beta h_i}\sum_{k=0}^{K_i}\alpha_k x^{[i-k]} = f(x^{[i]}, u^{[i]}, v, t^{[i]}); \quad i = 1, 2, \ldots, NS \tag{1.70}$$

where K_i is the order of the i-th integration step.

The structure of the discretized equations (1.70) is shown in Figure 1.2. The blocks in that matrix are as follows:

$$M_{ii} = \frac{\alpha_0}{\beta h_i} I - \frac{\partial f}{\partial x}(x^{[i]}, u^{[i]}, v, t^{[i]}); \quad i = 1, 2, \ldots, NS \tag{1.71}$$

$$M_{ii'} = \frac{\alpha_{i-i'}}{\beta h_i} I; \quad i - K_i \le i' < i; \quad i = 1, 2, \ldots, NS \tag{1.72}$$

$$M_{ii'} = 0; \quad \{i' < i - K_i\} \bigcup \{i' > i\}; \quad i = 1, 2, \ldots, NS \tag{1.73}$$

A K-stage *Runge-Kutta* (RK) method over step i is defined as follows:

$$t_j^{[i]} = t_0^{[i]} + c_j h_i \tag{1.74}$$

$$x_j^{[i]} = \tilde{x}_0^{[i]} + h_i \sum_{k=1}^{K} a_{jk}\dot{x}_k^{[i]} \tag{1.75}$$

$$\dot{x}_j^{[i]} = f(x_j^{[i]}, u_j^{[i]}, v, t_j^{[i]}) \tag{1.76}$$

$$j = 1, 2, \ldots, K; \quad i = 1, 2, \ldots, NS$$

Then the new value at the end of the element is:

$$\tilde{x}_K^{[i]} = \tilde{x}_0^{[i]} + h_i \sum_{k=1}^{K} b_k \dot{x}_k^{[i]} \tag{1.77}$$

where $\tilde{x}_0^{[i]}$ is the initial value for element i.

The scheme is explicit if the matrix A (of coefficients a_{jk}) is strictly lower-triangular, semi-implicit if A is lower-triangular, and fully implicit if A is full.

Examples of these cases for a 3-stage method over two consecutive time steps are shown in Figures 1.3–1.5.

The explicit case is shown in Figure 1.3. The blocks in the discretized Jacobian *within a single time step* are given by:

$$M_{jk}^{[i]} = -h_i \frac{\partial f}{\partial x}(x_k^{[i]}, u_k^{[i]}, v, t_k^{[i]}) a_{jk};$$

$$k = 1, 2, \ldots, j-1; \quad j = 1, 2, \ldots, K \tag{1.78}$$

$$M_{jj}^{[i]} = I; \quad j = 1, 2, \ldots, K \tag{1.79}$$

$$M_{jk}^{[i]} = 0; \quad k = j+1, j+2, \ldots, K; \quad j = 1, 2, \ldots, K \tag{1.80}$$

It is noted that the last block row corresponding to each time is the weighted summation yielding the value of the state variables x at the end of the time-step.

The semi-implicit case is shown in Figure 1.4. The blocks in the discretized Jacobian *within a single time step* are given by:

$$M_{jk}^{[i]} = -h_i \frac{\partial f}{\partial x}(x_k^{[i]}, u_k^{[i]}, v, t_k^{[i]}) a_{jk};$$

$$k = 1, 2, \ldots, j-1; \quad j = 1, 2, \ldots, K \tag{1.81}$$

$$M_{jj}^{[i]} = I - h_i \frac{\partial f}{\partial x}(x_j^{[i]}, u_j^{[i]}, v, t_j^{[i]}) a_{jk}; \quad j = 1, 2, \ldots, K \tag{1.82}$$

$$M_{jk}^{[i]} = 0; \quad k = j+1, j+2, \ldots, K; \quad j = 1, 2, \ldots, K \tag{1.83}$$

The fully-implicit case is shown in Figure 1.5. The blocks in the discretized Jacobian *within a single time step* are given by:

$$M_{jk}^{[i]} = -h_i \frac{\partial f}{\partial x}(x_k^{[i]}, u_k^{[i]}, v, t_k^{[i]}) a_{jk};$$

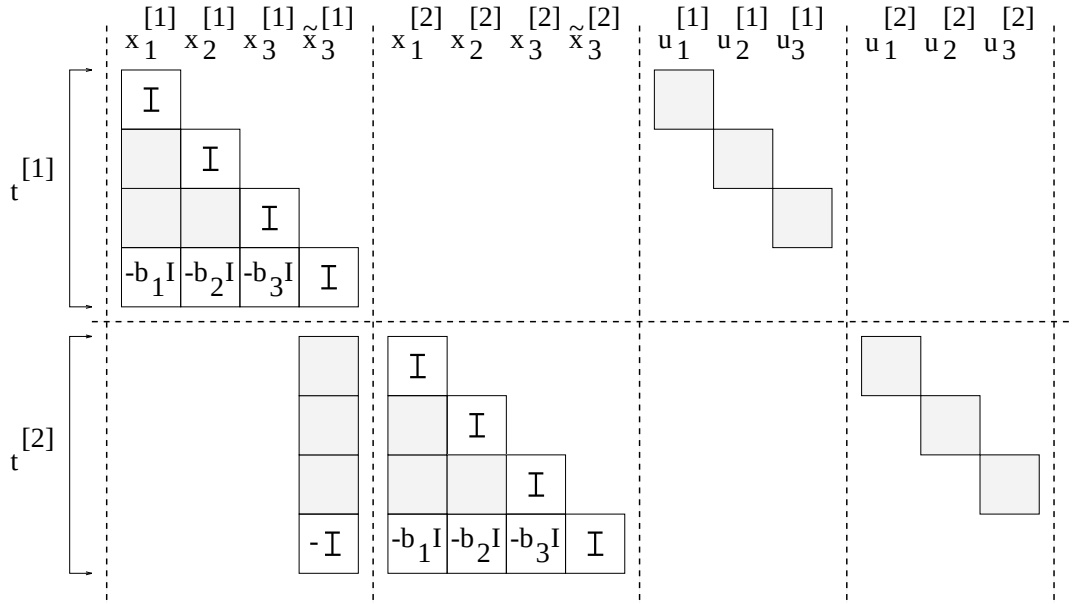

Figure 1.3. Explicit Runge-Kutta scheme

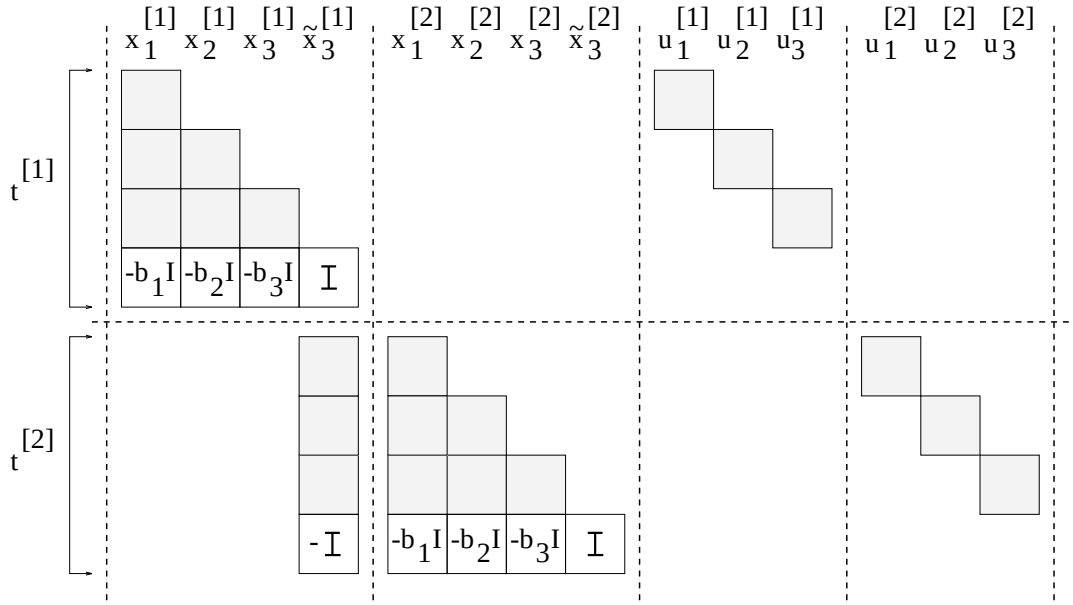

Figure 1.4. Semi-implicit Runge-Kutta scheme

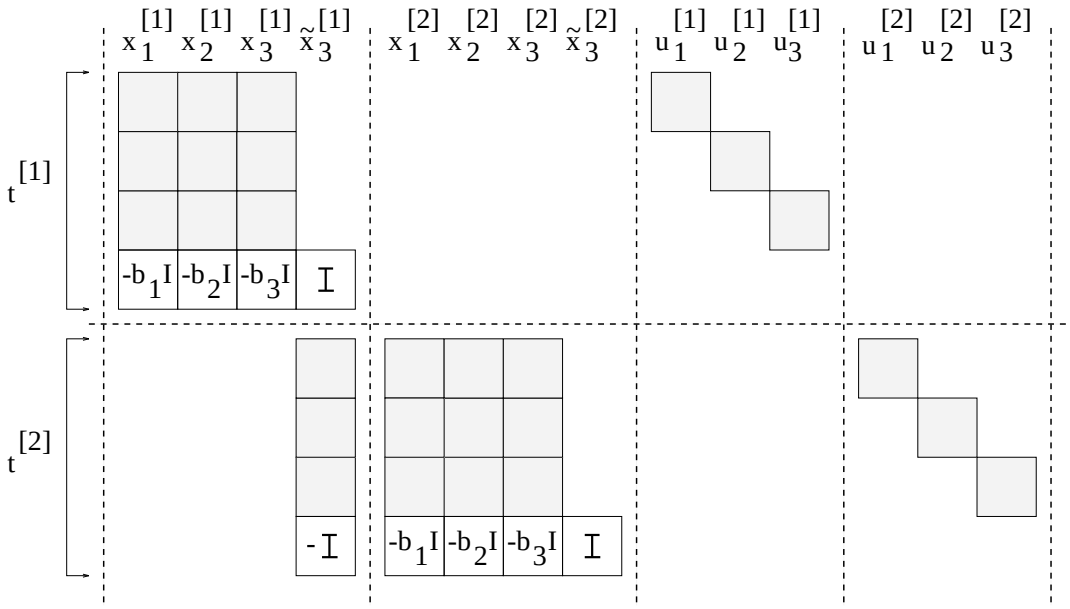

Figure 1.5. Fully implicit Runge-Kutta scheme

$$k = 1, 2, \ldots, j-1, j+1, \ldots, K; \quad j = 1, 2, \ldots, K \tag{1.84}$$

$$M_{jj}^{[i]} = I - h_i \frac{\partial f}{\partial x}(x_j^{[i]}, u_j^{[i]}, v, t_j^{[i]}) a_{jk}; \quad j = 1, 2, \ldots, K \tag{1.85}$$

Some observations can now be made:

1. The application of both LMM and RK integration methods to optimal control problems leads to large sets of discretized equations.

2. All discretized equation sets have a *bordered* block triangular structure—with the border corresponding to discretization of control variables.

3. If the variables involved in the border are given, we can solve the equations for the remaining variables. Because of the block-triangular structure, the system decomposes. For all implicit methods other than the fully-implicit Runge-Kutta (FI-RK) the blocks are of size $n \times n$. However, for the case of the FI-RK, the blocks are of size $(nK) \times (nK)$.

To show that collocation is equivalent to a RK scheme we define:

$$\dot{\Phi} = [\dot{\phi}_{jk}] \in \mathcal{R}^{K \times K}; j, k = 1, 2, \ldots, K \tag{1.86}$$

We define next a state variable matrix in column vector format:

$$X^{[i]} = \left[\begin{array}{cccccc} x_1^{[i]} & \dots & x_j^{[i]} & \dots & x_K^{[i]} \end{array} \right]; \quad x_j^{[i]} \in \mathcal{R}^n; \quad j = 1, 2, \dots, K \tag{1.87}$$

and also define it in row vector format:

$$X^{[i]} = \left[\begin{array}{c} \gamma_1^{[i]} \\ \vdots \\ \gamma_l^{[i]} \\ \vdots \\ \gamma_n^{[i]} \end{array} \right]; \quad (\gamma_j^{[i]})^T \in \mathcal{R}^K; \quad l = 1, 2, \dots, n \tag{1.88}$$

for all elements $i = 1, 2, \dots, NS$.

Similarly we define a state variable derivative matrix in column format:

$$\dot{X}^{[i]} = \left[\begin{array}{cccccc} \dot{x}_1^{[i]} & \dots & \dot{x}_j^{[i]} & \dots & \dot{x}_K^{[i]} \end{array} \right]; \quad \dot{x}_j^{[i]} \in \mathcal{R}^n; \quad j = 1, 2, \dots, K \tag{1.89}$$

and also define it in row vector format:

$$\dot{X}^{[i]} = \left[\begin{array}{c} \dot{\gamma}_1^{[i]} \\ \vdots \\ \dot{\gamma}_l^{[i]} \\ \vdots \\ \dot{\gamma}_n^{[i]} \end{array} \right]; \quad (\dot{\gamma}_j^{[i]})^T \in \mathcal{R}^K; \quad l = 1, 2, \dots, n \tag{1.90}$$

for all elements $i = 1, 2, \dots, NS$.

We may write eq. (1.50) as:

$$(\dot{\gamma}_l^{[i]})^T = \frac{1}{h_i} \left\{ \left[\begin{array}{c} \dot{\phi}_{10} \\ \vdots \\ \dot{\phi}_{j0} \\ \vdots \\ \dot{\phi}_{K0} \end{array} \right] (x_0^{[i]})_l + \dot{\Phi} (\gamma_l^{[i]})^T \right\} \tag{1.91}$$

$$l = 1, 2, \dots, n$$

where $x_0^{[i]} \in \mathcal{R}^n$ are the initial values of the state variables for element i. From

eq. (1.91) we can solve for $(\gamma_l^{[i]})^T$ to obtain:

$$(\gamma_l^{[i]})^T = \dot{\Phi}^{-1} \left\{ (\dot{\gamma}_l^{[i]})^T - \frac{1}{h_i} \begin{bmatrix} \dot{\phi}_{10} \\ \vdots \\ \dot{\phi}_{j0} \\ \vdots \\ \dot{\phi}_{K0} \end{bmatrix} (x_0^{[i]})_l \right\} \tag{1.92}$$

$$l = 1, 2, \ldots, n$$

From eq. (1.92) we can isolate the j-th element to yield:

$$(\gamma_l^{[i]})_j^T = \sum_{k=1}^{K} \left[\dot{\Phi}^{-1}\right]_{jk} \left\{ (\dot{\gamma}_l^{[i]})_k^T - \frac{1}{h_i} \dot{\phi}_{j0} (x_0^{[i]})_l \right\} \tag{1.93}$$

$$j = 1, 2, \ldots, K; \quad l = 1, 2, \ldots, n$$

but by the definition of matrices X, $\dot{X}$, we have:

$$(\gamma_l^{[i]})_j^T = (x_j^{[i]})_l \quad \text{and} \quad (\dot{\gamma}_l^{[i]})_k^T = (\dot{x}_k^{[i]})_l \tag{1.94}$$

which when substituted in eq. (1.93) yield:

$$(x_j^{[i]})_l = \sum_{k=1}^{K} \left[\dot{\Phi}^{-1}\right]_{jk} \left\{ (\dot{x}_k^{[i]})_l - \frac{1}{h_i} \dot{\phi}_{j0} (x_0^{[i]})_l \right\} \tag{1.95}$$

$$j = 1, 2, \ldots, K; \quad l = 1, 2, \ldots, n$$

Returning to vector expressions for $x_j^{[i]}$ and $\dot{x}_j^{[i]}$ we obtain:

$$x_j^{[i]} = \sum_{k=1}^{K} \left[\dot{\Phi}^{-1}\right]_{jk} \left\{ \dot{x}_k^{[i]} - \frac{1}{h_i} \dot{\phi}_{j0} x_0^{[i]} \right\} \tag{1.96}$$

$$j = 1, 2, \ldots, K$$

Then, eq. (1.96) is in the form of (1.75), while eq. (1.77) is just a repeat of the last row of eq. (1.96) for which $j = K$. If we wish to add an extra node and use extrapolation to determine $x_{K+1}^{[i]} = x_0^{[i+1]}$, then eq. (1.53) is the equivalent of eq. (1.77), with similar transformation, using eq. (1.96) to express $x_{K+1}^{[i]}$ in terms of $\dot{x}_j^{[i]}$, $j = 1, 2, \ldots, K$.

Of course, $\dot{\Phi}^{-1}$ is a full matrix, so eq. (1.96) corresponds to a fully implicit RK scheme.

In fact, extensive evidence in the ODE literature shows that RK methods are relatively inefficient when compared to other methods (such as BDF methods) (*e.g.* Feng *et al.*, 1984). In any case, the real difference between collocation and control parameterization is the treatment of discretized equations within the optimization algorithm. If equations are to be satisfied exactly during optimization (as for example advocated by Logsdon and Biegler, 1992), then it seems better to use an integrator (BDF or RK) to ensure that the correct number of steps are taken and that error criteria are satisfied.

1.6 Conclusions

In this chapter a general overview of existing techniques for the numerical solution of optimal control problems was presented.

The two general classes of techniques proposed in the literature are those that use the necessary conditions for optimality (either directly, thus solving a TPBVP, or indirectly, thus resorting to iterative schemes on approximations for the TPBVP), and those that use variable discretizations (either for the controls only, or for both the controls and the state variables).

Techniques resulting in TPBVPs may not be suitable for large problems as the solution of TPBVPs is still a very difficult numerical problem. On the other hand, discretization techniques have the advantage of transforming the dynamic optimization problems to finite dimensional NLPs which can be tackled by well-established large-scale nonlinear programming methods. Furthermore, the inclusion of various types of constraints is straightforward and the only special concern is how to provide their corresponding gradients to the optimization package.

Whether discretization should be applied to the control vector only, or to both the states and the controls is an issue which has been investigated by many researchers. The second option, although more appealing since it poses the problem as a simple NLP without the need for integrations of DAEs, can prove to be impractical due to the large number of equality constraints resulting from the discretization of the DAEs of the dynamic model. One can, of course, exploit the banded structure of these equations to reduce the problem to the space of the parameters discretizing the control variables, but then essentially this is equivalent to a simplified integration scheme. Another shortcoming of complete discretization is the requirement for a fixed number of discretization elements which are used in the state variable approximation. This is something that standard integration packages can adjust automatically so as to satisfy the DAEs within the prespecified tolerance. Thus no error constraints need be present in the optimization phase, and no ill-conditioning due to stiffness will be present in the resulting NLP.

In view of the last observation, it was decided to base this work on a control parameterization scheme. Furthermore, the use of sensitivity equations for gradient evaluation was preferred since, as noted in a previous section, their implementation is simpler than that of adjoints, and more importantly, the incorporation of interior point constraints in the formulations does not require multiple backward integrations as the use of adjoints does. Instead, a single forward integration suffices to evaluate all the required gradients.

CHAPTER 2

Formulation and Algorithm Development

As described in the previous chapter, the solution of differential-algebraic optimization problems may be tackled by numerical techniques that convert the problem into a nonlinear programming problem. The two general approaches that have appeared in the chemical engineering literature in the recent years are based either on control parameterization, or on parameterization of both control and state variables using collocation. Of the two, the first is selected and applied using a sequential optimization scheme consisting of two stages: the first stage involves state integrations to obtain function values, or state and sensitivity equation integrations to obtain function gradients. These are then supplied to the second stage which is optimization. The choices of individual components of the proposed formulation and solution procedure are explained and justified in this chapter. A new contribution to optimal control algorithms is the consideration of discrete events that cause differential states to jump so as to satisfy defined conditions. The DAE models considered are of index one; complications arising from higher index problems are not considered.

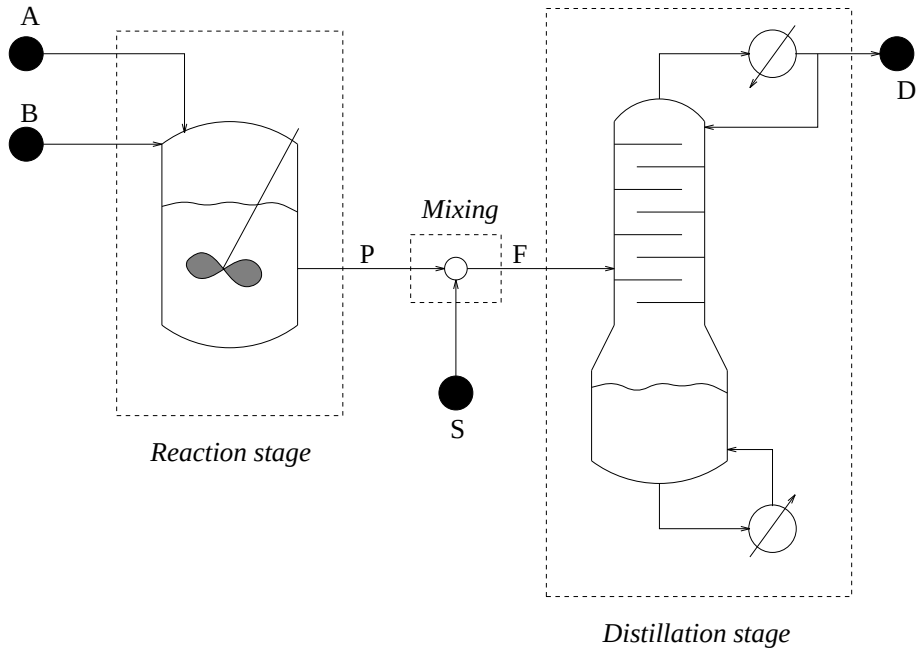

Figure 2.1. Two-stage process problem

2.1 Introduction and Model Definition

The overview in Chapter 1 presented optimal control problems for systems described by DAE formulations. The analysis that followed focused on *single-stage* systems. By "stage" we mean a well defined independent system of DAEs whose dynamics are active during a period $t_{i-1} \leq t < t_i$, $i = 1, 2, \ldots, NS$, within a time horizon $t_0 \leq t \leq t_f$. Systems with NS stages therefore have $t_f = t_{NS}$.

To motivate further discussion and to facilitate the presentation of terminology we consider a simple two-stage chemical engineering problem. In this, two feedstocks, A and B, are fed to a batch stirred tank reactor (CSTR) where reaction takes place during the period $0 \leq t < t_1$. On completion of the reaction, the product P is mixed instantaneously with some material S, and the mixture is fed to a distillation column which operates for a period $t_1 \leq t \leq t_2$ to produce D. The process is shown schematically in Figure 2.1.

In Figure 2.1 the two stages are clearly the reaction and the distillation. Each stage i is described by a different DAE system of the form:

$$f_i(\dot{x}_i(t), x_i(t), y_i(t), u_i(t), v_i, t) = 0; \quad t_{i-1} \leq t < t_i; \quad i = 1, 2, \ldots, NS \qquad (2.1)$$

where $x_i(t)$ are differential state variables, $\dot{x}_i(t)$ are their time derivatives, $y_i(t)$ are algebraic state variables, $u_i(t)$ are the time varying controls, v_i are the time-invariant design variables, and t is the time.

Although the mixing process could be viewed as a third stage, it is different to the other two in that it is assumed to take place instantaneously. Hence it is described by a set of algebraic material and energy balances that define the state of the feed to the distillation column in terms of the state of material leaving the reactor. In general, sets of such conditions may involve not only the differential variables $x_i(t)$ but also any of their time derivatives $\dot{x}_i(t)$ and the algebraic variables $y_i(t)$. A general form of *junction conditions* between stages i and $i+1$ is given by:

$$\begin{aligned} &J_i(\dot{x}_{i+1}(t_i^+), x_{i+1}(t_i^+), y_{i+1}(t_i^+), u_{i+1}(t_i^+), v_{i+1}, \\ &\dot{x}_i(t_i^-), x_i(t_i^-), y_i(t_i^-), u_i(t_i^-), v_i, t_i) = 0; \end{aligned} \tag{2.2}$$

$$i = 1, 2, \ldots, NS-1$$

The *initial condition* for the first stage can also be specified in implicit form:

$$J_0(\dot{x}_1(t_0), x_1(t_0), y_1(t_0), u_1(t_0), v_1, t_0) = 0 \tag{2.3}$$

The general junction conditions, including the initial condition at t_0 above, have to be considered simultaneously with the DAEs of the associated stage, *i.e.* the initialization of all variables involved will come from the solution of eqs. (2.1) and (2.2) (or (2.3)).

A different type of constraints are *path constraints*. These are constraints that have to be satisfied over the entire duration of the stage. They can be equalities and inequalities:

$$c_i^{[eq]}(\dot{x}_i(t), x_i(t), y_i(t), u_i(t), v_i, t) = 0 \tag{2.4}$$

$$c_i^{[in]}(\dot{x}_i(t), x_i(t), y_i(t), u_i(t), v_i, t) \leq 0 \tag{2.5}$$

$$t_{i-1} \leq t < t_i; \quad i = 1, 2, \ldots, NS$$

On the other hand, constraints imposed at specific times are termed *point constraints*. For convenience, it is assumed without loss of generality that such

constraints are enforced only at the boundaries of different stages, including the initial and final times, t_0, t_f:

$$r_i^{[eq]}(\dot{x}_i(t_i^-), x_i(t_i^-), y_i(t_i^-), u_i(t_i^-), v_i, t_i^-) = 0 \tag{2.6}$$

$$r_i^{[in]}(\dot{x}_i(t_i^-), x_i(t_i^-), y_i(t_i^-), u_i(t_i^-), v_i, t_i^-) \leq 0 \tag{2.7}$$

$$i = 1, 2, \ldots, NS-1$$

$$\left.\begin{array}{rcl} r_0^{[eq]}(\dot{x}_1(t_0), x_1(t_0), y_1(t_0), u_1(t_0), v_1, t_0) & = & 0 \\ r_{NS}^{[eq]}(\dot{x}_{NS}(t_N S), x_{NS}(t_{NS}), y_{NS}(t_{NS}), u_{NS}(t_{NS}), v_{NS}, t_{NS}) & = & 0 \end{array}\right\} \tag{2.8}$$

$$\left.\begin{array}{rcl} r_0^{[in]}(\dot{x}_1(t_0), x_1(t_0), y_1(t_0), u_1(t_0), v_1, t_0) & \leq & 0 \\ r_{NS}^{[in]}(\dot{x}_{NS}(t_N S), x_{NS}(t_{NS}), y_{NS}(t_{NS}), u_{NS}(t_{NS}), v_{NS}, t_{NS}) & \leq & 0 \end{array}\right\} \tag{2.9}$$

The stage switching times have also to be sequenced, so that they do not coincide or overlap. It is sufficient to consider:

$$t_{i-1} < t_i; \quad i = 1, 2, \ldots, NS \tag{2.10}$$

The objective function may involve the final state of each stage along with design parameters:

$$\min_{u_i(\cdot), v_i;\, i=1,2,\ldots,NS} C(\dot{X}, X, Y, U, V, T) \tag{2.11}$$

where the variables are:

$$\begin{aligned} X &= \begin{bmatrix} x_1(t_1) \\ x_2(t_2) \\ \vdots \\ x_{NS}(t_{NS}) \end{bmatrix}, \dot{X} = \begin{bmatrix} \dot{x}_1(t_1) \\ \dot{x}_2(t_2) \\ \vdots \\ \dot{x}_{NS}(t_{NS}) \end{bmatrix}, Y = \begin{bmatrix} y_1(t_1) \\ y_2(t_2) \\ \vdots \\ y_{NS}(t_{NS}) \end{bmatrix}, \\ U &= \begin{bmatrix} u_1(t_1) \\ u_2(t_2) \\ \vdots \\ u_{NS}(t_{NS}) \end{bmatrix}, V = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_{NS} \end{bmatrix}, T = \begin{bmatrix} t_1 \\ t_2 \\ \vdots \\ t_{NS} \end{bmatrix} \end{aligned} \tag{2.12}$$

The formulation for the generalized multistage optimal control problem is summarized below:

PROBLEM GMOCP

Objective function

$$\min_{u_i(\cdot),v_i;\, i=1,2,\ldots,NS} C(\dot{X}, X, Y, U, V, T) \tag{2.11}$$

subject to:

DAE equations of each stage

$$f_i(\dot{x}_i(t), x_i(t), y_i(t), u_i(t), t, v_i) = 0; \quad t_{i-1} \le t < t_i; \quad i = 1, 2, \ldots, NS \tag{2.1}$$

Initial conditions for first stage

$$J_0(\dot{x}_1(t_0), x_1(t_0), y_1(t_0), u_1(t_0), v_1, t_0) = 0 \tag{2.3}$$

Stage junction conditions

$$J_i(\dot{x}_{i+1}(t_i^+), x_{i+1}(t_i^+), y_{i+1}(t_i^+), u_{i+1}(t_i^+), v_{i+1},$$

$$\dot{x}_i(t_i^-), x_i(t_i^-), y_i(t_i^-), u_i(t_i^-), v_i, t_i) = 0; \quad i = 1, 2, \ldots, NS-1 \tag{2.2}$$

Equality and inequality path constraints for each stage

$$c_i^{[eq]}(\dot{x}_i(t), x_i(t), y_i(t), u_i(t), v_i, t) = 0 \tag{2.4}$$

$$c_i^{[in]}(\dot{x}_i(t), x_i(t), y_i(t), u_i(t), v_i, t) \le 0 \tag{2.5}$$

$$t_{i-1} \le t < t_i; \quad i = 1, 2, \ldots, NS$$

Equality and inequality point constraints

$$r_i^{[eq]}(\dot{x}_i(t_i^-), x_i(t_i^-), y_i(t_i^-), u_i(t_i^-), v_i, t_i^-) = 0 \tag{2.6}$$

$$r_i^{[in]}(\dot{x}_i(t_i^-), x_i(t_i^-), y_i(t_i^-), u_i(t_i^-), v_i, t_i^-) \le 0 \tag{2.7}$$

$$i = 1, 2, \ldots, NS-1$$

$$\left.\begin{array}{rcl} r_0^{[eq]}(\dot{x}_1(t_0), x_1(t_0), y_1(t_0), u_1(t_0), v_1, t_0) & = & 0 \\ r_{NS}^{[eq]}(\dot{x}_{NS}(t_N S), x_{NS}(t_{NS}), y_{NS}(t_{NS}), u_{NS}(t_{NS}), v_{NS}, t_{NS}) & = & 0 \end{array}\right\} \tag{2.8}$$

$$\left.\begin{array}{rcl} r_0^{[in]}(\dot{x}_1(t_0), x_1(t_0), y_1(t_0), u_1(t_0), v_1, t_0) & \le & 0 \\ r_{NS}^{[in]}(\dot{x}_{NS}(t_N S), x_{NS}(t_{NS}), y_{NS}(t_{NS}), u_{NS}(t_{NS}), v_{NS}, t_{NS}) & \le & 0 \end{array}\right\} \tag{2.9}$$

Control bounds

$$u_i^L \leq u_i(t) \leq u_i^U; \quad i = 1, 2, \ldots, NS \tag{2.13}$$

Design variable bounds

$$v_i^L \leq v_i \leq v_i^U; \quad i = 1, 2, \ldots, NS \tag{2.14}$$

Stage sequencing constraints

$$t_{i-1} < t_i; \quad i = 0, 1, \ldots, NS \tag{2.10}$$

The problems considered in this thesis are a subset of (GMOCP). Rather than having an entirely different DAE model in each stage, the model is assumed to contain the same number of variables and equations throughout. However, the equations are allowed to change structure from stage to stage. The model of this specialized class of multistage problems is described below:

PROBLEM OCP1

Objective function:

$$\min_{\dot{u}(\cdot), v, t_f} z = C(\dot{x}(t_f), x(t_f), y(t_f), u(t_f), v, t_f) \tag{2.15}$$

subject to:

DAE equations

$$f_i(\dot{x}(t), x(t), y(t), u(t), v, t) = 0 \tag{2.16}$$

$$i = 1, 2, \ldots, NS$$

Initial conditions

$$J_0(\dot{x}(t_0), x(t_0), y(t_0), u(t_0), v, t_0) = 0 \tag{2.17}$$

Junction conditions

$$J_i(\dot{x}(t_i^+), x(t_i^+), y(t_i^+), u(t_i^+), \dot{x}(t_i^-), x(t_i^-), y(t_i^-), u(t_i^-), v, t_i) = 0 \tag{2.18}$$

$$i = 1, 2, \ldots, NS - 1$$

Equality and inequality path constraints

$$c_i^{[eq]}(\dot{x}(t), x(t), y(t), u(t), v, t) = 0 \tag{2.19}$$

$$c_i^{[in]}(\dot{x}(t), x(t), y(t), u(t), v, t) \leq 0 \tag{2.20}$$

$$t_{i-1} \leq t < t_i; \quad i = 1, 2, \ldots, NS$$

Equality and inequality point constraints

$$r_i^{[eq]}(\dot{x}(t_i^-), x(t_i^-), y(t_i^-), u(t_i^-), v, t_i^-) = 0 \tag{2.21}$$

$$r_i^{[in]}(\dot{x}(t_i^-), x(t_i^-), y(t_i^-), u(t_i^-), v, t_i^-) \leq 0 \tag{2.22}$$

$$i = 1, 2, \ldots, NS$$

$$r_i^{[eq]}(\dot{x}(t_i), x(t_i), y(t_i), u(t_i), v, t_i) = 0 \tag{2.23}$$

$$r_i^{[in]}(\dot{x}(t_i), x(t_i), y(t_i), u(t_i), v, t_i) \leq 0 \tag{2.24}$$

$$i = (0, 1)$$

Control bounds

$$u^L \leq u(t) \leq u^U \tag{2.25}$$

Design variable bounds

$$v^L \leq v \leq v^U \tag{2.26}$$

Switching time constraints

$$\Delta + t_{i-1} \leq t_i; \quad \Delta > 0; \quad i = 1, 2, \ldots, NS \tag{2.27}$$

$$t_0 \leq t \leq t_f$$

The switching time strict sequencing inequalities eq. (2.10) are made stronger via a fixed positive value Δ so that existence of each stage will be guaranteed numerically.

A simple example of such a multistage fixed-DAE system problem is the case of a car which at given distances is loaded instantaneously with extra mass, thus defining a multistage system with the stages being defined by the given distances between loading stations, as shown in Figure 2.2. The objective could be the

minimization of the total time to reach distance s_3 and stop there, *i.e.* with zero velocity. Given three stages we have the following problem definition:

Objective function: minimize total travel time

$$\min_{u(\cdot),t_l} t_3 \tag{2.28}$$

ODEs: laws of motion

$$\dot{x}_1(t) = x_2(t) \tag{2.29}$$

$$\dot{x}_2(t) = u(t) \tag{2.30}$$

Initial conditions: car stationary at origin

$$x_1(0) = 0; \quad x_2(0) = 0 \tag{2.31}$$

Junction conditions: momentum conservation

$$m_i\, x_2(t_i^-) - m_{i+1}\, x_2(t_i^+) = 0; \quad i = 1, 2 \tag{2.32}$$

Interior and end-point constraints

$$x_1(t_1) = s_1 \tag{2.33}$$

$$x_1(t_2) = s_2 \tag{2.34}$$

$$x_1(t_3) = s_3 \tag{2.35}$$

$$x_2(t_3) = 0 \tag{2.36}$$

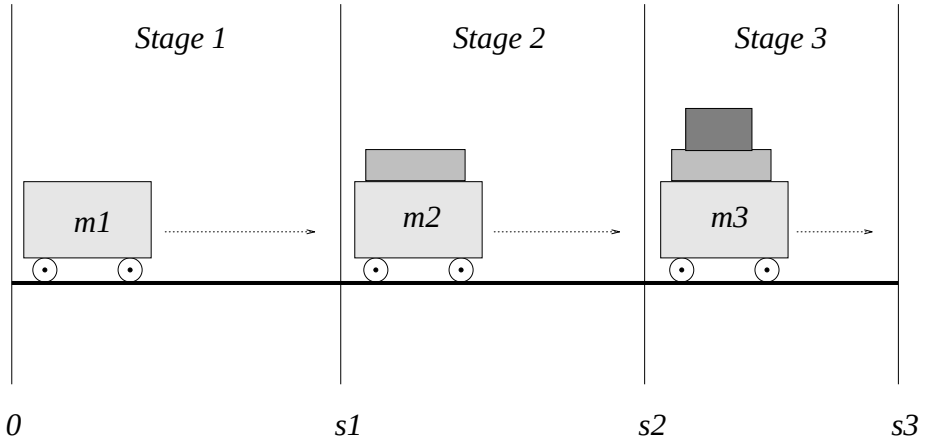

Figure 2.2. Multistage car example

Variables $x_1(t)$ and $x_2(t)$ represent the distance and the velocity, respectively, and $u(t)$ represents the acceleration which is the control variable. The initial conditions specify the starting point. The junction conditions correspond to the momentum conservation principle, and are necessary to define the new velocity assuming the extra mass is placed on the vehicle at zero velocity, thus corresponding to a perfectly inelastic collision.

2.2 Junction Conditions

For many optimal control problems, (2.17) is a complete and non-redundant initial condition specification that allows the computation of $x(t_0)$, $y(t_0)$, $\dot{x}(t_0)$ for given $u(t_0)$, v. The junction conditions (2.18) are similarly complete and non-redundant in that they permit the computation of $x(t_i^+)$, $y(t_i^+)$, $\dot{x}(t_i^+)$ given values of all the other quantities occurring in them.

In fact there is no need for either (2.17) or (2.18) to be complete and non-redundant. If they are underdetermined, then there is scope for optimizing the initial conditions at the first or later stages. If they are overdetermined, then they imply some restriction on $u(t)$, v, t_i.

We consider multistage systems described by index-1 DAEs of the form (2.16) such that $\begin{bmatrix} (f_i)_{\dot{x}} & (f_i)_y \end{bmatrix}$ is nonsingular. These equations are subject to a set of equality constraints at t_0:

$$E_0(\dot{x}(t_0), x(t_0), y(t_0), u(t_0), v, t_0) = 0 \tag{2.37}$$

other equality constraints at t_i, $i = 1, 2, \ldots, NS-1$:

$$E_i(\dot{x}(t_i^+), x(t_i^+), y(t_i^+), u(t_i^+), \dot{x}(t_i^-), x(t_i^-), y(t_i^-), u(t_i^-), v, t_i) = 0 \tag{2.38}$$

It should be noted that (2.37) comprise (2.17) while (2.38) comprise (2.18) for $i = 1, 2, \ldots, NS-1$.

The dimensions assumed for the variables are $x \in \mathcal{X} \subseteq \mathcal{R}^n$, $\dot{x} \in \mathcal{X}' \subseteq \mathcal{R}^n$, $y \in \mathcal{Y} \subseteq \mathcal{R}^m$, $u \in \mathcal{U} \subseteq \mathcal{R}^\pi$, $v \in \mathcal{V} \subseteq \mathcal{R}^q$, $t \in \mathcal{T} \equiv [t_0, t_f] \subseteq \mathcal{R}^1$, for the DAEs are

$f_i : \mathcal{X} \times \mathcal{X}' \times \mathcal{Y} \times \mathcal{U} \times \mathcal{V} \times \mathcal{T} \mapsto \mathcal{R}^{n+m}$. The dimensionality of constraints E_i, $i = 0, 1, \ldots, NS-1$ eqs. (2.37) and (2.38) is denoted by μ_i.

It should be noted that no restriction is imposed on the number μ_i of conditions E_i considered, nor do we require that the general matrices $\left[\begin{array}{ccc} \frac{\partial E_i}{\partial \dot{x}} & \frac{\partial E_i}{\partial x} & \frac{\partial E_i}{\partial y} \end{array} \right]$ be of full rank.

In the context of the control parameterization approach, the objective is normally to ensure that as many constraints as possible are taken into account by the *inner* problem (*i.e.* the DAE integration) so as to minimize the size of the optimization problem solved at the outer level. Of course, this requirement must be balanced against the difficulty of solving the inner problem.

On the other hand, this is not necessarily the case for general conditions E_i, $i = 0, 1, \ldots, NS-1$. Of course, we could, if we so wished, leave *all* of these to be handled by the optimization. Then we would have to treat all $x(t_i)$, $i = 0, 1, \ldots, NS-1$ as optimization decision variables, together with the parameters describing the u's, the v's and the times t_i, $i = 1, 2, \ldots, NS$. Because of the nonsingularity of $\left[\begin{array}{cc} (f_i)_{\dot{x}} & (f_i)_y \end{array} \right]$, $i = 1, 2, \ldots, NS$, we could then "initialize" each stage solving eq. (2.16) for $\dot{x}(t_{i-1})$ and $y(t_{i-1})$—which then allows us to integrate f_i from t_{i-1} to t_i, and to evaluate the constraints E_i.

However, such a scheme would result in a very large number of decision variables if n is large. Nevertheless, it might be advantageous in a multiple-instruction multiple-data (MIMD) computing environment since it allows all stage integrations to be performed in parallel.

Assuming it is advantageous to treat as many of the constraints E_i as possible during the DAE integration, we consider the matrix[1]:

$$A_i = \left[\begin{array}{ccc} (f_{i+1})_{\dot{x}} & (f_{i+1})_x & (f_{i+1})_y \\ (E_i)_{\dot{x}} & (E_i)_x & (E_i)_y \end{array} \right] \tag{2.39}$$

which is of size $(n + m + \mu_i) \times (2\,n + m)$. Let the rank of A_i be $\bar{\mu}_i$:

$$\bar{\mu}_i \leq \min(n + m + \mu_i, 2\,n + m) \tag{2.40}$$

[1]In the rest of this section, $(E_i)_{\dot{x}}, (E_i)_x, (E_i)_y$ for $i = 1, 2, \ldots, NS-1$ should be interpreted as $\frac{\partial E_i}{\partial \dot{x}(t_i^+)}$, $\frac{\partial E_i}{\partial x(t_i^+)}$, $\frac{\partial E_i}{\partial y(t_i^+)}$, respectively, *i.e.* as the partial derivatives of (2.38) with respect to its first three arguments.

Then there exists a lower triangular matrix L_i with non-zero diagonal, and row and column permutation matrices P_i, Q_i such that:

$$L_i\, P_i\, A_i\, Q_i = \begin{bmatrix} U_i & B_i \\ \cdot & 0 \end{bmatrix} \tag{2.41}$$

where U_i is an upper triangular $\bar{\mu}_i \times \bar{\mu}_i$ matrix with unit diagonal, B_i is a general $\bar{\mu}_i \times (2\,n + m - \bar{\mu}_i)$ matrix, and 0 is a $(n + m + \mu_i - \bar{\mu}_i) \times (2\,n + m - \bar{\mu}_i)$ null matrix.

Clearly the last $(n + m + \mu_i - \bar{\mu}_i)$ elements of:

$$P_i \begin{bmatrix} f_{i+1} \\ E_i \end{bmatrix} \tag{2.42}$$

cannot be handled in the context of the DAE initialization and must be left to the outer optimization procedure. However the DAE initialization is itself not well defined unless:

$$\bar{\mu}_i = 2\,n + m \tag{2.43}$$

If this is not the case, then the initial values of the last $(2\,n + m - \bar{\mu}_i)$ elements of:

$$Q_i \begin{bmatrix} \dot{x}(t_i^+) \\ x(t_i^+) \\ y(t_i^+) \end{bmatrix} \tag{2.44}$$

must be treated as optimization decision variables. This is equivalent to introducing sets of new time-invariant parameters $\bar{v}_i$ through the additional conditions:

$$\left\{ Q_i \begin{bmatrix} \dot{x}(t_i^+) \\ x(t_i^+) \\ y(t_i^+) \end{bmatrix} \right\}_j = \{\bar{v}_i\}_{j-\bar{\mu}_i}\,; \quad j = \bar{\mu}_i + 1, \ldots, 2\,n + m \tag{2.45}$$

In this case, the Jacobian matrix involved in the DAE initializations is the nonsingular matrix:

$$\begin{bmatrix} U_i & B_i \\ \cdot & I_i \end{bmatrix} \tag{2.46}$$

where I is a $(2\,n + m - \bar{\mu}_i) \times (2\,n + m - \bar{\mu}_i)$ unit matrix.

The nonsingularity of (2.46) ensures that the initial condition vector:

$$\begin{bmatrix} \dot{x}(t_0) \\ x(t_0) \\ y(t_0) \end{bmatrix} \tag{2.47}$$

can be calculated using a Newton iteration to solve for the first $\bar{\mu}_i$ elements of (2.42) together with the $(2\,n + m - \bar{\mu}_i)$ initial variable specifications set by the outer optimization procedure.

Of course, we may wish to restrict pivoting so that (2.47) are consistent, *i.e.* they satisfy the DAEs (2.16). This is easily done by restricting row pivoting on A_i (eq. (2.39)) to within f_{i+1} and E_i separately but *not across* both types of rows.

Similarly, we may prefer to have certain variables as initial conditions to be set by the optimization rather than others. This can be achieved by ordering these variables *last* and forbidding column pivoting between them and the rest. In particular, ensuring that only *differential* variables $x(t_0)$ are left to be specified outside is very easy: the nonsingularity of $\begin{bmatrix} (f_i)_{\dot{x}} & (f_i)_y \end{bmatrix}$ ensures that if $\{\dot{x}(t_i), y(t_i)\}$ are ordered first, *all* of them will be selected for inclusion in the inner DAE problem.

2.3 Derivation of the Control Parameterized Model

Control parameterization will be utilized to transform the original problem (OCP1) from infinite dimensional to a finite optimization problem.

The duration of each stage is subdivided into a number of finite elements, with the controls defined as low-order polynomials over each element. Figure 2.3 shows the subdivision of stages in finite elements.

For the sake of notational simplicity, *we treat each finite element as a separate stage*, and introduce junction conditions:

$$x(t_i^+) = x(t_i^-) \tag{2.48}$$

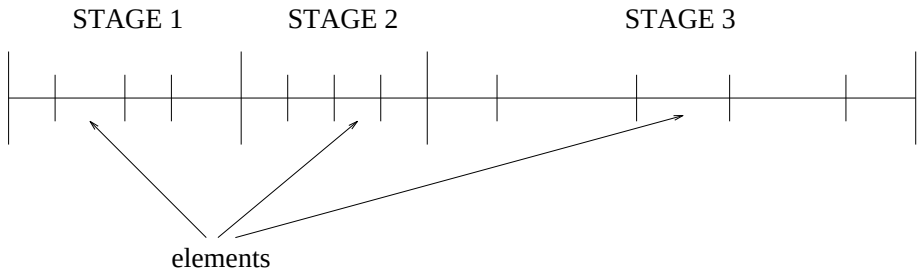

Figure 2.3. Subdivision of stages into finite elements

between them as in most physical systems the differential variables x represent conserved quantities. Therefore, we shall henceforth use NS to denote the *total* number of stages (real and artificial), and the range of the stage subscript i will be expanded accordingly.

For convenience, we use Lagrange polynomials. For control variable u_j over stage i, we employ an approximation:

$$u_j^{[i]}(t) = \sum_{k=1}^{M_j} u_{ijk}\phi_k^{(M_j)}(\tau^{[i]}) \tag{2.49}$$

$$t \in [t_{i-1}, t_i]$$

of order M_j (degree $< M_j$), where:

$$\begin{aligned} \phi_k^{(M_j)}(\tau) &= 1 \quad \text{if } M_j = 1; \\ \phi_k^{(M_j)}(\tau) &= \prod_{\substack{\kappa=1 \\ \kappa\neq k}}^{M_j} \frac{(\tau - \tau_\kappa)}{(\tau_k - \tau_\kappa)} \quad \text{if} \quad M_j \geq 2; \end{aligned} \tag{2.50}$$

and $\tau^{[i]}$ is normalized time over stage i:

$$\tau^{[i]} = \frac{t - t_{i-1}}{t_i - t_{i-1}} \tag{2.51}$$

$$i = 1, 2, \ldots, NS; \quad j = 1, 2, \ldots, \pi; \quad k = 1, 2, \ldots, M_j$$

The choice of the set of normalized points τ_κ does not affect the solution obtained as different polynomial approximations of the same order are equivalent. The choice of these points is to some extent coupled with the bounding of the control involved. If bounding is to be guaranteed then it is useful to have $\tau_1 = 0$ and $\tau_{M_j} = 1$ with τ_κ, $\kappa = 2, 3, \ldots, M_j - 1$ being chosen arbitrarily,

e.g. at orthogonal locations (Carey and Finlayson, 1975; Finlayson, 1980) or as equidistant. Control bounding is easy to guarantee for piecewise constant and piecewise linear approximations:

$$M_j = 1: \quad u^L \leq u_{j1} \leq u^U \tag{2.52}$$

$$M_j = 2: \quad u^L \leq u_{jk} \leq u^U, \quad k = 1, 2 \tag{2.53}$$

In fact, bounds can be enforced for quadratic or cubic functions as well, however these will involve functions of the parameters used to discretize the controls. With higher-order polynomials, such bounding may not be possible.

An important issue at this stage regards the continuity of the control variables. C^0-continuity for control variable j at stage boundaries can simply be enforced by the additional constraints:

$$u^{[i-1]}(t_{i-1}) = u^{[i]}(t_{i-1}), \quad \forall\, i = 2, 3, \ldots, NS \tag{2.54}$$

Expressing these constraints in terms of the discretization parameters is straightforward if the element end-points are included as approximation points as noted above:

$$u_{i-1,j,M_j} = u_{i,j,0} \quad \forall\, i = 2, 3, \ldots, NS \tag{2.55}$$

Examples of control profiles of various types are shown in Figure 2.4.

Higher-order continuity may be enforced by more complex constraints derived by differentiating the Lagrange polynomial approximations (2.49) with respect to time a sufficient number of times. Alternatively, a convenient way of enforcing higher-order continuity in a control variable $u(t)$ is to define an auxiliary set of equations:

$$\begin{aligned} \dot{u} &= z_1 \\ \dot{z}_1 &= z_2 \\ &\vdots \\ \dot{z}_{\nu-1} &= z_\nu \end{aligned} \tag{2.56}$$

where $z_1, z_2, \ldots, z_\nu$ are new variables. These equations are appended to the original DAE system, and variables u and z_l, $l = 1, 2, \ldots, \nu - 1$ are treated

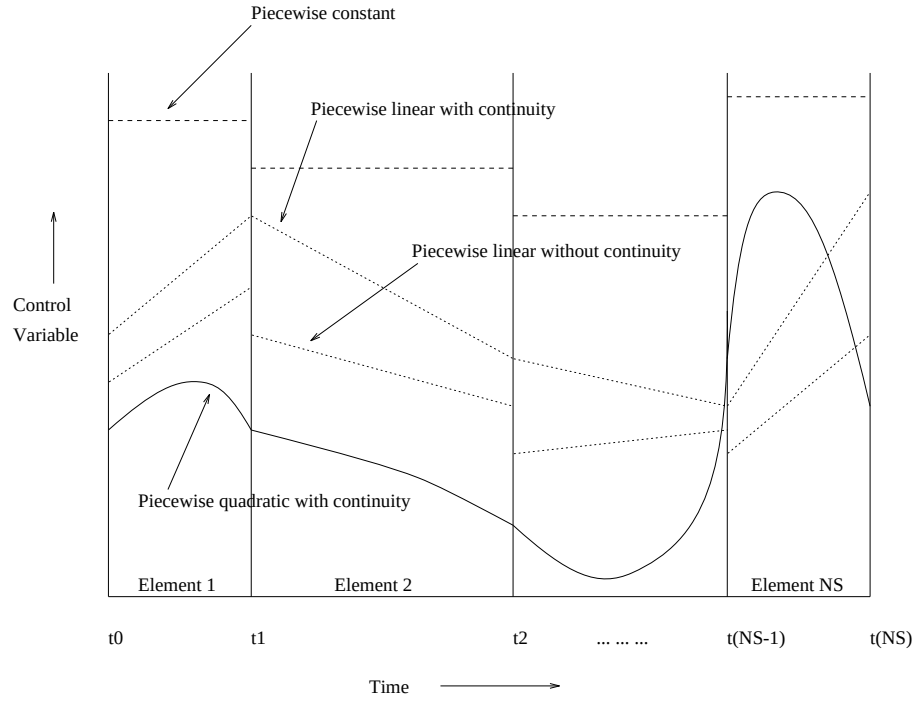

Figure 2.4. Control profile examples

as state variables while z_ν is treated as a control variable. It can easily be seen that z_l corresponds to $\frac{d^l u}{dt^l}$, and therefore, enforcing zero-order continuity on $z_\nu(t)$ ensures ν-th order continuity in $u(t)$.

This technique was suggested by Gritsis (1990) who also observed that control bounding is more difficult with this approach as it requires the presence of path constraints. It is noted that the initial conditions for the variables $u(t_0)$, $z_l(t_0), l = 1, 2, \ldots, \nu - 1$ above can be optimization parameters along with the set of parameters discretizing $z_\nu(t)$.

In any case, using the discretization of the control variables defined by equation (2.49), the infinite dimensional OCP1 is reduced to a finite dimensional optimization problem in terms of the set of parameters:

$$p = \{u_{ijk}, h_i, v\} \tag{2.57}$$

We assume that the techniques of section (2.2) have been applied to the junction conditions of OCP1 to ensure that (2.17) and (2.18) are complete and non-redundant sets of initial and junction conditions respectively (through the

definition of additional variables v representing unknown initial conditions if necessary). The optimization problem is then as follows:

PROBLEM OCP2

Objective function

$$\min_p C(\dot{x}(t_f,p), x(t_f,p), y(t_f,p), u^{[NS]}(t_f, u_{NS,j,k}, h_{NS}), v, t_f) \tag{2.58}$$

subject to:
Equality and inequality interior point constraints

$$r_i^{[eq]}(\dot{x}(t_i,p), x(t_i,p), y(t_i,p), u^{[i]}(t_i, u_{ijk}, h_i), v, t_i) = 0 \tag{2.59}$$

$$r_i^{[in]}(\dot{x}(t_i,p), x(t_i,p), y(t_i,p), u^{[i]}(t_i, u_{ijk}, h_i), v, t_i) \leq 0 \tag{2.60}$$

$$i = 0, 1, \ldots, NS$$

Control parameter bounds

$$u_j^L \leq u_j \leq u_j^U; \quad j = 1, 2, \ldots, \pi \tag{2.61}$$

Design variable bounds

$$v^L \leq v \leq v^U \tag{2.62}$$

Switching time constraints

$$h_i^L \leq h_i \leq h_i^U \tag{2.63}$$

$$t_i = t_{i-1} + h_i \tag{2.64}$$

$$i = 1, 2, \ldots, NS$$

where $u^{[i]}(t, u_{ijk}, h_i)$ denotes the control variable approximation (2.49).

The values of the variables $x(t,p)$, $y(t,p)$, $\dot{x}(t,p)$ are obtained by solving the DAE systems

$$f_i(\dot{x}(t_i,p), x(t_i,p), y(t_i,p), u^{[i]}(t, u_{ijk}, h_i), v, t) = 0; \quad i = 1, 2, \ldots, NS \tag{2.65}$$

subject to the initial conditions

$$J_0(\dot{x}(t_0,p), x(t_0,p), y(t_0,p), u(t_0,p), v, t_0) = 0 \tag{2.66}$$

and the junction conditions

$$J_i(\dot{x}(t_i^+,p), x(t_i^+,p), y(t_i^+,p), u^{[i+1]}(t_i^+, u_{i+1,j,k}, p),$$
$$\dot{x}(t_i^-,p), x(t_i^-,p), y(t_i^-,p), u^{[i]}(t_i^-, u_{ijk}, p), v, t_i) = 0 \qquad (2.67)$$
$$i = 1, 2, \ldots, NS-1$$

It is worth pointing out that the path constraints (2.19) and (2.20) have been omitted from OCP2. We shall return to them in Chapter 4 where it will be shown how they can be reformulated to point constraints.

2.4 Control Parameterization Algorithm Operation

In this section, the details of a CP algorithm based on the discretized model derived in the previous section are presented.

2.4.1 Algorithm Overview

Assuming that (2.65) is index-1 in (x, y), (2.66) is a complete and non-redundant set of initial conditions, and (2.67) is a complete and non-redundant set of junction conditions, *then* given values of p we can integrate from t_0 to t_f.

The first step is the initialization at t_0 solving (2.65) and (2.66) simultaneously to obtain the values of $\dot{x}, x, y$. Similarly at each stage starting point, a re-initialization is carried out using the appropriate junction condition from (2.67) and the stage DAE equations (2.65). After each initialization the DAEs (2.65) are integrated up to the end of the next stage.

Once t_f is reached, we can evaluate the objective function (2.58) and the constraints (2.59) and (2.60). The optimization, apart from requiring the objective function and constraint values, also requires their gradients. This issue is examined next.

2.4.2 Sensitivity Equations

By differentiating the stage equations (2.65) with respect to the parameters p, we obtain:

$$\frac{\partial f_i}{\partial \dot{x}}\dot{x}_p + \frac{\partial f_i}{\partial x}x_p + \frac{\partial f_i}{\partial y}y_p + \frac{\partial f_i}{\partial u}\frac{\partial u^{[i]}}{\partial p} + \frac{\partial f_i}{\partial v}\frac{\partial v}{\partial p} = 0 \tag{2.68}$$

where $x_p = \frac{\partial x}{\partial p}$, $y_p = \frac{\partial y}{\partial p}$, and $\dot{x}_p = \frac{\partial}{\partial t}\left(\frac{\partial x}{\partial p}\right)$.

Making use of eq. (2.68) requires the matrices $\frac{\partial u^{[i]}}{\partial p}$, and $\frac{\partial v}{\partial p}$, where we have already seen that vector p comprises three different types of parameters, namely u_{ijk}, h_i, and v. We deal with each type separately:

Gradients with respect to u_{ijk}

From eq. (2.49), it can be seen that:

$$\frac{\partial u_j^{[i]}}{\partial u_{i'j'k}} = \phi_k^{(M_j)}(\tau^{[i]})\,\delta_{ii'}\,\delta_{jj'}; \quad k = 1, 2, \ldots, M_j \tag{2.69}$$

where $\delta_{ii'}$, $\delta_{jj'}$ are Kronecker deltas.

Also obviously $\frac{\partial v}{\partial u_{ijk}} = 0$.

Gradients with respect to h_i

It is in fact, more convenient to derive the necessary gradients with respect to the switching times, t_i. From (2.49):

$$\frac{\partial u_j^{[i]}}{\partial t_i}(t) = \sum_{k=1}^{M_j} u_{ijk} \frac{d\phi_k^{(M_j)}}{d\tau^{[i]}} \frac{\partial \tau^{[i]}}{\partial t_i} \tag{2.70}$$

where the last factor may be obtained from (2.51) as:

$$\frac{\partial \tau^{[i]}}{\partial t_i} = -\frac{(t - t_{i-1})}{h_i^2} \tag{2.71}$$

overall leading to:

$$\frac{\partial u_j^{[i]}}{\partial t_i}(t) = -\frac{(t - t_{i-1})}{h_i^2} \sum_{k=1}^{M_j} u_{ijk} \frac{d\phi_k^{(M_j)}}{d\tau^{[i]}} \tag{2.72}$$

Similarly it can be shown that:

$$\frac{\partial u_j^{[i]}}{\partial t_{i-1}}(t) = -\frac{(t_i - t)}{h_i^2} \sum_{k=1}^{M_j} u_{ijk} \frac{d\phi_k^{(M_j)}}{d\tau^{[i]}} \tag{2.73}$$

The derivatives of $u^{[i]}$ with respect to all other switching times t_l, $l \neq i-1, i$ are zero.

Now, $t_i = \sum_{l=1}^{i} h_l$ and $t_{i-1} = \sum_{l=1}^{i-1} h_l$; therefore:

$$\frac{\partial u_j^{[i]}}{\partial h_i}(t) = \frac{\partial u_j^{[i]}}{\partial t_i} \frac{\partial t_i}{\partial h_i} = \frac{\partial u_j^{[i]}}{\partial t_i} \tag{2.74}$$

and

$$\begin{aligned} \frac{\partial u_j^{[i]}}{\partial h_l}(t) &= \frac{\partial u_j^{[i]}}{\partial t_i} \frac{\partial t_i}{\partial h_l} + \frac{\partial u_j^{[i]}}{\partial t_{i-1}} \frac{\partial t_{i-1}}{\partial h_l} = \frac{\partial u_j^{[i]}}{\partial t_i} + \frac{\partial u_j^{[i]}}{\partial t_{i-1}} \\ &= -\frac{1}{h_l} \sum_{k=1}^{M_j} u_{ijk} \frac{d\phi_k^{(M_j)}}{d\tau^{[i]}}; \quad \forall\, l = 1, 2, \ldots, i-1 \end{aligned} \tag{2.75}$$

All other gradients of $u^{[i]}$ with respect to h_l for $l > i$ are zero, and so are all

gradients of v with respect to h_l.

Gradients with respect to v

Clearly

$$\frac{\partial u^{[i]}}{\partial v} = 0 \tag{2.76}$$

while

$$\frac{\partial v}{\partial v} = I \tag{2.77}$$

Using the expressions derived above, it is now possible to evaluate the sensitivity equations (2.68). In fact, these are linear in the unknowns x_p, y_p. Their integration using implicit methods (*e.g.* based on BDF approximations of time derivatives) involves forming and factorizing the matrix:

$$\begin{bmatrix} \frac{\partial f_i}{\partial \dot{x}} & \frac{\partial f_i}{\partial x} & \frac{\partial f_i}{\partial y} \end{bmatrix} \tag{2.78}$$

However, this is exactly the same matrix as the Jacobian involved in the integration of the original DAE system, a fact that can be exploited to minimize the extra cost involved in evaluating sensitivities (Leis and Kramer, 1985; Caracotsios and Stewart, 1985).

2.4.3 Initial and Junction Conditions for Sensitivities

By differentiating the system initial conditions (2.66) with respect to the parameters p we obtain:

$$\frac{\partial J_0}{\partial \dot{x}}\dot{x}_p(t_0,p) + \frac{\partial J_0}{\partial x}x_p(t_0,p) + \frac{\partial J_0}{\partial y}y_p(t_0,p) + \frac{\partial J_0}{\partial u}\frac{\partial u^{[1]}}{\partial p} + \frac{\partial J_0}{\partial v}\frac{\partial v}{\partial p} = 0 \tag{2.79}$$

Equation (2.79) may be solved simultaneously with eq. (2.68) for $i = 1$ to obtain the initial values of the sensitivities:

$$\begin{bmatrix} \frac{\partial f_1}{\partial \dot{x}} & \frac{\partial f_1}{\partial x} & \frac{\partial f_1}{\partial y} \\ \frac{\partial J_0}{\partial \dot{x}} & \frac{\partial J_0}{\partial x} & \frac{\partial J_0}{\partial y} \end{bmatrix} \begin{bmatrix} \dot{x}_p(t_0,p) \\ x_p(t_0,p) \\ y_p(t_0,p) \end{bmatrix} = - \begin{bmatrix} \frac{\partial f_1}{\partial u} & \frac{\partial f_1}{\partial v} \\ \frac{\partial J_0}{\partial u} & \frac{\partial J_0}{\partial v} \end{bmatrix} \begin{bmatrix} \frac{\partial u^{[1]}}{\partial p} \\ \frac{\partial v}{\partial p} \end{bmatrix} \tag{2.80}$$

where $\frac{\partial u^{[1]}}{\partial p}$ and $\frac{\partial v}{\partial p}$ can be obtained as shown in section 2.4.2.

The matrix on the l.h.s. of (2.80) is the same as that involved in the initialization of the original DAE system, and its nonsingularity is a consequence of the completeness and non-redundancy of the initial conditions specifications of eq. (2.66).

In a similar fashion, the junction conditions eq. (2.67) may be differentiated with respect to the parameters p and the resulting equations can then be solved simultaneously with (2.68) to obtain the values of the sensitivities at the start of stage i, $i = 2, 3, \ldots, NS$. The corresponding system of equations is:

$$\begin{bmatrix} \frac{\partial f_{i+1}}{\partial \dot{x}} & \frac{\partial f_{i+1}}{\partial x} & \frac{\partial f_{i+1}}{\partial y} \\ \frac{\partial J_i}{\partial \dot{x}^+} & \frac{\partial J_i}{\partial x^+} & \frac{\partial J_i}{\partial y^+} \end{bmatrix} \begin{bmatrix} \dot{x}_p(t_i, p) \\ x_p(t_i, p) \\ y_p(t_i, p) \end{bmatrix} =$$

$$- \begin{bmatrix} \frac{\partial f_{i+1}}{\partial u} & \frac{\partial f_{i+1}}{\partial v} & 0 & 0 & 0 & 0 & \frac{\partial f_{i+1}}{\partial t} \\ \frac{\partial J_i}{\partial u^+} & \frac{\partial J_i}{\partial v} & \frac{\partial J_i}{\partial \dot{x}^-} & \frac{\partial J_i}{\partial x^-} & \frac{\partial J_i}{\partial y^-} & \frac{\partial J_i}{\partial u^-} & \frac{\partial J_i}{\partial t_i} \end{bmatrix} \begin{bmatrix} \frac{\partial u^{[i+1]}}{\partial p} \\ \frac{\partial v}{\partial p} \\ \frac{\partial \dot{x}^-}{\partial p} \\ \frac{\partial x^-}{\partial p} \\ \frac{\partial y^-}{\partial p} \\ \frac{\partial u^{[i]}}{\partial p} \\ \frac{\partial t_i}{\partial p} \end{bmatrix} \qquad (2.81)$$

$$i = 1, 2, \ldots, NS - 1$$

where $\frac{\partial u^{[i+1]}}{\partial p}$, $\frac{\partial v}{\partial p}$, $\frac{\partial u^{[i]}}{\partial p}$ can be obtained as shown in section 2.4.2, while $\frac{\partial \dot{x}^-}{\partial p}$, $\frac{\partial x^-}{\partial p}$, $\frac{\partial y^-}{\partial p}$ correspond to $\dot{x}_p(t_i^-)$, $x_p(t_i^-)$ and $y_p(t_i^-)$ respectively. The derivative $\frac{\partial t_i}{\partial p}$ is 1 if p is t_i, and 0 otherwise.

It should be noted that the matrix on the l.h.s. of (2.81) is the same as that involved in re-initializing the original DAE system at t_i which is assumed to be non-singular.

The values obtained by solving (2.81) are the correct sensitivities at a *fixed* time t_i, *i.e.* $\frac{\partial x}{\partial p}(t,p)$ for $t = t_i$. However, if the parameter p is t_i itself, we need to take this into account by considering the *total* derivative of $x(t_i,p)$. We note that, for any $t \in [t_i, t_{i+1})$:

$$x(t,p) = x(t_i,p) + \int_{t_i}^{t} \dot{x}(t,p)\, dt \tag{2.82}$$

differentiation of which with respect to t_i yields:

$$\frac{Dx(t,p)}{Dt_i} = \frac{\partial x(t_i)}{\partial t_i} - \dot{x}(t_i,p) + \int_{t_i}^{t} \frac{\partial \dot{x}(t,p)}{\partial t_i}\, dt \tag{2.83}$$

The result above simply defines the solution for the sensitivity equations. To obtain the correct initial condition for the switching time sensitivities the limit as $t \to t_i^+$ is taken:

$$\frac{Dx(t_i^+)}{Dt_i} = \frac{\partial x(t_i^+,p)}{\partial t_i} - \dot{x}(t_i^+,p) \tag{2.84}$$

This general expression for the initial condition of the switching time sensitivities has been derived without the need for any condition on functions f_i and J_i other than sufficient differentiability to guarantee the existence of all expressions used.

In many practical applications, the junction conditions (2.67) are of the less general form:

$$J_i(x(t_i^+,p), x(t_i^-,p), v, t_i) = 0; \quad i = 1, 2, \ldots, NS-1 \tag{2.85}$$

with $\frac{\partial J_i}{\partial x(t_i^+,p)}$ being nonsingular. In such cases, the values of $x(t_i^+,p)$ can be determined by solving the junction conditions (2.85) in isolation. And also (2.81) and (2.84) can be combined to yield:

$$\frac{Dx}{Dt_i}(t_i^+,p) = -\left[\frac{\partial J_i}{\partial x(t_i^+,p)}\right]^{-1} \left\{\frac{\partial J_i}{\partial x(t_i^-,p)}\dot{x}(t_i^-,p) + \frac{\partial J_i}{\partial t_i}\right\} - \dot{x}(t_i^+,p) \tag{2.86}$$

In the even more specialized (and very common) case of junction conditions of the form:

$$x(t_i^+,p) - x(t_i^-,p) = 0 \tag{2.87}$$

eq. (2.86) becomes:

$$\frac{Dx(t_i^+,p)}{Dt_i} = \dot{x}(t_i^-,p) - \dot{x}(t_i^+,p) \tag{2.88}$$

which is the same result as that reported by Rosen and Luus (1991).

2.5 Implementation

The ideas presented thus far in this chapter were implemented in the FORTRAN 77 subroutine DAEOPT (Vassiliadis, 1992). The integration code chosen is DASOLV (Jarvis and Pantelides, 1992a) which implements the idea of simultaneous solution of the sensitivity equations. The most important feature is the capability of handling large sparse DAE models very efficiently, as well as handling of explicit discontinuities such as those caused by the control switches from one element to another. The optimization code is SRQPD (Chen, 1988, Chen and Macchietto, 1989) which is an implementation of a reduced sequential quadratic programming algorithm code. This code is implemented in a reverse communication mode:

Set starting point p_0;
REPEAT
    Integrate DAEs;
    Call SRQPD;
UNTIL convergence;

The SRQPD code returns with a different flag depending on whether function values (line search) or gradient values are required (reformulation and solution of a new QP subproblem), or convergence has been achieved. If convergence is not achieved, then the appropriate type of integration is chosen (state only, or state and sensitivity) and the loop continues. If an integration for a specific parameter point p is infeasible, the step is cut and another line search is initiated at the new point.

In our implementation, simple constraints of the form $\pm x_n(t_i) \geq \gamma$, or $\pm x_n(t_i) = \gamma$, where γ is a constant and $t_i \in [0, t_f]$, are treated directly by the optimizer with their gradients being given by the sensitivities of the corresponding variables as calculated by the integrator.

More complex interior point constraints (including end-point constraints) and the objective are treated in two different groups. The first group, denoted as

{1}, consists of constraints that are interior point constraints containing state variables and possibly design variables. The second group, denoted as {2}, contains constraints that are explicit functions of the optimization parameters p only. Also the objective function is treated separately when it only depends on optimization parameters.

Constraints of type {1} and objective functions that involve state variables are assigned directly to the integrator as extra *algebraic* equations with one new state variable equal to their value. For example, for an inequality constraint ${}^{\{1\}}r_l^{[in]} \leq 0$ at some point t_i we define a new algebraic variable y_{n_y+1} through the algebraic equation:

$$y_{n_y+1}(t) - {}^{\{1\}}r_l^{[in]}(\dot{x}(t), x(t), y(t), u(t), p) = 0 \tag{2.89}$$

and then we pose the constraint $y_{n_y+1}(t_i) \leq 0$ to the optimization.

The reason for this transformation is that it is easier to model, and program, such problems. Of course, there is an extra cost as now the integrator is assigned a somewhat larger problem but for real applications, for which the size of the DAE model is significantly larger than the number of constraints, this entails relatively little additional computational effort.

Constraints of type {2} may represent design parameter constraints, or control parameter interrelationships. This enables the user to have direct access to the parameters determining the profiles of the controls and the remaining design variables and to express any type of constraint involving them.

For the solution, all the derivative matrices appearing in equations (2.80) and (2.81), are supplied through user-defined subroutines.

CHAPTER 3

Numerical Experiments

Optimal control problems arise frequently in engineering practice. Particularly in the field of chemical engineering, dynamic processes are often modelled as stiff differential equations with fast transients usually including algebraic constraints (*e.g.* physical property relations), thus forming systems of differential-algebraic equations (DAEs) describing the dynamic model of the problem. Such models are hard to solve, and the difficulty carries over to their optimization.

To test the algorithm as implemented in DAEOPT, a collection of problems were solved. Based on the experience obtained, several ways of speeding up the solution process as well as refining rough results have been devised and implemented successfully.

The issue of path constrained problems is dealt with by the following chapter as the difficulties encountered with such problems deserve individual attention.

The case studies to be presented in this section have been solved in most cases using various levels of tolerances for the integration and optimization stages in DAEOPT. The tolerance levels are given in Table 3.1. The integration package, DASOLV, uses the user specified tolerance to control the local error of the trajectory at each integration step. The optimization package, SRQPD, uses the user specified tolerance to control the satisfaction of constraints, bounds, and optimality with respect to the Lagrangian function.

Level	Integration	Optimization
1	10^{-5}	10^{-3}
2	10^{-7}	10^{-5}
3	10^{-9}	10^{-7}
4	10^{-12}	10^{-10}

Table 3.1. Tolerance levels

CPU time statistics are reported for each problem solved. The average cost ratio of a sensitivity and state integration with respect to a state only integration is given. QPs is the number of quadratic programming problems formulated and solved in the SQP algorithm (also equal to the number of sensitivity integrations), and LSs is the number of function evaluations required during the line search procedure (each corresponding to one state integration). All CPU times, unless otherwise stated, are reported for a SUN SPARC 2 workstation.

3.1 Discrete Charge Batch Reactor

Recently Levien (1992) proposed a methodology based on kinetic insight that optimizes the product distribution obtained from competing chemical reactions taking place in parallel. Kokossis and Floudas (1990) have considered the same problem as a case study for their superstructure approach for reactor networks based on continuous stirred tank reactor (CSTR) modules. This results in mixed integer nonlinear programming problems (MINLPs). An alternative approach is that of Achenie and Biegler (1988) who examined a continuous parameter superstructure for reactor networks resulting in optimal control problems. Unlike the Floudas and Kokossis method, this method cannot account for costs arising from existence or non-existence of units.

In any case, the optimization of a single batch reactor, or a fixed batch reactor train, does not require integer decisions. Therefore, as in the work of Achenie and Biegler, an optimal control model is the natural choice. In this way, any type of kinetics, non-idealities both in mixing and operation, as well as more detailed models involving partial differential equation models can be tackled very effectively.

Here, the Trambouze reaction scheme as tested by Levien (1992) and Kokossis and Floudas (1990) is examined in detail. The reaction scheme is shown in Figure 3.1, and the reaction kinetics lead to the following differential equations:

$$\begin{aligned} \dot{C}_A &= -\dot{C}_R - \dot{C}_S - \dot{C}_T \\ \dot{C}_R &= k_1\, C_A \\ \dot{C}_S &= k_2 \\ \dot{C}_T &= k_3\, C_A^2 \end{aligned} \tag{3.1}$$

The reactor is loaded initially with pure A, which corresponds to the initial condition:

$$\begin{aligned} C_A(0) &= 1000.0 \\ C_R(0) &= 0.0 \\ C_S(0) &= 0.0 \\ C_T(0) &= 0.0 \end{aligned} \tag{3.2}$$

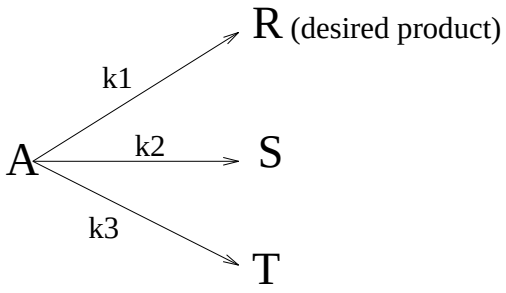

Figure 3.1. Trambouze reaction scheme

Further amounts of fresh feed may be added instantaneously during the reaction. If we denote the i-th such additional amount by d_i, the following instantaneous mass balances hold at the time of the addition, t_i:

$$\begin{aligned} V^+ C_A^+ &= V^- C_A^- + d_i C_{A0} \\ V^+ C_R^+ &= V^- C_R^- \\ V^+ C_S^+ &= V^- C_S^- \\ V^+ C_T^+ &= V^- C_T^- \\ V^+ &= V^- + d_i \\ i &= 1, 2, \ldots, N \\ \textstyle\sum_{i=1}^{N} d_i &= 0.1 \end{aligned} \tag{3.3}$$

where in the summation of charges the initial load is accounted for as well. For the first addition (at time $t = 0$), V^- is taken to be zero, hence $V^+ = d_1$.

The desired product is R. Two different objective functions were maximized separately, namely the *overall fractional yield* and the *overall product yield* in moles, Φ and $\mathcal{Y}$, respectively defined as:

$$\Phi = \frac{C_R(t_f) - C_{R0}}{C_{A0} - C_A(t_f)} \tag{3.4}$$

$$\mathcal{Y} = C_R(t_f)\, V(t_f) \tag{3.5}$$

To implement the overall fractional yield, Φ, as an objective function required some reformulation. The complication is that, in our implementation, end-point constraints as well as state dependent objective functions have to be defined by algebraic equations (*cf.* section 2.5) within the dynamic model. As can be

observed, Φ is not defined at $t = 0$. It was therefore replaced by $\hat{\Phi}$ defined as:

$$\hat{\Phi}(t) = \frac{C_R(t) - C_{R0}}{C_{A0} - C_A(t) + (t - t_f)^2} \tag{3.6}$$

Use of this variable is completely equivalent to the overall fractional yield at time t_f, as their values become equal. Moreover, the derivative of $\hat{\Phi}$ with respect to t_f and any *other* parameter p at $t = t_f$ is equal to the derivative of Φ, therefore no discrepancy will be caused:

$$\Phi(t_f) = \hat{\Phi}(t_f) \tag{3.7}$$

$$\frac{\partial \Phi(t_f)}{\partial t_f} = \frac{\partial \hat{\Phi}(t_f)}{\partial t_f} \tag{3.8}$$

$$\frac{\partial \Phi(t_f)}{\partial p} = \frac{\partial \hat{\Phi}(t_f)}{\partial p} \tag{3.9}$$

An alternative way of operating the reactor is to have a *continuous* feed instead of discrete charges. In this case, the model changes to:

$$\begin{aligned}
&V\,\dot{C}_A = -u\,C_A + u\,C_{A0} - V\,[k_1\,C_A + k_2 + k_3\,C_A^2]\,; && C_A(0) = 1000.0\\
&V\,\dot{C}_R = -u\,C_R + V\,k_1\,C_A; && C_R(0) = 0.0\\
&V\,\dot{C}_S = -u\,C_S + V\,k_2; && C_S(0) = 0.0\\
&V\,\dot{C}_T = -u\,C_T + V\,k_3\,C_A^2; && C_T(0) = 0.0\\
&\dot{V} = u; && V(0) = V_0\\
&0 \le u(t) \le 10\\
&V(t_f) = 0.1\\
&0 \le t \le t_f
\end{aligned} \tag{3.10}$$

In this case, the feed rate $u(t)$ was treated as a continuous control variable. Piecewise linear sections were used over 10 elements to approximate the control. The initial volume in the reactor was set to $10^{-5} m^3$. The control element sizes were bounded by 2.0 min and 20 min, initialized as 3.0 min each. The control profile was bounded by 0.0 m^3/min and 10.0 m^3/min, initially set to a uniform value of 0.01 m^3/min. The same data were used for optimizing both the overall fractional yield and the overall product yield. However, for the optimization of the overall product yield, the lower bound of the penultimate control element size was set to 0.3 min in order to allow a steep jump in the feed level.

	Discrete Charge Scheme						
	This Work				Levien		Kokossis & Floudas
Charge Scheme	2	5	10	continuous	2	10	10
Φ_{opt}	0.44751	0.47620	0.48954	0.49906	0.447513	0.4895	0.4872
t_f	8.59795	16.4915	25.2702	37.1713	8.491	25.323	20.287
CPU	23.6	154.3	302.3	640.2	—	—	134.7[1]
$\frac{\text{Sens.}}{\text{State}}$ CPU	4.9	12.2	9.2	8.8	—	—	—
# QPs	6	10	17	30	—	—	—
# LSs	8	17	33	49	—	—	—
$\mathcal{Y}_{\text{opt}}$	42.7636	45.2233	46.3276	47.0672	—	—	—
t_f	11.96403	20.41069	29.90683	42.40899	—	—	—

Table 3.2. Optimization performance indices

The optimization of the overall fractional yield and the overall product yield result in all cases in the same optimal final time and control profile up to the point where the reactor volume reaches 0.1 m^3. The optimization of the overall fractional yield stops there, whereas the optimization of the overall product yield takes the system further up to the point where the reactant concentration reaches zero, without any further addition of material.

Several optimization related indices are listed in Table 3.2. Reported there are the CPU time in seconds on a SUN SPARC IPX workstation, the average cost ratio of a sensitivity integration to a state-only integration, the number of quadratic programming subproblems (QPs) solved, and the number of line searches (LSs) required within the sequential quadratic programming algorithm used. The solution tolerances were set to Level 2 (see Table 3.1). Three discrete charge schemes involving 2, 5, and 10 charges respectively, were examined.

The values of the optimization parameters are listed in Table 3.3 for the discrete charge schemes. Figures 3.2, 3.3, and 3.4 depict the trajectories for C_A and C_R for each charge scheme respectively. The continuous feed scheme is depicted in Figures 3.5 and 3.6 where the flowrate and state profiles are given respectively.

[1] On a VAX 3200 workstation.

Discrete Charge Scheme						
Optimal Reaction Stage Durations (min)						
	This Work			Levien		Kokossis & Floudas
#	2	5	10	2	10	10
1	4.98329	4.99042	5.00099	4.891	4.981	6.691
2	3.61465	3.59945	3.62580	3.601	3.601	3.504
3	—	2.99764	3.00489	—	3.007	2.352
4	—	2.60574	2.59224	—	2.607	1.767
5	—	2.29831	2.27831	—	2.310	1.413
6	—	—	2.07455	—	2.079	1.178
7	—	—	1.88947	—	1.893	1.009
8	—	—	1.71083	—	1.739	0.883
9	—	—	1.59454	—	1.609	0.784
10	—	—	1.49857	—	1.497	0.706
Optimal Charge Levels (%)						
1	47.33	12.47	3.371	47.320	3.372	10.546
2	52.67	13.95	3.818	52.679	3.754	9.593
3	—	18.66	5.107	—	5.055	9.857
4	—	24.43	6.645	—	6.563	9.942
5	—	30.48	8.230	—	8.258	9.979
6	—	—	10.17	—	10.133	10.000
7	—	—	12.28	—	12.188	10.011
8	—	—	14.26	—	14.421	10.019
9	—	—	16.85	—	16.833	10.024
10	—	—	19.27	—	19.423	10.028

Table 3.3. Discrete charges—Optimal parameter values

It is observed that, with increasing number of material injections, the objective function is increased. The limit case is the continuous feed policy which yields the largest value for the objective function.

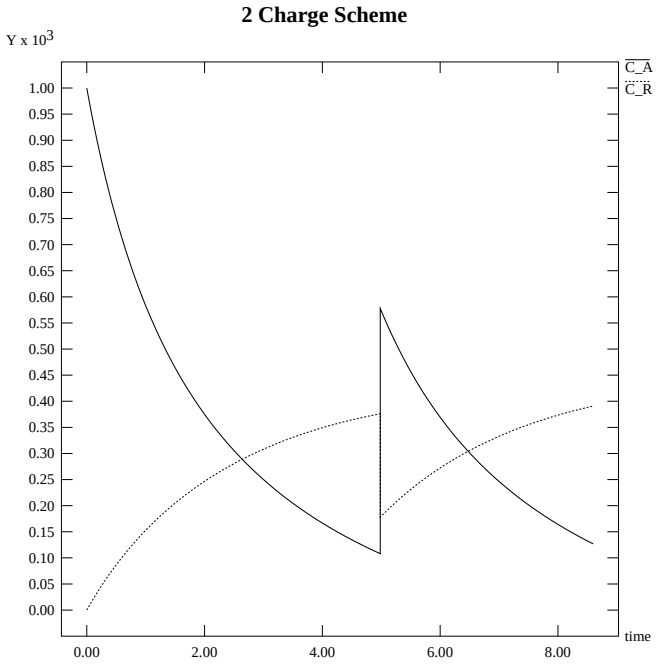

Figure 3.2. Concentration profiles for 2 charges

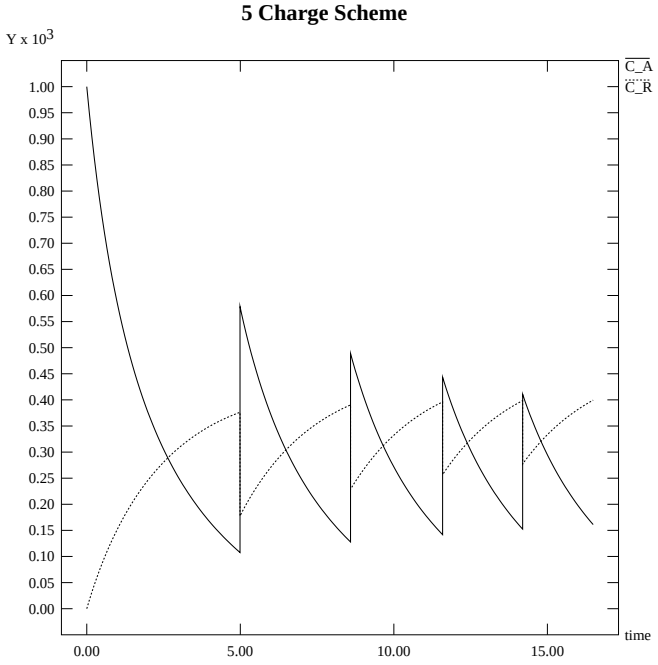

Figure 3.3. Concentration profiles for 5 charges

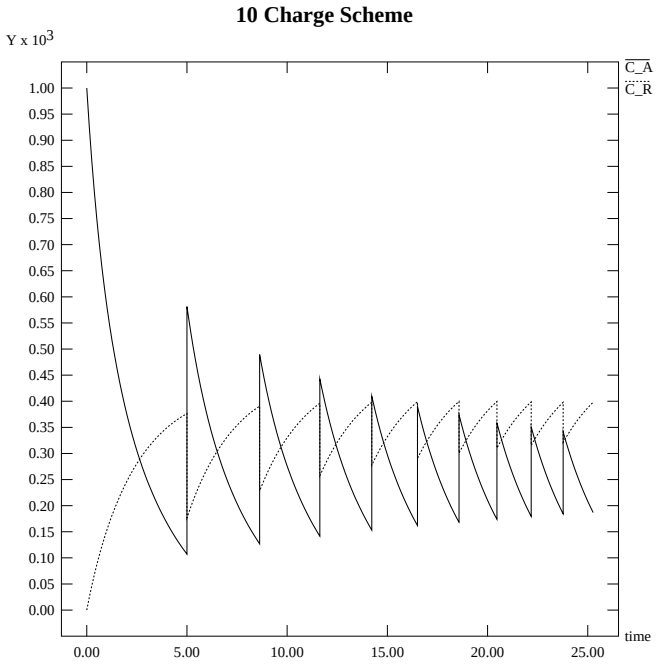

Figure 3.4. Concentration profiles for 10 charges

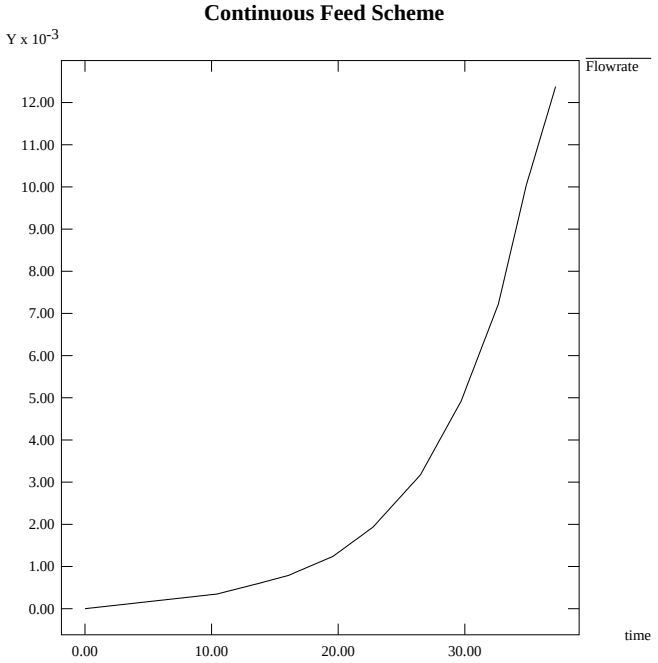

Figure 3.5. Feed flowrate profile

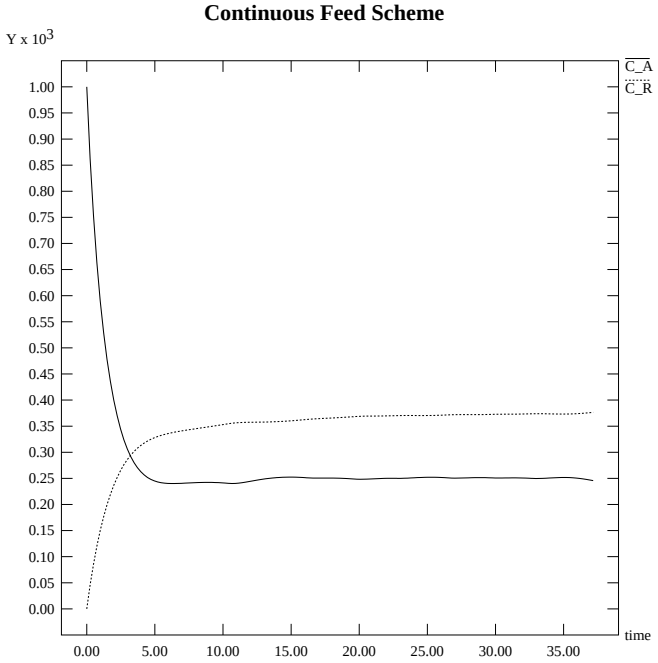

Figure 3.6. Concentration profiles for continuous feed

3.2 Hot-Spot Reactor

The problem is that solved by Vasantharajan and Biegler (1990), based on a model for a packed bed reactor by Finlayson (1971).

Two reactants, A and B, enter the reactor in a ratio 3:1 B/A where an exothermic reaction that produces C and D takes place. The reactor is jacketed and steam is produced as heat is removed. The product exchanges heat with the feed in order to preheat it. The process flowsheet is given in Figure 3.7.

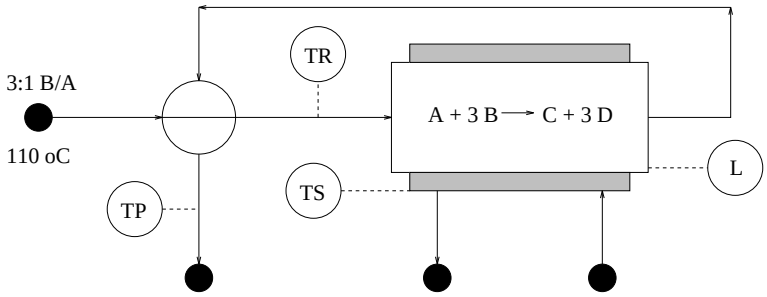

Figure 3.7. Hot-spot reactor flowsheet

Here we assume that the heat capacity of the input stream is the same as that of the reactor effluent.

The modelling equations are:
Objective function

$$\min_{t_f, T_R, T_S, T_P} t_f - P(t_f) \tag{3.11}$$

Mass balance (conversion q)

$$-\dot{q} + 0.3(1.0 - q)\exp(20.0 - 20.0/\bar{T}) = 0; \quad q(0) = 0 \tag{3.12}$$

Energy balance (normalized temperature $\bar{T}$)

$$-\dot{\bar{T}} - 1.5(\bar{T} - T_S/T_R) + 2.0\dot{q}/3.0 = 0; \quad \bar{T}(0) = 1.0 \tag{3.13}$$

Utility Profit (P)

$$-\dot{P} + \bar{T}\,T_R - T_S = 0; \quad P(0) = 0 \tag{3.14}$$

Heat exchanger energy balance constraint

$$(T_R - 110.0) - (\bar{T}(t_f)\,T_R - T_P) = 0 \tag{3.15}$$

				Optimum	
Parameter	Initial Value	L.Bound	U.Bound	This Work	Reference
T_P	250.0	120.0	1000.0	184.0	188.4
T_R	462.23	400.0	500.0	500.0	500.0
L	1.0	0.5	1.25	1.25	1.25
T_S	425.25	400.0	500.0	474.0	470.1

Table 3.4. Hot-spot reactor—Design parameters

Temperature crossover constraint

$$\bar{T}\,T_R - (T_R + 10.0) \geq 0 \tag{3.16}$$

The four time-invariant design parameters are: T_P the specified product temperature, T_R the reactor inlet temperature, L the reactor length[2], and T_S the steam sink temperature.

The starting point and the bounds on the optimization parameters were those used by Vasantharajan. Table 3.4 gives design parameter related values. Table 3.5 gives optimization run statistics for three different tolerance levels. CPU times are reported and also the average ratios of CPU time required to perform a sensitivity/state integration to the CPU time for state-only integration. The optimal parameter values obtained by Vasantharajan and Biegler were used in a simulation run using tolerance level 3, and the corrected objective value is also reported in Table 3.4. It is noted that the parameter values in this work produce a slightly better optimum. Figure 3.8 gives the optimal temperature and conversion profile obtained using level 3 tolerances.

[2]It is noted that L appears as the final time t_f in the equations above as the axial coordinate is used as the independent variable in this problem.

Tol. Level	Objective	Sens./St. CPU	Total CPU	# QPs	# LSs
1	-171.5226	5.9	19.4	5	4
2	-171.4879	4.0	29.1	7	6
3	-171.4874	4.5	53.1	7	6
Reference	-171.4600	—	—	—	—
Ref. corrected	-171.2340	—	—	—	—

Table 3.5. Hot-spot reactor—Solution statistics

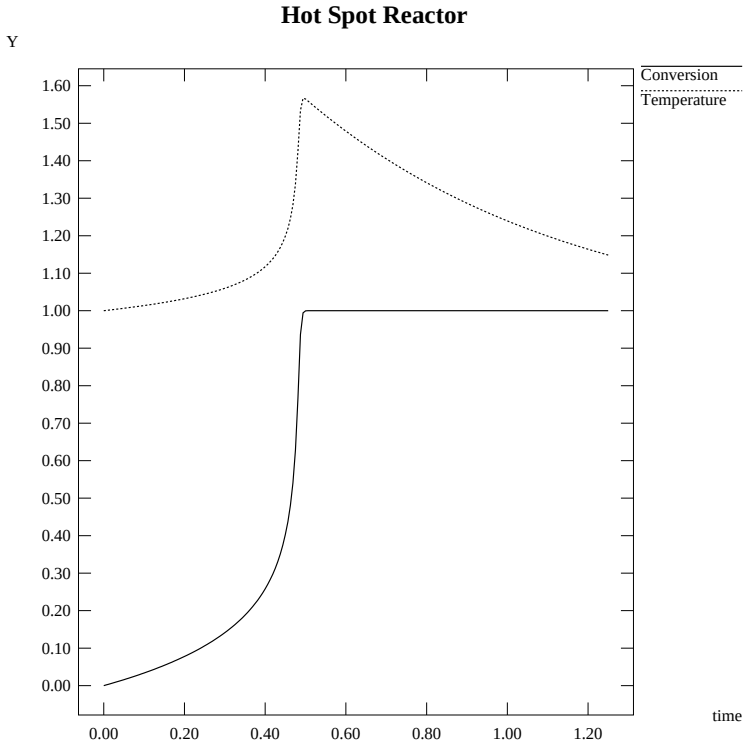

Figure 3.8. Conversion and temperature profiles for hot-spot reactor

3.3 Fed-Batch Reactor

This problem, described in Luus and Rosen (1991), considers protein production under fed-batch reactor conditions. The feed contains the nutrient for the microorganisms used and the product is the protein released by them. The feed rate is the control variable.

The volume of the fluid in the reactor is bounded, thus producing a final point constraint. An upper bound of 22 hr was imposed on the final time which was otherwise free to vary. The performance index to be maximized is the product of the concentration of protein and the culture volume, *i.e.* the total amount of protein produced. The control variable is the feed rate which can be varied between 0 and 10 units.

The modelling equations are:

Objective function

$$\min_{u(\cdot)} J = -x_1(t_f)\, x_5(t_f) \tag{3.17}$$

Differential Equations

$$\begin{aligned}
\dot{x}_1 &= g_1(x_4)\,(x_2 - x_1) - \frac{u}{x_5}\, x_1; && x_1(0) = 0.0 \\
\dot{x}_2 &= g_2(x_4)\, x_3 - \frac{u}{x_5}\, x_2; && x_2(0) = 0.0 \\
\dot{x}_3 &= g_3(x_4)\, x_3 - \frac{u}{x_5}\, x_3 && x_3(0) = 1.0 \\
\dot{x}_4 &= -7.3\, g_3(x_4)\, x_3 + \frac{u}{x_5}\,(20 - x_4); && x_4(0) = 5.0 \\
\dot{x}_5 &= u; && x_5(0) = 1.0 \\
0 &\le t \le t_f = 22.0
\end{aligned} \tag{3.18}$$

Auxiliary relations

$$g_3(x_4) = \frac{21.87\, x_4}{(x_4 + 0.4)(x_4 + 62.5)} \tag{3.19}$$

$$g_2(x_4) = \frac{x_4\, e^{-5\, x_4}}{0.1 + x_4} \tag{3.20}$$

$$g_1(x_4) = \frac{4.75\, g_3(x_4)}{0.12 + g_3(x_4)} \tag{3.21}$$

Control bounds

$$0 \leq u(t) \leq 10 \tag{3.22}$$

Volume final time constraint

$$x_5(t_f) \leq 14.35 \tag{3.23}$$

This case study was used to demonstrate use of different types of control variable approximations within the CP framework. As the evaluation of sensitivities becomes more expensive with an increasing number of parameters, one is forced to make as good an approximation as possible with the least number of control elements. In this spirit, a way is presented to handle difficulties such as bounding of the control variables without having to resort to high degrees of discretization to guarantee bounds satisfaction throughout the trajectory.

Four different cases are considered for the control profile with continuity across the boundary. The tolerance level was set to level 2 of Table 3.1. In all cases, 10 control elements were used, with their lengths being treated as optimization parameters, bounded between 0.1 and 12 hr.

For Case I, piecewise linear control profiles are used. The optimization is started from a uniform control profile having a value of 1.0. The 10 elements are initially of equal length, of 2.2 hr. The optimal control obtained is shown in Figure 3.9. It can be seen that it involves a "bang" section between 7 hr and 9 hr.

For Case II, piecewise quadratic profiles are used, with simple enforcement of the bounds on the control values at the discretization points. The optimization is started from a uniform control profile of 0.0 because it was difficult to locate a starting point that did not produce difficulties in the integration. As shown in Figure 3.10, this case resulted in a control profile violating the bounds, and therefore the solution is unacceptable. The spikes appearing in this profile can be attributed to a very small lower bound on the control element sizes, which may have caused numerical inaccuracies.

For Case III, piecewise quadratic control profiles are used. The optimization is started with an initial control profile obtained by an interpolation of the optimal

Element	Case I	Case II	Case III	Case IV
1	0.37378	0.12288	0.25850	1.99739
2	2.12514	7.56192	2.48663	1.35456
3	2.97047	0.17302	2.38558	0.77387
4	2.89874	1.67341	3.21776	1.20412
5	0.13182	0.10000	0.29994	0.76107
6	1.00796	0.10000	0.86180	2.37009
7	0.10000	9.93555	0.10000	0.30000
8	1.62353	1.75809	0.10000	0.79616
9	9.26891	0.21646	10.0310	11.3394
10	1.49965	0.35867	2.25843	1.10331

Table 3.6. Fed-batch reactor optimal switching element sizes

Index	Case I	Case II	Case III	Case IV
Objective	93.36127	93.59145	93.36215	93.23184
Total CPU	1317.5	3120.5	895.0	1984.31
Sens./St. CPU	9.4	12.5	12.4	9.0
# QPs	60	102	35	94
# LSs	84	175	39	165

Table 3.7. Fed-batch reactor solution statistics

profile of Case I; the "bang" portion of the control profile is constrained by fixing the values of the control discretization points defining the steep lines and the bang segment. The optimal control profile is shown in Figure 3.11.

Finally, Case IV uses piecewise constant profiles and is started from the same point as Case III, in order to check whether a similar profile to that of Cases I and III would be obtained. The optimal control profile thus obtained is shown in Figure 3.12.

In Table 3.6 the optimal switching length distribution is given for the four cases considered, while Table 3.7 summarizes solution statistics.

The best optimal objective function value is obtained by Case III, and the corresponding protein amount profile is shown in Figure 3.13. The value of

93.36215 is marginally better than that of 93.100 reported by Luus and Rosen (1991). One other difference is that, in our solution, the control profile identifies clearly a bang region, which does not appear in the profile of the reference. To investigate this aspect further, Case IV was constructed to see what a piecewise constant profile with variable control elements would look like. A bang region again became evident from about half way through the iterative procedure of the optimization. The objective function value for this case is still better than that reported in the reference which was derived using 20 fixed-step piecewise constant control elements.

The idea of increasing the polynomial order in transition from Case I to Case III is very effective in smoothing out the control profiles as can be seen from Figures 3.9 and 3.11. Also, to bypass the problem of enforcing control bounds with polynomials of higher order for which the control bounding cannot be guaranteed, the idea of fixing those regions of the profile which are more likely to cause a bound violation appears to have worked satisfactorily. In total, Case I and Case III together required 2212.5 CPU seconds, 95 QPs and 123 LSs. On the other hand, Case II required 3120.5 CPU seconds, 102 QPs, 175 LSs and produced an unacceptable solution at a greater expense.

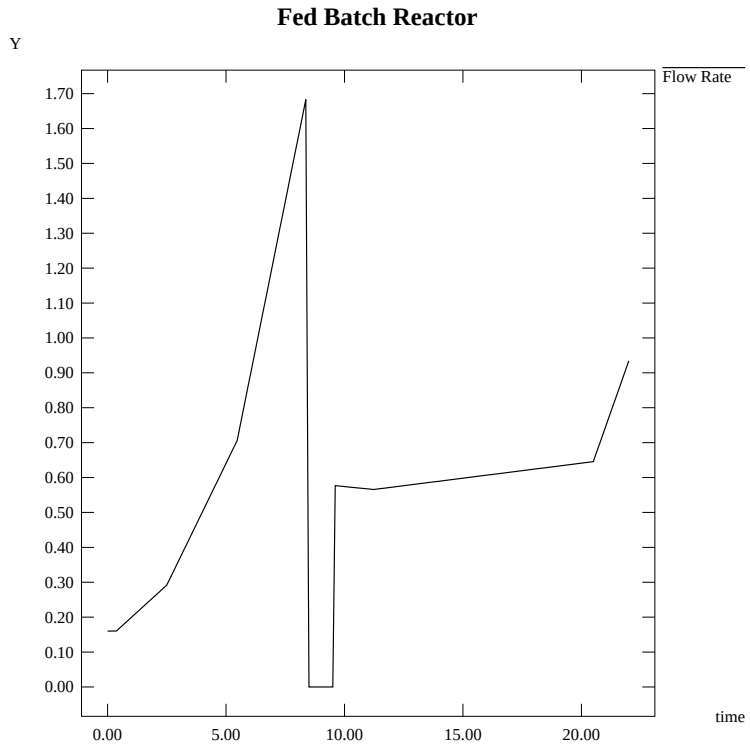

Figure 3.9. Fed batch reactor—Piecewise linear profile

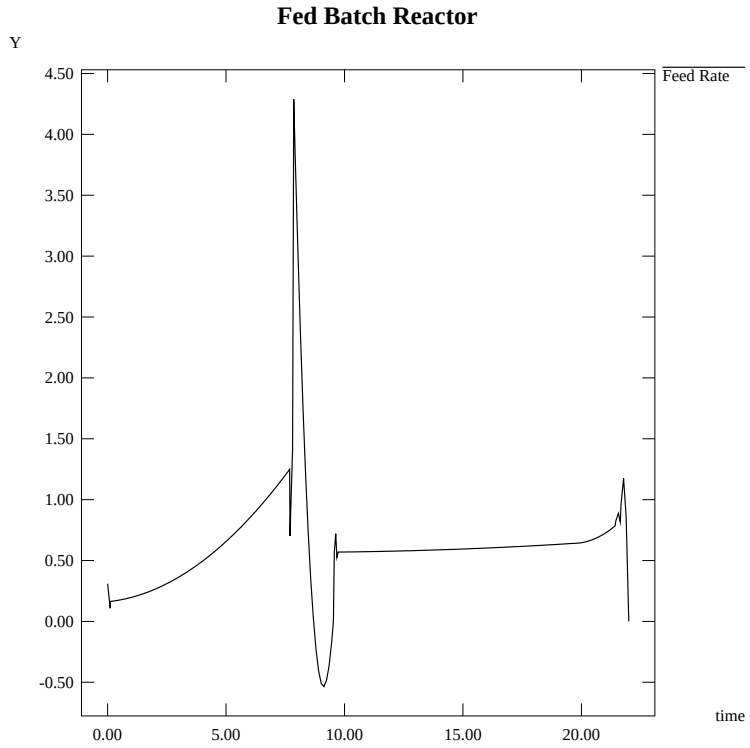

Figure 3.10. Fed batch reactor—Unrestricted piecewise quadratic profile

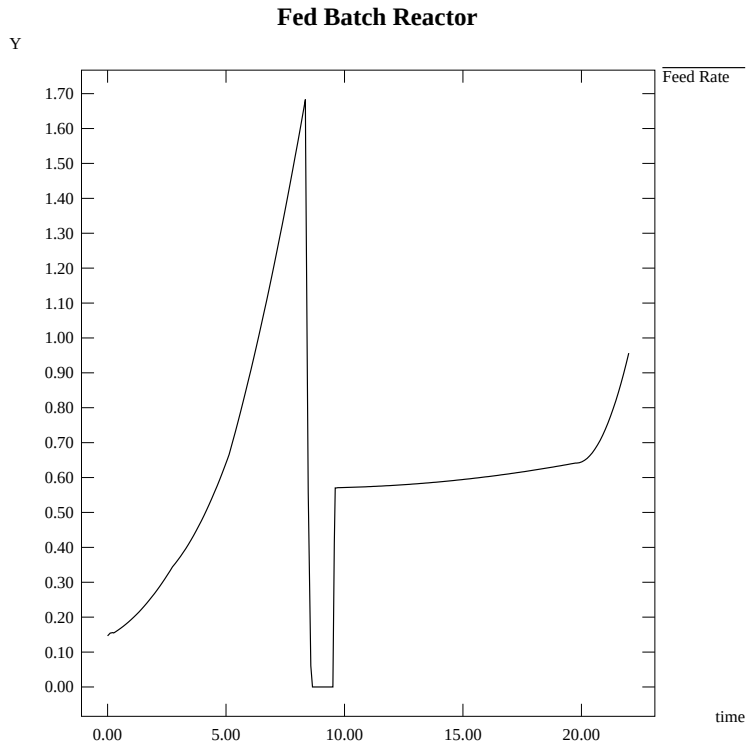

Figure 3.11. Fed batch reactor—Restricted piecewise quadratic profile

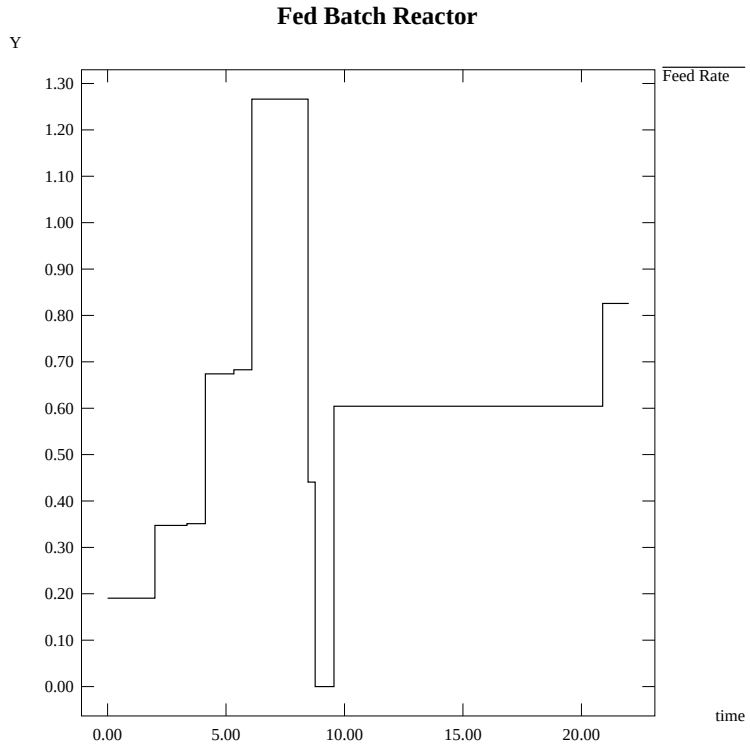

Figure 3.12. Fed batch reactor—Piecewise constant profile

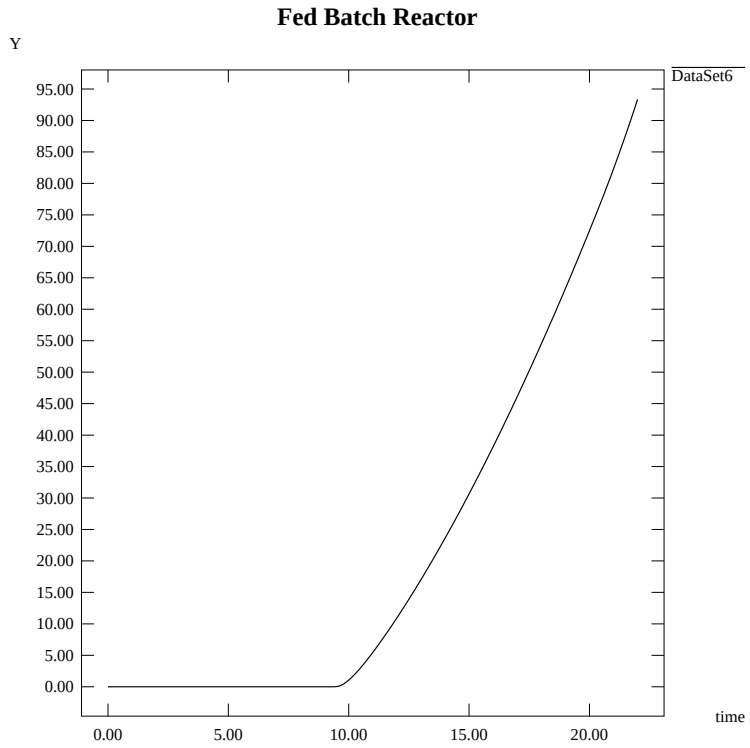

Figure 3.13. Fed batch reactor—Protein production profile

3.4 Boundary Value Problem in a Recycle Reactor

This boundary value problem (BVP) is taken from Kubíček and Hlaváček (1983). It represents a nonadiabatic tubular plug-flow recycle reactor in which a single first-order reaction occurs. The energy and mass balances are:

$$\frac{d\theta}{dx} = 0.40\,(1.0 - y)\,\exp^{\left(\frac{\theta}{1.0+0.05\,\theta}\right)} \tag{3.24}$$

$$\frac{dy}{dx} = 0.10\,(1.0 - y)\,\exp^{\left(\frac{\theta}{1.0+0.05\,\theta}\right)} \tag{3.25}$$

where θ is the dimensionless temperature, y the conversion, and x the dimensionless axial coordinate, $x \in [0, 1]$.

The boundary conditions are:

$$\theta(0) = 0.50\,\theta(1) \tag{3.26}$$

$$y(0) = 0.50\,y(1) \tag{3.27}$$

To solve this problem, the boundary conditions of eqs. (3.26) and (3.27) are simply introduced as end-point constraints of the form:

$$-\,\delta \leq \theta(0) - 0.5\,\theta(1) \leq \delta \tag{3.28}$$

$$-\,\delta \leq y(0) - 0.5\,y(1) \leq \delta \tag{3.29}$$

with the optimal control objective being the minimization of δ. The parameter δ and the initial conditions $\theta(0)$, $y(0)$ are treated as the decision variables.

The problem was solved using all four tolerance levels and a starting point of $[0.5, 0.5]^T$ for the initial conditions. Despite its nonlinearity, this was found to be a relatively easy problem, with only a small number of optimization iterations required for its solution. Table 3.8 summarizes the results obtained with DAEOPT.

Index	Level 1	Level 2	Level 3	Level 4	Reference
$\theta(0)$	0.97055372	0.97047572	0.97047362	0.97047373	0.9705
$y(0)$	0.24263843	0.24261893	0.24261841	0.24261843	0.2426
Total CPU	1.81	1.85	3.88	8.79	—
Sens./Stat.	4.2	3.1	3.8	4.1	—
# QPs	4	4	5	5	—
# LSs	3	3	4	4	—

Table 3.8. Boundary value problem solution statistics

3.5 Large Batch Distillation Problem

To demonstrate the ability of our algorithm and its implementation to handle realistically large problems, a simplified model of a batch distillation column was posed to it. The model is derived from the work of Mujtaba (1989) and describes the distillation of a ternary mixture of components with fixed relative volatilities: $\alpha_{13} = 8.0$, $\alpha_{23} = 4.0$, and $\alpha_{33} = 1.0$ in a column consisting of 18 equilibrium trays, a reboiler and a total condenser (see Figure 3.14). This results in a system of 65 differential and 59 algebraic equations.

The initial load of material in the reboiler was set to 10 $kmole$, and the molar fractions of the feed were $[0.3, 0.2, 0.5]^T$. The same composition was assumed for the trays, the reboiler, the condenser, and the distillate holdup tank.

The task is to minimize the time at which the first cut is produced. The product accumulated must be at least 2.0 $kmole$ and its molar fraction with respect to the first component must be at least 0.900. These two requirements are treated as end-point constraints.

The control variable is the *internal reflux ratio* which was discretized in four elements and was bounded between 0.2 and 0.99. Two cases were considered. For Case I, initially the control was taken to be piecewise constant, with a uniform initial profile equal to 0.90. The element lengths were set to 0.25 (yielding an initial time of 1.0). Case II used a piecewise linear approximation of the control with continuity across the element boundaries. The initial control profile was

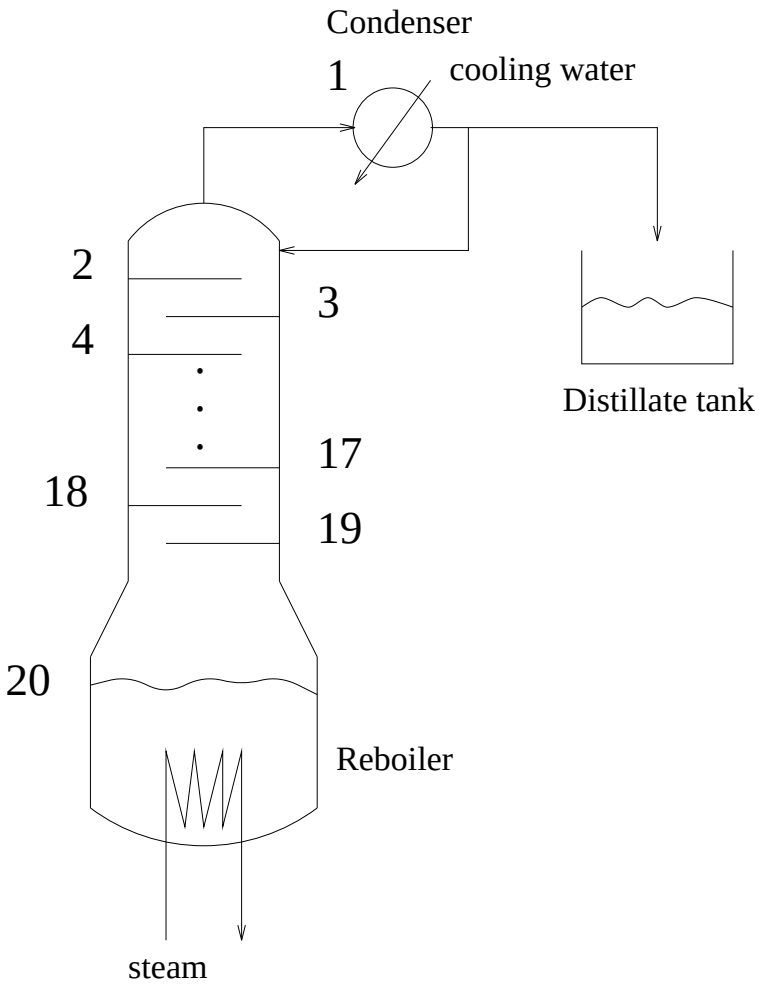

Figure 3.14. Batch distillation column

Element	Case I	Case II
1	0.2200	0.2850
2	1.1200	0.0500
3	0.0500	0.4960
4	0.3090	0.8500

Table 3.9. Large batch distillation problem—Optimal control-element sizes

Index	Case I	Case II
Objective	1.69850	1.68134
Total CPU	7299.0	7929.67
Sens./St. CPU	6.8	9.3
# QPs	40	36
# LSs	77	62

Table 3.10. Large batch distillation problem—Solution statistics

obtained by a rough interpolation of the optimal profile of case I.

The optimal control-element lengths are given in Table 3.9, while Table 3.10 summarizes run information for the two cases. For both cases, the tolerances were fixed at level 1 of Table 3.1, and the element lengths were bounded between 0.05 and 4.0.

The optimal control profiles obtained for the two cases are shown in Figures 3.15 and Figure 3.16 respectively. For case II, which gave a slightly improved performance index compared to that of case I, the molar holdup in the distillate accumulator is shown in Figure 3.17. Figure 3.18 shows the corresponding molar fractions for the accumulator. From these two last figures, it can be seen that the end-point constraints are satisfied very well.

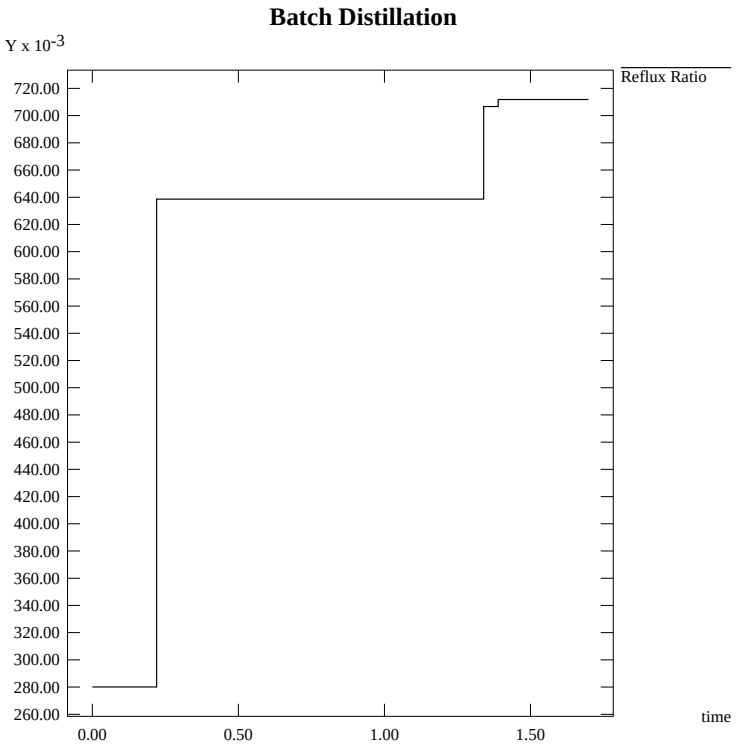

Figure 3.15. Batch distillation problem—Piecewise constant control profile

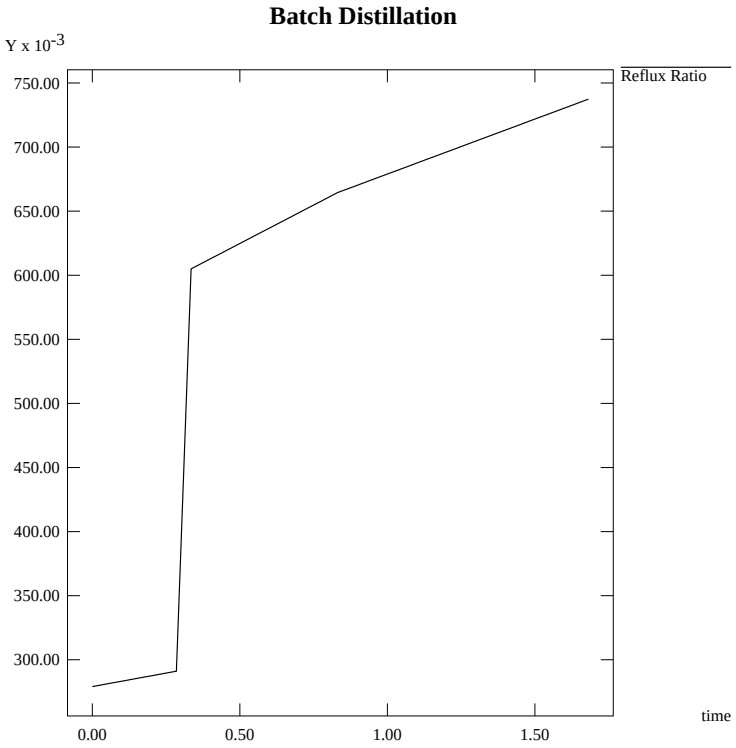

Figure 3.16. Batch distillation problem—Piecewise linear control profile

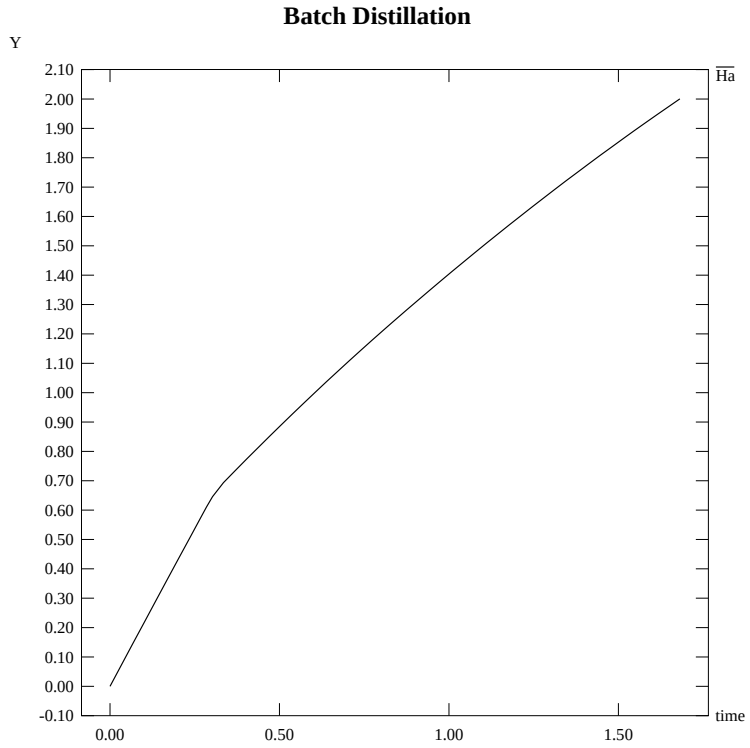

Figure 3.17. Batch distillation problem—Amount of first cut $(kmole)$

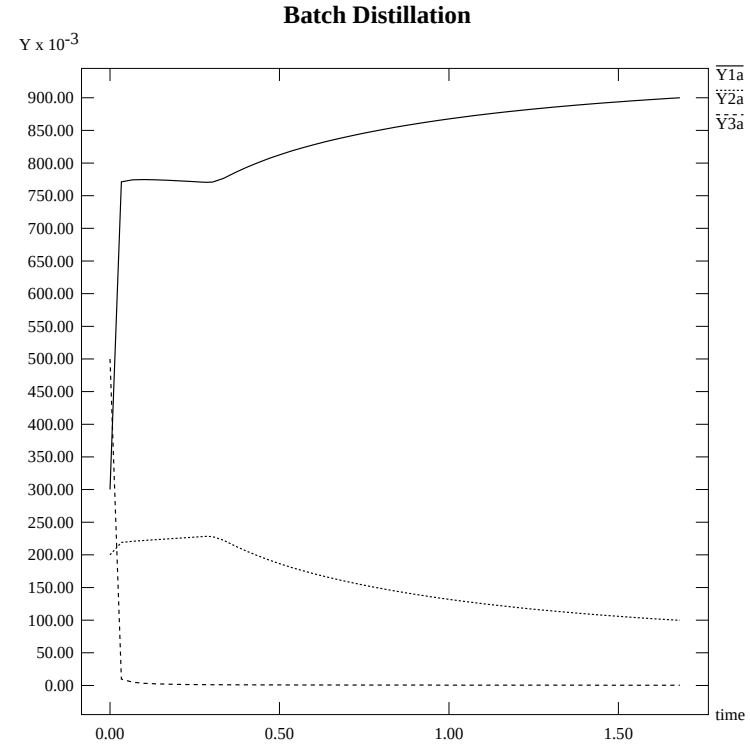

Figure 3.18. Batch distillation problem—Concentration profiles for first cut

3.6 Batch Reactor with Control Saturation

This reactor problem is found in Ray (1981) problem 3.1, pp. 128–129. It has been considered by Biegler (1984), Renfro (1986), and Logsdon and Biegler (1992).

The optimal control problem is:

$$\max_{u(\cdot)} y_2(1.0) \tag{3.30}$$

subject to:

$$\dot{y}_1 = -(u + u^2/2)\, y_1 \tag{3.31}$$

$$\dot{y}_2 = u\, y_1 \tag{3.32}$$

$$y_1(0) = 1.0, \quad y_2(0) = 0.0 \tag{3.33}$$

$$0.0 \leq u(t) \leq 5.0\,; \quad 0.0 \leq t \leq 5.0 \tag{3.34}$$

All tolerance levels of Table 3.1 were used for this problem, in order to study their effect on the solution. The starting point for the control was a uniform profile of 1.0. 10 piecewise linear continuous control elements were used whose lengths were bounded by $0.01 \leq h_i \leq 5.0$. Initially the elements were equispaced.

The solution results are summarized in Table 3.11, and the optimal control element distributions are given in Table 3.12. Figure 3.19 shows the control profile obtained for level 4 tolerances, and Figure 3.20 the corresponding state profiles. The interesting feature of this problem is that the control becomes saturated and the time of saturation found is $t = 0.951252$ for level 4 tolerances.

The results reported in Table 3.11 show that the CPU time for level 4 is quite excessive. However, the results obtained for Level 2 seem quite adequate for practical purposes. Nevertheless, the purpose here is to show how the implementation of the algorithm works in extreme situations, a theme that prevails in the examples to follow.

Level	Objective	Sens./St. CPU	Total CPU	# QPs	# LSs
1	0.5728287	4.5	32.0	15	29
2	0.5735300	5.4	93.2	26	46
3	0.5735387	6.1	218.6	40	61
4	0.5735429	7.3	1144.9	108	136
Biegler[3]	0.5726300	—	—	—	—
Biegler[4]	0.5734900	—	—	—	—
Biegler[5]	0.5691000	—	—	—	—
Renfro[6]	0.5700000	—	—	—	—

Table 3.11. Batch reactor with control saturation—Solution statistics

Element	Level 1	Level 2	Level 3	Level 4
1	0.1040	0.0100	0.0146	0.2310
2	0.1190	0.0100	0.0100	0.1890
3	0.1500	0.1850	0.2710	0.1500
4	0.1280	0.2610	0.1890	0.1180
5	0.0755	0.1670	0.1620	0.0904
6	0.0996	0.1390	0.1190	0.0675
7	0.0909	0.0579	0.0865	0.0485
8	0.0879	0.0726	0.0615	0.0338
9	0.0657	0.0520	0.0384	0.0230
10	0.0801	0.0444	0.0471	0.0487

Table 3.12. Batch reactor with control saturation—Element sizes

[3]Using global collocation on 5 points.

[4]Using control vector iteration (CVI).

[5]Using CP, with a quadratic profile.

[6]Using splines and 10 piecewise constant controls.

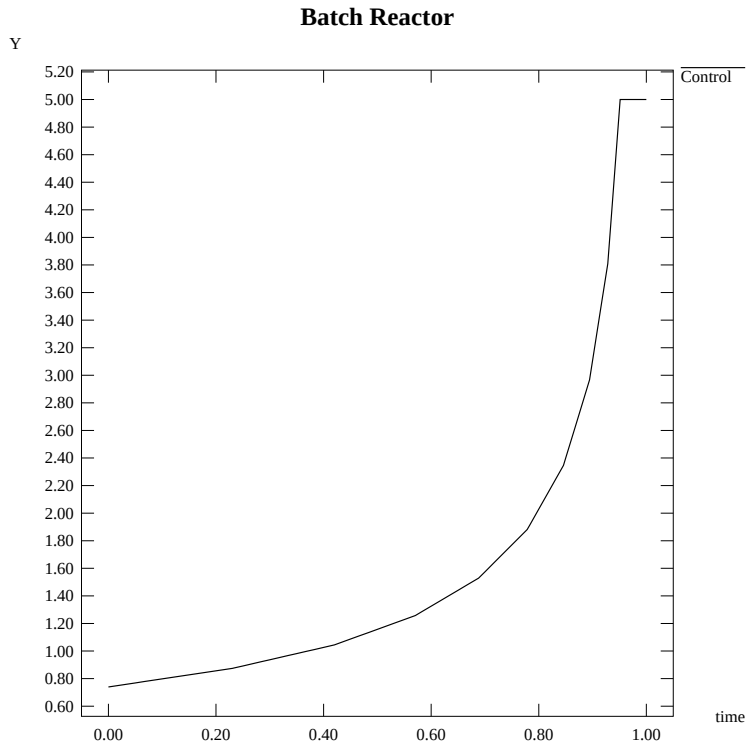

Figure 3.19. Batch reactor with control saturation—Control profile

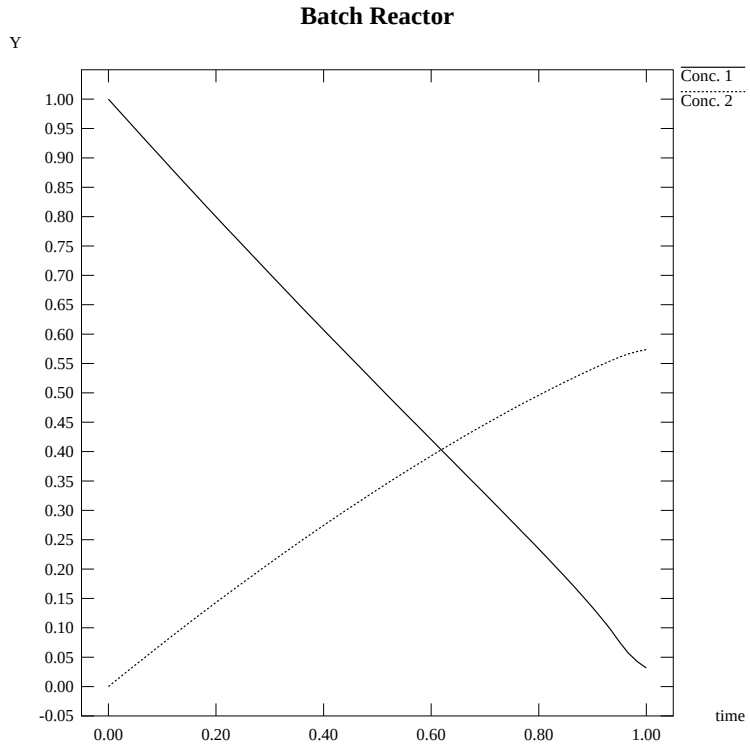

Figure 3.20. Batch reactor with control saturation—Concentration profiles

3.7 Catalyst Mixing Problem

Here, the optimal mixing policy of two catalysts along the length of a tubular reactor is to be determined. The control variable is the mixing ratio, and its profile exhibits a singular arc.

The problem was proposed by Gunn and Thomas (1964), and was solved numerically using orthogonal collocation on finite elements by Logsdon (1990):

$$\max_{u(\cdot)} y_3(1.0) \tag{3.35}$$

subject to:

$$\dot{y}_1 = u\,(10.0y_2 - y_1) \tag{3.36}$$

$$\dot{y}_2 = u\,(y_1 - 10.0y_2) - (1.0 - u)\,y_2 \tag{3.37}$$

$$y_3 = 1.0 - y_1 - y_2 \tag{3.38}$$

$$y_1(0) = 1.0, \quad y_2(0) = 0.0 \tag{3.39}$$

$$0.0 \leq u(t) \leq 1.0, \quad 0 \leq t \leq 1 \tag{3.40}$$

Six piecewise constant sections were used, and three cases were examined in order to see how well the algorithm matches the theoretical profile at various tolerances. The initial profile was taken to be uniform of value 0.170, and the elements were of equal length with lower bounds on their length equal to 0.01 and upper bounds 0.8. The first three tolerance levels of Table 3.1 were employed.

The results are given in Table 3.13, and the control element distributions in Table 3.14. Figures 3.21, 3.22, and 3.23 show the corresponding control profiles for tolerance levels 1–3 respectively. Figure 3.24 show the corresponding state profiles for level 3. It is observed that levels 1 and 2 are inadequate for resolving the singular arc, whereas level 3 produces the expected optimal policy. On the other hand, it is worth noting that, as far as the objective function value is concerned, the two suboptimal profiles are very close to the optimal one.

Level	Objective	Sens./St. CPU	Total CPU	# QPs	# LSs
1	0.0478645	3.8	5.42	5	6
2	0.0480469	4.3	14.4	8	9
3	0.0480557	4.7	38.1	14	15

Table 3.13. Catalyst mixing problem—Solution statistics

Element	Level 1	Level 2	Level 3
1	0.1630	0.1360	0.1360
2	0.1550	0.1690	0.1650
3	0.1430	0.1360	0.1280
4	0.1500	0.1540	0.1490
5	0.1660	0.1590	0.1470
6	0.2240	0.2460	0.2750

Table 3.14. Catalyst mixing problem—Optimal element sizes

Logsdon (1990) obtained a solution using an A-stable collocation method which required 47 major iterations of MINOS and 53.8 CPU seconds (VAX 6320), and the L-stable method used in that work required 42 major iterations and 113.9 CPU seconds. The singular arc is found to occur between 0.13632 and 0.72474 in this work, and 0.13674 and 0.72815 in Logsdon. The average value of the mixing ratio in the singular arc is 0.2273 in this work, and 0.2258 in Logsdon. The results mentioned for this work were obtained using tolerance level 3.

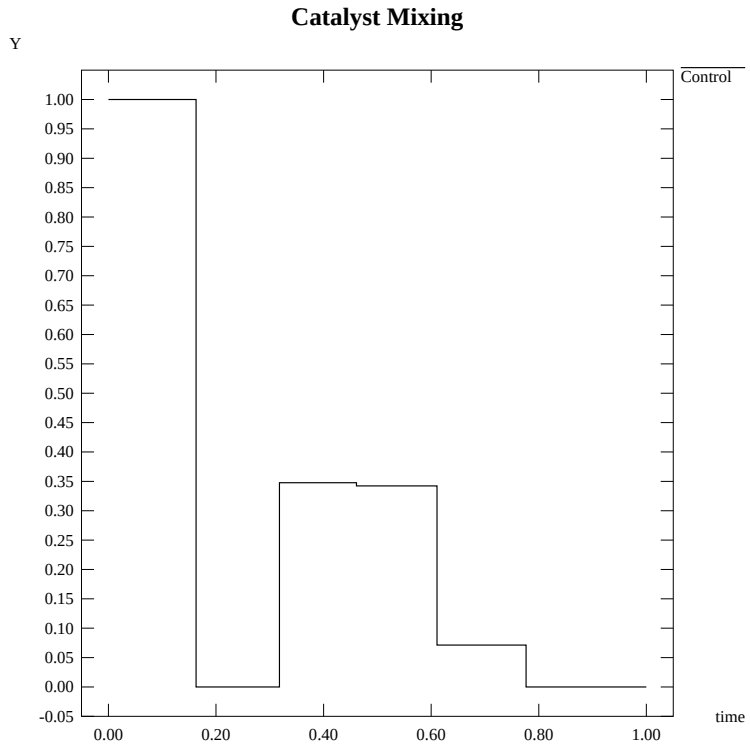

Figure 3.21. Catalyst mixing problem—Level 1 tolerance ratio profile

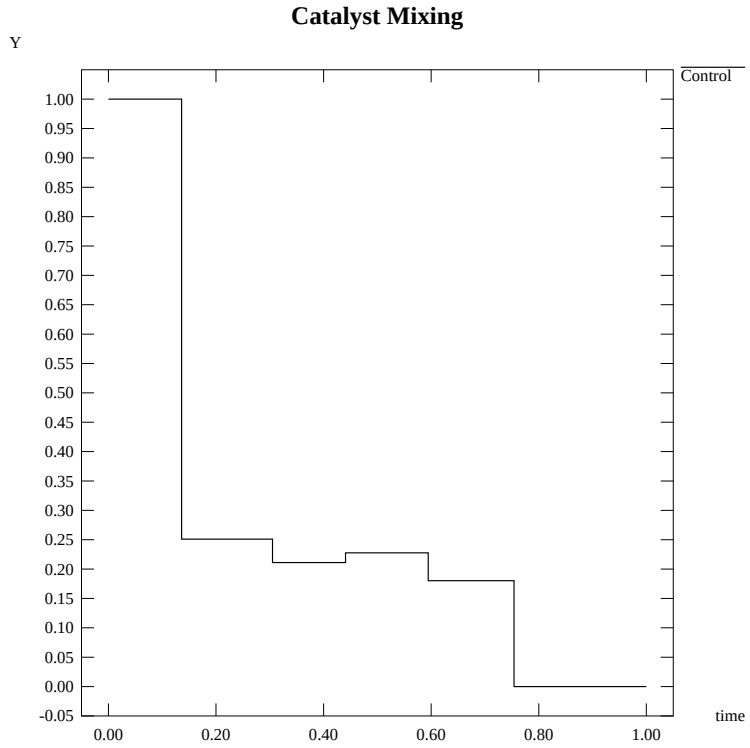

Figure 3.22. Catalyst mixing problem—Level 2 tolerance ratio profile

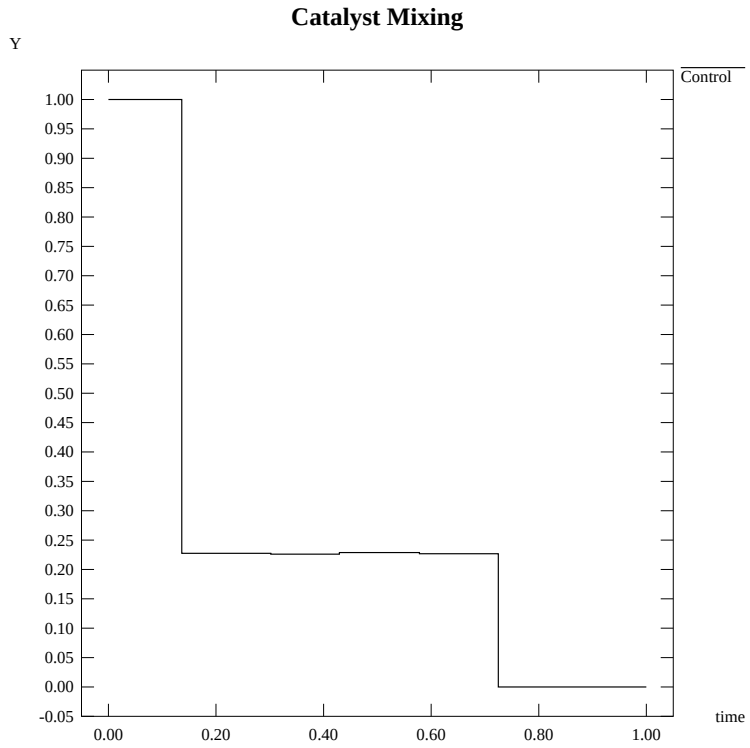

Figure 3.23. Catalyst mixing problem—Level 3 tolerance ratio profile

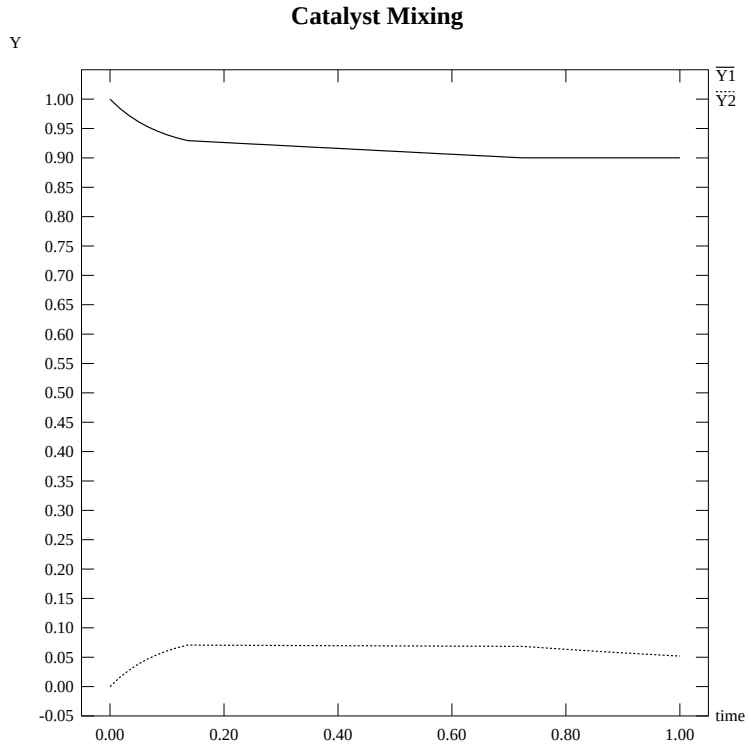

Figure 3.24. Catalyst mixing problem—Level 3 tolerance states

3.8 Consecutive Reaction Scheme Problem

In this problem, two first order reactions take place in series and the task is to maximize the concentration of the intermediate product. The control variable is the reaction temperature. The optimal control problem is:

$$\max_{u(\cdot)} y_2(25.0) \tag{3.41}$$

subject to:

$$\dot{y}_1 = -k_1\, y_1 \tag{3.42}$$

$$\dot{y}_2 = k_1\, y_1 - k_2\, y_2 \tag{3.43}$$

$$k_1 = 65.50 \exp^{-5027.7/u} \tag{3.44}$$

$$k_2 = 1970.0 \exp^{-8044.3/u} \tag{3.45}$$

$$600.0 \leq u(t) \leq 1000.0, \quad 0 \leq t \leq 25 \tag{3.46}$$

The problem was considered by Vassiliadis (1991) using a complete state and control variable discretization approach. First-order backward differences over 50 fixed elements were employed to approximate time derivatives, and an SQP algorithm was used to solve the resulting NLP.

For the solution with DAEOPT, all four tolerance levels were used. The controls were set to be piecewise linear profiles with continuity over 10 switching elements. The lower bounds on the element sizes were set at 1.0, 0.5, 0.1, and 0.05 respectively for the four tolerance levels, whereas the upper bounds were fixed at 15.0. The initial control profiles for all the cases were uniform at 800 K, and the control elements were of equal length.

Figure 3.25 depicts the profile for the control using level 1 tolerances. As can clearly be seen, it contains a spike which destroys the monotonicity of the profile. However, all other tolerance levels (2–4), for which the lower bounds on the element lengths were reduced, gave monotonic profiles. Figure 3.26 shows the control profile for level 4, and in Figure 3.27 the corresponding state profiles are presented. The solution results are summarized in Table 3.15, and the control element distributions are given in Table 3.16.

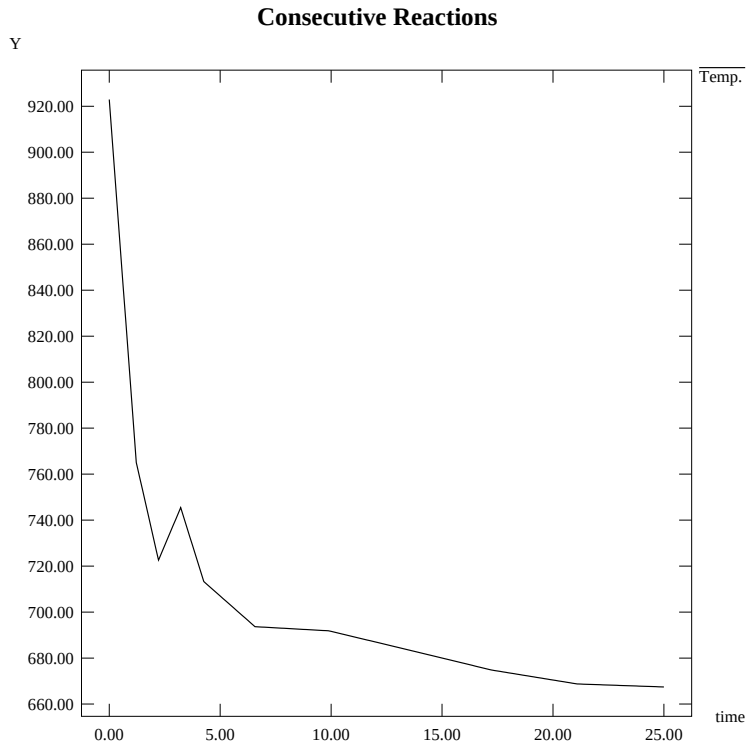

Figure 3.25. Consecutive reactions problem—Level 1 tol. control

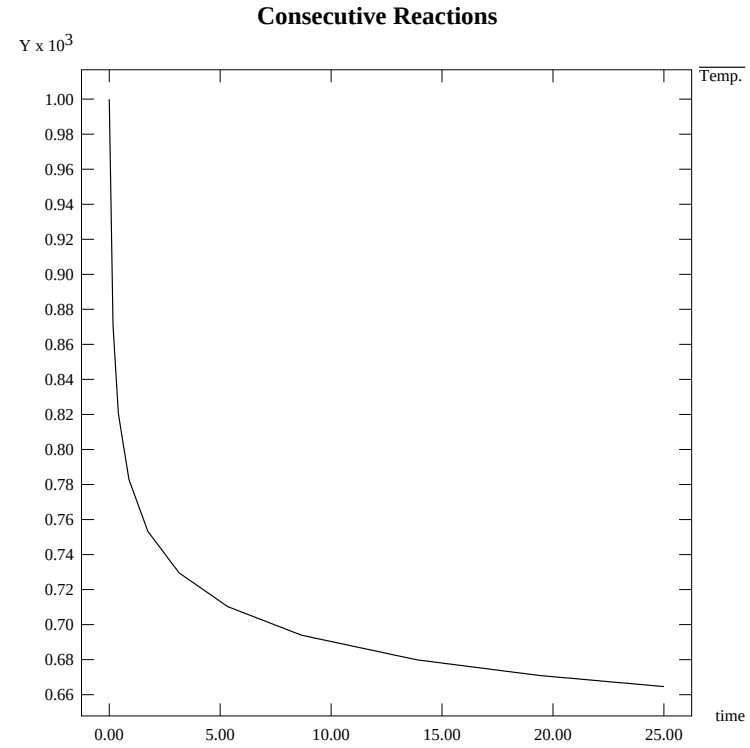

Figure 3.26. Consecutive reactions problem—Level 4 tol. control

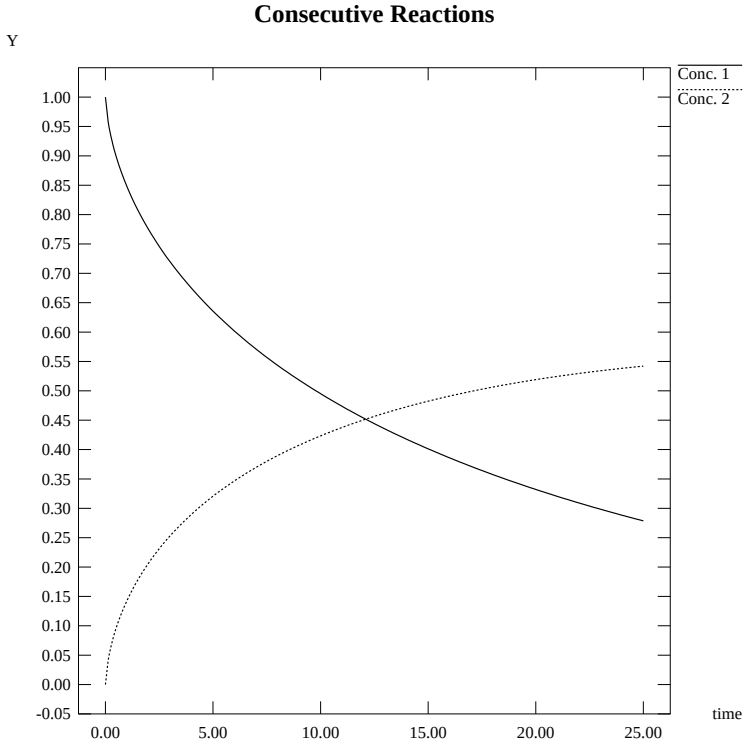

Figure 3.27. Consecutive reactions problem—Level 4 tol. concentrations

Level	Objective	Sens./St. CPU	Total CPU	# QPs	# LSs
1	0.5414405	5.6	26.6	10	12
2	0.5419570	6.6	92.4	18	20
3	0.5420010	7.8	222.3	27	31
4	0.5420100	9.0	3337.5	181	203
Vassiliadis (1991)	0.5348800	—	—	—	—

Table 3.15. Consecutive reactions problem—Solution statistics

Element	Level 1	Level 2	Level 3	Level 4
1	0.1220	0.5000	0.2290	0.1700
2	1.0000	0.6430	0.9710	0.2410
3	1.0000	0.5000	2.5300	0.4760
4	1.0400	0.9450	2.5600	0.8520
5	2.3100	3.3300	2.7500	1.4100
6	3.3200	3.2100	2.9700	2.1900
7	3.6100	3.5200	3.1200	3.3300
8	3.7200	4.0700	3.2400	5.2100
9	3.8700	4.2400	3.3100	5.5300
10	3.9100	4.0400	3.3200	5.5900

Table 3.16. Consecutive reactions problem—Optimal element sizes

It is observed that the algorithm automatically clusters the control elements in the steep region of the profile, thus providing a finer approximation there.

3.9 Nuclear Reactor Model

The problem is described by Sage and White (1977). It has been solved by Van Dooren (1989) and Luus and Rosen (1991).

The mathematical statement of the problem is:

$$\min_{u(\cdot)} y_3(1) \tag{3.47}$$

subject to:

$$\dot{y}_1 = 1000(u - 0.0064)\, y_1 + 0.1\, y_2 \tag{3.48}$$

$$\dot{y}_2 = 6.4\, y_1 - 0.1\, y_2 \tag{3.49}$$

$$y_1(0) = 0.5, \quad y_2(0) = 32.0 \tag{3.50}$$

$$y_1(1) = 5.0 \tag{3.51}$$

$$0.0 \leq u(t) \leq 12.0 \times 10^{-3}, \quad 0 \leq t \leq 1 \tag{3.52}$$

The upper bound on the control was not originally present in the model. The reason for adding it is that very large values of the control were found to result in an explosion in the state variables, thus rendering the integrations impossible. However, it should be noted that the control never actually reaches the imposed upper bound in the optimal solutions presented below.

The initial control profile was taken to be uniform with a value of 5.0×10^{-3}. 10 piecewise linear elements were used, with continuity across the element boundaries. They were initially set to be of equal length with lower bounds on their length equal to 0.05 and upper bounds 0.5. The first three tolerance levels were used to solve the problem. The solution is summarized in Table 3.17, and the corresponding element size distributions are given in Table 3.18. Figures 3.28 and 3.29 show the control and state profiles, respectively, for level 3 tolerances.

State variable y_1, at time t_f, has values 4.999107, 4.999999, and 5.000000 for the three tolerance levels respectively. Thus the end-point constraint (3.51) is satisfied very well at all levels.

Level	Objective	Sens./St. CPU	Total CPU	#QPs	# LSs
1	1.76714×10^{-5}	9.8	47.0	10	9
2	1.76736×10^{-5}	6.5	94.2	21	21
3	1.76736×10^{-5}	7.6	165.1	22	21
Van Dooren	1.767×10^{-5}	—	—	—	—
Luus & Rosen	1.7735×10^{-5}	—	—	—	—

Table 3.17. Nuclear reactor model—Solution statistics

Element	Level 1	Level 2	Level 3
1	0.0509	0.0724	0.0727
2	0.0510	0.0684	0.0687
3	0.0500	0.0640	0.0643
4	0.0503	0.0796	0.0799
5	0.1040	0.1060	0.1060
6	0.2430	0.1220	0.1200
7	0.2050	0.2020	0.2020
8	0.1220	0.1240	0.1240
9	0.0715	0.0866	0.0870
10	0.0521	0.0757	0.0758

Table 3.18. Nuclear reactor model—Optimal element sizes

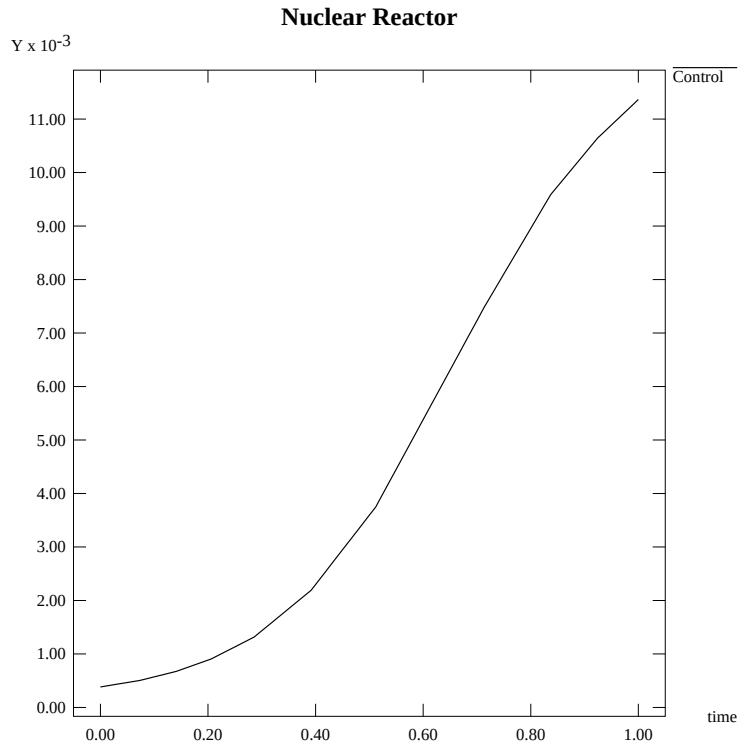

Figure 3.28. Nuclear reactor model—Control profile

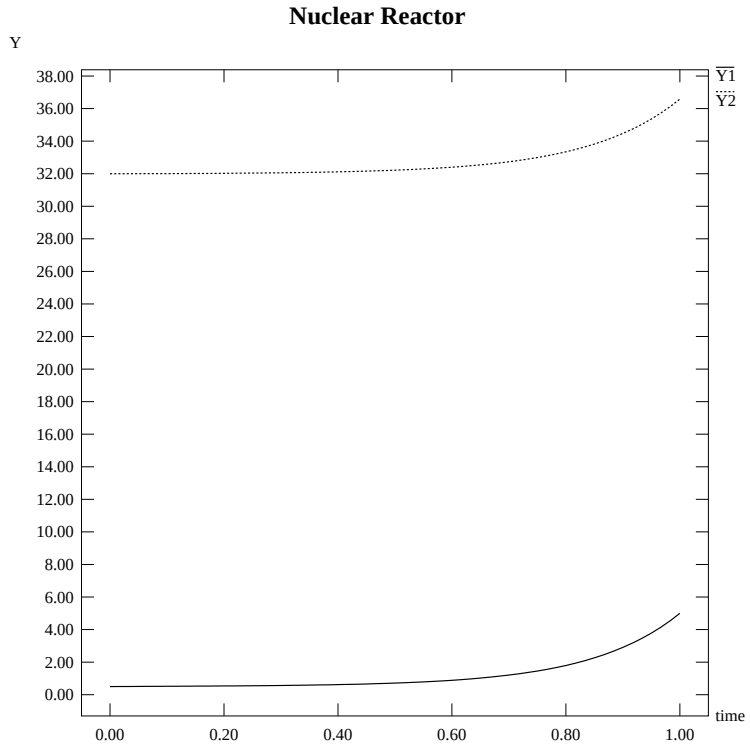

Figure 3.29. Nuclear reactor model—State profiles

3.10 Optimal Control of a CSTR

The model is that of a continuous stirred tank reactor (CSTR) given in Aris and Amudson (1958a&b) and used by Luus and Rosen (1991).

The problem is rewritten for the purposes of this work as:

$$\min_{u(\cdot)} y_3(0.78) \tag{3.53}$$

subject to:

$$\dot{y}_1 = -(y_1 + 0.25) + (y_2 + 0.5)\exp^{\frac{25y_1}{y_1+2}} - (1+u)(y_1 + 0.25) \tag{3.54}$$

$$\dot{y}_2 = 0.5 - y_2 - (y_2 + 0.5)\exp^{\frac{25y_1}{y_1+2}} \tag{3.55}$$

$$\dot{y}_3 = y_4 + 0.1u^2 \tag{3.56}$$

$$y_4 = y_1^2 + y_2^2 \tag{3.57}$$

$$y(0) = [0.09, 0.09, 0.0]^T \tag{3.58}$$

$$y_4(0.78) \leq \epsilon \tag{3.59}$$

$$0 \leq u(t) \leq 7, \quad 0 \leq t \leq 0.78 \tag{3.60}$$

In Luus and Rosen, no bounds on the control were given. The bounds shown above are simply imposed to restrict the control during the optimization procedure, without affecting the optimal control profiles. The parameter ϵ in this case is set to 10^{-6}. This is sufficient to guarantee that the end-point constraint will be satisfied by an order of magnitude closer to zero than the corresponding value obtained by Luus and Rosen of 2.73×10^{-5}. The initial control profile was set to be uniform with value 7.0. 10 control elements were used, initially equispaced with lower bounds on their length equal to 0.01 and upper bounds equal to 0.5. The control is approximated by piecewise linear segments with continuity. Table 3.19 gives a summary of the solution obtained with the first 3 tolerance levels, and Table 3.20 gives the control element distribution. Figures 3.30 and 3.31 show the control (u) and state profiles (y_1 and y_2), respectively.

From Table 3.19 it can be observed that the solution of Luus and Rosen gives a lower value for the objective function than that obtained here at level 3

Level	Objective	Sens./St. CPU	Total CPU	# QPs	# LSs
1	0.14992	10.3	185.0	30	40
2	0.14911	6.3	251.0	46	70
3	0.14838	7.1	661.0	78	102
Luus & Rosen	0.14490	—	—	—	—

Table 3.19. Optimal control of a CSTR—Solution statistics

Element	Level 1	Level 2	Level 3
1	0.0110	0.0100	0.0102
2	0.0100	0.0172	0.0133
3	0.0441	0.0100	0.0217
4	0.0100	0.0418	0.0100
5	0.0100	0.0100	0.0209
6	0.0100	0.0100	0.0242
7	0.1270	0.0100	0.0104
8	0.1450	0.0463	0.0877
9	0.1650	0.2100	0.1390
10	0.2470	0.4150	0.4430

Table 3.20. Optimal control of a CSTR—Optimal element sizes

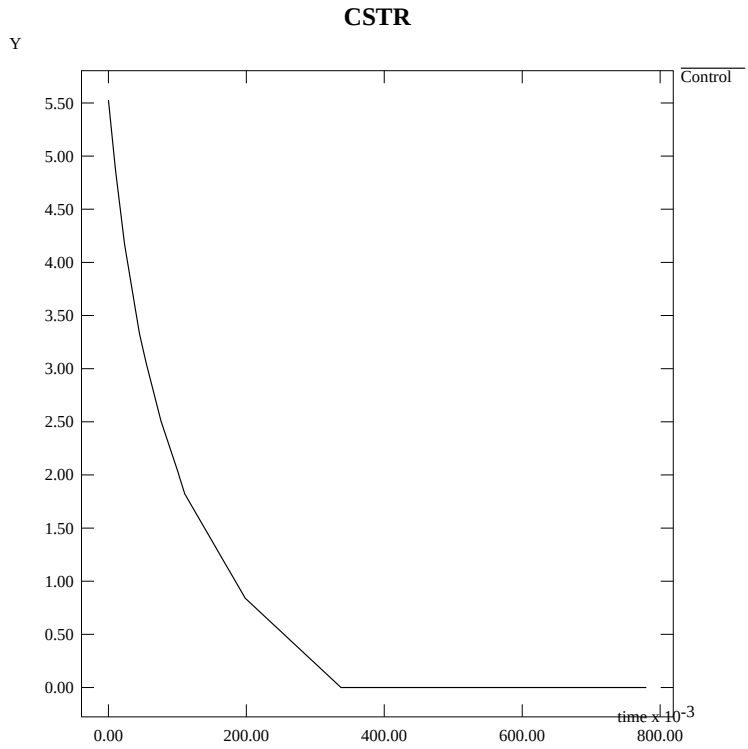

Figure 3.30. Optimal control of a CSTR—Control profile

tolerance. However the solution obtained in this work yields a lower value for the end-point constraint, thus a certain increase in the optimal performance index is to be expected. The time after which the control profile drops to zero is found to be $t = 0.337236$ for level 3, as opposed to $t = 0.39$ given in Luus and Rosen.

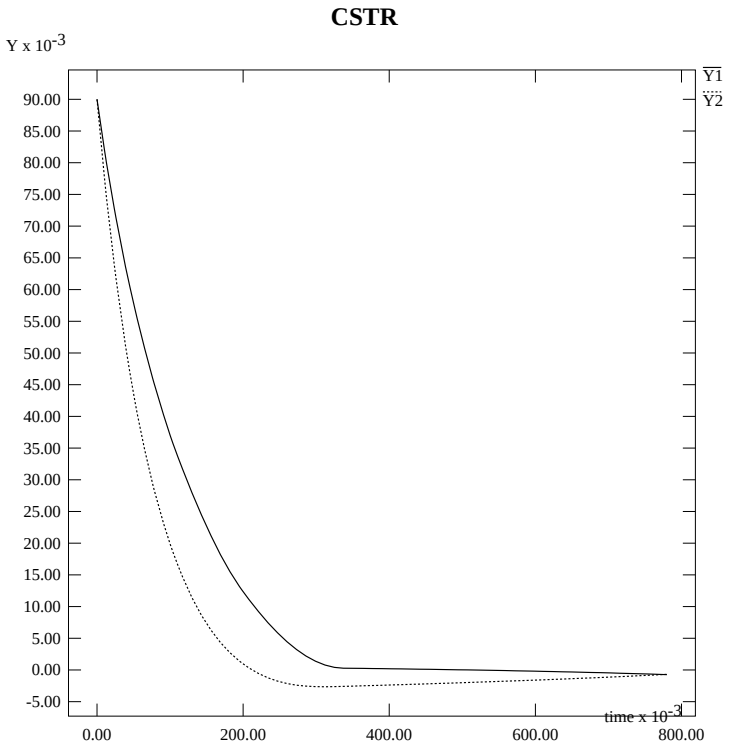

Figure 3.31. Optimal control of a CSTR—State profiles

3.11 Isoperimetric Problem

This is a classical problem, *e.g.* as described in Luenberger (1979) as example x2, on pp. 408–409. A curve is to be determined that connects two fixed points on the t-axis, has a fixed arc length, and encloses the maximum area between itself and the t-axis, as shown in Figure 3.32. The optimal control problem is formulated as follows:

$$\max_{u(\cdot)} y_2(1) \tag{3.61}$$

subject to:

$$\dot{y}_1 = u \tag{3.62}$$

$$\dot{y}_2 = y_1 \tag{3.63}$$

$$\dot{y}_3 = \sqrt{1.0 + u^2} \tag{3.64}$$

$$y(0) = [0, 0, 0]^T, \quad y_1(1) = 0.0, \quad y_3(1) = 1.5 \tag{3.65}$$

$$-100 \leq u(t) \leq +100, \quad 0 \leq t \leq 1 \tag{3.66}$$

The problem here is specified to have an arc length 1.5, and to return the curve to the t-axis at the final time. The variables are: y_1 the curve height above the horizontal axis; y_2 the area under the curve; y_3 the arc length; u the slope of the curve (the control variable).

The solution to this problem is known to be a circular arc. It may be obtained directly by solving the following nonlinear equation for the radius R (see

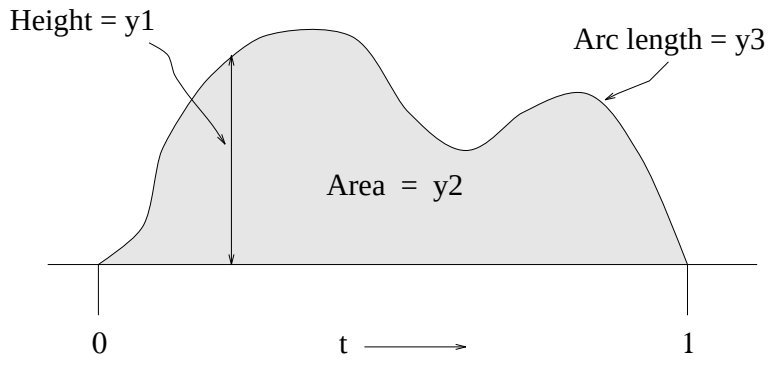

Figure 3.32. Isoperimetric problem description

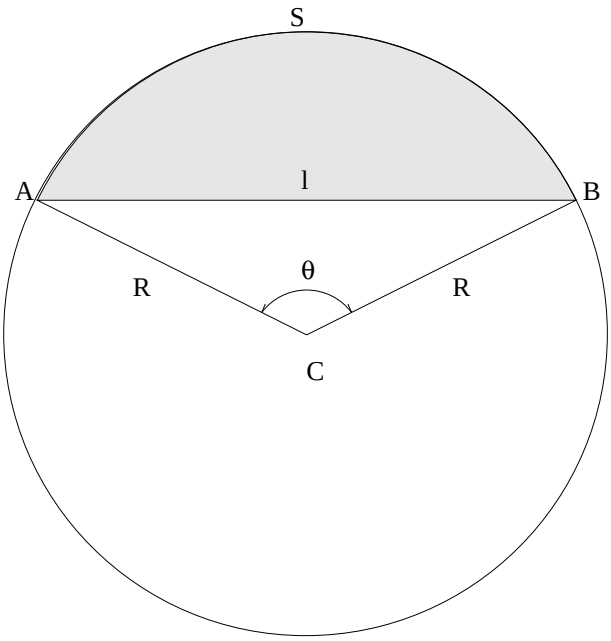

Figure 3.33. Area between circular arc and chord

Figure 3.33):

$$l = 2\,R\,\sin\left[\frac{2\,\pi\,R - S}{2\,R}\right] \tag{3.67}$$

where l is the length of the chord and S is the length of the corresponding arc. The angle to the endpoints of the arc is:

$$\theta = \frac{S}{R} \tag{3.68}$$

Upon substitution of $l = 1.0$ and $S = 1.5$, the solution of the two equations above yields $R = 0.50141$ and $\theta = 2.991564$. The arc length involved is that which is smaller than the half-circle. Therefore the desired area is given by:

$$A = \frac{1}{2}\,\theta\,R^2 - R\,\cos\left[\frac{2\,\pi - \theta}{2}\right] \tag{3.69}$$

and the calculated value is 0.3572687. The value of the diameter $(2\,R)$ is very close to the specified chord length. This was a deliberate choice, as the shape of the arc will almost be a half-circle and consequently the slope at the end-points will be very steep, in this way providing a test for the algorithm in regions of steep variations of the controls.

To test the efficiency of the algorithm when fewer control elements are supplied, only five elements were provided. Initially, they were equidistributed and

Level	Objective	Sens./St. CPU	Total CPU	# QPs	# LSs
1	0.3557471	5.2	100.3	46	84
2	0.3564474	4.5	130.8	49	55
3	0.3569133	4.9	303.4	75	92
3-quadr.	0.3571153	7.9	562.4	49	74
3-cubic.	0.3572240	7.7	1047.3	82	120
Analytical	0.3572687	—	—	—	—

Table 3.21. Isoperimetric problem—Solution statistics

Element	Level 1	Level 2	Level 3	Level 3-quadr.	Level 3-cubic
1	0.0780	0.0500	0.0500	0.0500	0.0716
2	0.0783	0.1170	0.1500	0.1980	0.4830
3	0.2070	0.4410	0.6000	0.5500	0.2850
4	0.5830	0.2780	0.1500	0.1520	0.1110
5	0.0538	0.1140	0.0500	0.0500	0.0500

Table 3.22. Isoperimetric problem—Optimal element sizes

bounded by $0.05 \leq h_i \leq 0.6$. The control profiles were specified to be piecewise linear with continuity at the element boundaries. The initial control profile was a uniform one of value 0.2. The problem was then solved using tolerance levels 1–3 of Table 3.1.

Following the solution using linear profiles, the control approximation was changed to piecewise quadratic with continuity, and finally to piecewise cubic with continuity. These two cases were solved at level 3 tolerances, using an interpolation of the previous solution as an initial point.

The solutions are summarized in Table 3.21, and the control element size distributions are given in Table 3.22.

The optimal control profiles obtained are depicted in several plots: Figures 3.34, 3.35, 3.36 show the results for tolerances 1–3, respectively, using the linear control approximations; Figures 3.37 and 3.38 show the control profiles obtained using the quadratic and cubic approximations, respectively; Figure 3.39 shows

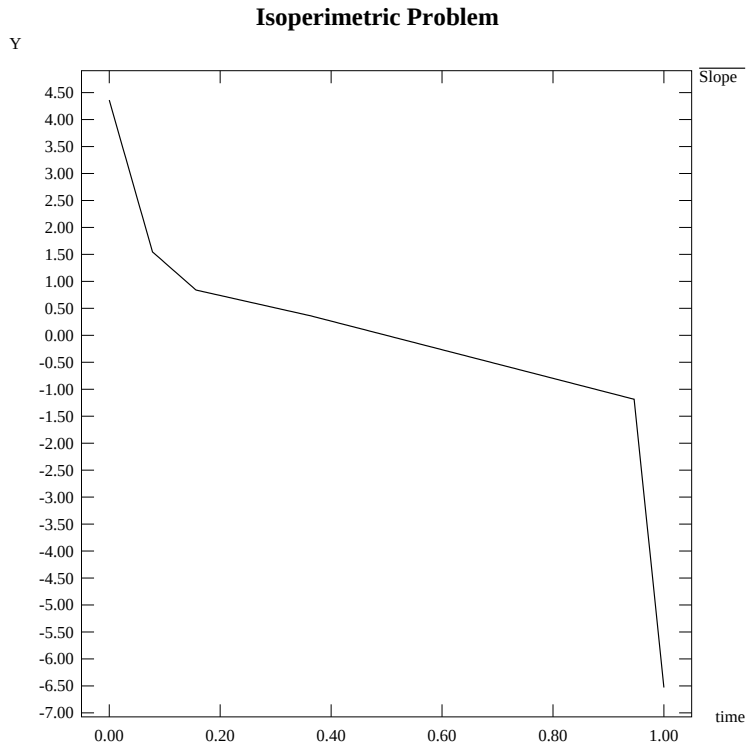

Figure 3.34: Isoperimetric problem—Control for level 1 piecewise linear profile

the state profiles for the case of cubic approximations.

The use of interpolated solutions obtained at lower orders to initialize the higher order optimal control calculation seems to have worked well, as the number of major iterations (QPs) was comparable among the iterations needed for level 3 linear, quadratic, and cubic approximations. Starting directly at these higher order approximations would have required more iterations to converge, as extra degrees of freedom are introduced into the optimization problem.

Low tolerance levels can be observed to produce poor profiles in Figures 3.34 and 3.35. This is crucial when few control elements are provided as in this case, and has an impact on the solutions obtained. Finally, the cubic approximation, although outperforming the others in terms of the performance index, shows some non-smoothness which distorts the control profile. This is possibly a good indication that the number of elements has to be increased to provide more flexibility.

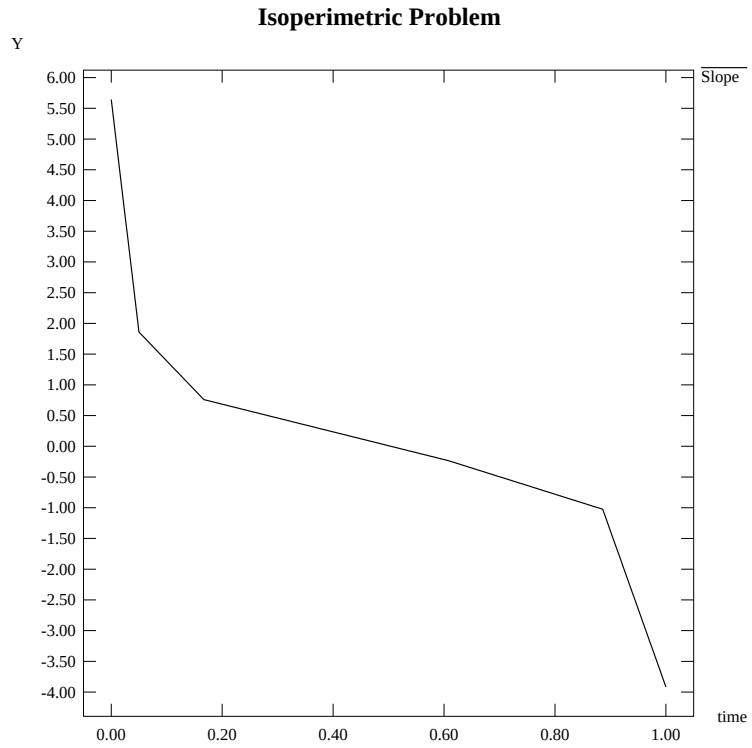

Figure 3.35: Isoperimetric problem—Control for level 2 piecewise linear profile

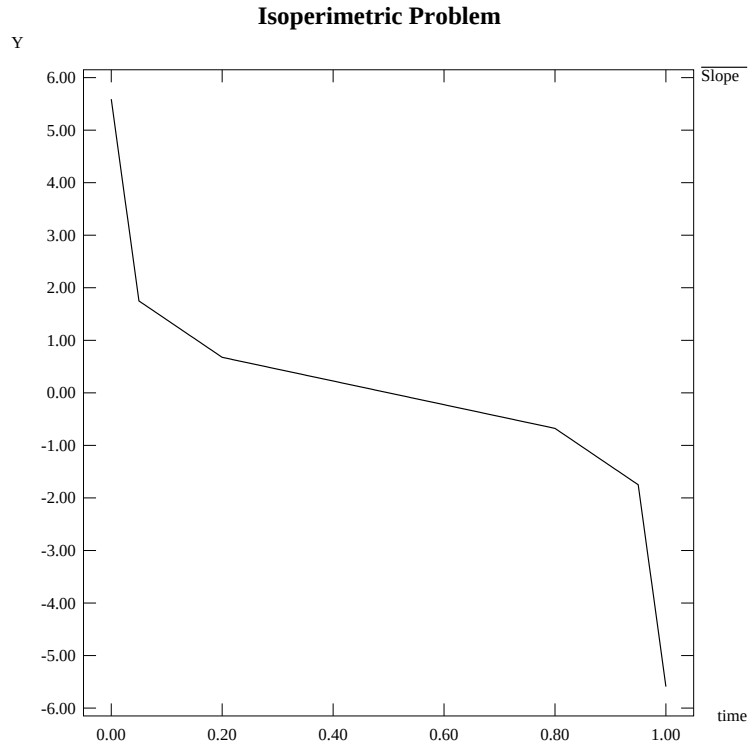

Figure 3.36: Isoperimetric problem—Control for level 3 piecewise linear profile

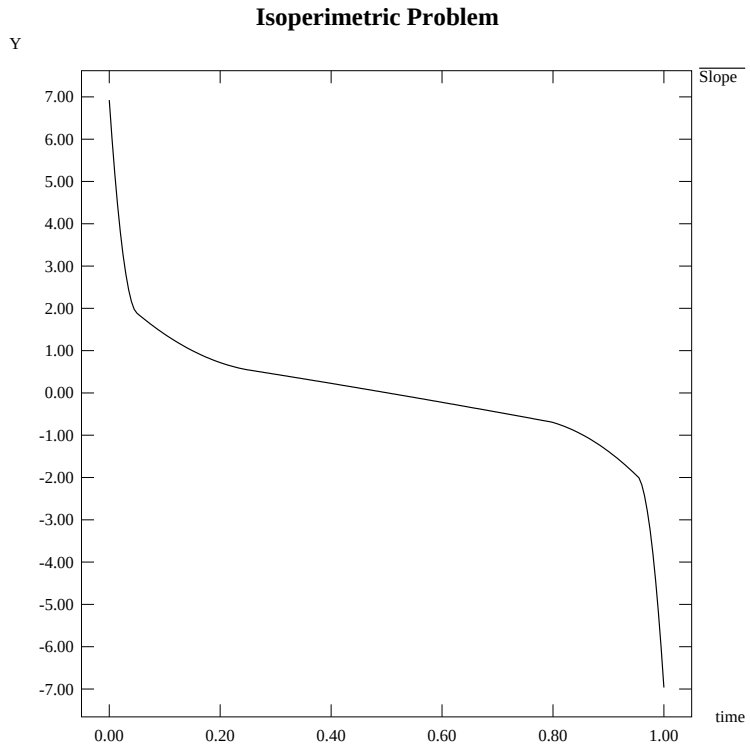

Figure 3.37: Isoperimetric problem—Control for level 3 piecewise quadratic profile

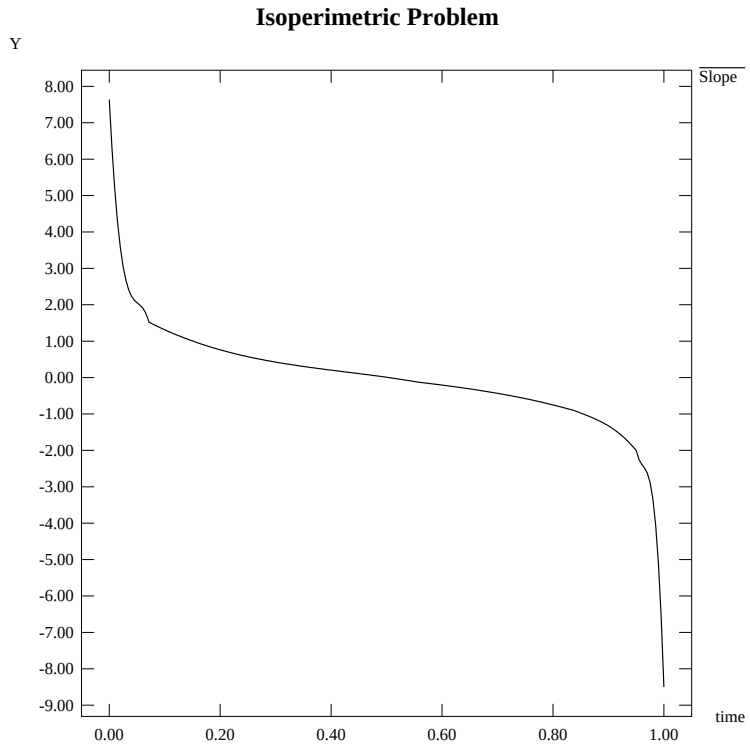

Figure 3.38: Isoperimetric problem—Control for level 3 piecewise cubic profile

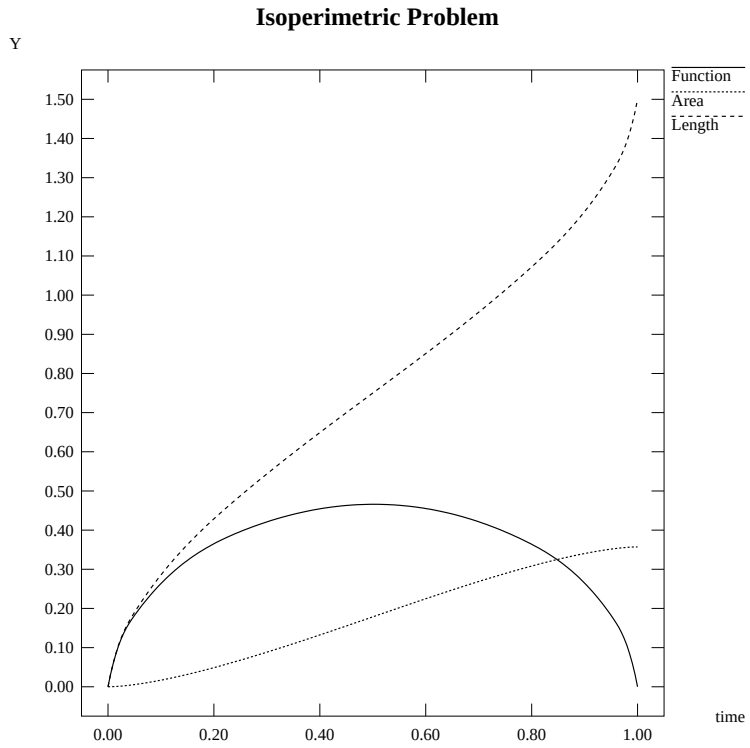

Figure 3.39. Isoperimetric problem—State profiles

3.12 Minimum Time Orbit Transfer Problem

This is a trajectory problem involving the determination of the thrust angle history of a rocket which minimizes the time required to transfer the vehicle from the orbit of Earth to that of Mars. The model description is taken from Vlassenbroeck and Van Dooren (1988). The problem is as follows:

$$\min_{u(\cdot),t_f} t_f \tag{3.70}$$

subject to:

$$\dot{y}_1 = y_2 \tag{3.71}$$

$$\dot{y}_2 = \frac{y_3^2}{y_1} - \frac{1.0}{y_1^2} + \frac{0.1405 \sin(u)}{(1.0 - 0.07487t)} \tag{3.72}$$

$$\dot{y}_3 = \frac{y_2\, y_3}{y_1} + \frac{0.1405 \cos(u)}{(1.0 - 0.07487t)} \tag{3.73}$$

$$y(0) = [1.0, 0.0, 1.0]^T, \quad y(t_f) = [1.525, 0.0, 0.8098]^T \tag{3.74}$$

$$0.0 \leq u(t) \leq 6.283; \quad 0 \leq t \leq t_f \tag{3.75}$$

The variables are: y_1 is the distance of the rocket from the Sun; y_2 is the radial velocity; y_3 is the tangential velocity; u is the thrust angle in radians. The three state variables are normalized to earth related values.

The problem was attacked directly using level 4 tolerances. It was assumed that this is a high precision problem, as an error would misdirect the rocket from its target orbit. Ten control elements were employed in the solution. Initially, piecewise linear approximations were assumed, with continuity at the element boundaries. Their lengths were bounded by $0.05 \leq h_i \leq 0.5$ and the initial final time estimate was taken to be 2.5 units. The control elements were equidistributed over the time horizon.

The initial control profile was set to be equal to $\frac{\pi}{2} \approx 1.57$ radians for the 5 first elements, and $\frac{3\pi}{2} \approx 4.71$ radians for the last 5 elements. This reflects a policy that attempts to drive the vehicle perpendicularly away from the Earth's

Level	Objective	Sens./St. CPU	Total CPU	# QPs	# LSs	Maximum Boundary Condition Error
Linear	3.31941	10.4	2337.2	114	158	1.5×10^{-13}
Quadratic	3.31939	12.1	1507.0	60	111	4.8×10^{-13}
Vlass.&V.Door.	3.31874	—	—	—	—	$< 10^{-13}$

Table 3.23. Minimum time orbit transfer problem—Solution statistics

orbit pointing outwards (at least initially), and at the end reverses the thrust so that in the limit this is perpendicular to the orbit of Mars pointing inwards to stabilize the vehicle in that orbit.

After the optimal solution was obtained, the problem was re-solved using quadratic approximations with continuity at the element boundaries. In this case, the control profile was initialized using an interpolation of the previous optimal profile. The initial values of the control element lengths were also set to the optimal values obtained earlier. However, the bounds on the optimization parameters were changed, in essence providing error bars around the values of the new starting point. This reduces the search space and excludes values too far away from what is known to be a satisfactory solution. Such tightening of the bounds is a well known method of speeding up optimizations of highly nonlinear problems (Amarger *et al.*, 1992). The control bounds were changed to ± 0.10 radians and the element length bounds were changed by ± 0.05 time-units around their starting point values.

The solutions are reported in Table 3.23, and the corresponding control element size distributions are given in Table 3.24. CPU times are given for a SUN SPARCstation IPX workstation.

The control profiles for the two cases are shown in Figures 3.40 and 3.41. Figures 3.42, 3.43, and 3.44 show the trajectories of the state variables for the case with the quadratic approximations. The end-point constraints were satisfied within 10^{-11}.

The idea of restricting the bounds around an optimal profile for a lower order

Element	Linear	Quadratic
1	0.5000	0.5029
2	0.5000	0.5008
3	0.4615	0.4626
4	0.1630	0.1637
5	0.1236	0.1086
6	0.0919	0.0936
7	0.1834	0.1850
8	0.3356	0.3375
9	0.4603	0.4634
10	0.5000	0.5013

Table 3.24. Minimum time orbit transfer problem—Optimal element sizes

approximation has worked very well. The solution with the quadratic approximation required 60 QPs as opposed to 114 for the linear case, and also the solution CPU time was reduced by a factor of 1.55. It is also observed that even though the quadratic approximation is smoother than the linear approximation, the objective is affected very little by this, and the boundary conditions are slightly less well satisfied. This shows that the 10 piecewise constant sections served as a very good approximation for the control.

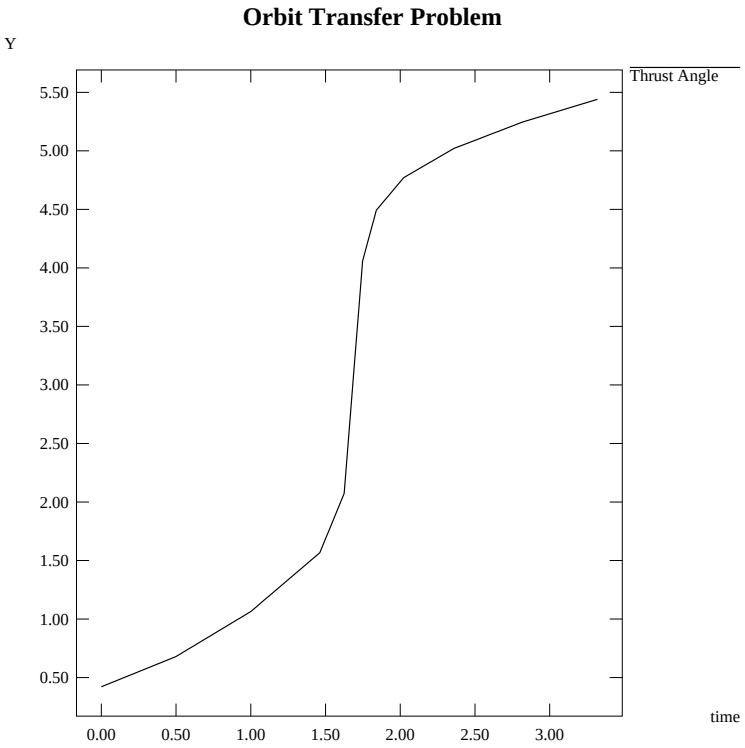

Figure 3.40. Min. time orbit transfer—Piecewise linear profile

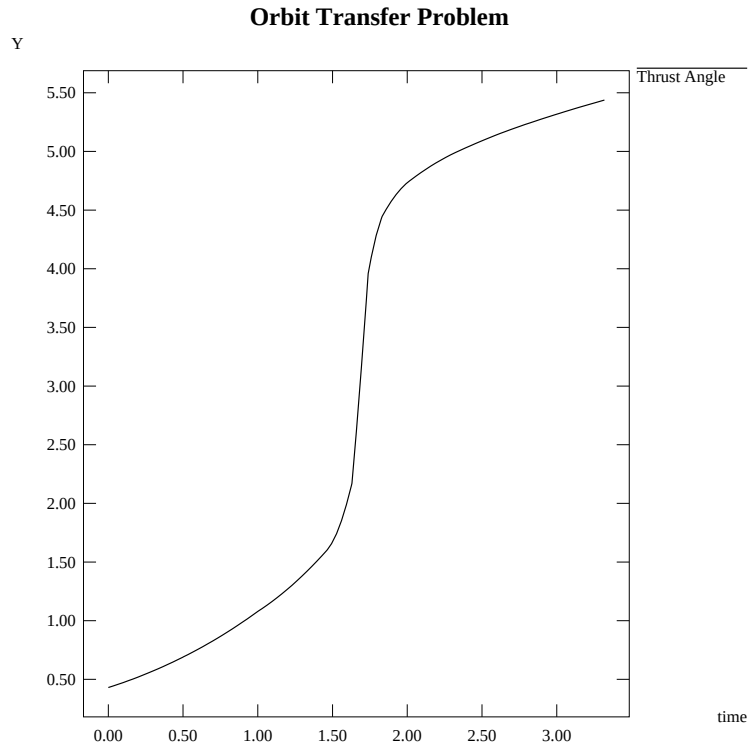

Figure 3.41. Min. time orbit transfer—Piecewise quadratic profile

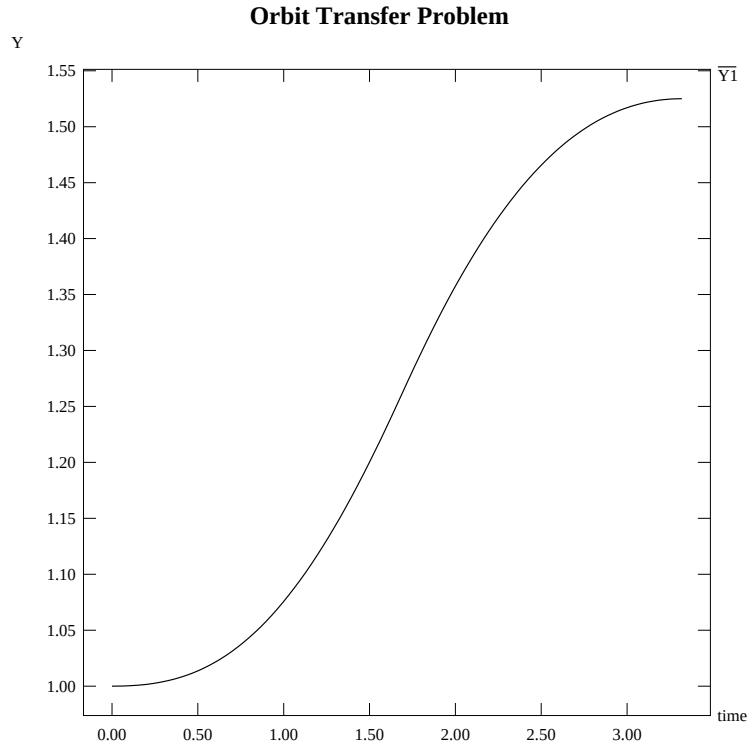

Figure 3.42. Min. time orbit transfer—Distance from Sun

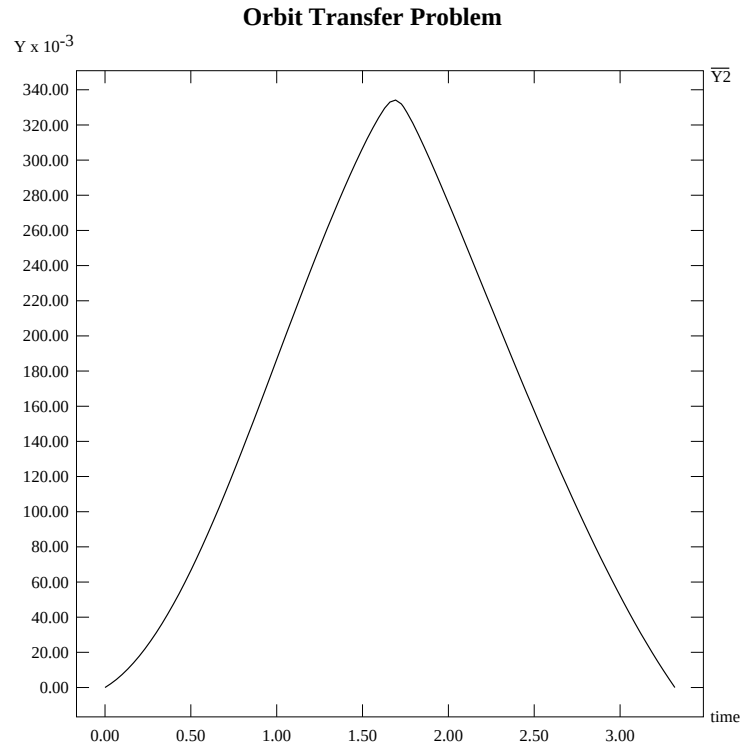

Figure 3.43. Min. time orbit transfer—Radial velocity

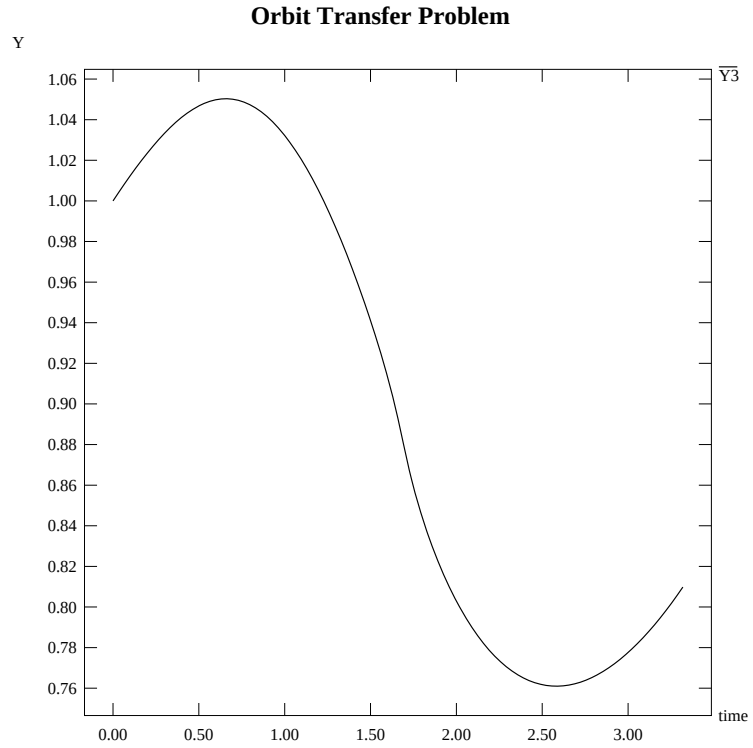

Figure 3.44. Min. time orbit transfer—Tangential velocity

3.13 Car Problem

This is a classical optimal control example. A car is to travel a fixed distance in minimum time, starting and finishing at rest. The modelling equations are taken from Logsdon and Biegler (1989). The problem is defined as follows:

$$\min_{u(\cdot),t_f} t_f \tag{3.76}$$

subject to:

$$\dot{y}_1 = u \tag{3.77}$$

$$\dot{y}_2 = y_1 \tag{3.78}$$

$$y(0) = [0,0]^T, \quad y(t_f) = [0,300]^T \tag{3.79}$$

$$-2.0 \leq u(t) \leq +1.0, \quad 0.0 \leq t_f \leq 50.0, \quad 0 \leq t \leq t_f \tag{3.80}$$

where y_1 is the velocity, y_2 is the distance, and u is the acceleration. The latter is treated as the control variable.

To solve this problem, three control elements were used over which the control (acceleration) was assumed to be piecewise constant. Their lengths were bounded by $0.01 \leq h_i \leq 25$, and initially were set to be all equal. The initial control profile was assumed uniform having value 0.0. Table 3.25 summarizes the solution results for all four tolerance levels. Table 3.26 gives the optimal control element size distributions. Figures 3.45, 3.46, and 3.47 show the optimal profiles for the acceleration ($u(t)$), the velocity ($y_1(t)$) and the distance ($y_2(t)$) for level 4 tolerances. It should be noted that no significant difference existed among the solutions for the 4 tolerance levels.

Level	Objective	Sens./St. CPU	Total CPU	# QPs	# LSs
Level 1	30.000100	9.5	2.5	5	4
Level 2	30.000001	7.7	2.5	5	4
Level 3	30.000000	6.1	2.9	5	6
Level 4	30.000000	4.1	2.6	5	6

Table 3.25. Car problem—Solution statistics

Element	Level 1	Level 2	Level 3	Level 4
1	10.0001	10.0000	9.9999	9.9999
2	10.0000	9.9999	10.0000	10.0001
3	10.0000	10.0000	10.0000	10.0000

Table 3.26. Car problem—Optimal element sizes

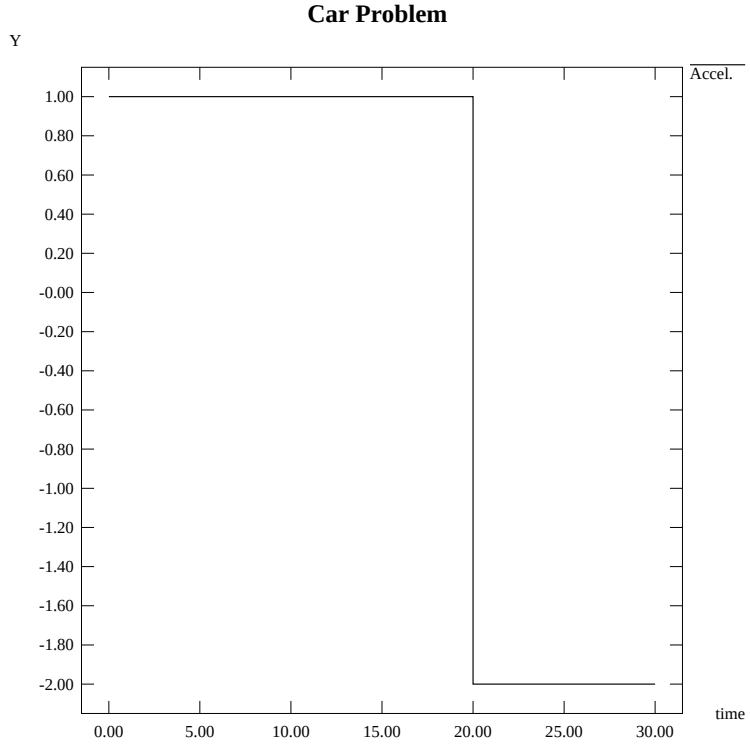

Figure 3.45. Car problem—Acceleration profile

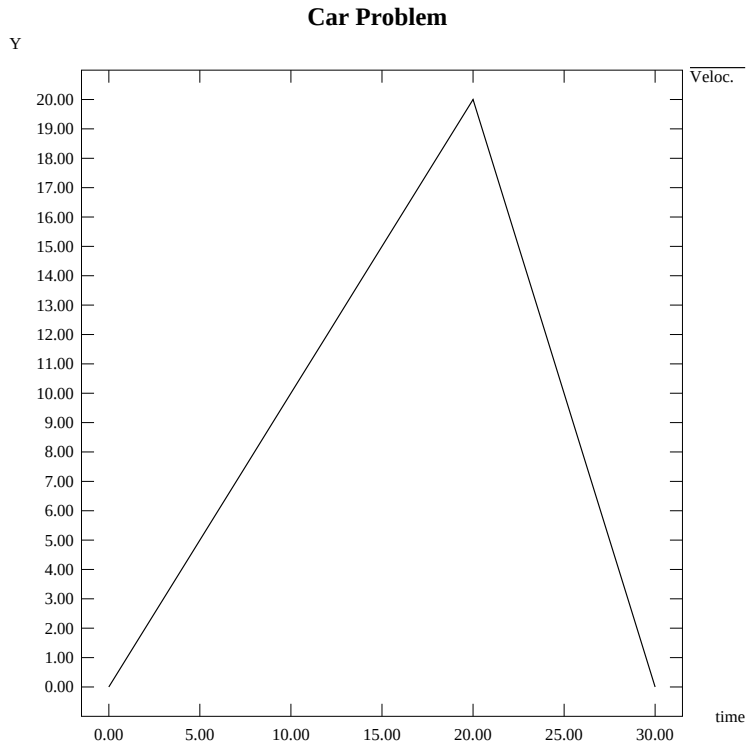

Figure 3.46. Car problem—Velocity profile

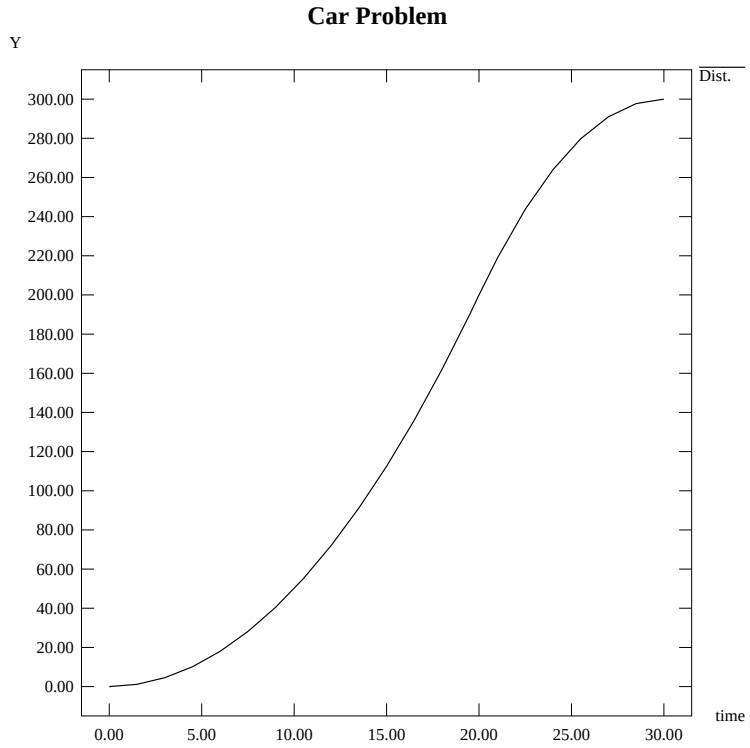

Figure 3.47. Car problem—Distance profile

3.14 Van der Pol Oscillator

The model for this problem was taken from Gritsis (1990). The optimal control problem is:

$$\min_{u(\cdot)} y_3(5.0) \tag{3.81}$$

subject to:

$$\dot{y}_1 = (1 - y_2^2)\, y_1 - y_2 + u \tag{3.82}$$

$$\dot{y}_2 = y_1 \tag{3.83}$$

$$\dot{y}_3 = y_1^2 + y_2^2 + u^2 \tag{3.84}$$

$$y(0) = [0, 1, 0]^T \tag{3.85}$$

$$-0.3 \leq u(t) \leq 1.0, \quad 0 \leq t \leq 5 \tag{3.86}$$

The problem was solved using the first 3 tolerance levels of Table 3.1. 10 piecewise linear elements were used with continuity of the control profile enforced at the element boundaries. The element sizes were bounded by $0.05 \leq h_i \leq 4.5$, and initially were all equal. The initial control profile was taken to be uniform of value 0.70. The bounds on the control are imposed in this work. The solutions obtained are reported in Table 3.27, along with solutions obtained by Gritsis (1990) and Morison (1984), and the corresponding control element size distributions are given in Table 3.28. Figure 3.48 shows the optimal control profile, and Figure 3.49 shows the corresponding state profiles, obtained for level 3 tolerances. It should be noted that Morison gives a slightly lower value of the objective function than those obtained here, however he used a quadratic approximation for the control profiles.

Level	Objective	Sens./St. CPU	Total CPU	# QPs	# LSs
Level 1	2.8691116	5.6	44.50	15	25
Level 2	2.8685702	6.5	105.0	20	30
Level 3	2.8680949	5.6	449.8	49	58
Gritsis	2.868	—	—	—	—
Morison	2.867	—	—	—	—

Table 3.27. Van der Pol oscillator—Solution statistics

Element	Level 1	Level 2	Level 3
1	0.6010	0.6110	0.5860
2	0.1300	0.1400	0.2810
3	0.3760	0.3590	0.1790
4	0.1170	0.1250	0.3680
5	0.4380	0.4150	0.2240
6	0.9880	1.0000	1.2100
7	0.8100	0.8070	0.4770
8	0.6100	0.6060	0.6170
9	0.4760	0.4770	0.5340
10	0.4540	0.4570	0.5230

Table 3.28. Van der Pol oscillator—Optimal element sizes

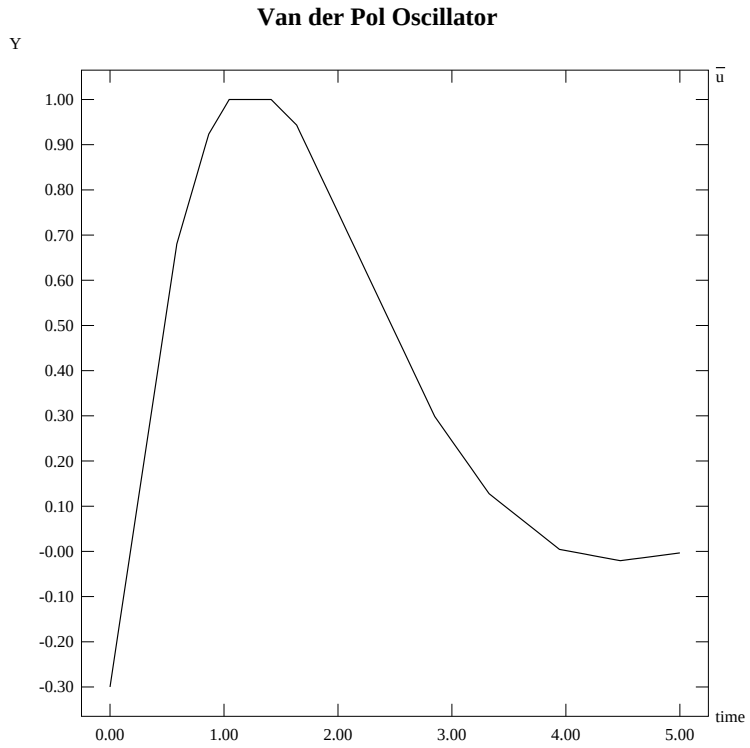

Figure 3.48. Van der Pol oscillator—Control profile

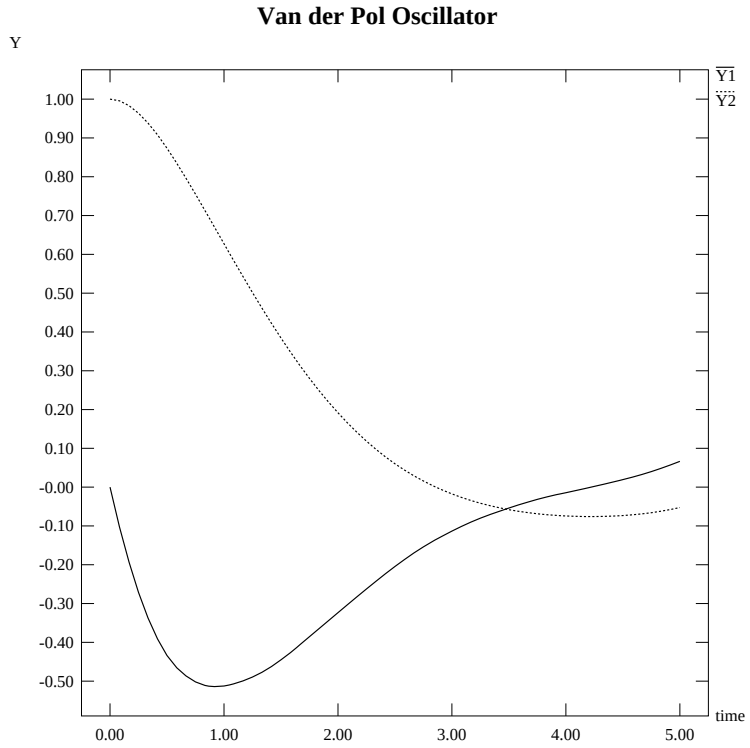

Figure 3.49. Van der Pol oscillator—State profiles

Level	Objective	Sens./St. CPU	Total CPU	# QPs	# LSs
Level 1	8.2864×10^{-2}	5.3	30.3	13	13
Level 2	7.9556×10^{-2}	6.2	115.6	29	29
Level 3	7.9437×10^{-2}	7.1	353.3	60	60
Gritsis	8.1×10^{-2}	—	—	—	—

Table 3.29. Jacobson and Lele problem—Solution statistics

3.15 Jacobson and Lele Problem

This problem is taken from Gritsis (1990), and is adapted from one given in Jacobson and Lele (1969). The optimal control problem is:

$$\min_{u(\cdot)} y_3(1) \tag{3.87}$$

subject to:

$$\dot{y}_1 = y_2 \tag{3.88}$$

$$\dot{y}_2 = y_2 + u \tag{3.89}$$

$$\dot{y}_3 = y_1^2 + y_2^2 + 0.005u^2 \tag{3.90}$$

$$y(0) = [0, -1, 0]^T \tag{3.91}$$

$$-3.0 \leq u(t) \leq +15.0, \quad 0 \leq t \leq 1 \tag{3.92}$$

It must be pointed out that the second equation in the original Jacobson and Lele problem is of the form $\dot{y}_2 = -y_2 + u$. However we use the form employed by Gritsis.

The first three tolerance levels of Table 3.1 were used to solve this problem. 10 control elements were initially of equal length and bounded by $0.01 \leq h_i \leq 0.90$. The initial control profile was set to a uniform value of 6.0. The results are summarized in Table 3.29, and the element size distributions are given in Table 3.30. Figures 3.50 and 3.51 give the control and the state profiles, respectively, obtained for level 3 tolerances.

Element	Level 1	Level 2	Level 3
1	0.0951	0.0841	0.0676
2	0.0763	0.0822	0.0981
3	0.8870	0.0878	0.0814
4	0.0971	0.0934	0.0892
5	0.1010	0.1010	0.1010
6	0.1050	0.1060	0.1090
7	0.1070	0.1100	0.1130
8	0.1090	0.1120	0.1150
9	0.1100	0.1130	0.1150
10	0.1090	0.1110	0.1120

Table 3.30. Jacobson and Lele problem—Optimal element sizes

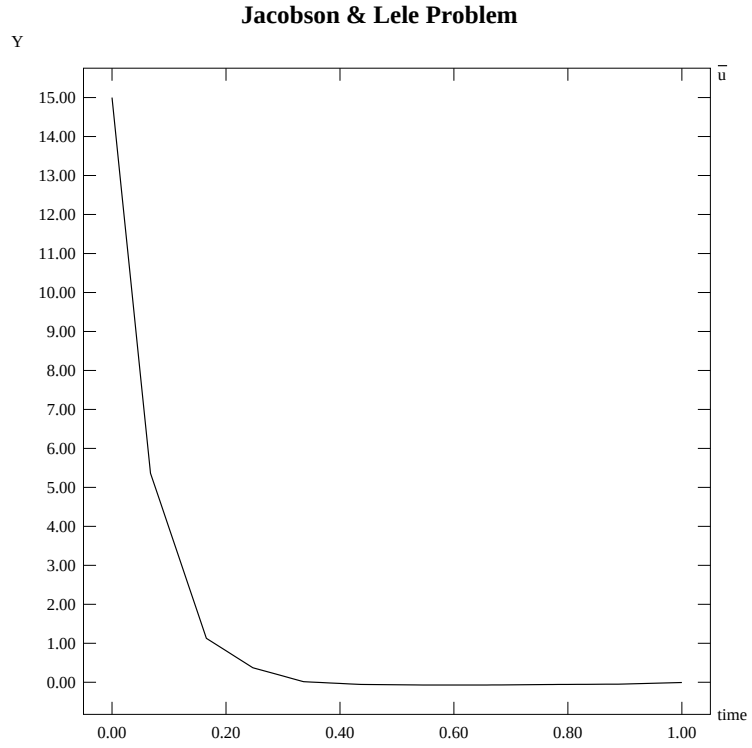

Figure 3.50. Jacobson and Lele problem—Control profile

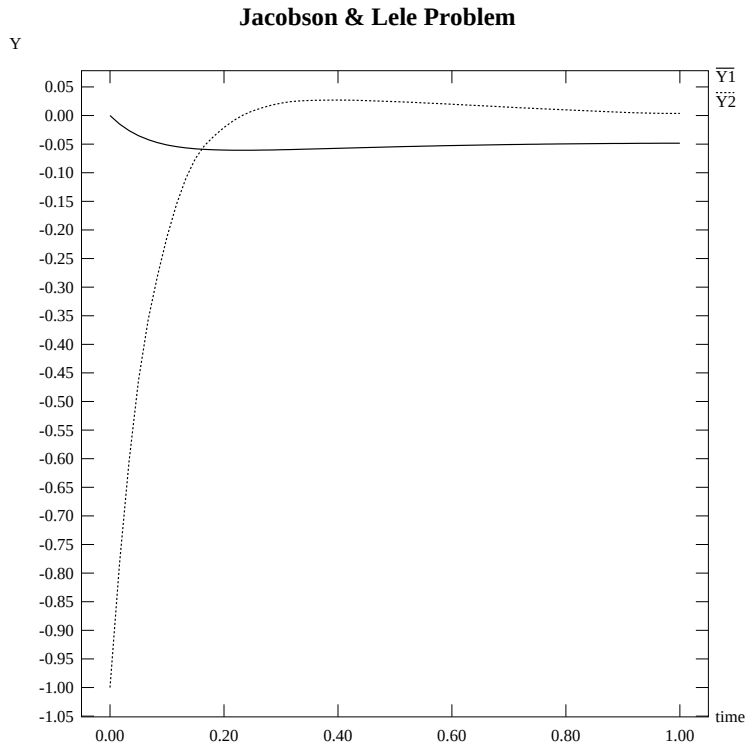

Figure 3.51. Jacobson and Lele problem—State profiles

3.16 Conclusions

The examples presented in this chapter have demonstrated the wide applicability and reliability of the proposed approach and its implementation. Most of the problems examined were small in size, except the large batch distillation problem (section 3.5) which was chosen to show the ability of the algorithm to tackle realistically large case studies.

End-point constrained problems are handled effectively, as demonstrated the examples given in sections 3.1, 3.2, 3.3, 3.4, 3.5, 3.9, 3.10, 3.11, 3.12, and 3.13.

Most problems considered are single stage, however the algorithm was designed for multistage problems and such an application is given in section 3.1.

Highly nonlinear problems, such as those given in sections 3.3, 3.4, 3.10, 3.12, were also considered and solved effectively.

Finally, the ability of our approach to handle singular arc problems is demonstrated in section 3.7.

Modelling issues were also addressed. In particular, the use of low order solutions to initialize a high order one was demonstrated to be very effective in reducing the overall number of iterations. In the transition from the low order to the high order problem, the bounds around the low order optimal controls and design variables were tightened, thus reducing the search space and consequently the required iterations for the higher order problem to be solved.

CHAPTER 4

Path Constraints

Path constraints require special handling within the framework of the control parameterization (CP) approach.

In the following sections, path constraints will be distinguished into equality and inequality constraints and various techniques for handling them will be presented. A unified approach to equality path constraints, and a new hybrid approach for handling inequality path constraints will be presented, together with numerical experiments illustrating the effectiveness of the techniques.

4.1 Equality Path Constraints

The general form of equality path constraints is:

$$c(\dot{x}(t), x(t), y(t), u(t), v, t) = 0 \tag{4.1}$$

There are two common approaches for handling such constraints within CP algorithms.

4.1.1 Penalty Methods and Transformation to End-Point Constraints

The use of an *integral penalty function* (Bryson and Ho, 1975) is a computational strategy that satisfies the constraints approximately. In the case of a scalar constraint, the augmented objective function is:

$$J' = J + K \int_{t_0}^{t_f} c^2(\dot{x}, x, y, u, v, t)\, dt \tag{4.2}$$

where K is a large number. This approach, in common with other penalty approaches in optimization of constrained problems, may cause numerical difficulties, and also requires $K \to \infty$ in order to satisfy the constraint exactly.

Based on the same principle is another method (Sargent and Sullivan, 1977) which reformulates the problem so that path constraints are replaced by a single end-point constraint, namely the integral of the constraint violation:

$$\Psi(t_f) = \int_{t_0}^{t_f} c^T(\cdot) c(\cdot)\, dt = 0 \tag{4.3}$$

A variant is to consider instead an l-dimensional end-point constraint:

$$\Psi_i(t_f) = \int_{t_0}^{t_f} c_i^2(\cdot)\, dt = 0; \quad i = 1, 2, \ldots, l \tag{4.4}$$

where l is the number of path constraints.

The advantage of eq. (4.4) over eq. (4.3) is that more information is supplied to the optimizer during the optimization stage. However, both have the disadvantage that at the solution the gradients of the end-point constraints derived above with respect to the optimization decision variables are zero, which in turn means that the rate of convergence is reduced for algorithms based on finding a solution to the first order necessary conditions of optimality. The success of such algorithms relies very strongly in this case on the line search merit function used (Goh and Teo, 1988).

4.1.2 Handling Equality Path Constraints Within the DAE System

We consider the following DAE system:

$$f(\dot{x}(t), x(t), y(t), u(t), v, t) = 0 \tag{4.5}$$

and a set of equalities that must be satisfied along the solution trajectory:

$$c(\dot{x}(t), x(t), y(t), u(t), v, t) = 0 \tag{4.6}$$

where $x \in \mathcal{X} \subseteq \mathcal{R}^n$, $\dot{x} \in \mathcal{X}' \subseteq \mathcal{R}^n$, $y \in \mathcal{Y} \subseteq \mathcal{R}^m$, $u \in \mathcal{U} \subseteq \mathcal{R}^\pi$, $v \in \mathcal{V} \subseteq \mathcal{R}^d$, $t \in \mathcal{T} \equiv [t_0, t_f] \subseteq \mathcal{R}^1$, and $f : \mathcal{X}' \times \mathcal{X} \times \mathcal{Y} \times \mathcal{U} \times \mathcal{V} \times \mathcal{T} \mapsto \mathcal{R}^{n+m}$ and $c : \mathcal{X}' \times \mathcal{X} \times \mathcal{Y} \times \mathcal{U} \times \mathcal{V} \times \mathcal{T} \mapsto \mathcal{R}^l$. We note that indeed no fundamental difference exists between these two, *i.e.* they may be combined into a single DAE system. We are therefore led to consider a unified treatment for all these equality constraints at the DAE solution level. In this case, some controls may become state variables as degrees of freedom may be lost by the introduction of (4.6). For the problem to be well-posed, *i.e.* not to be overspecified, the following must hold:

$$l \leq \pi \tag{4.7}$$

Unfortunately the index of the combined DAE system (4.5) and (4.6) may be greater than one. Most current state-of-the-art integrators can handle DAE systems of index only up to one. Therefore, the desired strategy is to derive from (4.5) and (4.6) a maximal subset of equations and controls that have index 1. Any remaining equalities will be treated as path constraints, left to be handled by some approximation at the optimization level in an appropriate fashion.

A further complication arises from the fact that, in many applications, the space of practically implementable control functions for certain elements of the control vector $u(t)$ is restricted (*e.g.* to piecewise constant or linear functions). As the solutions of DAE systems do not, in general, belong to such function spaces, it is clear that the corresponding control variables must be left to be determined at the optimization level.

Furthermore, it is unlikely that any direct manipulation of the state variables $x(t)$, $y(t)$ will be possible in practice (*e.g.* in a chemical plant, one cannot vary directly compositions or temperatures). Hence it is highly desirable that these variables be left to be determined by the solution of the DAE system, rather than be treated as controls to be determined by the optimization.

We denote the system obtained by combining (4.5) and (4.6) by:

$$F(\dot{x}(t), x(t), y(t), u(t), v, t) = 0 \tag{4.8}$$

$F : \mathcal{X}' \times \mathcal{X} \times \mathcal{Y} \times \mathcal{U} \times \mathcal{V} \times \mathcal{T} \mapsto \mathcal{R}^{n+m+l}$. The control vector u is assumed to be partitioned into (u_1, u_2) of respective dimensions π_1 and π_2, such that u_2 *must* be treated as controls, but u_1 may be treated similarly to the rest of the unknowns. According to this distinction, it is assumed that *any* functional form of $u_1(t)$ can be implemented using perfect control, whereas $u_2(t)$ is restricted to the simple piecewise low-order polynomial forms considered in section 2.3.

We form the following $(n + m + l) \times (n + m + \pi_1)$ Jacobian matrix:

$$A = \begin{bmatrix} \dfrac{\partial f}{\partial \dot{x}} & \dfrac{\partial f}{\partial y} & \dfrac{\partial f}{\partial u_1} \\ \dfrac{\partial c}{\partial \dot{x}} & \dfrac{\partial c}{\partial y} & \dfrac{\partial c}{\partial u_1} \end{bmatrix} \tag{4.9}$$

Through the application of a set of row and column pivoting and row operations we effect the transformation:

$$L\,P\,A\,Q = \begin{bmatrix} U & B \\ \cdot & 0 \end{bmatrix} \tag{4.10}$$

where L is a lower triangular matrix, and P and Q are row and column permutation matrices of the form:

$$P = \begin{bmatrix} P_1 & \\ & P_2 \end{bmatrix}; \qquad Q = \begin{bmatrix} Q_1 & \\ & Q_2 \end{bmatrix} \tag{4.11}$$

where P_1, Q_1 are $(n + m) \times (n + m)$ permutation matrices, P_2 is $l \times l$, and Q_2 is $\pi_1 \times \pi_1$. On the right hand side, U is a $r \times r$ upper triangular matrix.

Since the DAE system (4.5) is of index 1, $[f_{\dot{x}}\ f_y]$ is nonsingular and hence $r \geq n + m$. Furthermore, the first r elements of the permuted equation vector PF forms an index-1 system with respect to the first r elements of the permuted variable vector:

$$Q \begin{bmatrix} x \\ y \\ u_1 \end{bmatrix} \tag{4.12}$$

Because of the special form (4.11) of the permutation matrices, this index-1 system involves all equations (4.5) and all variables (x, y), as well as the first $(r - n - m)$ elements of $P_2\, c$ and the first $(r - n - m)$ elements of $Q_2\, u_1$.

The latter are now treated as ordinary algebraic variables within the expanded DAE system. The remaining $(n + m + l - r)$ elements of c, and $(n + m + \pi_1 - r)$ elements of u_1 are left to be handled at the optimization level as path constraints and control variables respectively. All π_2 elements of $u_2(t)$ are also handled as control variables.

We illustrate the procedure described in this section with the help of a small example. We consider the following linear DAE system with three controls u_1, u_2, u_3, and three equations:

$$-\dot{x}_1 + x_1 + x_2 + 2y_1 + u_1 = 0 \tag{4.13}$$

$$-\dot{x}_2 + x_1 - x_2 - u_2 + u_3 = 0 \tag{4.14}$$

$$\dot{x}_1 + 3x_1 + 2x_2 - y_1 + 3u_1 + 5u_2 = 0 \tag{4.15}$$

to which the following equality path constraints are added:

$$x_1 + x_2 - 1 = 0 \tag{4.16}$$

$$2\dot{x}_1 + x_2 - 2u_2 - u_3 = 0 \tag{4.17}$$

We assume that no restrictions are imposed on the functional form of $u_1(t)$, $u_2(t)$, and $u_3(t)$ hence they all belong to the first partition of the control vector $u(t)$.

The Jacobian of the above constraints with respect to $\dot{x}_1$, $\dot{x}_2$, y_1, u_1, u_2, and u_3 is:

$$A = \begin{pmatrix} -1 & 0 & 2 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & -1 & 1 \\ 1 & 0 & -1 & 3 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & -2 & 0 & -1 \end{pmatrix} \tag{4.18}$$

which can be transformed to:

$$\begin{bmatrix} U & B \\ \cdot & 0 \end{bmatrix} = \left(\begin{array}{cccc|cc} -1 & 0 & 2 & 0 & 1 & 0 \\ 0 & -1 & 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & 5 & 4 & 0 \\ 0 & 0 & 0 & -20 & -16 & -1 \\ \hline 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right) \tag{4.19}$$

using the matrices:

$$P = \left(\begin{array}{ccc|cc} 1 & 0 & 0 & & \\ 0 & 1 & 0 & & \\ 0 & 0 & 1 & & \\ \hline & & & 0 & 1 \\ & & & 1 & 0 \end{array}\right) \tag{4.20}$$

$$Q = \left(\begin{array}{ccc|ccc} 1 & 0 & 0 & & & \\ 0 & 1 & 0 & & & \\ 0 & 0 & 1 & & & \\ \hline & & & 0 & 1 & 0 \\ & & & 1 & 0 & 0 \\ & & & 0 & 0 & 1 \end{array}\right) \tag{4.21}$$

and

$$L = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ -2 & 0 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{4.22}$$

Clearly the rank of A is 4 and hence the original DAE system (4.13–4.15) can be augmented by one more equation and variable. The form of the matrices P and Q suggest that this augmentation will involve the second path constraint (4.17) and the second control variable $u_2(t)$. The latter will then be treated as an algebraic variable within the DAE.

On the other hand, the optimizer will manipulate $u_1(t)$ and $u_3(t)$ as control variables, with (4.16) being treated as an equality path constraint using one of the techniques of section 4.1.1.

4.2 Inequality Path Constraints

Here we consider constraints of the form:

$$c(\dot{x}(t), x(t), y(t), u(t), v, t) \leq 0 \tag{4.23}$$

The treatment of inequality path constraints is very similar to that of equality constraints.

4.2.1 Penalty Methods and Transformation to End-Point Constraints

Several numerical methods rely on the reformulation of the path constraint as an end-point constraint using a measure of the violation of that constraint at any time t:

$$r_i = \| \, c_i(\cdot) \, \|_t \tag{4.24}$$

For example, the norm used can be:

$$r_i = \max[c_i, 0] \tag{4.25}$$

As in the case of equality constraints, the constraint violation can be appended as a penalty term to the objective function (*cf.* eq. 4.2)

$$J' = J + K \int_{t_0}^{t_f} r^T(\cdot) r(\cdot) \, dt \tag{4.26}$$

In practice, the integral can be evaluated by introducing an additional variable x_{n+1} defined by the additional ODE:

$$\dot{x}_{n+1}(t) = r^T(\cdot) r(\cdot), \quad x_{n+1}(t_0) = 0 \tag{4.27}$$

Alternatively, the violation may be forced to zero through the end-point constraint $x_{n+1}(t_f) = 0$. Instead of using a single end-point constraint for all the path constraints, individual end-point constraints can be used for each path constraint, thus defining the following ODE system:

$$\dot{x}_{n+i}(t) = r_i^p(\cdot), \quad x_{n+i}(t_0) = 0, \quad x_{n+i}(t_f) = 0, \quad i = 1, 2, \ldots, l \tag{4.28}$$

where the exponent p can be either 1 or 2. Although $p = 1$ is sufficient, using $p = 2$ yields a function $[\max(w, 0)]^2$ which has first order continuity at $w = 0$.

The analysis of the advantages and disadvantages of the methods presented thus far is the same as that given for equality constraints in the previous section. However, the use of norm (4.25) introduces a discontinuity as the maximum function has zero gradient. This implies that when the trajectory is feasible, the constraint conveys no information to the subproblem yielding a search direction, along which constraint violations may appear. Nevertheless, some numerical experience does exist to support the use of this metric and as noted by Goh and Teo, 1988, as mentioned in section 4.1.1, the success of such implementations relies very strongly on the line search merit function used in the optimization algorithm.

As has already been noted, the major disadvantage of this approach is the discontinuity that occurs as the constraint moves from active to inactive status and vice versa. To avoid this problem, Walsh (1991) introduced the idea of ϵ-relaxation of the end-point constraint. According to this, the rigid end-point constraint $x_{n+1}(t_f) = 0$ is relaxed to

$$x_{n+1}(t_f) \leq \epsilon \tag{4.29}$$

where ϵ is a small tolerance. Numerical experience indicates that this simple expedient may result in significant performance improvements.

4.2.2 Slack Variable Approach

This is also known as Valentine's method and is described by Jacobson and Lele (1969). It was originally developed for constraints of the form:

$$c(x(t), t) \geq 0 \tag{4.30}$$

A slack variable $a(t)$ is used to transform eq. (4.30) to an equality constraint:

$$c(x(t), t) - \frac{1}{2}a^2(t) = 0 \tag{4.31}$$

A control variable becomes a state variable, as the new equality constraint is introduced to the DAE system. The resulting DAE system has higher index since

eq. (4.31) is an algebraic constraint on the differential variables. This equation is therefore differentiated with respect to time as many times as required to obtain an expression for some control variable, say u_j, in terms of x, y, v, the remaining control variables u_i, $i \neq j$, $a(t)$, and its k derivatives, where k is the number of differentiations required. The problem is then reformulated by elimination of u_j, setting $a^{(k)}(t)$ as the new control variable, and setting the $k-1$ previous derivatives of $a(t)$ as differential variables.

Overall, this procedure eliminates the path constraints while ensuring that they are strictly satisfied; thus feasibility is maintained at every stage of the solution procedure.

This method can be extended to cover also the general case of inequality path constraints given by eq. (4.23).

4.2.3 Bounding the Minimum Value of a Constraint

This approach (Gritsis, 1990) considers path constraints of the form:

$$c(x(t), y(t), u(t), v, t) \geq 0 \tag{4.32}$$

A new function Ψ is defined to be the minimum value of c over the interval $I \equiv [t_0, t_f]$:

$$\Psi(x, y, u, v) = \min_{t \in I} [c(x, y, u, v, t)] \tag{4.33}$$

If there are many, or infinite, points at which this value is attained, then point t_m is chosen to be the earliest time at which the minimum value is attained:

$$t_m = \min \{t \mid t \in T_m\} \tag{4.34}$$

where T_m is the set of points for which the minimum is attained. The basic idea of this approach is to enforce the path constraint as a point constraint at t_m:

$$\Psi(x(t_m), y(t_m), u(t_m), v) \geq 0 \tag{4.35}$$

thereby ensuring that (4.33) is satisfied everywhere in the time domain.

Gritsis derives gradient information for this constraint, as well as a method for the calculation of t_m.

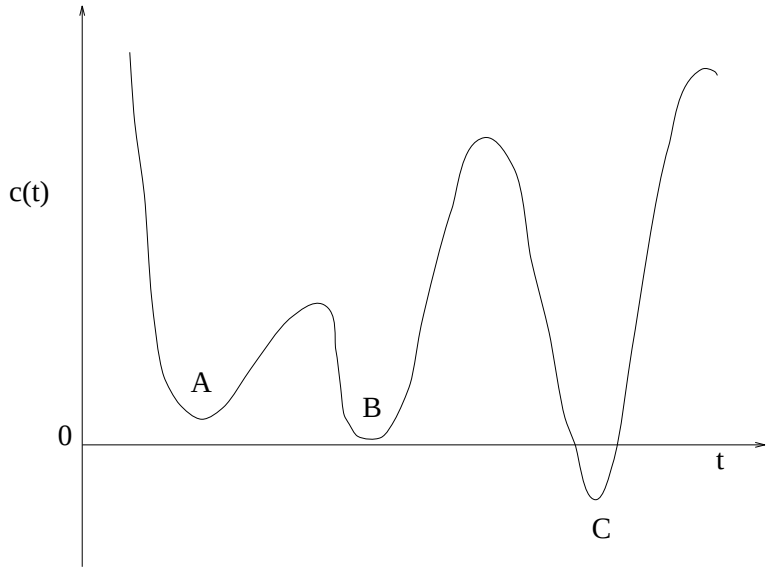

Figure 4.1. Minimum path constraint value—First discontinuity

This method has the desirable characteristic of using the minimum number of discretization points in enforcing the path constraint, and not introducing any discontinuity involving the maximum function. However, it has other underlying discontinuities.

The first type of discontinuity arises from a switch in the minimum tracked by the algorithm. Should the global minimum switch from one local minimum to another, then a gradient discontinuity will emerge. For instance, as in the path constraint depicted in Figure 4.1, a discontinuity will occur when the global minimum of $c(t)$ switches from point C to point B.

The second type of discontinuity will come from the fact that the interior point constraint is a free-floating constraint. Such constraints will most certainly cause a discontinuity in control parameterization as the parameters affecting the value of a function change from one control element to another. For example, in Figure 4.2 when the constraint falls out of element 1 into element 2, its value is affected by the parameters involved in both elements rather than those of the first element only as was previously the case.

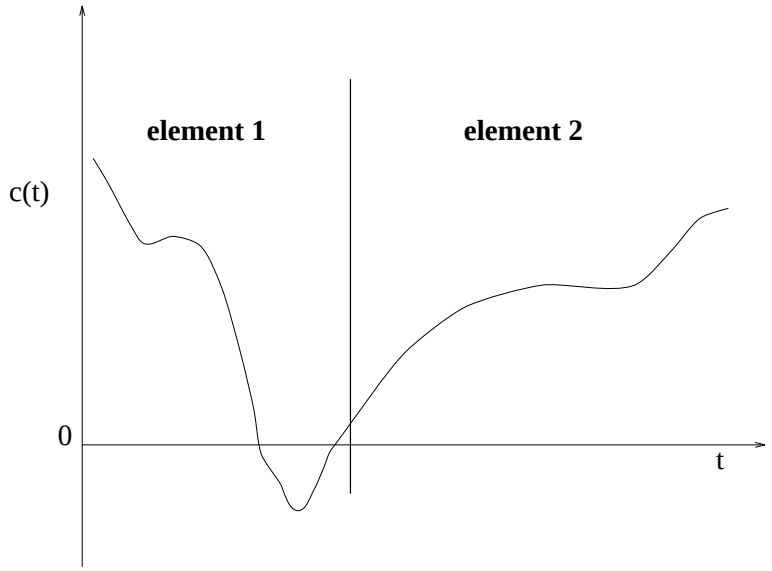

Figure 4.2. Minimum path constraint value—Second discontinuity

4.3 A New Hybrid Approach for Path Constraints

Optimal control problems with path constraints have thus far been handled mostly using the squared maximum formulation (*cf.* eqs. (4.24), (4.28)). This is a convenient way of dealing with the path constraint, with the final value of the end-point constraint (4.28) measuring globally the satisfaction of the constraint.

One major disadvantage of methods converting path constraints to end-point constraints is that the violations of the path constraint are lumped together through the integral involved.

At the other end of the spectrum of methods for handling path constraints lies discretization of the constraint using collocation schemes, as advocated, for instance, by Biegler and co-workers (*e.g.* Cuthrell and Biegler, 1987; Logsdon and Biegler, 1989). However, should one decide to use only discretization to bound the path constraint everywhere, then the number of constraints may have to be very large. More importantly, an insufficient discretization may not guarantee feasibility of the solution with respect to the constraint.

In this section, a hybrid of the two extreme approaches is proposed, such that

local information will be conveyed to the optimizer using constraints of the form:

$$\rho(c(t_i)) \leq \epsilon_i\,, \quad i = 0, 1, 2, \ldots, NS \tag{4.36}$$

These constraints are applied as interior point constraints, t_i. Several possibilities exist for the selection of appropriate functions $\rho(c)$. Some of these are examined below.

4.3.1 Scheme 1: Bounding of the Error Accumulated in Each Element

This approach attempts to bound the path constraint violation over each element separately, instead of simply bounding it over the entire time horizon.

If $x^*(t)$ denotes a cumulative integral violation (defined by equations of type (4.27) or (4.28)) between t_0 and t_f, then the appropriate constraint can be written as:

$$x^*(t_i) - x^*(t_{i-1}) \leq \epsilon_i; \quad i = 1, 2, \ldots, NS \tag{4.37}$$

The values of ϵ_i can be made dependent on the element size by setting $\epsilon_i = \frac{\epsilon\, h_i}{t_f - t_0}$, h_i being the element length, and ϵ some constant tolerance. The idea is that larger elements may be allowed larger violations.

4.3.2 Scheme 2: Escalated Tolerance for Path Constraint Value

A modification of the previous scheme is to consider adding the interior point constraints defined in eq. (4.37) to derive:

$$x(t_i^*) \leq \sum_{k=1}^{i} \epsilon_k; i = 1, 2, \ldots, NS \tag{4.38}$$

Thus, this formulation requires that the accumulated error at each element end-point be less than a progressively increasing tolerance. The scheme allows in this way a small error in one interval to compensate large error in another.

4.3.3 Scheme 3: Fixed Tolerance for Path Constraint Value

An even greater relaxation of accumulated error schemes is to replace the right hand sides of constraints (4.38) by the fixed tolerance ϵ for the end-point.

One would expect this scheme to be essentially the same as imposing the path constraint as an end-point constraint only. This may have some benefit in that information is provided on when the constraint became violated (*i.e.* earlier information).

4.3.4 Scheme 4: Direct Discretization of the Path Constraint

This scheme is expected to provide local information in its raw form, as it examines locally the value of the constraint without any transformation. The resulting constraints are of the form:

$$c(t_i) \leq 0\,, \quad i = 0, 1, 2, \ldots, NS \tag{4.39}$$

and are imposed along with the end-point constraint (4.29).

The number of interior points is relatively small; indeed it can be equal to or smaller than the number of control elements (plus the initial point), thus minimizing the size effects on the resulting NLPs, as opposed to complete discretization. In any case, for most problems of realistic complexity, the CPU time devoted to the NLP solution is insignificant in comparison to the integration times, and therefore any algorithm that reduces the number of function and gradient evaluations (consequently integrations of the DAEs) is going to be more efficient.

The idea of incorporating discretization points simultaneously with the usual end-point constraint resulting from the path constraint violation norm has the benefit of conveying more information to the optimizer as to the status of the path constraint, even when the latter is not violated. The presence of the violation norm end-point constraint, on the other hand, is important as it controls the global satisfaction of the path constraint in the time horizon.

4.3.5 Scheme 5: Bounding the Integral of the Constraint

Another scheme that may be used to provide local information is to bound the value of the integral of the constraint at interior intervals. The resulting constraints are of the form:

$$\int_{t_{i-1}}^{t_i} c(t)\,dt \leq 0, \quad i = 1, 2, \ldots, NS \tag{4.40}$$

and are imposed along with the end-point constraint (4.29).

It should be noted that, like their simpler counterparts (4.39), these constraints are necessary but not sufficient conditions for the original path constraint to be satisfied throughout the time horizon.

4.4 Numerical Experiments

Six test problems were set up and solved in order to provide a comparison of the various techniques presented in the previous section.

The tolerance levels were fixed to level 2 presented in Table 3.1 of Chapter 3. All CPU times are in seconds reported for a SUN SPARC IPX workstation, unless otherwise stated. SUN SPARC 2 workstation CPU times were found by tests to be slightly longer (by about 6% at most), so a fair comparison can be made between the two where the need arises.

4.4.1 Van der Pol Oscillator

To the problem presented in section 3.14, the following path constraint is appended:

$$-y_1(t) - 0.4 \leq 0 \tag{4.41}$$

The starting point for the optimization and the bounds are the same as those reported for the unconstrained case.

This problem was used as a guiding example to see whether the five schemes would produce results reflecting their expected character. A notation of the form (N)(TYPE) is used in the first column of Table 4.1 which summarizes the results for all the runs. (TYPE) corresponds to the two norms used: the maximum squared (symbolized by $\max^2$), and the maximum norm alone (max). (N) is the scheme number, with scheme (0) being the direct application of the integral norms without any type of discretization. Schemes 1–5 of section 4.3 are similarly coded as (1)–(5). When ($\max^2$) was used the total error allowed over the entire horizon was set to 10^{-7}, but when (max) was used the total error was set to 10^{-5}, so as to account for the fact that the latter norm is stricter.

In Table 4.1, the first run reported corresponds to the unconstrained problem (*i.e.* without the path constraint). The following run is the one given in Gritsis (1990) using the $\max^2$ integral norm alone. The next 12 runs correspond to the coded scheme number and norm type as described above. Finally, the last entry on the table corresponds to discretization only of the path constraints over the element end-points without the enforcement of any integral norm.

The information produced is very illuminating. The most unexpected result is that of the squared maximum norm coupled with discretization (scheme (4)($\max^2$)). The result is obtained *faster* than the unconstrained optimum in terms of QP iterations. One should compare the methods using this as the main criterion, CPU time being second, as it reflects the ease of optimization.

The result produced with constraint discretization alone (last row of Table 4.1) is, in fact, unacceptable, being closer to that of the unconstrained case than to the one with the path constraint. Indeed the path constraint was hardly enforced

Scheme & norm	Objective	Sens./St. CPU	CPU	#QPs	#LSs	ϵ or ϵ_i
Unconstrained	2.86857	6.5	105.0[1]	20	30	—
Gritsis $\max^2$ (1990)	2.9600	—	—	27	—	—
(0) ($\max^2$)	2.95436	7.4	217.4	34	51	10^{-7} (ϵ)
(0) (max)	2.95496	7.2	223.9	32	54	10^{-5} (ϵ)
(1) ($\max^2$)	2.95555	7.3	314.0	51	77	0.1×10^{-7} (ϵ_i)
(1) (max)	2.95678	7.2	213.8	31	62	0.1×10^{-5} (ϵ_i)
(2) ($\max^2$)	2.95506	7.2	241.4	40	52	0.1 to 1.0×10^{-7} (ϵ_i)
(2) (max)	2.95491	7.2	200.6	30	45	0.1 to 1.0×10^{-5} (ϵ_i)
(3) ($\max^2$)	2.95421	7.3	246.7	40	56	10^{-7} (ϵ)
(3) (max)	2.95486	7.3	194.4	29	45	10^{-5} (ϵ)
(4) ($\max^2$)	2.95474	7.4	99.8	16	25	10^{-7} (ϵ)
(4) (max)	2.95539	7.3	96.8	18	25	10^{-5} (ϵ)
(5) ($\max^2$)	2.95502	7.5	131.8	19	24	10^{-7} (ϵ)
(5) (max)	2.95517	7.6	288.2	40	91	10^{-5} (ϵ)
Discretization only	2.88884	7.3	94.4	15	20	—

Table 4.1. Results for Van der Pol constrained problem

using this scheme, an indication for the need of more discretization points.

Figure 4.3 shows the optimal control profile and Figure 4.4 shows states y_1 and y_2. These plots correspond to scheme (4)($\max^2$).

[1] Obtained on a SUN SPARC 2 workstation.

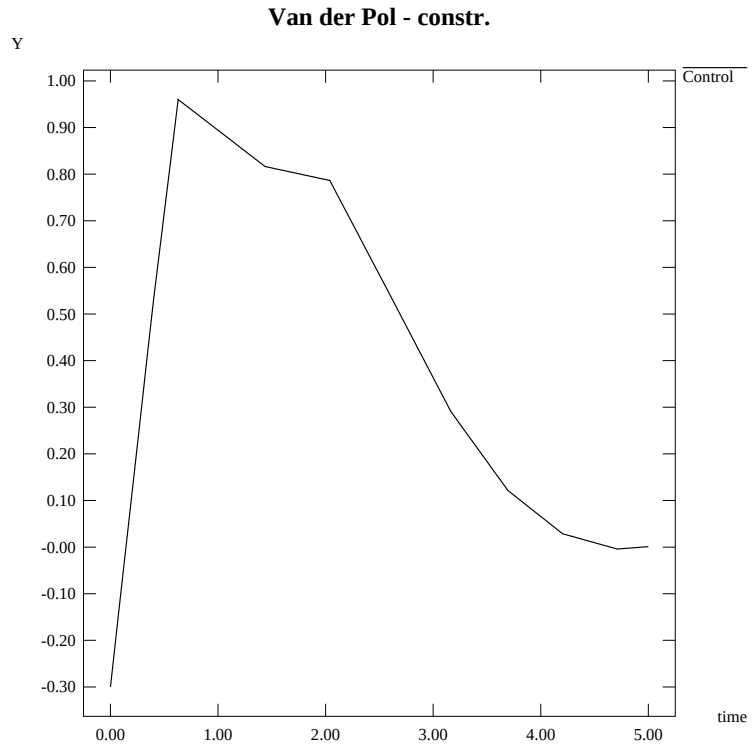

Figure 4.3. Van der Pol with path constraint—Control profile

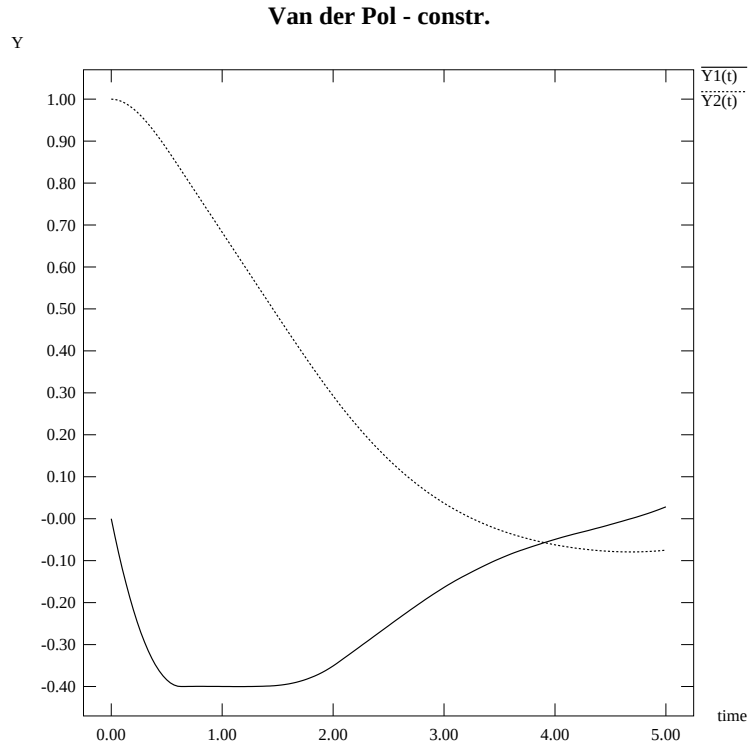

Figure 4.4. Van der Pol with path constraint—State profiles

Scheme & norm	Objective	Sens./St. CPU	CPU	#QPs	#LSs	ϵ
Unconstrained	7.9556×10^{-2}	6.2	115.6[2]	29	29	—
Gritsis (1990)	0.18040	—	—	—	—	—
(0) ($\max^2$)	0.17989	15.1	675.0	39	46	10^{-7}
(0) (max)	0.18028	15.2	860.8	49	64	10^{-5}
(4) ($\max^2$)	0.18006	10.9	285.2	25	25	10^{-7}
(4) (max)	0.18018	10.1	344.2	33	70	10^{-5}

Table 4.2. Results for Jacobson and Lele constrained problem

4.4.2 Jacobson and Lele Problem

Following Gritsis (1990), the following path constraint is added to the problem considered in section 3.15:

$$y_2(t) - 8\,(t - 0.5)^2 + 0.5 \leq 0 \tag{4.42}$$

The schemes considered here are only (0) and (4), as the latter has proved to be most effective. The starting point and the bounds are exactly the same as those used in the unconstrained case. Table 4.2 summarizes the run results.

Again the discretization scheme with the squared maximum norm solves the problem in fewer QP iterations, even fewer than in the unconstrained case. The corresponding control profile is shown in Figure 4.5 and the profiles of states y_1 and y_2 are shown in Figure 4.6.

[2]Obtained on a SUN SPARC 2 workstation.

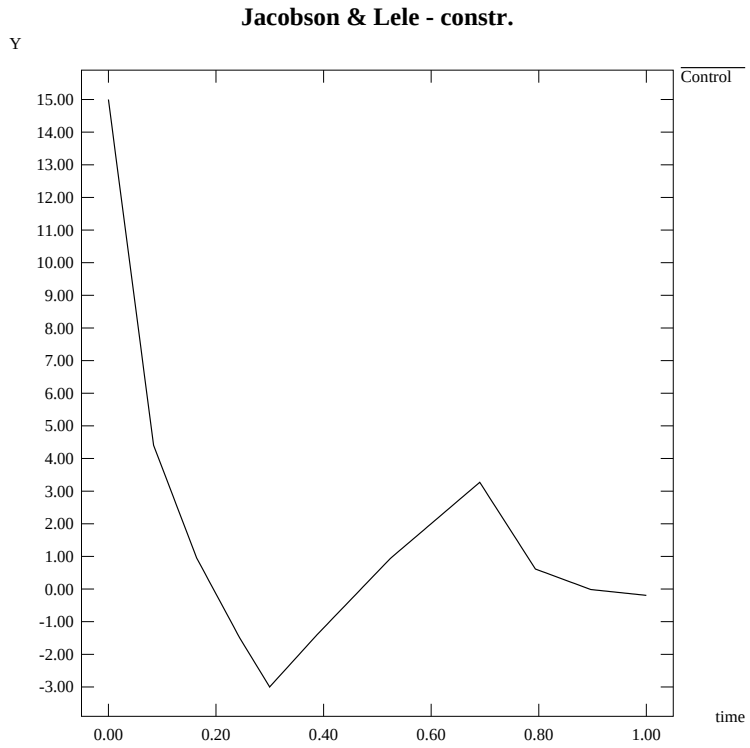

Figure 4.5. Jacobson and Lele with path constraint—Control profile

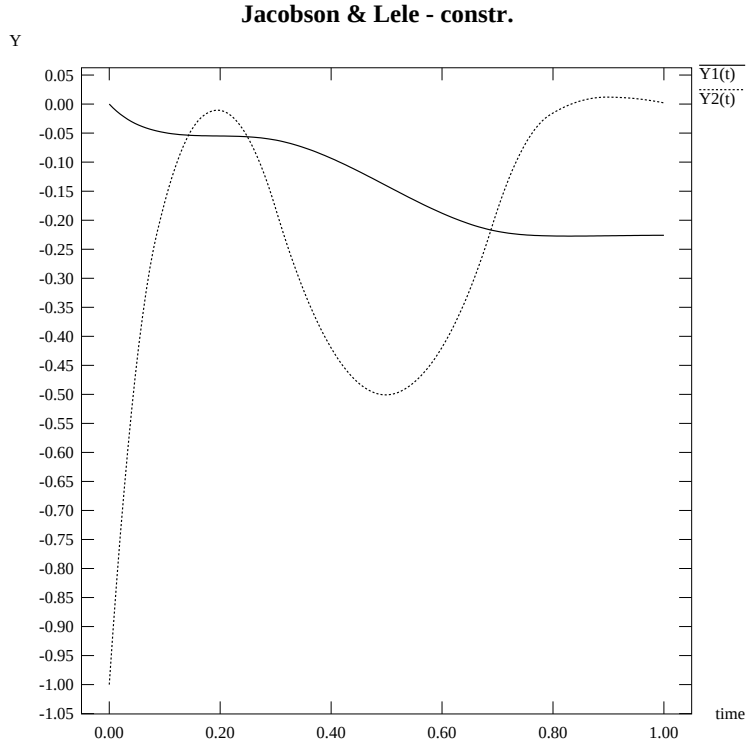

Figure 4.6. Jacobson and Lele with path constraint—State profiles

Scheme & norm	Objective	Sens./St. CPU	CPU	#QPs	#LSs	ϵ
Unconstrained	30.000001	7.7	2.5	5	4	—
(0) ($\max^2$)	37.500167	4.8	46.3	40	60	10^{-7}
(0) (max)	39.893559[3]	3.6	125.3	128	154	10^{-5}
(4) ($\max^2$)	37.500002	7.5	9.2	8	9	10^{-7}
(4) (max)	37.500001	6.6	19.7	19	22	10^{-5}

Table 4.3. Results for car problem with path constraint

4.4.3 Car Problem

To the same problem presented in section 3.13, the following path constraint is enforced on the velocity of the vehicle:

$$y_1(t) - 10.0 \leq 0 \tag{4.43}$$

The optimal solution found for the unconstrained case was used as the starting point for the constrained problem, as convergence from other starting points was found to be difficult. The lower bound on the element sizes was increased to 1.0 from 0.01 as the latter is very small, while their number remains 3. The discretized path constraint was enforced at the two interior points, since the starting point satisfies this constraint and the final point has an equality constraint of zero on the value of the velocity. The results for the various runs are summarized in Table 4.3.

The squared maximum norm with discretization outperforms all other methods, keeping the number of iterations very close to those of the unconstrained case. The use of the maximum norm alone resulted in oscillation and converged, after a long run, to a suboptimal point. The optimal acceleration profile is shown in Figure 4.7, the velocity profile in Figure 4.8 and finally the optimal distance profile in Figure 4.9. The optimal lengths for the three elements are 10.000, 22.500, and 5.000 respectively.

[3]This result is suboptimal.

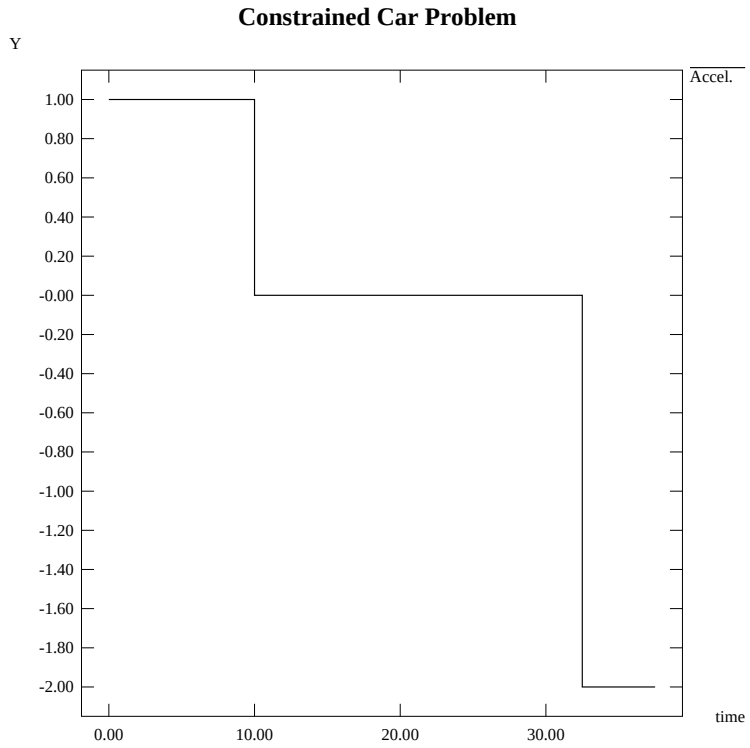

Figure 4.7. Constrained car problem—Acceleration profile

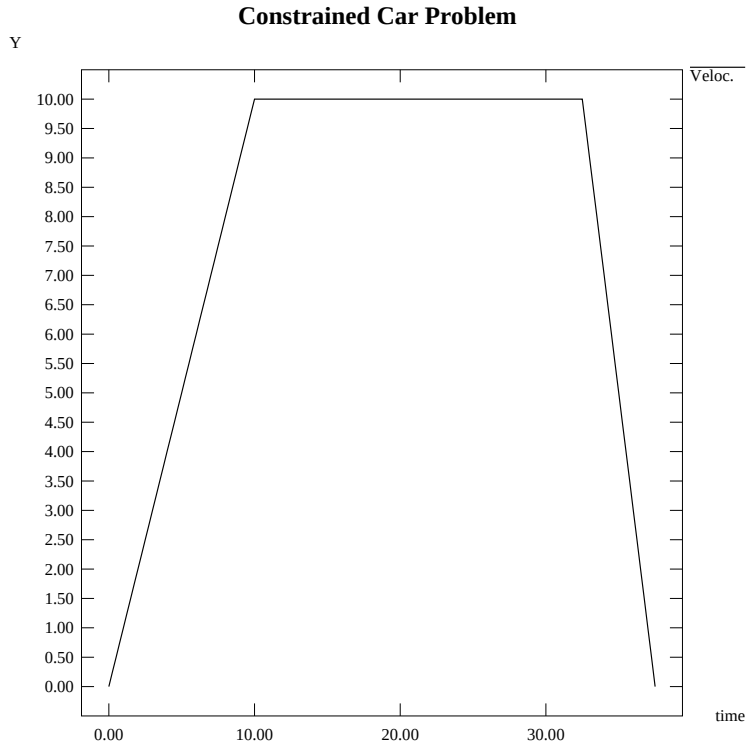

Figure 4.8. Constrained car problem—Velocity profile

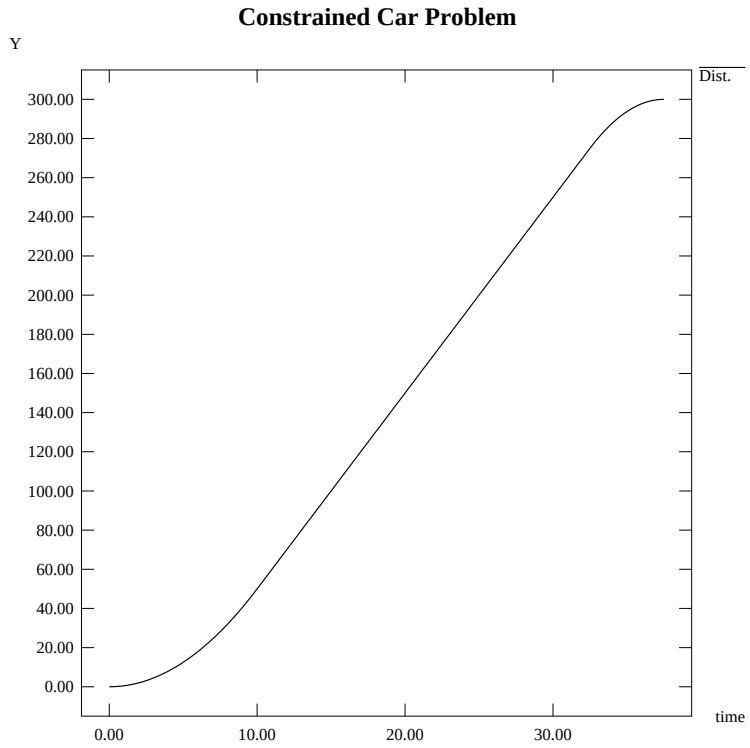

Figure 4.9. Constrained car problem—Distance profile

4.4.4 Hot-Spot Reactor Problem

This problem is the same as the one examined in section 3.2, with the addition of a path constraint on the temperature in the tubular reactor, namely:

$$T(t) \leq 1.45 \tag{4.44}$$

This is the constraint considered by Vasantharajan and Biegler (1990). The starting point and the bounds on the optimization parameters are the same as those used in the unconstrained case.

The element sizes, which determine the positions of the point constraints discretizing the path constraint, were kept equal by appropriate constraints during the optimization. However, the total length of the time horizon (*i.e.* t_f) was free to vary. Design parameter related values are given in Table 4.4 and the solution statistics are summarized in Table 4.5. The optimal design parameters reported in this work correspond to the run with the $\max^2$ norm and discretization (scheme (4)($\max^2$)).

The result reported by Vasantharajan and Biegler is actually suboptimal. The optimal parameter values they derived were used in a simulation run and the correct value of the objective function is given in Table 4.4. Their values result in an inactive path constraint with the temperature peaking at 1.44 normalized units, whereas the profile in this work has the constraint active with a peak value of 1.454 units (which violates the constraint slightly).

				Optimum	
Parameter	Initial Value	Lower Bound	Upper Bound	This Work	Reference
T_P	250.0	120.0	1000.0	213.6	232.1
T_R	462.23	400.0	500.0	500.0	500.0
L	1.0	0.5	1.25	1.25	1.25
T_S	425.25	400.0	500.0	451.5	450.5

Table 4.4. Constrained hot-spot reactor—Design parameters

Scheme & norm	Objective	Sens./St. CPU	CPU	#QPs	#LSs	ϵ & ϵ_i
Unconstrained	-171.4879	4.0	29.1[4]	7	6	—
(0)(max^2)	-152.8485	4.0	37.2	10	10	10^{-7} (ϵ)
(0)(max)	-152.2742	3.7	25.0	7	7	10^{-5} (ϵ)
(4)(max^2)	-152.8518	6.7	76.8	9	8	10^{-7} & 3.2×10^{-4} (ϵ_i)
(4)(max)	-152.2788	13.8	152.24	8	7	10^{-5} & 3.2×10^{-4} (ϵ_i)
Reference	-148.4730	—	—	—	—	—
Ref. corrected	-148.0847	—	—	—	—	—

Table 4.5. Constrained hot-spot reactor—Solution statistics

The conversion and temperature profiles are shown in Figure 4.10.

In contrast to the earlier problems considered earlier in this chapter, the use of path constraint discretization (scheme 4) does not result in any performance improvement in comparison to the simpler scheme 1. In fact, the CPU times for the problems with discretization are unnecessarily long due to the evaluation of the sensitivities with respect to the control element lengths; even though the latter were equal, they are always considered as optimization parameters in the optimal control code DAEOPT. This feature may be undesirable at times; however this is merely an implementation aspect.

It must be noted that the convergence of the squared maximum norm problem with discretization was found to be difficult when the discrete constraints were enforced with upper bound 1.45 temperature units. It was, in fact, necessary to relax the bound to 1.5 temperature units, without any effect on the solution,

[4]Obtained on a SUN SPARC 2 workstation.

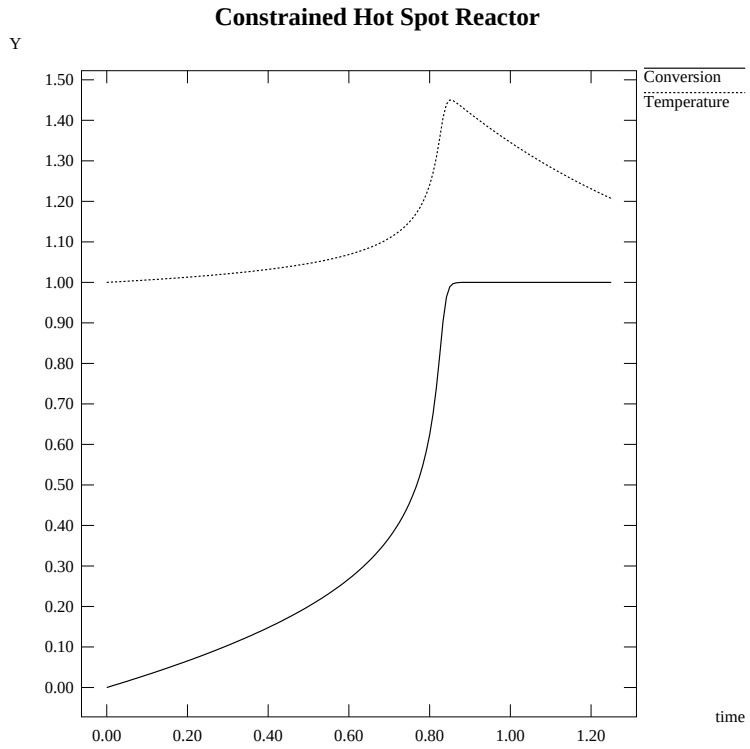

Figure 4.10: Conversion and temperature profiles for constrained hot-spot reactor

the feasibility of which is ensured by the end-point constraint.

4.4.5 Pendulum Problem

This problem originates from a high-index DAE formulation of a simple pendulum. The original problem is described by the following index-3 DAE system (Jarvis and Pantelides, 1992b) :

$$\dot{y}_1 = y_3\,; \quad y_1(0) = 1.0 \tag{4.45}$$

$$\dot{y}_2 = y_4\,; \quad y_2(0) = 0.0 \tag{4.46}$$

$$\dot{y}_3 = -y_5\, y_1\,; \quad y_3(0) = 0.0 \tag{4.47}$$

$$\dot{y}_4 = -y_5\, y_2 + 1.0\,; \quad y_4(0) = 0.0 \tag{4.48}$$

$$y_1^2 + y_2^2 = 1.0 \tag{4.49}$$

It can be verified that the initial conditions given above are consistent.

Jarvis and Pantelides proposed the reformulation of such problems as equality state constrained optimal control problems. This involves a state variable becoming a control, with its profile being optimized to minimize some norm of violation of one of the algebraic equations in the system.

In the particular case of the pendulum problem described above, the algebraic variable y_5 becomes a control:

$$u(t) = y_5(t) \tag{4.50}$$

and the algebraic constraint (4.49) is enforced by minimizing the integral of its squared residual. Thus, the optimal control problem is:

$$\min_{u(\cdot)} \int_0^{t_f} \left(y_1^2 + y_2^2 - 1.0\right)^2 \, dt \tag{4.51}$$

subject to the ODEs (4.45)–(4.48).

The problem was run with $t_f = 3.60$, using a starting profile that rises linearly from 0.5 to 3.0 and then descends to 0.0. This captures qualitatively the behaviour of the expected optimal profile, but, even so, convergence from it was found to be difficult. 10 control elements were employed, initially being equidistributed in the time horizon.

Scheme & norm	Objective	Sens./St. CPU	CPU	#QPs	#LSs	ϵ
no int. constr.	4.6×10^{-7}	11.4	1320.3	59	72	—
$c^2 \leq \epsilon$	1.8×10^{-6}	12.5	784.2	30	46	10^{-7}
$-\sqrt{\epsilon} \leq c \leq \sqrt{\epsilon}$	4.7×10^{-7}	9.1	216.3	12	13	3.2×10^{-4}

Table 4.6. High-index pendulum problem—Solution statistics

Element	No constr.	$c^2 \leq \epsilon$	$-\sqrt{\epsilon} \leq c \leq \sqrt{\epsilon}$
1	0.33996	0.20498	0.25919
2	0.33938	0.54444	0.40263
3	0.33230	0.35497	0.21413
4	0.41234	0.28678	0.31090
5	0.40701	0.31055	0.35967
6	0.35639	0.38608	0.41084
7	0.30766	0.33691	0.42717
8	0.30709	0.29890	0.41045
9	0.39070	0.42168	0.37371
10	0.40717	0.45471	0.43131

Table 4.7. High-index pendulum problem—Optimal element sizes

To test the effect of local information on the convergence, two more problems were designed. The first imposed constraints on the squared value of the violation of the equality path constraint at interior points, and the second imposed constraints on the absolute value of the violation directly:

$$\left(y_1^2(t_i) + y_2^2(t_i) - 1.0\right)^2 \leq \epsilon\,; \quad i = 0, 1, 2, \ldots, NS \tag{4.52}$$

$$-\sqrt{\epsilon} \leq y_1^2(t_i) + y_2^2(t_i) - 1.0 \leq \sqrt{\epsilon}\,; \quad i = 0, 1, 2, \ldots, NS \tag{4.53}$$

where ϵ was set to 10^{-7}. The results are summarized in Table 4.6. All runs were conducted using level 3 tolerances of Table 3.1. The optimal control element sizes are given in Table 4.7.

The second type of interior point constraints eq. (4.53) is found to improve the convergence dramatically, with only 12 QPs being required to converge to the optimum. The use of the squared violation interior constraints eq. (4.52)

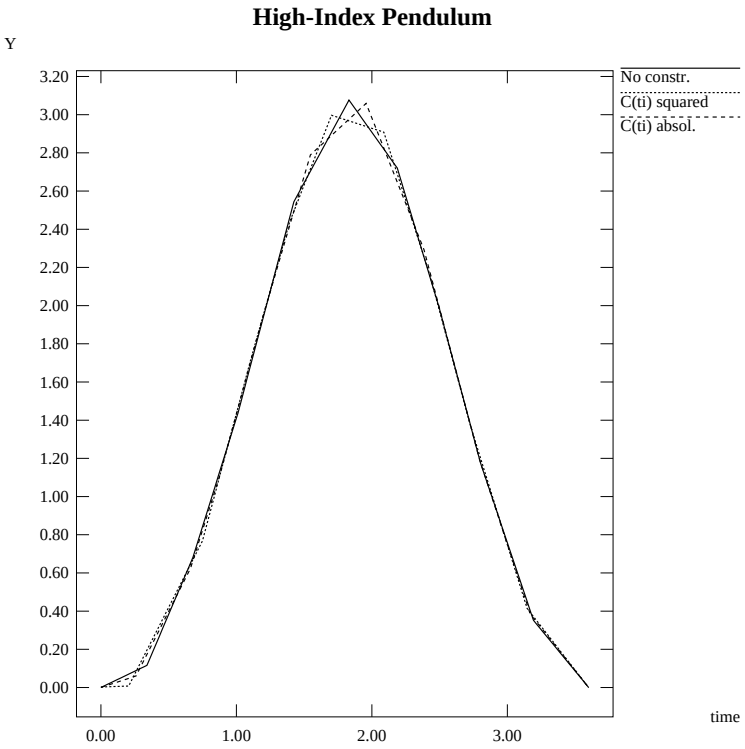

Figure 4.11. High-index pendulum—Control profiles

did accelerate convergence, but resulted in a higher value of the objective. Also the control profile of that run did not exhibit the same trend as the others, particularly in the first 2 elements. The control profiles are shown for all three cases superimposed in Figure 4.11. The four state variable profiles obtained from the third run are shown in Figure 4.12.

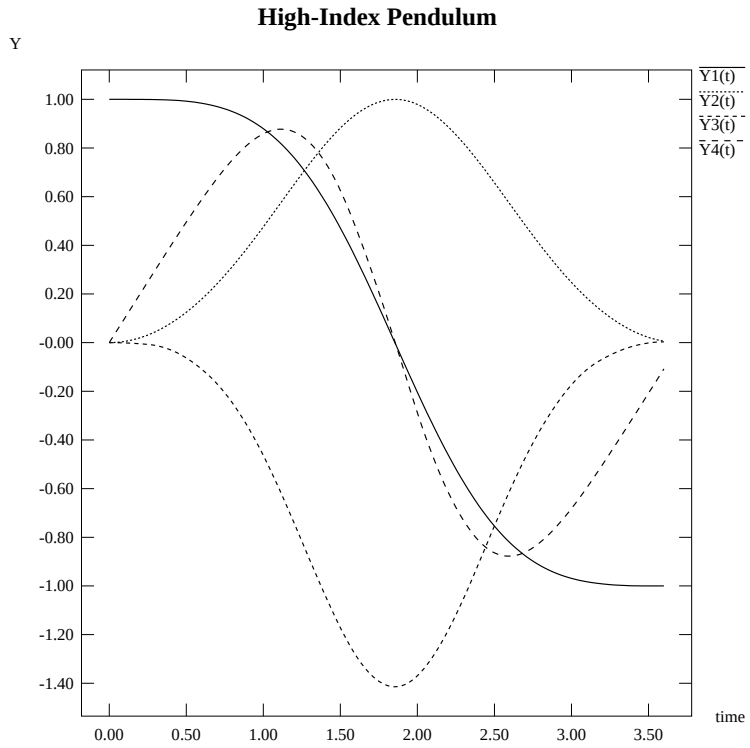

Figure 4.12. High-index pendulum—State profiles

4.4.6 Isoperimetric Problem Revisited

The results of the examples presented thus far showed that the speed of convergence for the constrained problems was in almost all cases higher than for the unconstrained ones. To test whether artificial path constraints would have the same effect on unconstrained problems, provided they are inactive at the optimum, the isoperimetric problem of section 3.11 was considered.

By analyzing the problem, the following loose bounds on the length of arc and the value of the arc function can be derived:

$$y_3(t) \leq 1.5 \tag{4.54}$$

$$y_1(t) \leq 0.56 \tag{4.55}$$

The first one is merely the equality end-point constraint on the length of arc transformed to an inequality constraint and enforced at interior times, while the second is derived by considering the isosceles triangle having as third side the chord of length 1.0. This triangle is the one with maximum height, therefore maximum possible $y_1(t)$. Both constraints will be inactive at the optimum.

First the problem was run with discretization of the constraints only, at the element end-points of the 5 elements used. The second run had enforced additionally to the discretized constraints the maximum squared violation norm of each of eqs. (4.54) and (4.55) as well, both bounded by $\epsilon = 10^{-4}$, which is one order of magnitude smaller than the optimization tolerance. All starting points were the same as reported in section 3.11, and the tolerances were at level 2 of Table 3.1. Table 4.8 summarizes the results for the two cases considered here along with that for the original free problem. Constraint (4.54) was not enforced at the final point of the horizon since it already is required to be an equality at that time in the original formulation of the problem.

No significant difference existed between the optimal control profiles and element sizes obtained here, and those for the unconstrained problem reported in section 3.11. On the other hand, the convergence of the last case shows that the combination of the maximum squared norm and discretization managed to almost halve the QPs required to reach the optimum.

Scheme & norm	Objective	Sens./St. CPU	CPU	#QPs	#LSs	ϵ
free	0.3564474	4.5	130.8[5]	49	55	—
discr. only	0.3565579	4.7	104.5	36	41	—
discr. & $\max^2$	0.3563809	5.1	98.7	26	36	10^{-4}

Table 4.8. Isoperimetric problem revisited—Solution statistics

This reduction may be due to the redundant constraints being active during the optimization procedure, thus providing additional guidance to the search. Although this was achieved at the expense of solving a much larger QP at each iteration, the dominant cost in optimal control is, by far, that of the integration of the states and sensitivity equations—and hence the overall effect on performance is quite beneficial.

4.5 Conclusions

In this chapter the treatment of general equality constraints was considered in the context of the control parameterization approach to the optimization of dynamic systems described by mixed sets of differential and algebraic equations (DAEs) of index not exceeding 1.

Equality path constraints are handled by incorporating as many of them as possible within the DAE system itself without increasing its index. This allows a subset of the control variables to be determined from the solution of the augmented DAE system. The issues involved in establishing an appropriate partitioning of the control variable vector are examined.

Inequality and equality path constraints are handled through several schemes that introduce discrete point constraints either on their own or supplementing an integral measure of their violation which is forced to zero. The most successful scheme was found to be one using the squared maximum violation norm integral as an end-point ϵ relaxed constraint supplemented by constraints derived from

[5]Obtained on a SUN SPARC 2 workstation.

the discretization of the path constraint at a small number of points in the time horizon. This scheme was found to be the most reliable and efficient on all 6 case studies considered.

CHAPTER 5

Conclusions

The theme of this thesis has been the development of an efficient computational framework for the solution of general optimal control problems. Four major areas were addressed:

- Development of a general purpose optimal control algorithm.
- Handling of a special class of multistage problems.
- Efficient solution of path constrained problems.
- Modelling aspects and solution enhancement techniques.

All these are examined in some detail in the sections to follow.

5.1 Optimal Control Algorithm

To obtain a discretization of the optimal control problem, thus transforming it to a finite dimensional optimization problem, control parameterization was preferred over discretization of all variables mainly for the reason that the latter results in excessively large optimization problems when the underlying original dynamic model is large.

The optimal control algorithm designed and implemented in the FORTRAN subroutine DAEOPT focused exclusively on index one DAE systems which can be solved using standard integration codes. It is interesting to note, however, that, through reformulation, it is possible to solve high-index DAE problems as optimal control problems of index one, as the example of the pendulum given in section 4.4.5.

The control profiles were assumed to be piecewise continuous over finite time-elements with the possibility of discontinuity at the element boundaries. The basis functions are selected to be Lagrange polynomials which allow an easy and physically meaningful bounding of the discretization parameters which are the values of the approximated controls at the discretization points in each element.

To obtain gradients, sensitivities were selected over adjoints because they are easily and efficiently calculated directly by the integration procedure within a BDF algorithm. Moreover, and more importantly, the implementation of interior point constraints is straightforward, as no backward integrations are required. Sensitivities are also easy to map across state discontinuity points, and more generally, across different systems in a multiperiod optimal control problem. Also, sensitivities are directly translated into gradients for the constraints. Adjoints, on the other hand require additional calculations to derive gradients.

To get an estimate of the average cost of a sensitivity integration relative to a state only integration as the number of parameters increases, the case studies examined in this work have been used to yield the information shown in Figure 5.1.

An estimate of the number of integrations that would be required to evaluate gradients using forward (or backward) differences is used as a measure of size. Time invariant parameters will require a full integration, whereas the evaluation of gradients with respect to parameters occurring in a certain element require only an integration that starts from the beginning of that element. In this way the number of complete integrations required for a forward (or backward) finite difference scheme becomes:

$$INT = \frac{NS+1}{2} \times NP + ND \tag{5.1}$$

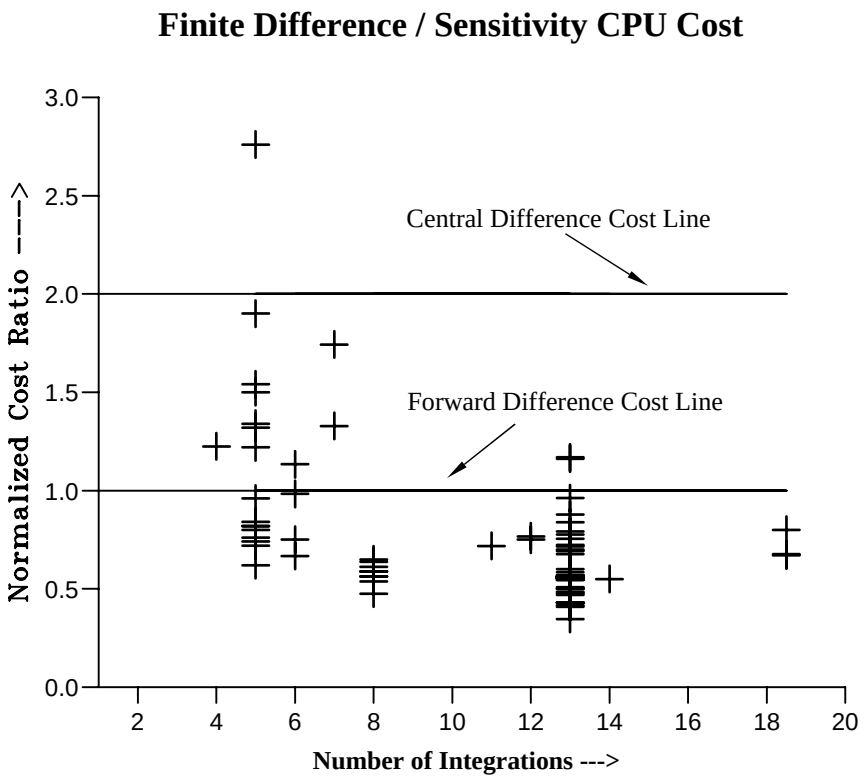

Figure 5.1. Sensitivity evaluation cost comparisons

where NS is the total number of elements, NP is the number of parameters per element resulting from the discretization of the controls and including the element size parameter, and ND is the number of design parameters, for example initial conditions to be optimized, problem design variables, and control profile initial values. Assuming that the cost of integrating the equations over each control element is roughly the same, the above is a better measure of computational cost than the total number of parameters ($NS \times NP + ND$).

The vertical axis shows the value of the ratio of CPU time ratios for one sensitivity integration with respect to a state integration, over the corresponding estimate INT derived above. In this way forward and backward differences have value of 1.0 for each INT and central differences requiring two perturbations per parameter have a value of 2.0. Forward (or backward) differences will not require an additional base calculation since the information is available during the last line search that selected the new parameter point. From the results displayed, which are obtained for 79 runs conducted in the course of this work, the forward difference approach appears cheaper for almost half the case studies with $INT < 8$. The sensitivity approach is cheaper for $INT \geq 8$ than the use of finite difference schemes. Central differences are totally non-competitive for all cases with a single exception. It appears that high levels of discretization of the control favour sensitivities which become cheaper.

5.2 Multistage Problems

A new contribution in optimal control problem modelling and solution is the consideration of a special class of multistage systems, which have generalized junction and initial conditions between stages and retain the same number of DAEs from stage to stage (*cf.* the discrete charge reactor problem of section 3.1). Indeed, such problems often arise in real models and current trends in dynamic simulation are now starting to take them into account (Pantelides and Barton, 1992).

The research within the scope of this work has provided the first unified method for optimizing problems with generalized junction and initial condi-

tions involving DAE states at a specified time. The essential steps are a re-initialization of the DAEs at the time of discontinuity and the mapping of sensitivities across the discontinuity.

In general, the starting point of each element used in the discretization of the control is a candidate for the application of generalized junction conditions that, coupled with the DAEs of the system, can be solved to provide the condition of the system at the start of the new element. This formulation also allows the DAE system to change structurally, provided the number of equations and variables involved does not change. In this sense, the problem considered is a special subset of the general case of multistage optimal control problems.

5.3 Path Constrained Problems

Another interesting outcome in this work is the proposed technique for handling inequality path constraints. Earlier techniques for handling such constraints were notorious for their slow convergence, and the solution of constrained problems was generally considered in the literature to be much more difficult than that of the solution of unconstrained counterparts.

Feasible path methods such as control parameterization resorted to the use of the maximum squared norm violation integral, which was required to be equal, or nearly equal, to zero at the final time, thus providing a measure of the global feasibility of the state trajectory with respect to the constraint. This approach suffered from ill-conditioning and a slow convergence when the optimum was approached. The idea of relaxing the constraint to an inequality with an ϵ tolerance (Walsh, 1991) is very effective in avoiding the region of oscillation (Goh and Teo, 1988) and discontinuity of such constraints.

Infeasible path methods, *i.e.* those based on collocation, on the other hand claim faster convergence but are in many cases impractical to implement due to the large size of the resulting NLP. No satisfactory analysis of the numerical behaviour of such methods with path constrained problems has been conducted up to now. A fundamental shortcoming is the inability of such a scheme to

provide a measure of the global feasibility of the trajectories, and to enforce it. Furthermore, in practice enforcing path constraints in difficult problems will require very fine discretizations.

Examining both approaches led to the development of a technique that indeed shows that path constrained problems are not more difficult to solve than the unconstrained ones. The technique consists of a scheme that enforces the path constraint using the squared maximum violation norm integral as an end-point ϵ relaxed inequality constraint and also enforces a small number of constraints derived from a discretization of the path constraint in the interior of the time domain. The points selected for this discretization were the element end-points, thus keeping the number of interior point constraints induced in this fashion relatively small. The essential advantage of this method is the provision of local trajectory information to the optimizer.

The technique has been found to result in fast convergence in terms of the number of required QPs in the SQP optimization algorithm employed. In fact, the incorporation of redundant path constraints (derived from analysis of the underlying dynamic model) that are inactive at the solution has been found to result in the acceleration of the solution of an originally unconstrained problem. This is because path constraints actually reduce the feasible search space, thus accelerating convergence to the solution.

Overall, interior discretizations provide the speed which is most beneficial in the initial steps of the optimization process while far from the solution point. At later stages, global feasibility is controlled by the violation norm which takes over. Finally, use of ϵ-relaxation alleviates oscillation around the optimum.

A case where, even with this approach, obtaining the solution may be problematic is when the violation occurs near a sharp peak of the value of the constraint, as was the case with the hot-spot reactor example (section 4.4.4). The only possible remedy within this framework is to increase the number of interior point constraints, so that the chances of capturing the peak become greater.

5.4 Modelling Aspects

Several techniques for reducing solution time and enhancing accuracy for the derived solution have been implemented in the course of this research.

One method that was found very effective in refining control profiles is to start the approximations at a low order polynomial, using piecewise linear profiles or even piecewise constant. Experience has shown that piecewise linear profiles are more than sufficient for achieving high accuracies in most cases. Nevertheless, should smoother profiles be required it is possible to restart the optimization for a higher order polynomial using as a starting point an interpolation of the previous optimal profile obtained with the lower order polynomial (Gill *et al.*, pp. 266–267, 1988).

In this work a further step was taken in which the bounds on the control discretization points were reduced around the old profile so as to produce an envelope within which the new optimum is sought. This provides a significant reduction of the search space and usually convergence is achieved in a fraction of the QP iterations required for the lower order control profile approximation.

Bounding the control variables was also found to be a major issue. While piecewise constant and piecewise linear approximations can guarantee the satisfaction of control bounds directly, higher approximations may require special attention if the profile approaches the bounds in the optimum. One way to overcome such problems is to solve the problem initially using piecewise linear approximations in order to identify regions over which the controls are at, or near, their bounds. Subsequently, the problem may be re-solved using higher order polynomial approximations, with the controls kept fixed at their bounds over the critical region(s). Of course, the solution obtained in this way may be suboptimal for the particular higher order approximation, but it will still be better than the piecewise linear one.

5.5 Directions for Future Work

In our implementation of the optimal control algorithm, all DAE integrations are performed using the same tolerance. However, the automatic adjustment of the integration tolerance during the various stages of the optimization procedure may well speed up the solution procedure of a feasible path algorithm. Successive integrations should have their tolerances adjusted so that the convergence properties of the optimization algorithm are not impaired by numerical noise in gradients and function values, at the same time trying to keep accuracies as low as possible for the integrations to proceed fast. This issue requires further investigation.

Furthermore, the DAE integrator employed automatically adjusts the step length and order of integration for each integration carried out. The re-use of the step history from a previous integration could possibly speed up a subsequent integration stage, especially for integrations close to the region in which the convergence of the optimization is superlinear.

Parallelization of the sensitivity evaluations will help significantly in the reduction of CPU time. In particular, the sensitivity of the state vector with respect to any optimization parameter uses the same Jacobian factorization, and therefore the back-substitutions can be performed in parallel.

One issue that has received little attention in the literature of optimal control algorithms is the problem of handling discrete decisions. This would be a mixed integer optimal control problem (MIOCP) and there could be scope to examine its solution in the light of existing mixed integer nonlinear programming techniques. Such problems occur in a number of process engineering applications. For example, it may be impossible to use a reactor unless the volume of its contents exceeds a certain lower bound; also controls are often allowed to take only integer values.

REFERENCES

[1] Achenie, L. K. E., and L. T. Biegler, "A Superstructure Based Approach to Chemical Reactor Network Synthesis", *Comput. chem. Engng.*, **14** (1), 23–40 (1990).

[2] Amarger, R. J., L. T. Biegler, and I. E. Grossmann, "An Automated Modelling and Reformulation System for Design Optimization", *Comp. chem. Engng.*, **16** (7), 623–636 (1992).

[3] Aris, R., and N. R. Amudson, "An Analysis of Chemical Reactor Stability and Control—I", *Chem. Eng. Sci.*, **7**, 121–131 (1958).

[4] Aris, R., and N. R. Amudson, "An Analysis of Chemical Reactor Stability and Control—II", *Chem. Eng. Sci.*, **7**, 132–147 (1958).

[5] Ascher, U., "Collocation for Two-Point Boundary Value Problems Revisited", *SIAM J. Numer. Anal.*, **23** (3), 596–609 (1986).

[6] Bartholomew-Biggs, M. C., L. C. W. Dixon, S. E. Hersom, and Z. A. Maany, "The Solution of Some Difficult Problems in Low-Thrust Interplanetary Trajectory Optimization", *Opt. Cont. Appl. & Meth.*, **9**, 229–251 (1988).

[7] Bellman, R., *Dynamic Programming*, Princeton University Press, Princeton, New Jersey (1957).

[8] Biegler, L. T., "Solution of Dynamic Optimization Problems by Successive Quadratic Programming and Orthogonal Collocation", *Comput. Chem. Engng.*, **8** (3), 243–248 (1984).

[9] Brenan, K. E., S. L. Campbell, and L. R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, Elsevier, New York (1989).

[10] Bryson, A. E., and Y.-C. Ho, *Applied Optimal Control*, Hemisphere, Washington (1988).

[11] Bulirsch, R., *Die Mehrzielmethode zur numerischen Lösung von nichtlinearen Randwertproblemen und Aufgaben der optimalen Steuerung*, Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt, Oberpfaffenhofen, Federal Republic of Germany, report of the Carl-Cranz Gesellschaft (1971).

[12] Canon, M. D., C. D. Cullum, and E. Polak, *Theory of Optimal Control and Mathematical Programming*, McGraw-Hill (1970).

[13] Caracotsios, M., and W. E. Stewart, "Sensitivity Analysis of Initial Value Problems with Mixed ODEs and Algebraic Equations", *Comput. chem. Engng.*, **9**, 359–365 (1985).

[14] Carey, G. F., and B. A. Finlayson, "Orthogonal Collocation on Finite Elements", *Chem. Engng. Sci.*, **30**, 587–596 (1975).

[15] Chen, C. L., "A Class of Successive Quadratic Programming Methods for Flowsheet Optimization", Ph.D. Thesis, University of London (1988).

[16] Chen, C. L., and S. Macchietto, "Successive Reduced Quadratic Programming —SRQP— User's Guide", Dept. of Chem. Eng., Imperial College, London (1989).

[17] Cuthrell, J. E., and L. T. Biegler, "On the Optimization of Differential-Algebraic Process Systems", *AIChE J.*, **33** (8), 1257–1270 (1987).

[18] Dixon, L. C. W., and M. C. Bartholomew-Biggs, "Adjoint-Control Transformations for Solving Practical Optimal Control Problems", *Opt. Cont. Appl. & Meth.*, **2**, 365–381 (1981).

[19] Dunker, A. M., "The Decoupled Direct Method for Calculating Sensitivity Coefficients in Chemical Kinetics", *J. Chem. Phys.*, **81** (5), 2385–2393 (1984).

[20] Edgar, T. F., and D. M. Himmelblau, *Optimization of Chemical Processes*, McGraw-Hill, Singapore (1989).

[21] Feng, A., C. D. Holland, and S. E. Gallun, "Development and Comparison of a Generalized Semi-Implicit Runge-Kutta Method with Gear Method for Systems of Coupled Differential and Algebraic Equations", *Comp. chem. Engng.*, **8** (1), 51–59 (1984).

[22] Ferziger, J. H., *Numerical Methods for Engineering Application*, Wiley, New York (1981).

[23] Finlayson, B. A., "Packed Bed Reactor Analysis by Orthogonal Collocation", *Chem. Eng. Sci.*, **26**, 1081–1091 (1971).

[24] Finlayson, B. A., *Nonlinear Analysis in Chemical Engineering*, McGraw-Hill, New York (1980).

[25] Fletcher, R., *Practical Methods of Optimization*, John Wiley & Sons (1991).

[26] Gear, C. W., "Simultaneous Numerical Solution of Differential-Algebraic Equations", *IEEE Trans. on Circ. Th.*, **CT-18** (1), 89–95 (1971).

[27] Gill, P. E., W. Murray, and M. H. Wright, *Practical Optimization*, Academic Press, Sand Diego (1988).

[28] Goh, C. J., and K. L. Teo, "Control Parameterization: A Unified Approach to Optimal Control Problems with General Constraints", *Automatica*, **24** (1), 3–18 (1988).

[29] Golub, G. H., and C. F. Van Loan, *Matrix computations*, John Hopkins University Press, Baltimore (1989).

[30] Gritsis, D., "The Dynamic Simulation and Optimal Control of Systems Described by Index Two Differential-Algebraic Equations", Ph.D. Thesis, University of London (1990).

[31] Gunn, D. J., and W. J. Thomas, "Mass Transport and Chemical Reaction in Multifuctional Catalyst Systems", *Chem. Eng. Sci.*, **20**, 89–100 (1965).

[32] Horwitz, L. B., and P. E. Sarachik, "Davidon's Method in Hilbert Space", *SIAM J. Appl. Math.*, **16** (4), 676–695 (1968).

[33] Jacobson, D. H., and M. M. Lele, "A Transformation Technique for Optimal Control Problems with a State Variable Inequality Constraint", *IEEE Trans. Autom. Cont.*, **AC-14**, 457–464 (1969).

[34] Jacobson, D. H., and D. Q. Mayne, *Differential Dynamic Programming*, American Elsevier, New York (1970).

[35] Jarvis, R. B., and C. C. Pantelides, "DASOLV, User's Manual, Version 1.0", Centre for Process Systems Engineering, Imperial College, London, January, (1992a).

[36] Jarvis, R. B., and C. C. Pantelides, "A Differentiation-Free Algorithm for Solving High-Index DAE Systems", *AIChE Annual Meeting, Miami Beach, 1–6 November*, (1992b).

[37] Keller, H. B., *Numerical Methods for Two-Point Boundary Value Problems*, Blaisdell, Waltham, Mass. (1968).

[38] Kelley, H. J., "Gradient Theory of Optimal Flight Paths", presented at Am. Rocket Soc. Semi-Annual Meeting, Los Angeles, California, May 9–12, 1960; *ARS Journal*, **30**, 947–954 (1960).

[39] Kelley, H. J., "Method of Gradients", in *Optimization Techniques with Applications to Aerospace Systems*, G. Leitman (ed.), Academic Press, New York, 205–254 (1962).

[40] Kelley, H. J., "A Trajectory Optimization Technique Based Upon the Theory of the Second Variation", *Progress in Astronautics and Aeronautics*, **14**, 559–582 (1964).

[41] Kokossis, A. C., and C. A. Floudas, "Optimization of Complex Reactor Networks—I. Isothermal Operation", *Chem. Eng. Sci.*, **45** (3), 595–614 (1990).

[42] Kraft, D., "On Converting Optimal Control Problems into Nonlinear Programming Problems", *NATO ASI Series, Comp. Math. Prog. (Schitkowski, K., Ed.)*, **15**, 261–280 (1985).

[43] Kubíček, M., and V. Hlaváček, *Numerical Solution of Nonlinear Boundary Value Problems with Applications*, Prentice Hall, Englewood Cliffs, (1983).

[44] Lasdon, L. S., Mitter, S. K., and A. D. Warren, "The Conjugate Gradient Method for Optimal Control Problems", *IEEE Tr. Aut. Contr.*, **AC-12**, 132–138 (1967).

[45] Leis, J. R., and M. A. Kramer, "Sensitivity Analysis of Systems of Differential and Algebraic Equations", *Comput. chem. Engng.*, **9** (3), 93–96 (1985).

[46] Levien, K. L., "Maximizing the Product Distribution in Batch Reactors: Reactions in Parallel", *Chem. Eng. Sci.*, **47** (7), 1751–1760 (1992).

[47] Logsdon, J. S., and L. T. Biegler, "Accurate Solution of Differential-Algebraic Optimization Problems", *Ind. Eng. Chem. Res.*, **28**, 1628–1639 (1989).

[48] Logsdon, J. S., "Efficient Determination of Optimal Control Profiles for Differential Algebraic Systems", Ph.D. thesis, Carnegie Mellon University, October (1990).

[49] Logsdon, J. S., and L. T. Biegler, "Decomposition Strategies for Large-Scale Dynamic Optimization Problems", *Chem. Eng. Sci.*, **47** (4), 851–864 (1992).

[50] Luenberger, D. G., *Introduction to Dynamic Systems,*, Wiley, New York (1979).

[51] Luenberger, D. G., *Linear and Nonlinear Programming*, Addison-Wesley, Reading, Massachusetts (1984).

[52] Luus, R., "Optimal Control by Dynamic Programming Using Systematic Reduction in Grid Size", *Int. J. Control*, **51** (5), 995–1013 (1990a).

[53] Luus, R., "Application of Dynamic Programming to High-Dimensional Non-Linear Optimal Control Problems", *Int. J. Control* **52** (1), 239–250 (1990b).

[54] Luus, R., and O. Rosen, "Application of Dynamic Programming to Final State Constrained Optimal Control Problems", *Ind. Eng. Chem. Res.*, **30** (7), 1525–1530 (1991).

[55] Luus, R., "Piecewise Linear Continuous Optimal Control by Iterative Dynamic Programming", *Ind. Eng. Chem. Res.*, **32** (5), 859–865 (1993).

[56] Mengel, A. S., "Optimum Trajectories", Proceedings of Project Cyclone Symposium 1 on REAC Techniques, Reeves Instrument Corp. and U.S. Navy Special Devices Center, New York (March 15–16), 7–13 (1951).

[57] Miele, A., Pritchard, R. E., and J. N. Damoulakis, "Sequential Gradient-Restoration Algorithm for Optimal Control Problems", *JOTA*, **5** (4), 235–282 (1970).

[58] Miele, A., Damoulakis, J. N., Cloutier, J. R., and J. L. Tietze, "Sequential Gradient-Restoration Algorithm for Optimal Control Problems with Non-differential Constraints", *JOTA*, **13** (2), 218–255 (1974).

[59] Miele, A., "Recent Advances in Gradient Algorithms for Optimal Control Problems", *JOTA*, **17** (5/6), 361–430 (1975).

[60] Miele, A., Mohanty B. P., Venkataraman, P., and Y. M. Kuo, "Numerical Solution of Minimax Problems of Optimal Control, Part 1", *JOTA*, **38** (1), 97–109 (1982a).

[61] Miele, A., Mohanty B. P., Venkataraman, P., and Y. M. Kuo, "Numerical Solution of Minimax Problems of Optimal Control, Part 2", *JOTA*, **38** (1), 111–135 (1982b).

[62] Morison, K. R., and R. W. H. Sargent, "Optimization of Multistage Processes Described by Differential-Algebraic Equations", *4th IIMAS Workshop on Numerical Analysis*, Guanajuato, July (1984).

[63] Morison, K. R., "Optimal Control of Processes Described by Systems of Differential-Algebraic Equations", Ph.D. Thesis, University of London (1984).

[64] Mujtaba, I. M., "Optimal Operational Policies in Batch Distillation", Ph.D. thesis, University of London (1989).

[65] Pagurek, B., and C. M. Woodside, "The Conjugate Gradient Method for Optimal Control Problems with Bounded Control Variables", *Automatica*, **4**, 337–349 (1968)

[66] Pantelides, C. C., and P. I. Barton, "Equation-Oriented Dynamic Simulation: Current Status and Future Perspectives", *ESCAPE 2, 5/7 Oct. Toulouse, France* (1992).

[67] Polak, E., *Computational Methods in Optimization*, Academic Press, New York (1971).

[68] Pollard, G. P., and R. W. H. Sargent, "Off Line Computation of Optimum Controls for a Plate Distillation Column", *Automatica*, **6**, 59–76 (1970).

[69] Ray, W. H., *Advanced Process Control*, McGraw-Hill, New York (1981).

[70] Renfro, J. G., "Computational Studies in the Optimization of Systems Described by Differential/Algebraic Equations" Ph.D. Thesis, University of Houston, University Park (1986).

[71] Renfro, J. G., A. M. Morshedi, and O. A. Asbjornsen, "Simultaneous Optimization and Solution of Systems Described by Differential/Algebraic Equations", *Comput. chem. Engng.*, **11** (5), 503–517 (1987).

[72] Roberts, S. M., and J. S. Shipman, *Two-Point Boundary Value Problems: Shooting Methods*, American Elsevier, New York (1972).

[73] Rosen, O., and R. Luus, "Evaluation of Gradients for Piecewise Constant Optimal Control", *Comput. chem. Engng.*, **15** (4), 273–281 (1991).

[74] Russell, R. D., and J. Christiansen, "Adaptive Mesh Strategies for Solving Boundary Value Problems", *SIAM J. Numer. Anal.*, **15** (1), 59–80 (1978).

[75] Sage, A. P., and C. C. White, *Optimum Systems Control*, Prentice-Hall, Englewood Cliffs, NJ (1977).

[76] Sargent, R. W. H., and G. R. Sullivan, "The Development of an Efficient Optimal Control Package", *Proc. 8th IFIP Conf. Optimization Tech. Pt. 2*, (1977).

[77] Stoer, J., and R. Bulirsch, *Introduction to Numerical Analysis*, Springer-Verlag, New York (1980).

[78] Sullivan, G. R., "Development of Feed Changeover Policies for Refinery Distillation Units", Ph.D. thesis, University of London (1977).

[79] Synge, J. L., and B. A. Griffith, *Principles of Mechanics*, McGraw-Hill, Tokyo (1982).

[80] Tsang, T. H., D. M. Himmelblau, and T. F. Edgar, "Optimal Control via Collocation and Non-linear Programming", *Int. J. Control*, **21** (5), 763–768 (1975).

[81] Van Dooren, R. A., "A Chebyshev Technique Applied to a Controlled Nuclear Reactor System", *Optim. Cont. Appl. Meth.*, **10**, 285–291 (1989).

[82] Vasantharajan, S., and L. T. Biegler, "Simultaneous Strategies for Optimization of Differential-Algebraic Systems with Enforcement of Error Criteria", *Comput. chem. Engng.*, **14** (10), 1083–1100 (1990).

[83] Vassiliadis, V., "Numerical Optimal Control of Processes Described by General Differential-Algebraic Constraints", Ph.D./M.Phil. Transfer Report, Department of Chemical Engineering, Imperial College, London, February, (1991).

[84] Vassiliadis, V. S., "DAEOPT, User's Manual, Version 1.0", Technical Report, Centre for Process Systems Engineering, Imperial College, London, February, (1992).

[85] Vlassenbroeck, J., and R. van Dooren, "A Chebyshev Technique for Solving Nonlinear Optimal Control Problems", *IEEE Trans. Aut. Con.*, **33** (4), 333–340 (1988).

[86] Walsh, S. P., "Integrated Design of Chemical Effluent Treatment Systems", Ph.D./M.Phil. Transfer Report, Centre for Process Systems Engineering, Imperial College, London, September, (1991).

[87] Wright, S. J., "Structured Interior Point Methods for Optimal Control", *Proc. 30th CDC, Brighton, England, December*, (1991).

INDEX