IN-15
19763
P.225

NASA Contractor Report 4376

Time-Domain Finite Elements in Optimal Control With Application to Launch-Vehicle Guidance

Robert R. Bless

GRANT NAG1-939
MAY 1991

NASA Contractor Report 4376

Time-Domain Finite Elements in Optimal Control With Application to Launch-Vehicle Guidance

Robert R. Bless
Georgia Institute of Technology
Atlanta, Georgia

Prepared for
Langley Research Center
under Grant NAG1-939

NASA
National Aeronautics and
Space Administration
Office of Management
Scientific and Technical
Information Division

1991

ACKNOWLEDGEMENTS

The author has had the privilege of having Dr. Dewey H. Hodges as his thesis advisor. The author gratefully acknowledges his help, support and effective guidance over the years and hopes to continue to do work with him in the years to come.

The author thanks Dr. Anthony Calise who has served as a secondary advisor and has taught the author much about optimal control. The author also appreciates the time and effort of the other members of his thesis committee, Dr. David Peters and Dr. J.V.R. Prasad of the School of Aerospace Engineering, and Dr. Al Ferri of the School of Mechanical Engineering.

Special thanks goes to Martin Leung. He has provided all the multiple shooting results in this proposal, along with insight to some optimal control problems. Also, Dr. Eric Corban has been of great assistance in helping the author unravel the mysteries of inequality constraints. Special thanks are also extended to Dr. Dan Moerder at NASA Langley who was the technical monitor of the grant which supported this work. Best wishes are also extended to the author's friends and colleagues over the years including Dr. Erian Armanios, Mimi Armanios, Dr. Ali Atilgan, Chris Ayers, Ashraf Badir, Bill Calhoon, Carlos Cesnik, Dr. Bryan Fortson, Mark Fulton, Dr. Andy Kurdila, Jian Li, Dave Palmer, Dr. Levend Parnas, Leon Phillips, Daryl Reece and Allen Siu.

The author is grateful to his family for their prayers, support and concern. Last, but not least, the author lovingly thanks his wife Cleta and daughter Megan. They have encouraged and supported the author's desire to obtain a doctorate, and patiently endured the author's preoccupation with the problems of optimal control.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

A time-domain finite element method is developed for optimal control problems. The theory derived is general enough to handle a large class of problems including optimal control problems that are continuous in the states and controls, problems with discontinuities in the states and/or system equations, problems with control inequality constraints, problems with state inequality constraints, or problems involving any combination of the above. The theory is developed in such a way that no numerical quadrature is necessary regardless of the degree of nonlinearity in the equations. Also, the same shape functions may be employed for every problem because all strong boundary conditions are transformed into natural or weak boundary conditions. In addition, the resulting nonlinear algebraic equations are very sparse. Use of sparse matrix solvers allows for the rapid and accurate solution of very difficult optimization problems. The formulation is applied to launch-vehicle trajectory optimization problems, and results show that real-time optimal guidance is realizable with this method. Finally, a general problem solving environment is created for solving a large class of optimal control problems. The algorithm uses both FORTRAN and a symbolic computation program to solve problems with a minimum of user interaction. The use of symbolic computation eliminates the need for user-written subroutines which greatly reduces the setup time for solving problems.

CHAPTER 1

INTRODUCTION

The calculus of variations was born at the very end of the seventeenth century through the work of such great mathematicians as Newton and Leibniz. With the birth of the calculus of variations came optimal control theory for continuous systems. The newly discovered methods of differential calculus were used at once to solve many important and practical maximum and minimum problems. The first optimization problem solved by the calculus of variations was set up and solved by Newton in 1686. The problem, of interest to aerospace engineers even today, was to choose a nose shape for minimum drag in hypersonic flow [1]. Another optimization problem was presented by John Bernoulli in 1696. Bernoulli posed the classical brachistochrone problem defined as: Among all lines connecting two given points, find the curve traversed in the shortest time by a material body under the influence of gravity [2]. This problem was solved independently by John and James Bernoulli, L'Hôpital, Leibniz, and Newton. For years, work was done on solving separate variational problems. However, the tremendous intellectual feat of creating a single method for solving variational problems belongs to the great Swiss mathematician Leonhard Euler. At the astonishingly young age of 25, Euler published his work "General Solution of the Isoperimetric Problem Taken in Its Most General Sense" [3].

Since the birth of the calculus of variations, optimization problems have been a topic of research. The optimal control problem of interest in this thesis may be described as follows. Consider a system that is completely defined by a finite number of states, *i.e.*, quantities that describe the current status of the system. The status of the system is determined by a set of first-order ordinary differential equations. The states are influenced by a finite number of control variables. The optimization problem is to choose the control variables to satisfy the given boundary conditions while minimizing (or maximizing) a given performance index, or cost functional. Use of the calculus of variations results in a multi-point boundary-value problem. Unfortunately, not many analytical solutions to these types of optimal control problems have been found beyond that which the great minds of the 17^{th} and 18^{th} centuries solved. However, the appearance of practical, high-speed digital computers in the 1950's revolutionized the field of optimal control. Computers and numerous numerical methods are now the tools for dealing with the nonlinear and complex systems of today. Still, with the ever-increasing attention given both space exploration and space travel, even more reliable and efficient numerical methods are required. This thesis deals with a new type of numerical method based on finite elements in time for solving optimal control problems. Particular emphasis is given to the computation of optimal trajectories for advanced launch vehicles.

1.1. Background

If the United States is to maintain its position as a world leader among space-faring nations, then cheaper and more reliable means for transporting people and cargo to and from space must be developed. The current Space Shuttle, as technically successful as it is, will not meet all the future needs of the United States. Studies indicate that the projected transportation needs will be best served by a mix of expendable and reusable vehicles. Specifically, the functions of one-way cargo transport to orbit and two-way passenger transport should be separated [4].

The Air Force turned to industry in May of 1987 to help meet the goals of the advanced launch system (ALS) program. The goals of the ALS program are to (1) place large payloads (in excess of 100,000 pounds) into low Earth orbit and at an order of magnitude lower cost per pound, and (2) make space launch operations, including the manufacturing, processing, and actual launch of booster vehicles, significantly more routine compared to present methods and procedures [5].

Currently, an extensive amount of ground support (typically weeks) is required to prepare the guidance system of the Space Shuttle for launch. To meet the operational requirements of the ALS program, ground support for pre-mission activities must be drastically reduced. On-board algorithms must maximize system performance as measured by autonomy, mission flexibility, in-flight adaptability, reliability, accuracy, and payload capacity. For real-time trajectory optimization to be realizable, the algorithms must be computationally efficient, robust, self-starting, and capable of functioning independently of ground control. Furthermore, the algorithms must be designed with the anticipation that the launch vehicle will undergo evolutionary growth [6].

On-board, real-time trajectory optimization algorithms are required to meet the needs of the ALS program. Such algorithms promise to (1) reduce the cost of designing flight profiles, (2) reduce the time required to respond to changed payload or mission requirements, and (3) improve vehicle performance. The diverse mission requirements of a general-purpose launch vehicle require new approaches to trajectory optimization. This work, conducted under NASA Grant NAG-1-939 of which Dr. Daniel D. Moerder is the technical monitor, was chiefly concerned with the development of a method for calculating optimal trajectories of these advanced launch vehicles.

1.2. Previous Work

There are scores of methods available for solving optimal control problems. The method chosen is dependent upon, among other things, the type of problem to be solved and the resources (in terms of both software and money) available to the user. This section will give a brief outline of some of the methods now being used to solve optimal control problems.

1.2.1. The LQ Problem

One type of problem considered frequently is one in which the system is defined by a set of linear differential equations while the performance index is a quadratic functional. This is the so-called LQ (linear-quadratic) problem. Over the past twenty years, a new technique has been established for the solution of LQ problems using orthogonal functions. The main characteristic of this technique is that the differential equations involved in the problem are reduced to algebraic equations, thus greatly simplifying the problem solution. The technique calls for the differential equations to first be converted to integral equations. Subsequently, the various unknowns involved in the integral equation are approximated by truncated orthogonal series. The key idea of this technique is to derive an integral operational matrix to convert integral expressions into algebraic equations. The form of the operational matrix is dependent upon the choice of orthogonal functions used. Various functions have been used to parameterize the system including Walsh functions [7, 8], block-pulse functions [9, 10], Chebychev functions [11], Hermite series [12], polynomial series basis vectors [13, 14], and Legendre polynomials [15].

Another approach to LQ problems is presented in [16]. The method transforms the canonical equations to a set of algebraic equations and allows approximating functions that need not satisfy the initial conditions *a priori*. This enlarges the space from which the approximating functions can be chosen. Furthermore, a Lagrange multiplier technique is used to enforce the terminal conditions on the states. Orthogonal polynomials are then used to solve the LQ problem. The idea of setting up approximating functions that do not need to satisfy boundary conditions is one of the key ideas in this current work, as will be seen in Chapter 2.

1.2.2. The Nonlinear Problem

Methods available for the solution of optimal control problems generally fall into two distinct categories: direct and indirect. Direct techniques seek to directly minimize the performance index, or cost functional, by prudent choices of the free parameters in the system. Indirect techniques, on the other hand, seek to minimize the performance index indirectly by satisfying the first-order necessary conditions for optimality as established from the calculus of variations.

The direct approach to the solution of optimal control problems first requires parameterization of the control and state time histories. The choice of parameterization schemes is not unique and success of the direct methods has been achieved using Hermite polynomials [17], Chebychev polynomials [18, 19], single-term Walsh series [20], splines [21], and the like.

Once the parameterization scheme is chosen, a parameter optimization algorithm is then used to improve the initial guess of the free parameters. These algorithms are in common use today and include variable metric techniques or quasi-newton methods [22] and variations on gradient methods. Gradient methods [1, 23] were developed to surmount the "initial guess" difficulty associated with other methods such as Newton-Raphson. They are characterized by iterative algorithms for improving estimates of the control histories, in order to come closer to satisfying the optimality conditions and the boundary conditions. First-order gradient methods usually show rapid improvements when sufficiently far from the optimal solution. However, the rate of convergence drastically decreases in the neighborhood of the solution. Second-order gradient methods have excellent convergence characteristics near the optimal solution, similar to a Newton-Raphson method. Conjugate gradient methods are very powerful because they combine the first-order and second-order gradient methods. Ref. [24] contains a thorough description of the gradient method and many other algorithmic methods in optimal control. It is noted that direct methods have been successfully used to solve trajectory optimization problems [17, 18, 25, 26].

The indirect approach to the solution of optimal control problems attempts to satisfy the necessary conditions of optimality as derived from the calculus of variations. These conditions result in multi-point boundary-value problems. Analytical solutions to such problems are generally unobtainable except for the simplest of problems. Therefore, numerical methods are usually employed.

The two main techniques for solving nonlinear multi-point boundary-value problems are shooting methods and quasilinearization methods. Shooting methods [27, 28, 29] are frequently used and can be described as follows: The initial conditions and the differential equations are satisfied at each stage of the process while the final conditions are sacrificed somewhat. A nominal solution is generated by guessing the missing initial conditions and forward integrating the differential equations. The intent is to reduce the error in the final conditions at each iteration. Quasilinearization techniques [1, 30] involve choosing nominal functions for the states and costates that satisfy as many of the boundary conditions as possible. The control is then found by using the optimality conditions. The system equations and costate equations are then linearized about the nominal values and a succession of nonhomogeneous, linear two-point boundary value problems are solved to modify the solution until the desired accuracy is obtained. Other indirect techniques include the method of adjoints, Newton-Raphson methods, and continuation methods [31, 32, 33]

1.2.3. Finite Element Methods

Finite element methods, which include the Rayleigh-Ritz and Galerkin methods [21], as well as the method of collocation have been used to solve optimal control problems [34]. Ref. [35] appears to be one of the first papers using finite elements to solve optimal control problems. Therein, the authors considered the application of a modified Ritz-Trefftz direct method to the so-called state regulator control problem. This is an LQ problem and the method leads to an approximation of the performance index of order h^7 where h is the time step involved. The modified Ritz-Trefftz method used in [35] was later extended to include problems with terminal conditions on the states [36]. Other examples of the use of the Ritz method can be found in [37] and [38]. An application of the collocation method is found in [17].

Only a few of these papers are listed here because these methods, although very accurate and useful, suffer from the same computational problems. These methods require that the approximating functions satisfy all the strong boundary conditions. Thus, in practice, certain equations are eliminated depending on the boundary conditions present for a particular problem. Another drawback of these methods is that numerical quadrature is required. This can introduce error and greatly increase the computational effort required to solve a problem. The finite element method described in this thesis avoids these two pitfalls, yielding a computationally efficient and versatile algorithm.

1.2.4. General Programs

In closing this section, a few of the commercially available programs for solving optimal control problems are mentioned. The first three programs mentioned are general-purpose problem solvers, whereas the last two are particularly designed to optimize point-mass trajectories.

The Chebychev Trajectory Optimization Program (CTOP) has been found to be useful in a wide variety of practical applications [18]. This program uses a direct technique for solving problems and parameterizes the functions using Chebychev polynomials. Penalty functions are used to enforce the equations of motion and path constraints. The Nonlinear Programming for Direct Optimization of Trajectories (NPDOT) uses piecewise polynomials and collocation to satisfy the differential equations. Results presented in [17] show that NPDOT runs much more quickly than does CTOP. A FORTRAN program called MISER (the origin of the acronym is unknown) is presented in [39]. This general-purpose software utilizes a unified computational approach to solve a wide range of optimal control problems subject to general constraints. This program appears very useful; however, a substantial amount of user programming is involved. Specifically, the user must provide a series of FORTRAN subroutines which evaluate the right-hand side of the state and costate equations. Furthermore, the user must transform the problem into the canonical form described by the authors.

POST, or Program To Optimize Simulated Trajectories, provides the capability to target and optimize point mass trajectories for a powered or unpowered vehicle operating near a rotating oblate planet [40]. POST offers the solution to a wide range of flight problems including aircraft performance, orbital maneuvers, and injection into orbit. The user can select the optimization variable, the dependent variables, and the independent variables from a list of more than 400 program variables. POST is also operational on several computer systems. Another much-used program is OTIS, Optimal Trajectories by Implicit Simulation. OTIS is a three degree of freedom (point mass) simulation program for multiple vehicles [17]. The user can simulate a wide variety of vehicles such as aircraft, missiles, re-entry vehicles, and hypervelocity vehicles. The methods used were chosen to improve speed, convergence and applicability of OTIS over existing performance programs. Both POST and OTIS are very reliable and accurate programs, but are limited in scope as compared to the three programs listed above.

1.3. Present Approach

This thesis describes in detail a time-domain finite element approach for solving optimal control problems. The so-called weak principle for optimal control problems is based on Hamilton's principle, which has traditionally been used in *analytical* mechanics as a method of obtaining the equations of motion for dynamical systems. Bailey [41] followed by several others [42, 43, 44] obtained direct solutions to dynamics problems using a form of Hamilton's principle known as the law of varying action, thus opening the door for its use in *computational* mechanics.

More recently, it has been shown that expression of Hamilton's law as a weak form (commonly referred to as Hamilton's weak principle or HWP) provides a powerful alternative to numerical solution of ordinary differential equations in the time domain [45, 46]. The accuracy of the time-marching procedure derived in [45, 46] is competitive with standard ordinary differential equation solvers. Further computational advantages were obtained in so-called mixed formulations of HWP in which the generalized coordinates and momenta appear as independent unknowns [47]. Therein, an unconditionally stable algorithm emerges for the linear oscillator with exact element quadrature. HWP also has shown to be an ideal tool for obtaining periodic solutions for autonomous systems, as well as finding the corresponding transition matrix for perturbations about the periodic solution [48]. These are complex two-point boundary value problems; its utility for these problems and its superior performance in mixed form strongly suggest that it could be used in optimal control problems.

Chapter 2 develops the weak principle for optimal control theory for problems in which the states and controls are continuous. In Chapter 3, the theory developed in Chapter 2 is tested on a single-stage rocket trajectory optimization problem. The weak principle is then extended in Chapter 4 to handle problems with discontinuities in the states and system equations. A realistic two-stage rocket problem is then solved in Chapter 5. The weak principle is further extended in Chapters 6 and 7 with the inclusion of control and state inequality constraints. A final demonstration of the trajectory optimization capabilities of the weak principle is demonstrated in Chapter 8.

Chapter 9 of this thesis gives a brief study of error estimates for the weak principle. The subject of generating initial guesses to solve the discretized equations is dealt with in Chapter 10. Chapter 11 is the main contribution of the work. It describes a general code for the solution of optimal control problems. Conclusions and future research are discussed in Chapter 12.

There are three appendices to the thesis. Appendix A discusses the solution of dynamics problems using Hamilton's weak principle. Appendix B deals with the solution of initial-value ordinary differential equations by using the weak principle.

Finally, Appendix C describes how simple beam problems can be cast in the form of optimal control problems and solved using the weak principle.

CHAPTER 2

WEAK PRINCIPLE FOR OPTIMAL CONTROL

It is desired to develop a solution strategy for optimal control problems based on finite elements in time. Finite elements have been used in the past to solve optimal control problems and two-point boundary-value problems in general (see [21] and [35-38]); however, these methods all require numerical quadrature. In addition, a different choice of shape functions must be made for each problem depending on the strong boundary conditions to be enforced. These two obstacles are overcome with the weak principle derived below.

A weak principle based on the variation of the performance index will be formulated [49, 50]. When deriving this formulation, two things must be remembered. First, the resulting formulation *must* satisfy the Euler-Lagrange equations and boundary conditions that have already been established in optimal control theory [1]. Second, in an attempt to make the solution scheme as general as possible all strong boundary conditions will be transformed into natural or weak boundary conditions.

The boundary conditions are all cast in the form of weak boundary conditions so that the shape functions used for the test functions can be chosen from a less restrictive class of functions. For example, if there is a strong boundary condition on one of the states at the initial time (*i.e.*, an initial condition) then the shape function chosen for the variation of that state must equal zero at the initial time [51]. It would be advantageous if one could choose the same shape functions for every optimal control problem. This is possible if there are no strong boundary conditions that must be satisfied by the shape functions.

The idea of transforming strong boundary conditions to natural boundary conditions [52] revolves around adjoining a constraint equation to the performance index with an unknown Lagrange multiplier. The variation of the performance index is then taken in a straightforward manner. Through appropriate integration by parts, one may show that the Euler-Lagrange equations are identical to those derived in classical textbooks [1] and that the boundary conditions are the same, only stated weakly instead of strongly.

As is shown below, the weak principle for optimal control reduces the necessary conditions for optimality to a set of nonlinear algebraic equations. These algebraic equations can be derived prior to specifying the problem to be solved. It is this feature in particular that makes the weak principle so powerful.

2.1. General Development

Consider a system defined by a set of n states x and a set of m controls u. Furthermore, let the system be governed by a set of state equations of the form $\dot{x} = f(x, u, t)$. In this chapter, the class of problems is restricted to those where x, u, and f are continuous. We may denote elements of the performance index, J_0, with an integrand $L(x, u, t)$ and discrete functions of the states and time $\phi[x(t), t]$ defined only at the initial and final times t_0 and t_f. In addition, any constraints imposed on the states and time at the initial and final times may be placed in sets of functions $\psi[x(t), t]$. These constraints may be adjoined to the performance index by discrete Lagrange multipliers ν defined at t_0 and t_f. Finally, we may adjoin the state equations to the performance index with a set of Lagrange multiplier functions $\lambda(t)$ which will be referred to as costates. For variable t_f, this yields a performance index of the form

$$J_0 = \int_{t_0}^{t_f} \left[L(x,u,t) + \lambda^T (f - \dot{x}) \right] \, dt + \Phi|_{t_0}^{t_f} \tag{2.1–1}$$

where $\Phi = \phi[x(t),t] + \nu^T \psi[x(t),t]$. The constraints to be adjoined to J_0 above are simply that the states be continuous at the initial and final times. Introducing

$$x|_{t_0} \triangleq \lim_{t \to t_0^+} x(t) \quad \text{and} \quad x|_{t_f} \triangleq \lim_{t \to t_f^-} x(t) \tag{2.1–2}$$

and

$$\hat{x}_0 = \hat{x}|_{t_0} \triangleq x(t_0) \quad \text{and} \quad \hat{x}_f = \hat{x}|_{t_f} \triangleq x(t_f) \tag{2.1–3}$$

continuity is weakly enforced by adjoining $\alpha^T(x - \hat{x})|_{t_0}^{t_f}$ to J_0 where α is a set of discrete unknown Lagrange multipliers defined only at t_0 and t_f. The new performance index is

$$J = \int_{t_0}^{t_f} \left[L(x,u,t) + \lambda^T (f - \dot{x}) \right] \, dt + \Phi|_{t_0}^{t_f} + \alpha^T (x - \hat{x})|_{t_0}^{t_f} \tag{2.1–4}$$

To derive the weak principle, it is necessary to determine dJ, the first variation of J. Denoting with $\delta x(t_f)$ and $\delta\lambda(t_f)$ the variations of x and λ at $t = t_f$ when holding t_f fixed, and letting $dx(t_f)$ and $d\lambda(t_f)$ be the variations of x and λ at $t = t_f$ when t_f is allowed to be free, then the variations at $t = t_f$ can be expressed by the linear equations (see [1] or [2])

$$\delta x(t_f) = dx(t_f) - \dot{x}|_{t_f} dt_f \quad \text{and} \quad \delta\lambda(t_f) = d\lambda(t_f) - \dot{\lambda}|_{t_f} dt_f \tag{2.1–5}$$

The first variation of J is

$$
\begin{aligned}
dJ = \int_{t_0}^{t_f} \Bigg\{ & \delta\lambda^T(f-\dot{x}) - \delta\dot{x}^T\lambda + \delta x^T\left[\left(\frac{\partial L}{\partial x}\right)^T + \left(\frac{\partial f}{\partial x}\right)^T\lambda\right] \\
& + \delta u^T\left[\left(\frac{\partial L}{\partial u}\right)^T + \left(\frac{\partial f}{\partial u}\right)^T\lambda\right]\Bigg\} dt \\
& + dt_f\left[L + \lambda^T(f-\dot{x}) + \frac{\partial\Phi}{\partial t}\right]\Bigg|_{t_f} + \delta\nu^T\psi\Big|_{t_0}^{t_f} \\
& + dx^T\left(\frac{\partial\Phi}{\partial x}\right)\Bigg|_{t_0}^{t_f} + \delta\alpha^T(x-\hat{x})\Big|_{t_0}^{t_f} + \alpha^T(dx-d\hat{x})\Big|_{t_0}^{t_f}
\end{aligned}
\tag{2.1–6}
$$

A necessary condition for an extremal of J is that the first variation be zero. Also, the admissible variations of the states must be continuous at the initial and final times and therefore $(dx - d\hat{x})|_{t_0}^{t_f} = 0$. For notational convenience we will define

$$
\hat{\lambda}\Big|_{t_0} = \frac{\partial\Phi}{\partial x}\Bigg|_{t_0} \quad \text{and} \quad \hat{\lambda}\Big|_{t_f} = \frac{\partial\Phi}{\partial x}\Bigg|_{t_f}
\tag{2.1–7}
$$

As an aside, to ensure that no necessary conditions have been altered, the $\delta\dot{x}$ term will be integrated by parts. This results in

$$
\begin{aligned}
\int_{t_0}^{t_f} \Bigg\{ & \delta\lambda^T(f-\dot{x}) + \delta x^T\left[\dot{\lambda} + \left(\frac{\partial L}{\partial x}\right)^T + \left(\frac{\partial f}{\partial x}\right)^T\lambda\right] \\
& + \delta u^T\left[\left(\frac{\partial L}{\partial u}\right)^T + \left(\frac{\partial f}{\partial u}\right)^T\lambda\right]\Bigg\} dt \\
& + dt_f\left[L + \lambda^T(f-\dot{x}) + \frac{\partial\Phi}{\partial t}\right]\Bigg|_{t_f} + \delta\nu^T\psi\Big|_{t_0}^{t_f} \\
& + dx^T\hat{\lambda}\Big|_{t_0}^{t_f} + \delta\alpha^T(x-\hat{x})\Big|_{t_0}^{t_f} - \delta x^T\lambda\Big|_{t_0}^{t_f} = 0
\end{aligned}
\tag{2.1–8}
$$

Using Eq. (2.1–5) and noting that $\delta x|_{t_0} = dx|_{t_0}$ since t_0 is fixed, then the above equation can be simplified to the following.

$$
\begin{aligned}
\int_{t_0}^{t_f} \Bigg\{ & \delta\lambda^T(f-\dot{x}) + \delta x^T\left[\dot{\lambda} + \left(\frac{\partial L}{\partial x}\right)^T + \left(\frac{\partial f}{\partial x}\right)^T \lambda\right] \\
& + \delta u^T \left[\left(\frac{\partial L}{\partial u}\right)^T + \left(\frac{\partial f}{\partial u}\right)^T \lambda\right]\Bigg\} dt \\
& + dt_f \left(L + \lambda^T f + \frac{\partial \Phi}{\partial t}\right)\bigg|_{t_f} + \delta\nu^T \psi\big|_{t_0}^{t_f} \\
& + dx^T(\hat{\lambda}-\lambda)\big|_{t_0}^{t_f} + \delta\alpha^T(x-\hat{x})\big|_{t_0}^{t_f} = 0
\end{aligned}
\tag{2.1-9}
$$

The Euler-Lagrange equations from the above will now be compared with the well-known optimal control equations presented in [1]. The coefficients of $\delta\lambda^T$, δx^T, and δu^T in the integrand, when set equal to zero, correspond to Eqs. (2.8.15 – 2.8.17) from [1]. There are also four trailing terms in Eq. (2.1–9) from which the boundary conditions of [1] can be determined. Namely, the requirement for the coefficient of dt_f to vanish is equivalent to Eq. (2.8.20). The requirement for the coefficient of $\delta\nu^T$ to vanish at $t = t_f$ yields Eq. (2.8.21). The requirement for the coefficient of dx^T to vanish at $t = t_f$ shows that the value of $\lambda|_{t_f}$ equals $\hat{\lambda}|_{t_f}$ as given in Eq. (2.1–7), which corresponds to Eq. (2.8.19). Finally, the requirement for the coefficient of $\delta\alpha^T$ to vanish at $t = t_0$ requires the value of $x|_{t_0}$ to equal $\hat{x}|_{t_0}$, in accordance with Eq. (2.8.18).

Three additional boundary conditions are present in the above formulation. One is the requirement for the coefficient of $\delta\alpha^T$ to vanish at $t = t_f$ which demands that the value of $x|_{t_f}$ equal $\hat{x}|_{t_f}$. The second is the requirement for the coefficient of dx^T to vanish at $t = t_0$ which demands that the value of $\lambda|_{t_0}$ equal $\hat{\lambda}|_{t_0}$ as given in Eq. (2.1–7). These two conditions enforce continuity of the states at the final time and continuity of the costates at the initial time. The third and last boundary condition is the requirement for the coefficient of $\delta\nu^T$ at $t = t_0$ to vanish which demands that $\psi[x(t_0), t_0] = 0$. Again, all boundary conditions were cast in the form of natural boundary conditions so that the shape functions chosen for δx and $\delta\lambda$ will not have to satisfy any particular boundary conditions.

Having satisfied the requirement that none of the fundamental equations are altered, the weak formulation is now derived from Eq. (2.1–9). By noting that α is a Lagrange multiplier whose only restriction is that $\delta\alpha$ be independent of dx, $\delta\nu$, and dt_f, and that α has the units of the costates, we then choose $\delta\alpha = d\lambda$. (Note that there is no unique choice for $\delta\alpha$, but this one will lead to a successful solution strategy.) Also, the dx and $d\lambda$ terms can be eliminated from Eq. (2.1–9) using Eq. (2.1–5) resulting in

$$\begin{aligned}
&\int_{t_0}^{t_f}\left\{\delta\lambda^T(f-\dot{x})+\delta x^T\left[\dot{\lambda}+\left(\frac{\partial L}{\partial x}\right)^T+\left(\frac{\partial f}{\partial x}\right)^T\lambda\right]\right.\\
&\quad\left.+\delta u^T\left[\left(\frac{\partial L}{\partial u}\right)^T+\left(\frac{\partial f}{\partial u}\right)^T\lambda\right]\right\}dt\\
&\quad+dt_f\left(L+\lambda^T f+\frac{\partial\Phi}{\partial t}\right)\Big|_{t_f}+\delta\nu^T\psi\Big|_{t_0}^{t_f}\\
&\quad+\delta x^T(\hat{\lambda}-\lambda)\Big|_{t_0}^{t_f}+\delta\lambda^T(x-\hat{x})\Big|_{t_0}^{t_f}\\
&\quad+(\hat{\lambda}-\lambda)^T\dot{x}\Big|_{t_f}dt_f-(\hat{x}-x)^T\dot{\lambda}\Big|_{t_f}dt_f=0
\end{aligned}\qquad (2.1\text{–}10)$$

Finally, the $\dot{x}$ and $\dot{\lambda}$ terms are integrated by parts so that no time derivatives of x or λ appear in the weak formulation for optimal control. This allows for the simplest possible shape functions to be chosen which in turn eliminates the need for numerical quadrature. The resulting equation is

$$\begin{aligned}
&\int_{t_0}^{t_f}\left\{\delta\dot{\lambda}^T x-\delta\dot{x}^T\lambda+\delta x^T\left[\left(\frac{\partial L}{\partial x}\right)^T+\left(\frac{\partial f}{\partial x}\right)^T\lambda\right]\right.\\
&\quad\left.+\delta\lambda^T f+\delta u^T\left[\left(\frac{\partial L}{\partial u}\right)^T+\left(\frac{\partial f}{\partial u}\right)^T\lambda\right]\right\}dt\\
&\quad+dt_f\left(L+\lambda^T f+\frac{\partial\Phi}{\partial t}\right)\Big|_{t_f}+\delta\nu^T\psi\Big|_{t_0}^{t_f}\\
&\quad+\delta x^T\hat{\lambda}\Big|_{t_0}^{t_f}-\delta\lambda^T\hat{x}\Big|_{t_0}^{t_f}+(\hat{\lambda}-\lambda)^T\dot{x}\Big|_{t_f}dt_f-(\hat{x}-x)^T\dot{\lambda}\Big|_{t_f}dt_f=0
\end{aligned}\qquad (2.1\text{–}11)$$

After noting that the last two terms are equal to zero in accordance with the natural boundary conditions (see the coefficients of dx and $\delta\alpha$ in Eq. 2.1–9), one can discard those terms without changing the necessary conditions. Also, we note that for most problems, the initial conditions are given for all n states and thus, in accordance with Eq. (2.1–7), all the initial costates are unknown. Therefore, instead of treating elements of ν at $t = t_0$ as unknowns and replacing $\hat{\lambda}|_{t_0}$ with these unknowns, we will instead treat $\hat{\lambda}|_{t_0}$ as unknowns and eliminate the $\delta\nu|_{t_0}$ equations from the weak principle. We hasten to point out that the elements of $\hat{x}|_{t_0}$ are the initial conditions. The final form of the weak principle is then given as

$$\begin{aligned}
\int_{t_0}^{t_f} \Bigg\{ & \delta\dot{\lambda}^T x - \delta\dot{x}^T\lambda + \delta x^T \left[\left(\frac{\partial L}{\partial x}\right)^T + \left(\frac{\partial f}{\partial x}\right)^T \lambda \right] \\
& + \delta\lambda^T f + \delta u^T \left[\left(\frac{\partial L}{\partial u}\right)^T + \left(\frac{\partial f}{\partial u}\right)^T \lambda \right] \Bigg\} dt \\
& + dt_f \left(L + \lambda^T f + \frac{\partial \Phi}{\partial t} \right) \Bigg|_{t_f} + \delta\nu^T \psi\big|_{t_f} + \delta x^T \hat{\lambda}\big|_{t_0}^{t_f} - \delta\lambda^T \hat{x}\big|_{t_0}^{t_f} = 0
\end{aligned} \qquad (2.1\text{–}12)$$

This is the governing equation for the weak Hamiltonian method for optimal control problems of the form specified. It will serve as the basis for the finite element discretization described below for constructing of candidate solutions (*i.e.*, solutions which satisfy all the necessary conditions). It should be noted that normally one will encounter various types of discontinuities in the states and state equations, as well as inequality constraints on the controls and states in problems that deal with optimal control. These aspects will be treated in Chapters 4, 6 and 7 respectively.

2.2. Finite Element Discretization

Let us break the time interval from t_0 to t_f into N elements. The nodal values of these elements are t_i for $i = 1, \ldots, N+1$ where $t_0 = t_1$ and $t_f = t_{N+1}$. Now, define a nondimensional elemental time τ as

$$\tau = \frac{t - t_i}{t_{i+1} - t_i} = \frac{t - t_i}{\Delta t_i} \tag{2.2–1}$$

Note in Eq. (2.1–12) that time derivatives of δx and $\delta\lambda$ are present. However, no time derivatives of x and λ exist. Therefore, it is possible to implement linear shape functions for δx and $\delta\lambda$ and constant shape functions for x and λ within each element. The linear shape functions for the virtual states and costates are

$$\begin{aligned} \delta x &= \delta x_i(1-\tau) + \delta x_{i+1}\tau \\ \delta\lambda &= \delta\lambda_i(1-\tau) + \delta\lambda_{i+1}\tau \end{aligned} \tag{2.2–2}$$

For the states and costates, piecewise constant shape functions are taken to be

$$x = \begin{cases} \hat{x}_i & \text{if } \tau = 0 \\ \bar{x}_i & \text{if } 0 < \tau < 1 \\ \hat{x}_{i+1} & \text{if } \tau = 1 \end{cases} \tag{2.2–3}$$

and

$$\lambda = \begin{cases} \hat{\lambda}_i & \text{if } \tau = 0 \\ \bar{\lambda}_i & \text{if } 0 < \tau < 1 \\ \hat{\lambda}_{i+1} & \text{if } \tau = 1 \end{cases} \tag{2.2–4}$$

It is important to understand that the equalities $\hat{x}_1 = x(t_0), \hat{\lambda}_1 = \lambda(t_0), \hat{x}_{N+1} = x(t_f)$, and $\hat{\lambda}_{N+1} = \lambda(t_f)$ are enforced as natural (weak) boundary conditions. In other words, the hatted values of x and λ at the beginning and end of the time interval *are* the discrete values of x and λ that are needed in the weak formulation of Eq. (2.1–12). This is clarified below in Figure 2.1 where the time line is broken into elements and the nodes are labelled appropriately.

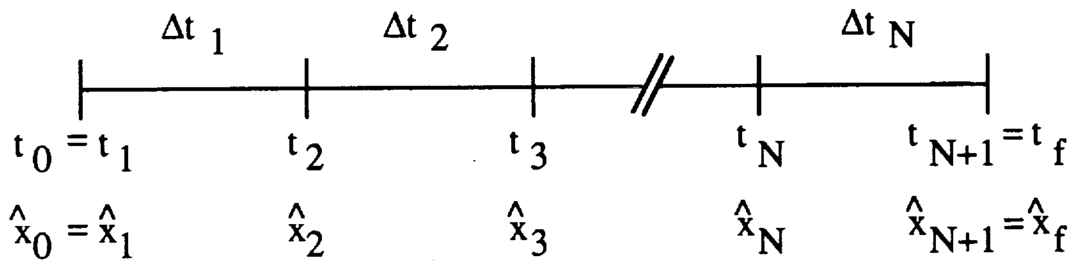

Fig. 2.1: Time line broken into elements and labelling of nodes

Finally, note that the time derivatives of u and δu do not appear in the formulation. Thus, constant shape functions are chosen for *both* u and the variation of u. These shape functions are

$$\begin{aligned} u &= \bar{u}_i \\ \delta u &= \delta \bar{u}_i \end{aligned} \tag{2.2–5}$$

Plugging in the shape functions described for x, λ, and u, substituting $t = t_i + \tau \Delta t_i$, and carrying out the element quadrature over τ from 0 to 1 results in

$$
\begin{aligned}
\sum_{i=1}^{N} \Bigg\langle & \delta x_i^T \left[\bar{\lambda}_i + \frac{\Delta t_i}{2} \left(\frac{\partial \bar{f}}{\partial \bar{x}}\right)_i^T \bar{\lambda}_i + \frac{\Delta t_i}{2} \left(\frac{\partial \bar{L}}{\partial \bar{x}}\right)_i^T \right] - \delta \lambda_i^T \left(\bar{x}_i - \frac{\Delta t_i}{2} \bar{f}_i \right) \\
& - \delta x_{i+1}^T \left[\bar{\lambda}_i - \frac{\Delta t_i}{2} \left(\frac{\partial \bar{f}}{\partial \bar{x}}\right)_i^T \bar{\lambda}_i - \frac{\Delta t_i}{2} \left(\frac{\partial \bar{L}}{\partial \bar{x}}\right)_i^T \right] + \delta \lambda_{i+1}^T \left(\bar{x}_i + \frac{\Delta t_i}{2} \bar{f}_i \right) \\
& + \delta \bar{u}_i^T \left\{ \Delta t_i \left[\left(\frac{\partial \bar{L}}{\partial \bar{u}}\right)_i^T + \left(\frac{\partial \bar{f}}{\partial \bar{u}}\right)_i^T \bar{\lambda}_i \right] \right\} \Bigg\rangle \qquad (2.2\text{–}6) \\
& + dt_f \left(\hat{L} + \hat{\lambda}_f^T \hat{f} + \frac{\partial \phi}{\partial t} + \nu^T \frac{\partial \psi}{\partial t} \right)\Bigg|_{t_f} + \delta \nu^T \psi \big|_{t_f} \\
& - \delta x_1^T \hat{\lambda}_0 + \delta \lambda_1^T \hat{x}_0 + \delta x_{N+1}^T \hat{\lambda}_f - \delta \lambda_{N+1}^T \hat{x}_f = 0
\end{aligned}
$$

where $\bar{f}_i = f(x = \bar{x}_i, u = \bar{u}_i)$, $\bar{L}_i = L(x = \bar{x}_i, u = \bar{u}_i)$. This is the general algebraic form of the weak formulation for all optimal control problems of the form specified. Note that if the time t does not appear explicitly in the problem formulation then all integration is *exact* and can be done by inspection. If t does appear explicitly, then t may be approximated by a constant value over each element (as are x, λ, and u) and the integration may still be done by inspection. Note that the elements must be assembled over the entire time interval for this two-point boundary-value problem. Only the nodal values (the hatted quantities) of the states and costates at the initial and final times appear in the algebraic equations. These remaining hatted quantities are the discrete values of the states and costates which appear in Eq. (2.1–12).

Eq. (2.2–6) is a set of nonlinear algebraic equations whose size depends on the number of elements N. In fact, if $\psi|_{t_f}$ is a $q \times 1$ column matrix and there are N elements, then there are $2n(N+1)$ (for δx_i and $\delta \lambda_i$) + mN (for δu_i) + q (for $\delta \nu$) + 1 (for dt_f) equations and $2n(N+2)$ (for $\bar{x}_i$, $\hat{x}_0$, $\hat{x}_f$, $\bar{\lambda}_i$, $\hat{\lambda}_0$, and $\hat{\lambda}_f$) + mN (for $\bar{u}_i$) + q (for ν) + 1 (for t_f) unknowns. Therefore, $2n$ of the $4n$ endpoint values for the states and costates ($\hat{x}_0, \hat{\lambda}_0, \hat{x}_f$, and $\hat{\lambda}_f$) must be specified. In general, $\hat{x}_0$ (the initial conditions) is known in accordance with physical constraints. Also, $\hat{\lambda}_f$

can be specified in terms of other unknowns with the use of Eq. (2.1–7). Now we have the same number of equations as unknowns. These equations may be used for any optimal control problem of the form specified. One simply needs to substitute the appropriate f, L, ϕ, ψ and boundary conditions into Eq. (2.1–12) for a given problem. Note that there is never a need to eliminate any of the equations (except the dt_f equation for fixed-time problems) as is usual in standard finite element practice where strong boundary conditions are enforced by the virtual quantities.

Normally, Eq. (2.2–6) can be solved by expressing the Jacobian explicitly and using a Newton-Raphson solution procedure. For the example problems which follow, the iteration procedure will converge quickly for a small number of elements with a trivial initial guess. Then, the answers obtained for a small number of elements can be used to generate initial guesses for a higher number of elements. Thus, a large number of elements can be solved with a very efficient run-time on the computer.

Although the nodal values $\hat{x}_i$ and $\hat{\lambda}_i$ for $2 < i < N$ (on the interior of the time interval) do not appear in the algebraic equations, their values can be easily recovered after the solution is found. This is most easily seen by looking at the following ordinary differential equation multiplied by a test (or weighting) function $\delta\lambda$ and integrated over some time interval where the integral makes sense. A constraint to transform the strong boundary conditions to weak ones has been adjoined via a discrete multiplier which has been identified as $\delta\lambda$.

$$\int_{t_1}^{t_2} \delta\lambda[f(x,t) - \dot{x}]\,dt + \delta\lambda(x - \hat{x})\Big|_{t_1}^{t_2} = 0 \qquad (2.2\text{–}7)$$

After an integration by parts, using the linear shape function for $\delta\lambda$ defined in Eq. (2.2–2), using the piecewise constant shape function for x defined in Eq. (2.2–3), and substituting τ for t as given in Eq. (2.2–1), then the following equation is obtained from Eq. (2.2–7).

$$\delta\lambda_1(-\bar{x}_1 + \frac{\Delta t}{2}\bar{f} + \hat{x}_1) + \delta\lambda_2(\bar{x}_1 + \frac{\Delta t}{2}\bar{f} - \hat{x}_2) = 0 \tag{2.2–8}$$

With arbitrary $\delta\lambda_1$ and $\delta\lambda_2$, the coefficients must vanish, forming two equations of the form

$$\begin{aligned} -\bar{x}_1 + \frac{\Delta t}{2}\bar{f} + \hat{x}_1 &= 0 \\ \bar{x}_1 + \frac{\Delta t}{2}\bar{f} - \hat{x}_2 &= 0 \end{aligned} \tag{2.2–9}$$

Now, by subtracting the second equation from the first, it is seen that

$$\bar{x}_1 = \frac{\hat{x}_1 + \hat{x}_2}{2} \tag{2.2–10}$$

or, in words, that the interior value (the bar value) is simply the average of the surrounding nodal (or hatted) quantities. Once the solution is found, all the midpoint values and the end nodal values are known, and thus all other nodal values can be recovered by repeatedly using Eq. (2.2–10). In fact, the nodal values are really the best approximation to the solution and thus are the only ones plotted for the state and costates in this thesis.

Although the shape function for the control u only defines a constant value within the element, values of u at additional points are available. For instance, once the nodal values for the states and costates are found, then one may use the optimality condition ($\partial H/\partial u = 0$) to solve for u at a nodal point. In fact, this is how one finds a value for $\hat{u}_f$ to use in the δt_f equation. Also, if the states and costates are approximated by some continuous curve fit through the nodal values obtained from the solution, then the control could be approximated at any instant in time by using the optimality condition.

2.3. Example: A Fixed-Final-Time Problem

As the first optimal control problem, the transfer of a particle to a rectilinear path will be examined (see Fig. 2.2). This is an example taken from article 2.4 of [1]. Let $x_{(1)}$ and $x_{(2)}$ denote the position of the particle at a given time and $x_{(3)}$ and $x_{(4)}$ denote the particle's velocity at a given time. (A subscripted number in parentheses refers to the state index to avoid confusion with the element index.) The thrust angle u is the control and the particle has mass m and a constant acceleration a.

The state equations are defined as

$$\dot{x} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} x + \begin{Bmatrix} 0 \\ 0 \\ a\cos u \\ a\sin u \end{Bmatrix} = f \tag{2.3–1}$$

The final time T is fixed and the problem is to maximize the final horizontal component of velocity. Thus,

$$\begin{aligned} L &= 0 \\ \phi &= \lfloor 0 \quad 0 \quad 1 \quad 0 \rfloor \hat{x}_f \end{aligned} \tag{2.3–2}$$

The optimality condition $\partial H/\partial u = 0$ yields an expression for the control of $\tan u = \lambda_{(4)}/\lambda_{(3)}$. There are also two terminal constraints on the states. These are that the particle arrive with a fixed final height (h) and that the final vertical component of velocity be zero. The final horizontal component of position is free. These constraints can be stated analytically as

$$\psi = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \hat{x}_f - \begin{Bmatrix} h \\ 0 \end{Bmatrix} \tag{2.3–3}$$

The initial conditions are $x(0) = \hat{x}_0 = \lfloor 0 \quad 0 \quad 0 \quad 0 \rfloor^T$. Finally, the unknown $\hat{\lambda}_f$'s are eliminated by writing it in terms of other unknowns. In accordance with Eq. (2.1–7)

$$\hat{\lambda}_f = \begin{Bmatrix} 0 \\ \nu_1 \\ 1 \\ \nu_2 \end{Bmatrix} \tag{2.3-4}$$

The f, L, ϕ, ψ, and boundary conditions to substitute directly into Eq. (2.2–6) have now been defined.

These equations are solved by choosing $\Delta t_i = \Delta t = t_f/N$ for all i, expressing the Jacobian explicitly and using a Newton-Raphson algorithm. For $N = 2$, suitable initial guesses for the nonlinear iterative procedure can be found by simply choosing element values that are not too different from the boundary conditions. The results from solving the $N = 2$ equations are then used to obtain the initial guesses for arbitrary N by simple interpolation. In all results obtained to date for this problem, no additional steps are necessary to obtain results as accurate as desired.

Representative numerical results for all four states versus dimensionless time t/T are presented in Figs. 2.3 – 2.6. For this example, $h = 100$, $T = 20$, and $4h/aT^2 = 0.8897018$. (This last number is chosen to yield a value of 75° for the initial control angle of the exact solution available in [1]) The results for 2, 4, and 8 elements are plotted against the exact solution. It can easily be seen that $N = 8$ gives acceptable results for all the states. Amazingly enough, even the very crude 2 element mesh yields a decent approximation to the answer.

In Fig. 2.7, the control angle u versus dimensionless time t/T is presented. Once again, the results are seen to be excellent for $N = 8$. Note the extra data points which are available for the control when we make use of the optimality condition at the nodal and midpoint values of the elements. This is of great value since it is the control variable which is of the most interest.

Three of the four costates are constants for all time and this method yields two of these exactly. The third costate is very close to the exact answer. The fourth costate corresponding to the vertical component of velocity $\lambda_{(4)}$ is shown in Fig. 2.8. The results compare nicely with the exact results.

A plot of the relative error of the performance index $J = \hat{x}_{f,(3)}$ and the endpoint multiplier ν_1 versus the number of elements is shown in Fig. 2.9. It is seen to be nearly a straight line on a log-log scale. The slope of the line is about -2 which indicates that the error varies inversely with the square of N, similar to *a-posteriori* error bounds as formulated in usual finite element applications [53]. Notice in Fig. 2.9 that there is a bend in the endpoint multiplier curve. It is not unusual for mixed formulations to have an error curve that is not monotonically decreasing. It should be noted that developments of mathematical proofs for convergence and expressions for error bounds are not state-of-the-art for mixed methods. However, some initial error estimate studies for the weak principle are given in Chapter 9.

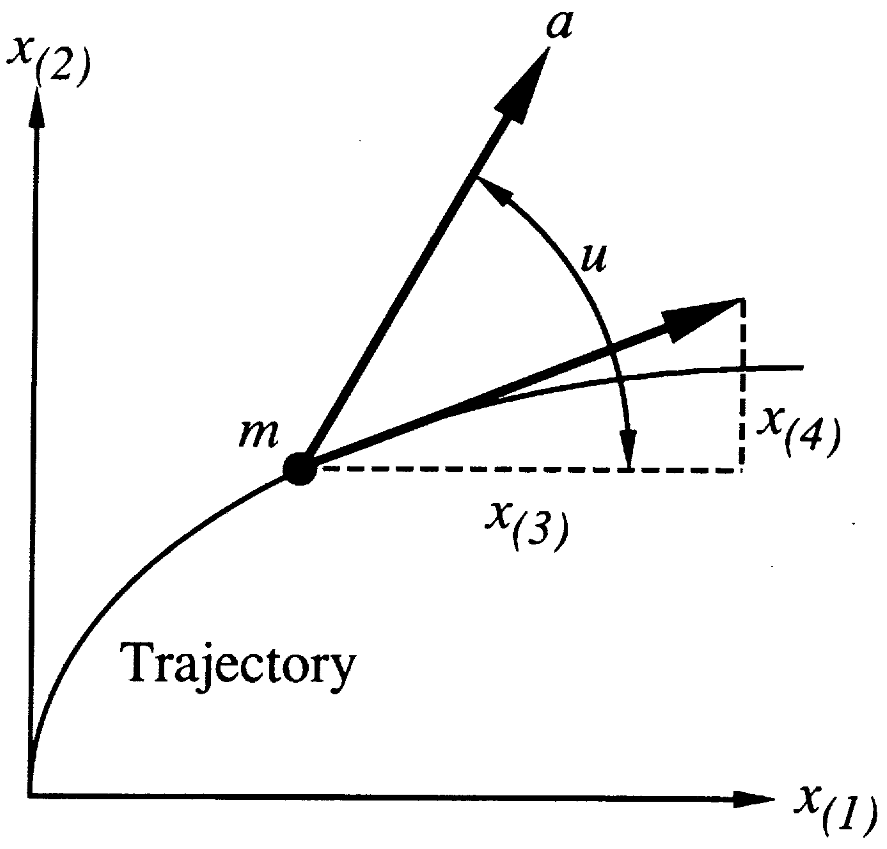

Fig. 2.2: Nomenclature for Examples

Transfer of a particle of mass m to a rectilinear path using a constant acceleration a

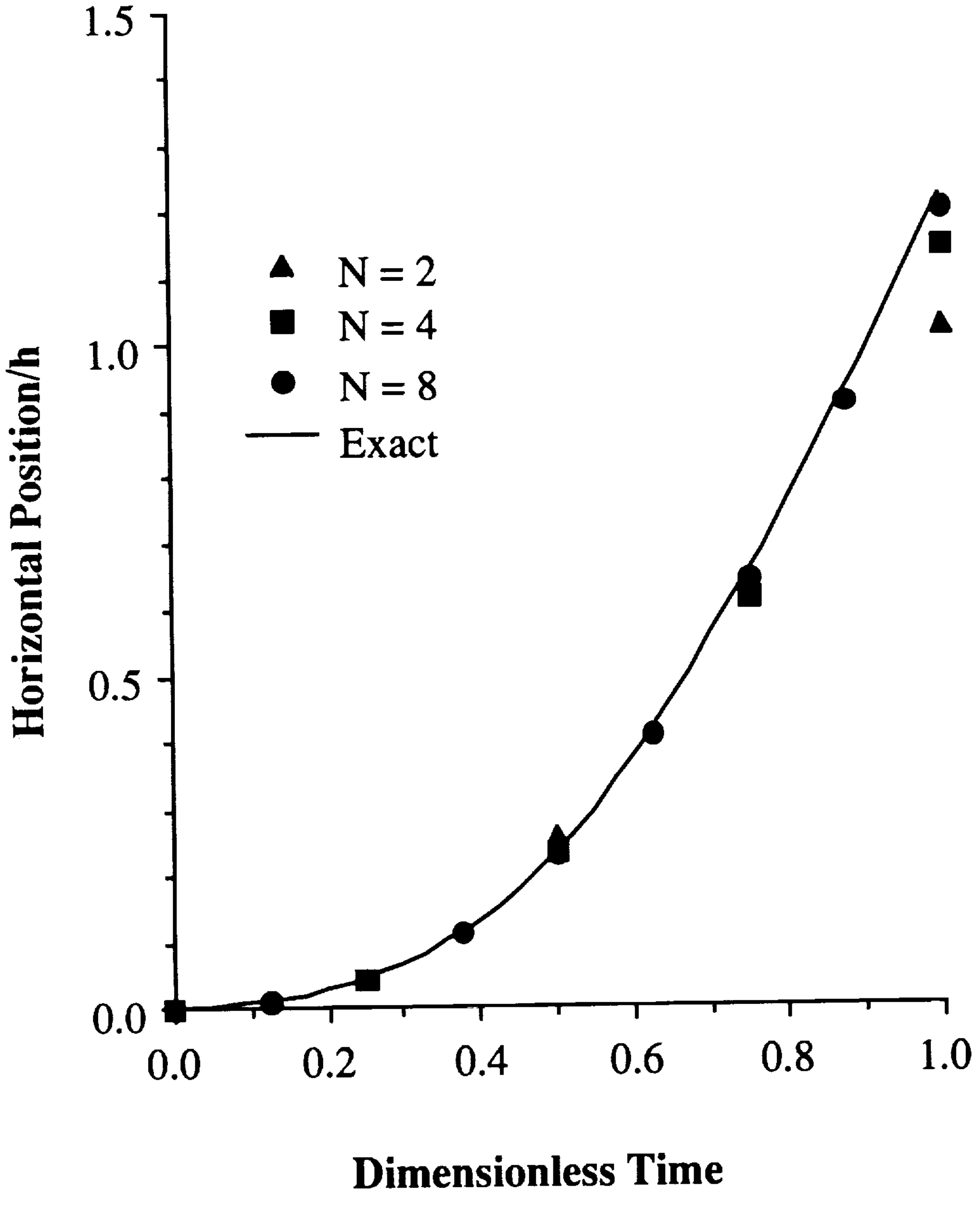

Fig. 2.3: Dimensionless horizontal position $x_{(1)}/h$ vs. t/T
Note that the final horizontal component of position is not specified

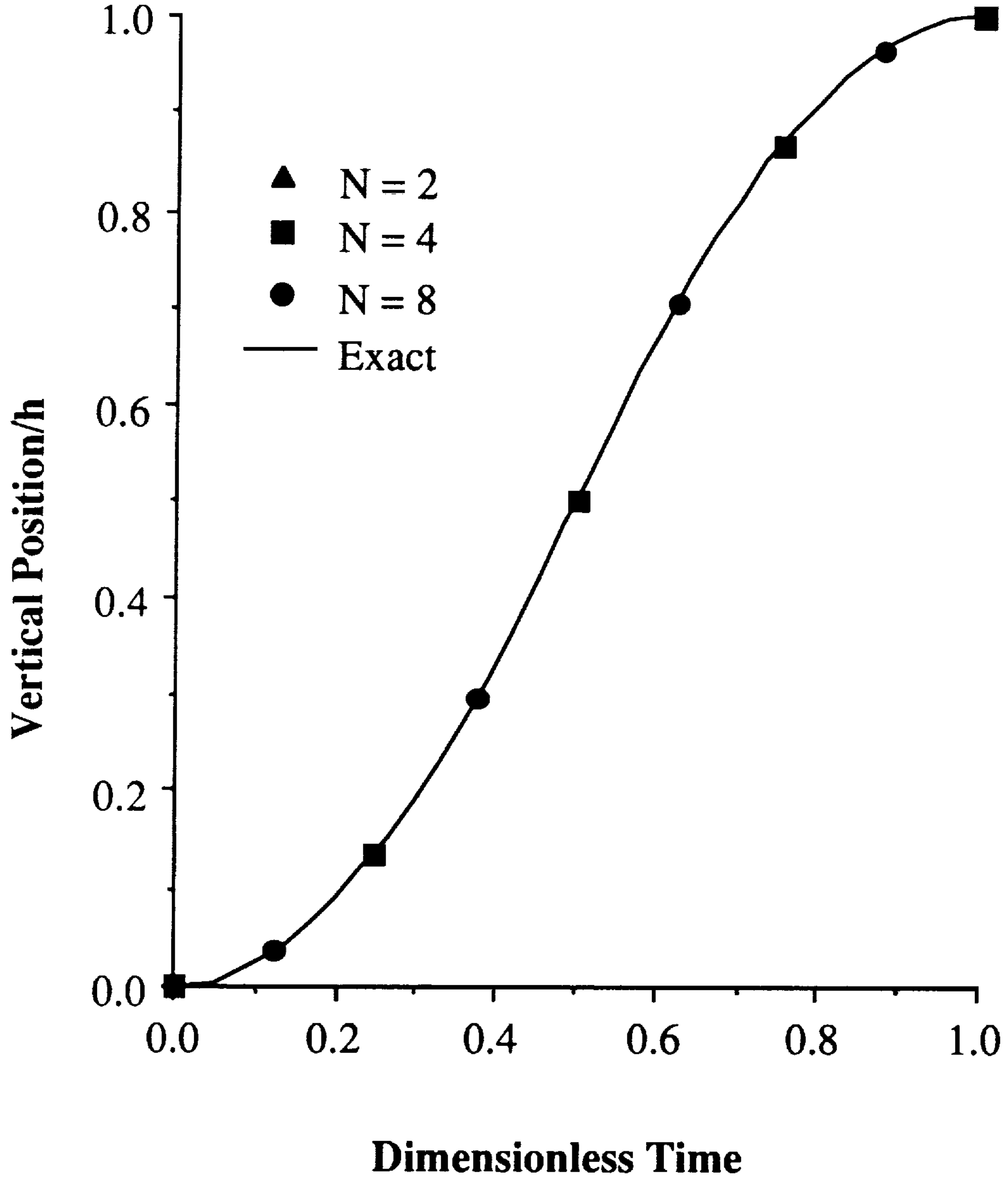

Fig. 2.4: Dimensionless vertical position $x_{(2)}/h$ vs. t/T

The final height is constrained to be h at the final time

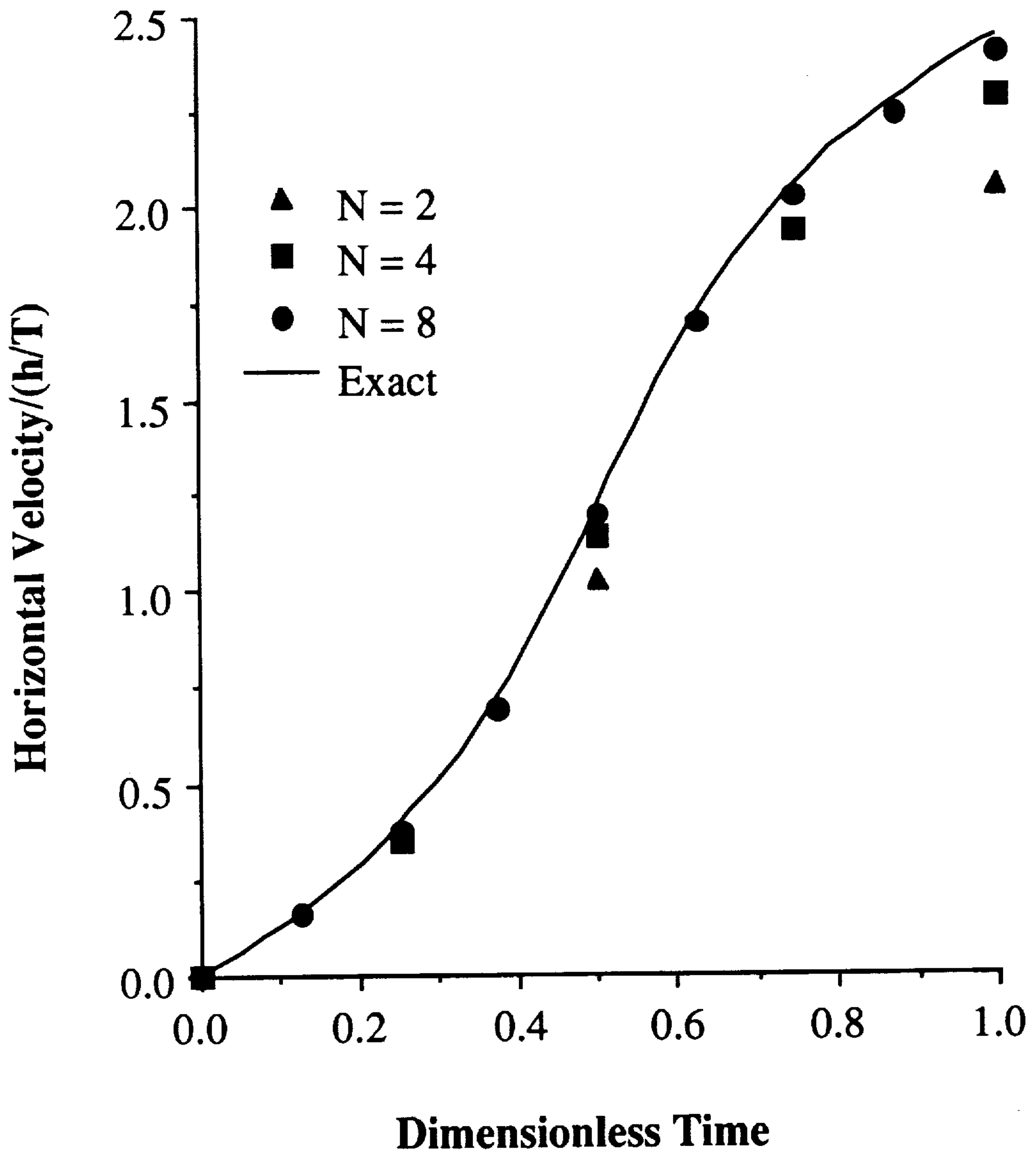

Fig. 2.5: Dimensionless horizontal velocity $x_{(3)}T/h$ vs. t/T
Note that the performance index $J = x_{(3)}(T)$

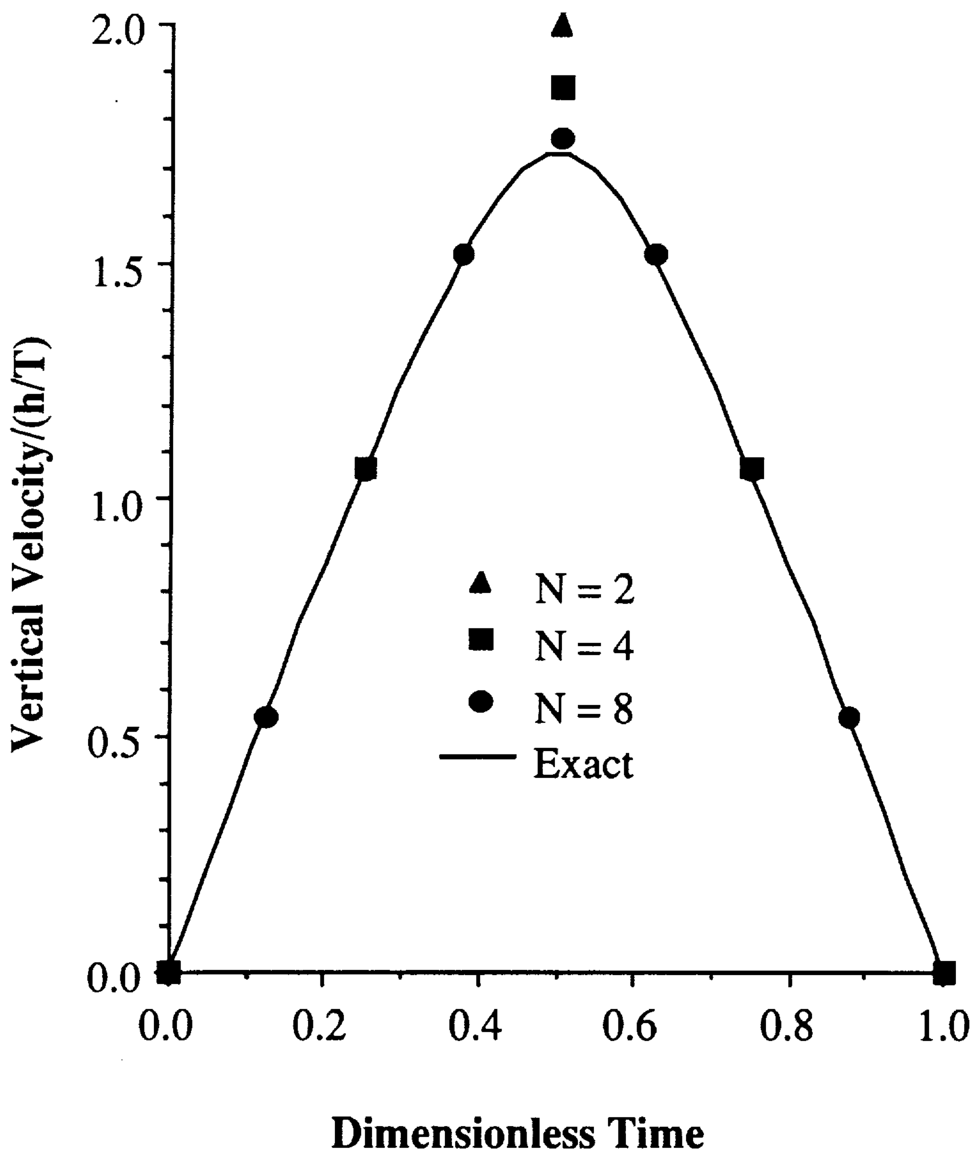

Fig. 2.6: Dimensionless vertical velocity $x_{(4)}T/h$ vs. t/T
The final vertical component of velocity is constrained to be zero at the final time

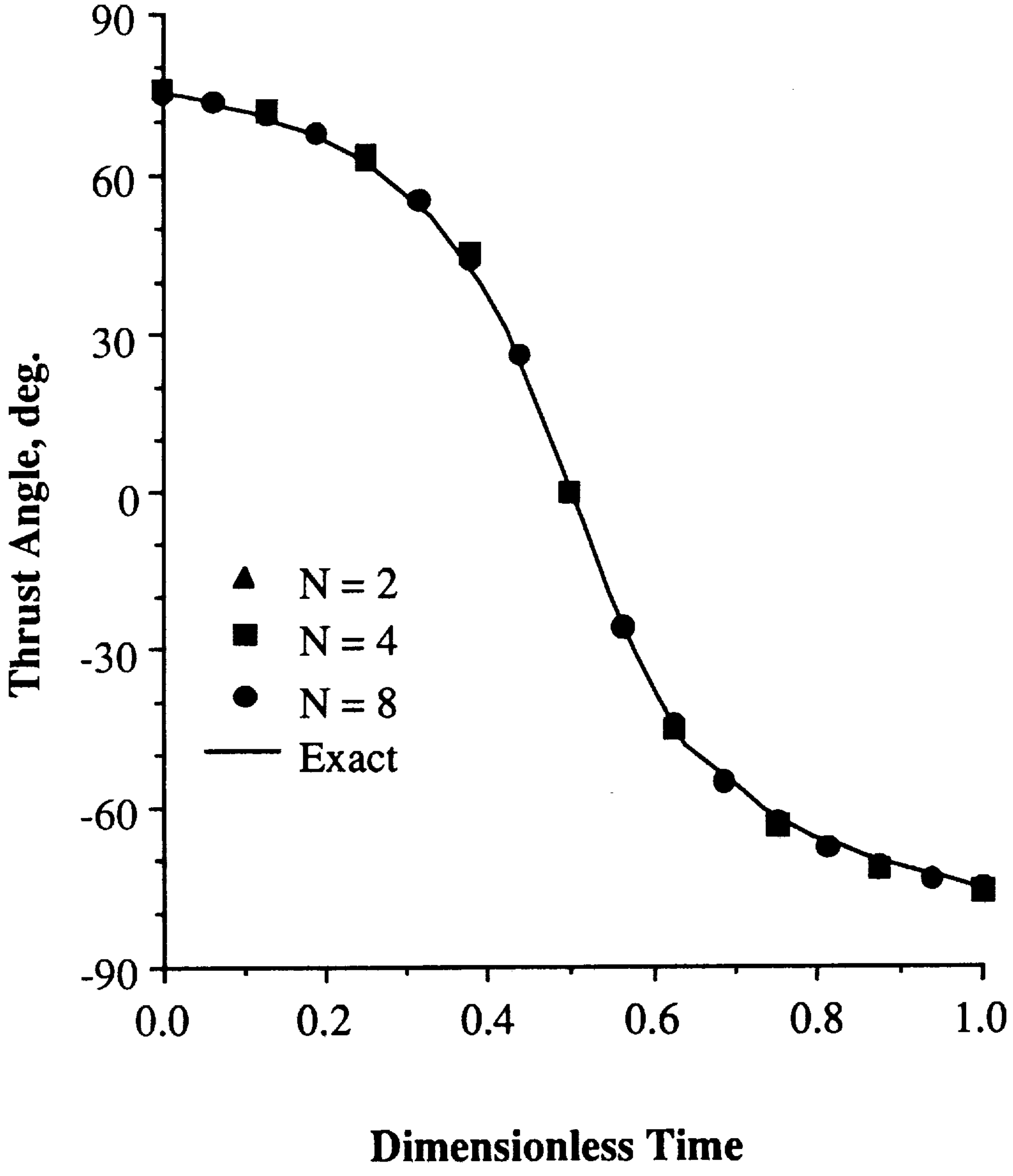

Fig. 2.7: Control angle u vs. t/T

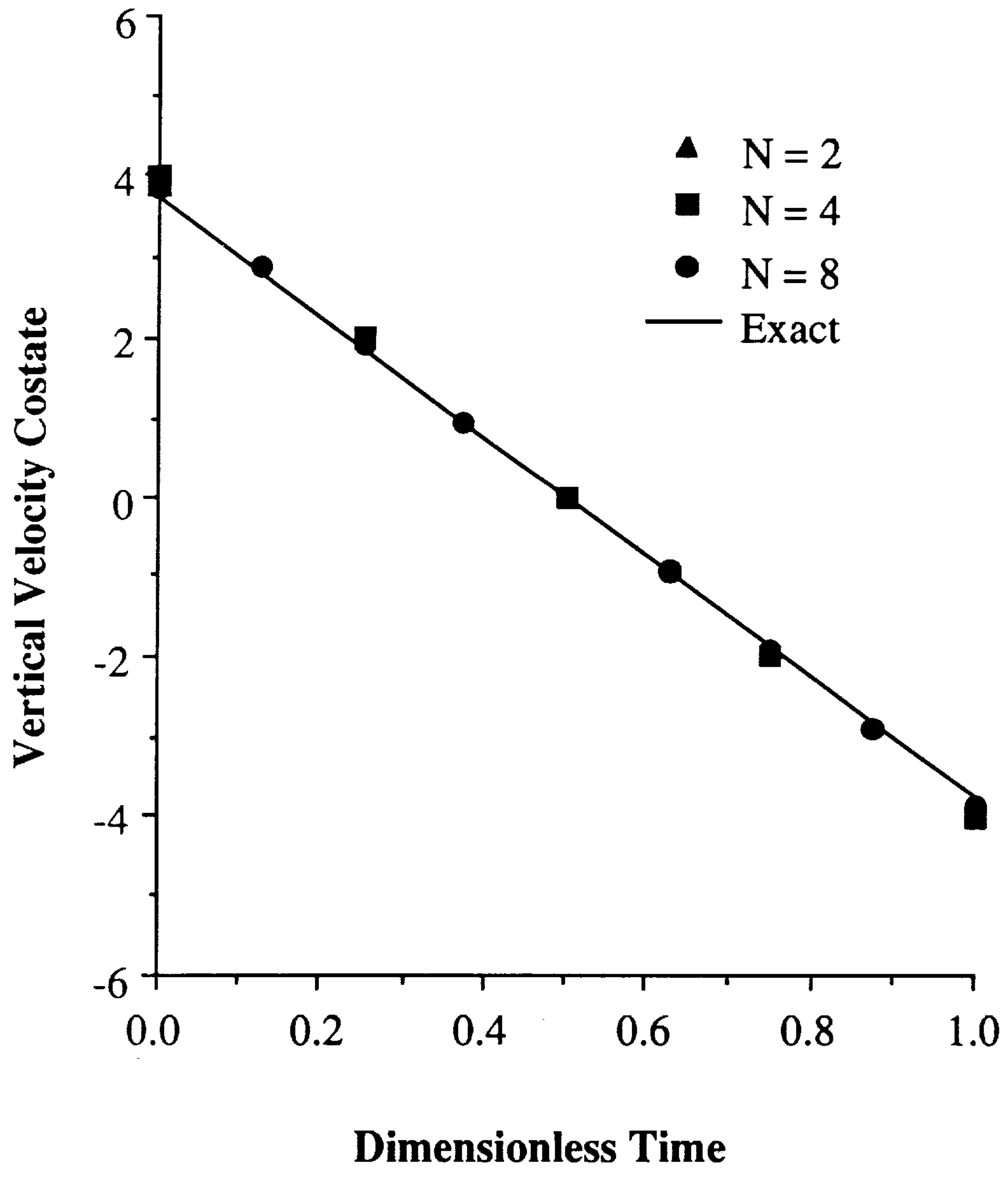

Fig. 2.8: Vertical velocity costate $\lambda_{(4)}$ vs. t/T

The results for this costate are the least accurate of all the costates

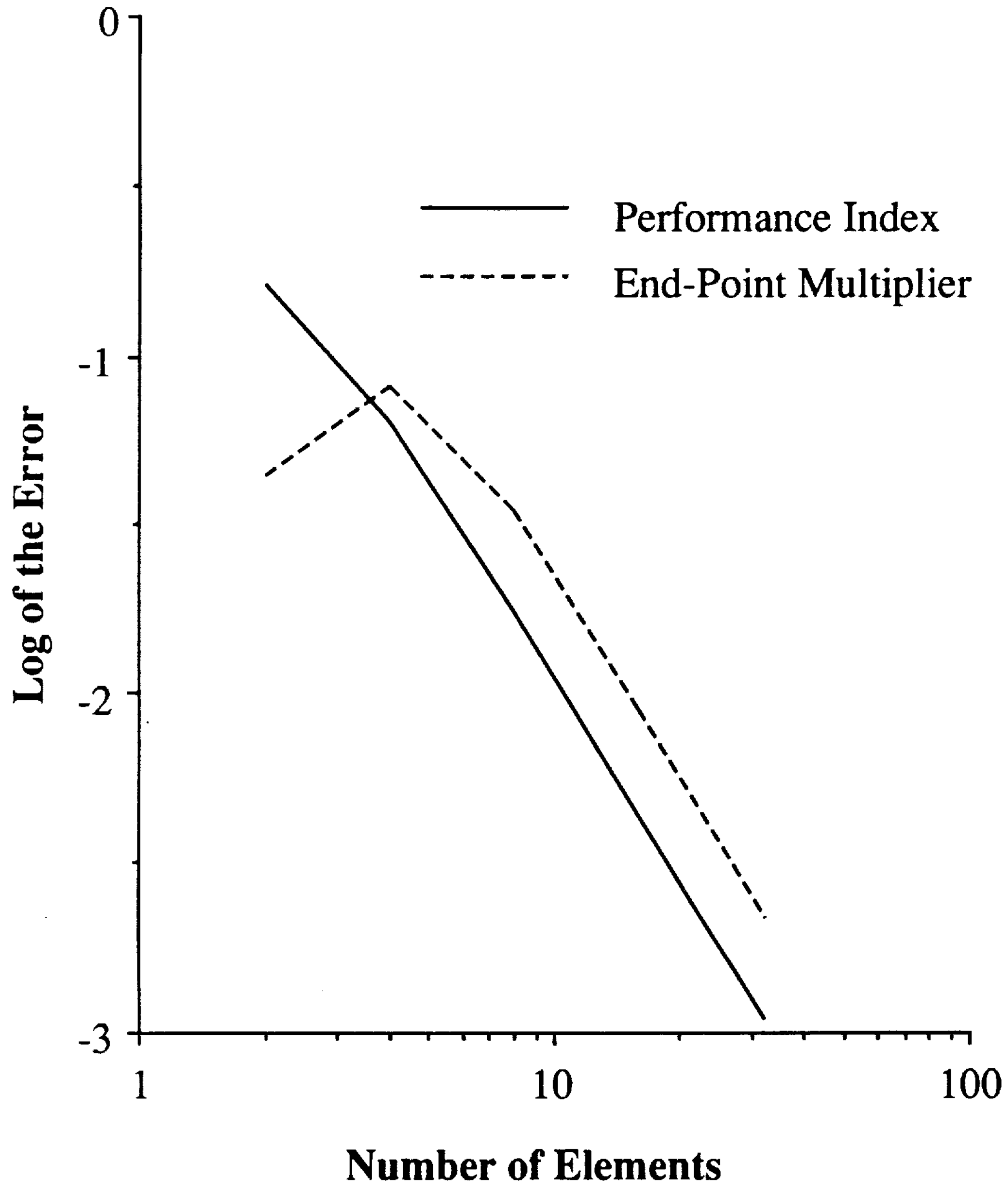

Fig. 2.9: Relative error of the performance index and the endpoint multiplier ν_1 vs. N

2.4. Example: A Free-Final-Time Problem

The second optimal control problem is similar to the previous example except that now the final time is free and we would like to obtain a given horizontal component of velocity (U) in the minimum time (see [1], problem 9, article 2.7). The algorithm from the preceding example is readily modified to fit this problem by noting the following changes. The performance index is now the final time T; so $\phi = 0$ and $L = 1$. Also, there is an additional endpoint constraint on the states; namely that $x_{(3)} = U$. With these changes

$$\hat{\lambda}_f = \begin{Bmatrix} 0 \\ \nu_1 \\ \nu_2 \\ \nu_3 \end{Bmatrix} \tag{2.4-1}$$

and

$$\psi = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \hat{x}_f - \begin{Bmatrix} h \\ U \\ 0 \end{Bmatrix} \tag{2.4-2}$$

Along with these changes to the equations, one additional equation is added from the coefficient of dt_f. The new system of equations is solved in the same manner described previously. Again, initial guesses not too far from the boundary conditions are satisfactory for $N = 2$, and these answers are used to obtain initial guesses for arbitrary N.

Representative numerical results for all four states versus dimensionless time t/T are presented in Figs. 2.10 – 2.13 for a case with $ah/U^2 = 0.75$. The results for 2, 4, and 8 elements are plotted against the exact solution available in [1]. It can easily be seen that $N = 8$ gives acceptable results for all the states.

The control angle u versus dimensionless time t/T is presented in Fig. 2.14. Once again, the results are seen to be excellent for $N = 8$. In Table 2.1, the initial control $u(t_0)$ (which can be easily shown to be related to ν_1, ν_2, and ν_3) and the normalized final time $\frac{aT}{U}$ are shown to converge quite rapidly as N is increased. Note, however, that the $N = 2$ and $N = 4$ approximations for $u(t_0)$ are neither upper nor lower bounds. This is a common characteristic of mixed formulations.

In Fig. 2.15, the vertical velocity costate is shown. The agreement of the finite element solution and the exact solution is excellent, even for 2 elements. The plot of the relative error of the performance index T versus the number of elements N is shown in Fig. 2.16. Again, the slope of the line is about -2 indicating that the error varies inversely with the square of N.

Table 2.1: Convergence of $u(t_0)$ and $\frac{aT}{U}$ versus N

N	$u(t_0)$ (degrees)	$\frac{aT}{U}$
2	72.586	1.8819
4	74.736	1.8531
8	75.027	1.8413
16	74.969	1.8380
32	74.950	1.8372
exact	74.944	1.8369

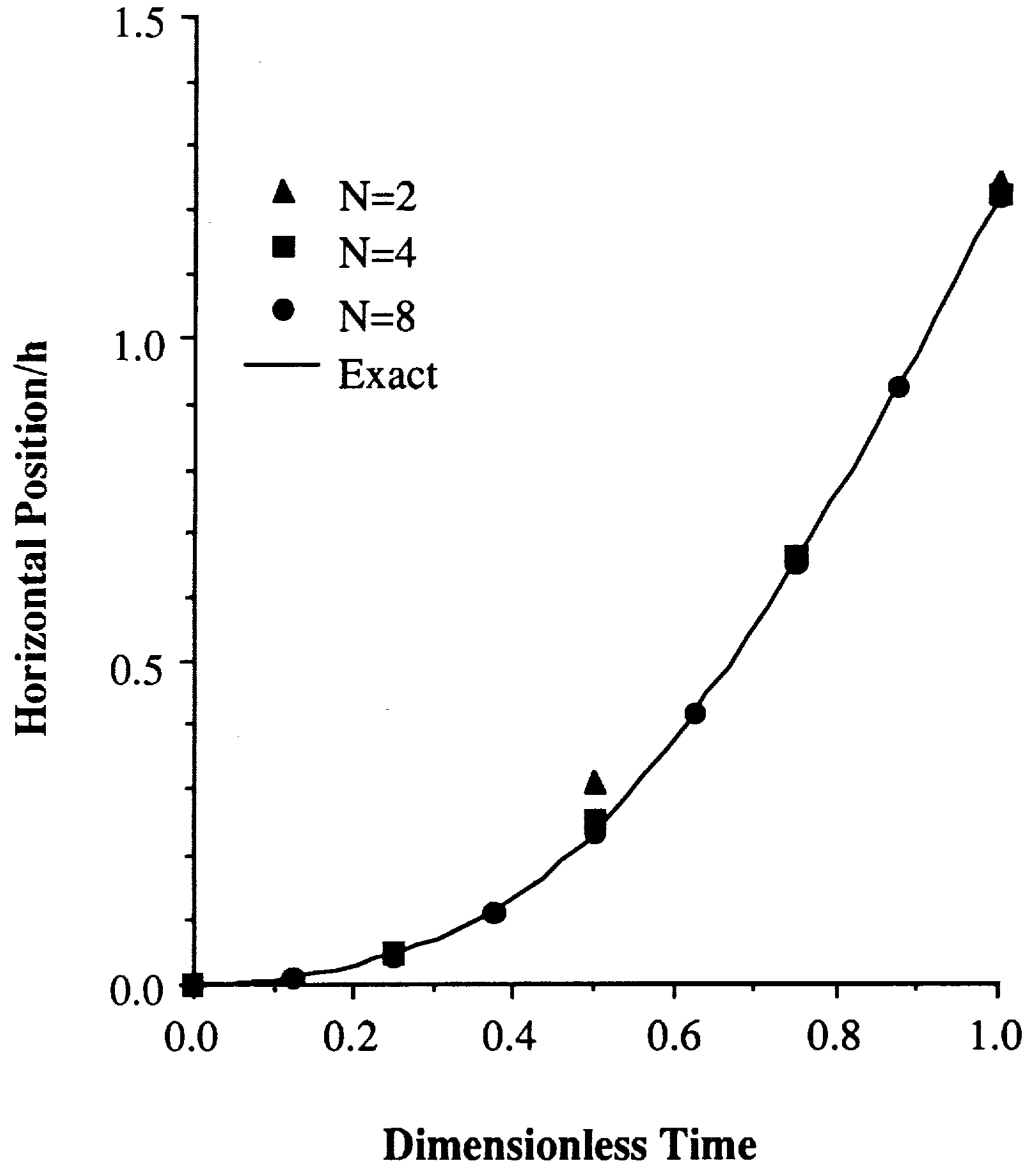

Fig. 2.10: Dimensionless horizontal position $x_{(1)}/h$ vs. t/T
Note that the final horizontal component of position is not specified

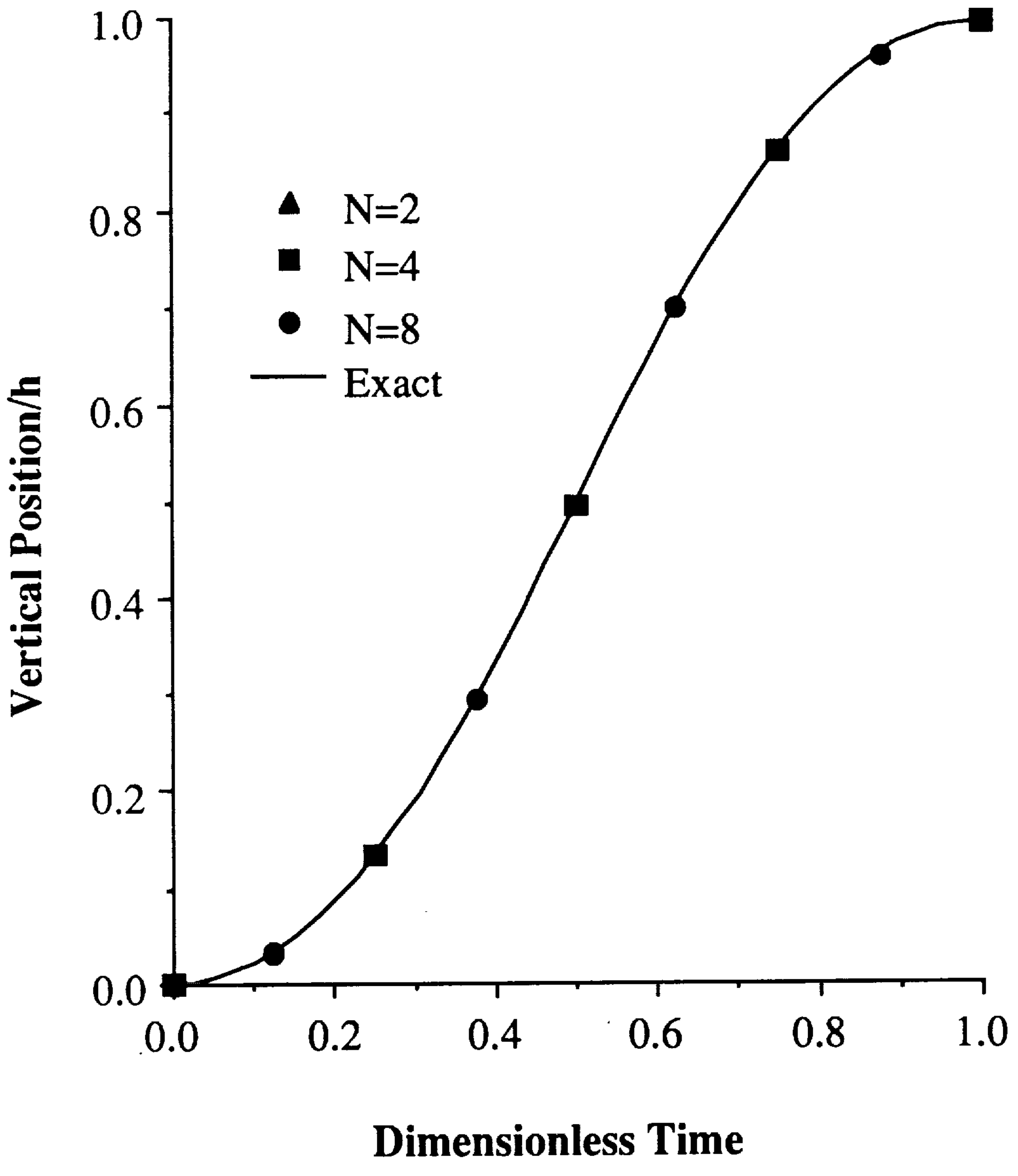

Fig. 2.11: Dimensionless vertical position $x_{(2)}/h$ vs. t/T
The final height is constrained to be h at the final time

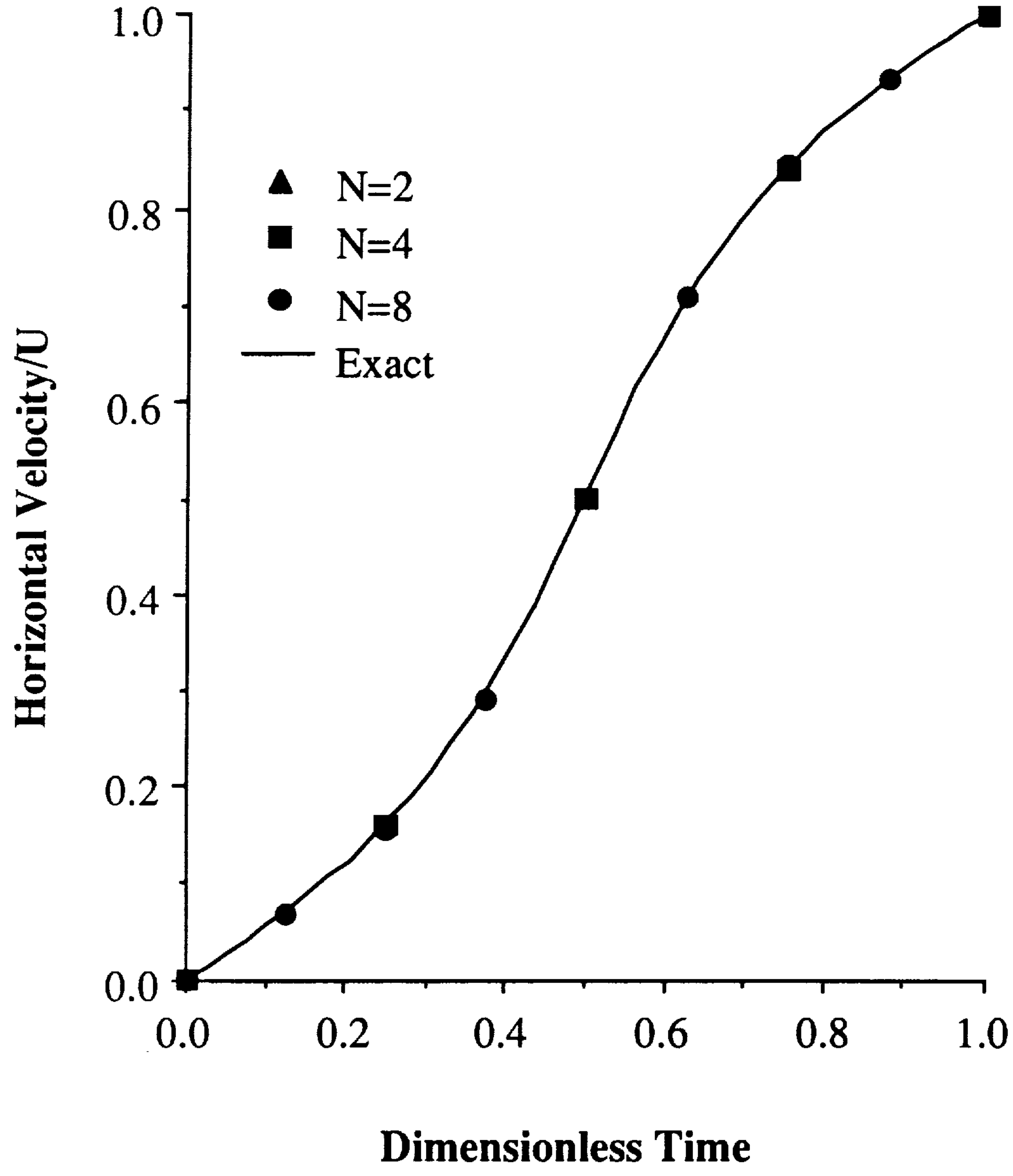

Fig. 2.12: Dimensionless horizontal velocity $x_{(3)}/U$ vs. t/T

The final horizontal component of velocity is constrained to be U

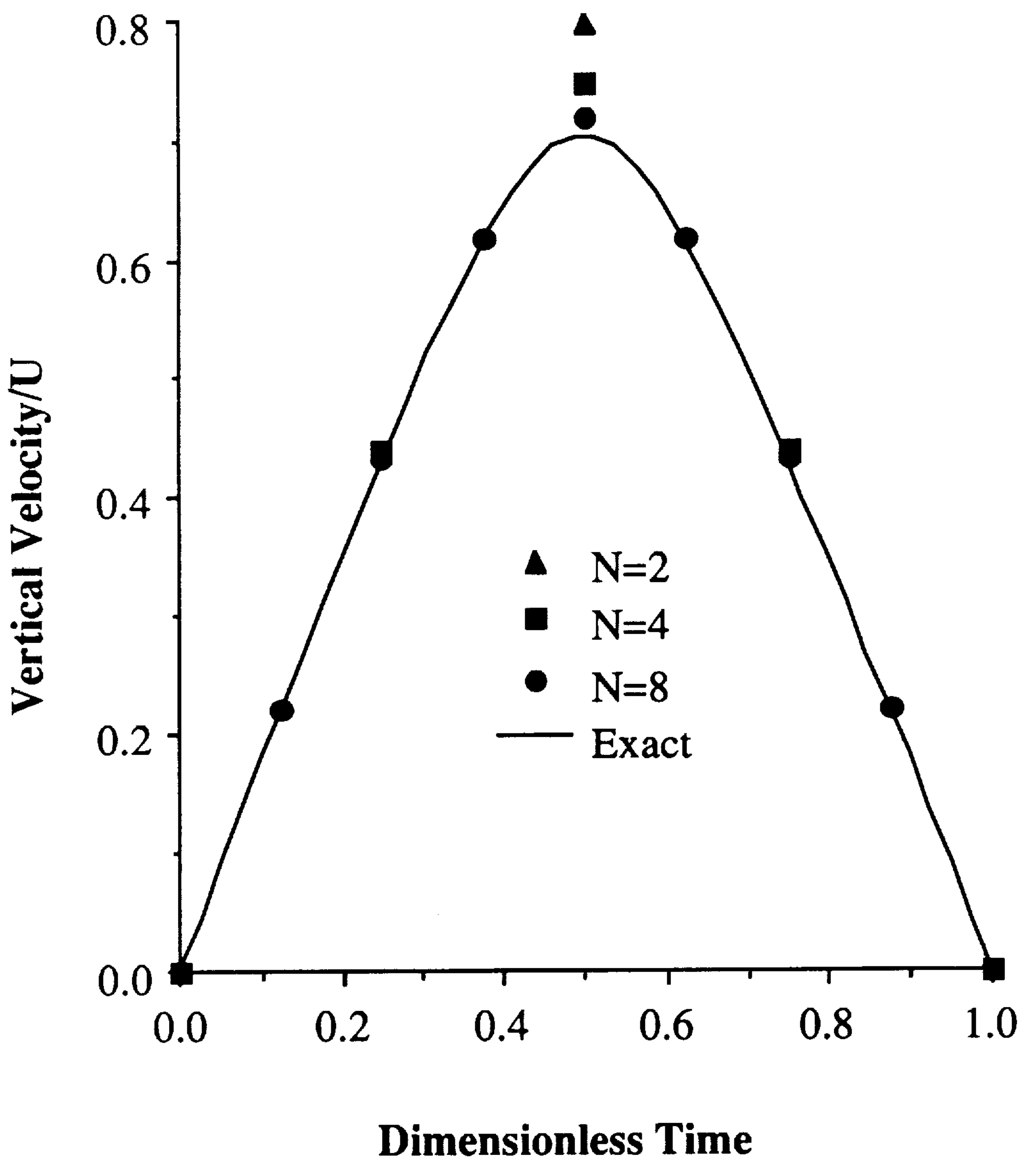

Fig. 2.13: Dimensionless vertical velocity $x_{(4)}/U$ vs. t/T

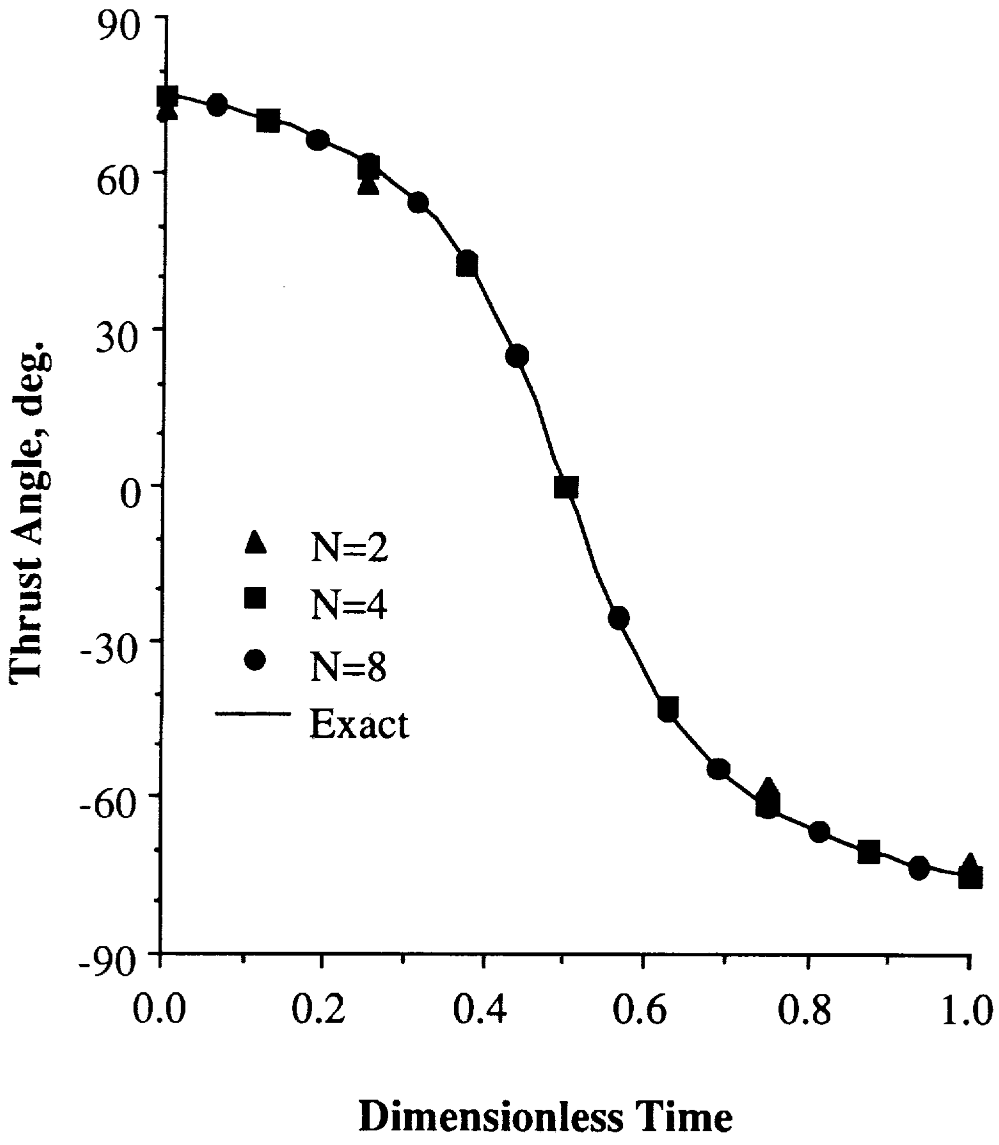

Fig. 2.14: Control angle u vs. t/T

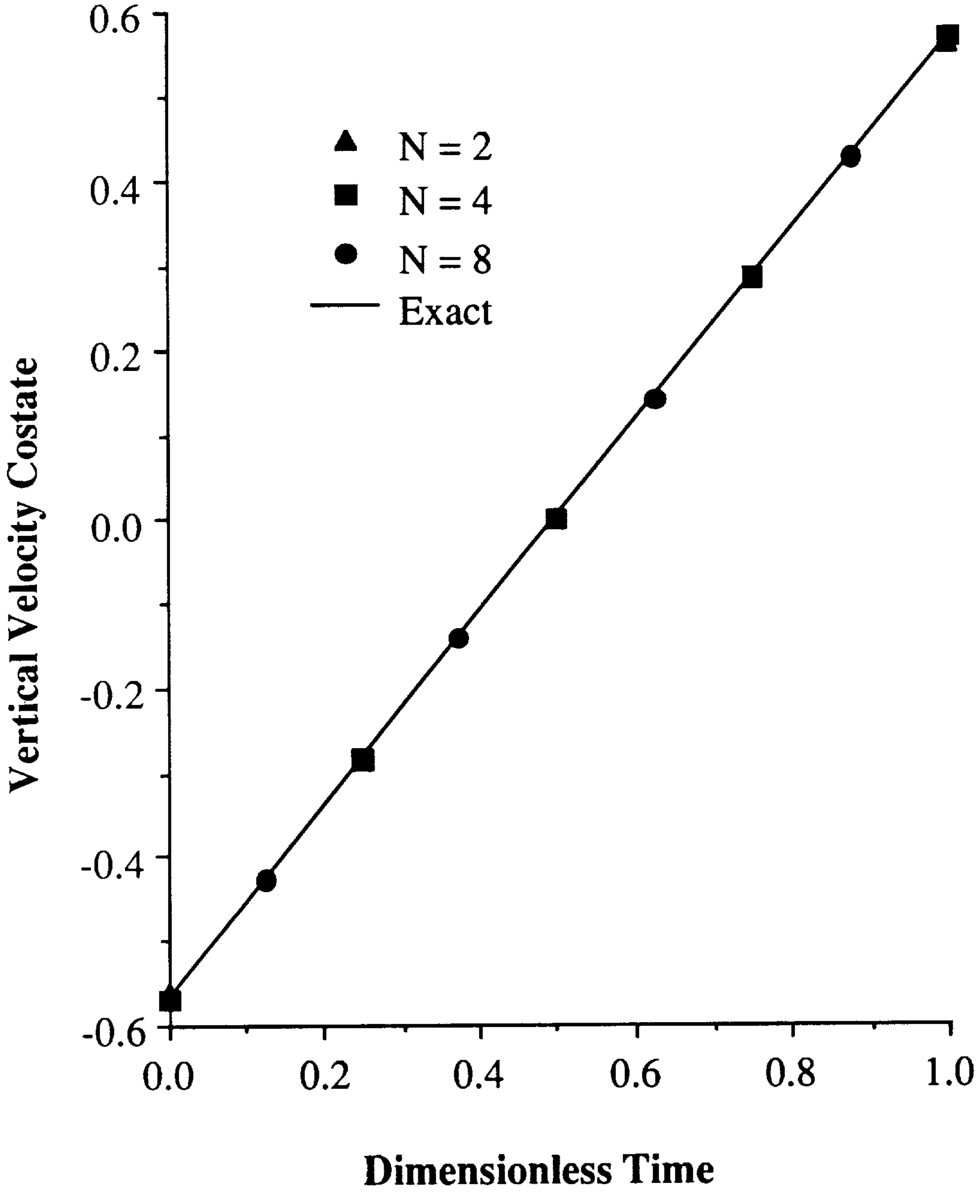

Fig. 2.15: Vertical velocity costate $\lambda_{(4)}$ vs. t/T

The results for this costate are the least accurate of all the costates

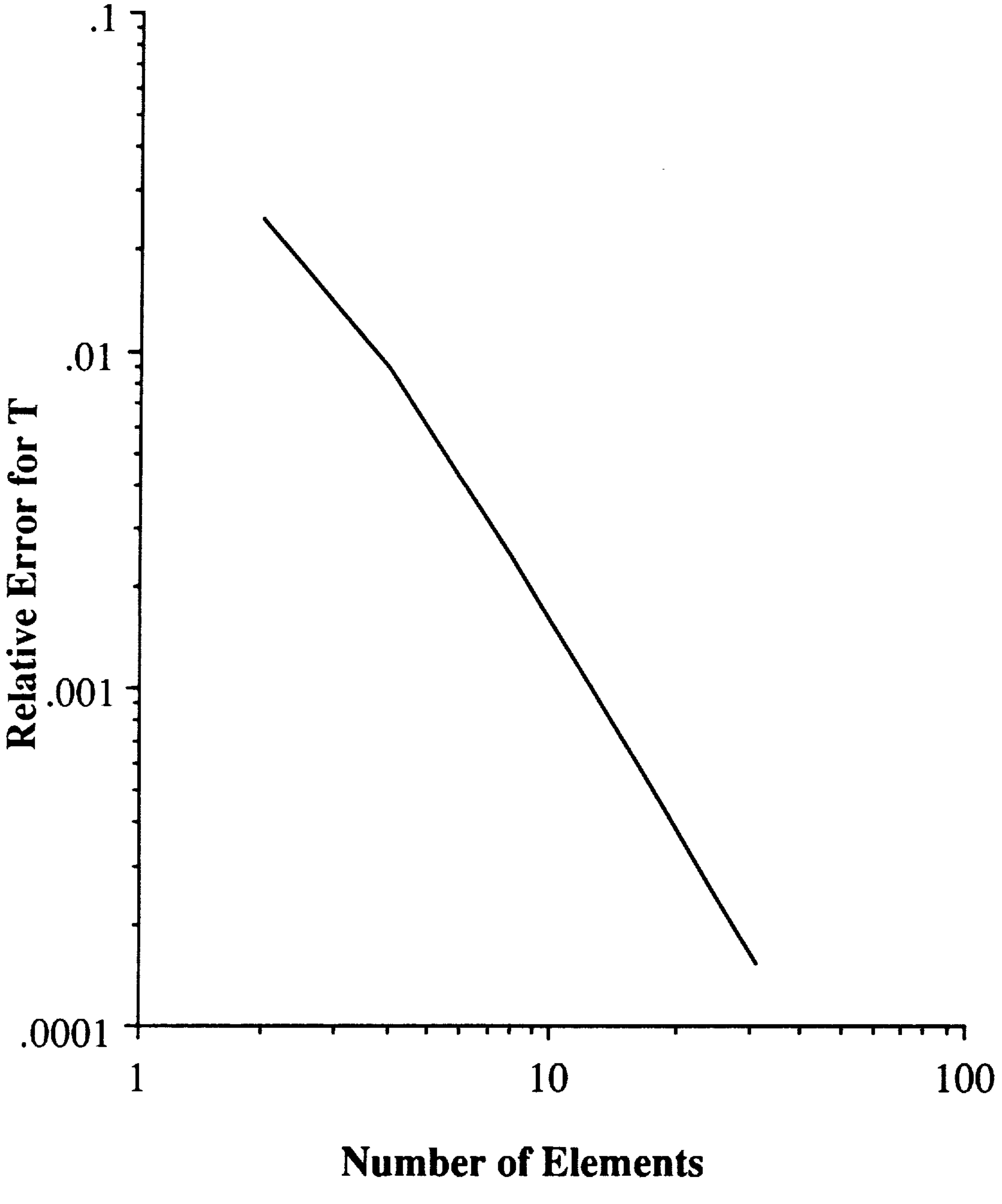

Fig. 2.16: Relative error of the performance index T vs. N

CHAPTER 3

SATURN ONE-STAGE ROCKET MODEL

In this chapter, a model is presented which is suitable for evaluating the potential usefulness of the weak principle for practical problems in optimal control [50]. The weak principle will be applied to a one-stage model of the Saturn IB rocket. In this case an analytical solution is not available, and the accuracy of the solution will be compared with a solution obtained using a multiple shooting method. While there would normally exist several inequality constraints in this problem, the constraints are not included in this application. However, Chapter 5 contains a two-stage Saturn model involving discontinuities in the system equations and in the states, and Chapter 8 has a model of an advanced launch vehicle which includes staging and inequality constraints.

3.1. A Four-State Model

Consider a vehicle confined to vertical plane dynamics and flying over a spherical, non-rotating earth as depicted in Fig. 3.1. This results in the following state model for the states m (mass), h (height), E (energy per unit mass), and γ (flight-path angle):

$$
\begin{aligned}
\dot{m} &= -\frac{T_{vac}}{9.81 I_{sp}} \\
\dot{h} &= V \sin\gamma \\
\dot{E} &= \left(\frac{T-D}{m}\right) V \\
\dot{\gamma} &= \left(\frac{T + qSC_{L\alpha}}{mV}\right)\alpha + \left(\frac{V}{r} - \frac{\mu}{r^2 V}\right)\cos\gamma
\end{aligned}
\tag{3.1–1}
$$

where T is the thrust, T_{vac} is the thrust in a vacuum (a constant), D is the drag, and V is the velocity. Here α, the angle of attack, has been adopted as a control variable. The atmospheric model is given by the following equations:

$$
\begin{aligned}
p &= p_0(1 - 0.00002255h)^{5.256} && \text{for } h \leq 11000\text{m} \\
p &= p_{11}\exp\left(-\frac{h-11000}{6350}\right) && \text{for } h \geq 11000\text{m} \\
\rho &= \rho_0 \exp\left(-\frac{h}{6700}\right) \\
a &= a_0\sqrt{1 - 0.00002255h} && \text{for } h \leq 11000\text{m} \\
a &= 295.03\text{ms}^{-2} && \text{for } h > 11000\text{m}
\end{aligned}
\tag{3.1–2}
$$

The aerodynamic and propulsion models are given by the following equations:

$$
\begin{aligned}
T &= T_{vac} - A_e p \\
r &= R_e + h \\
V &= \sqrt{2\left(E + \frac{\mu}{r}\right)} \\
q &= \frac{\rho V^2}{2} \\
D &= qS\left[C_{D0}(M) + \alpha^2 C_{N\alpha}(M)\right] \\
C_{L\alpha}(M) &= C_{N\alpha}(M) - C_{D0}(M) \\
M &= \frac{V}{a}
\end{aligned}
\tag{3.1–3}
$$

The vehicle parameters chosen for this model are based on a Saturn IB launch vehicle SA-217 [54] and are

$$\begin{aligned} S &= 33.468\,\mathrm{m}^2; \quad I_{sp} = 263.4\,\mathrm{s} \\ T_{vac} &= 8155800\,\mathrm{N}; \quad A_e = 8.47\,\mathrm{m}^2 \end{aligned} \tag{3.1–4}$$

The aerodynamic coefficient data $C_{N\alpha}$ and C_{D0} are presented as functions of the Mach number M in Tables 3.1 and 3.2 and shown graphically in Figs. 3.2 and 3.3. The physical constants used in the above model are the earth's gravitational constant $\mu = 3.9906 \times 10^{14}$ m^{3}s^{-2}, the earth's mean radius $R_e = 6.378 \times 10^6$ m, the sea-level atmospheric pressure $p_0 = 101320$ Nm^{-2}, the atmospheric pressure at 11 km $p_{11} = 22637$ Nm^{-2}, the sea-level density of air $\rho_0 = 1.225$ kg m^{-3}, and the sea-level speed of sound in air $a_0 = 340.3$ ms^{-1}.

The initial conditions specified are $m(0) = 5.2 \times 10^5$ kg, $h(0) = 1800$ m, $E(0) = -6.25 \times 10^7$ m^{2}s^{-2}, and $\gamma(0) = 75°$. The final energy is specified as $E(t_f) = -4.25 \times 10^7$ m^{2}s^{-2}.

The performance index is

$$J = \phi|_{t_f} = m|_{t_f} \tag{3.1–5}$$

and the final time t_f is open. Note that $L = 0$.

From Eqs. (2.1–7) and (3.1–5) it is seen that $\hat{\lambda}_f$ is given by $\lfloor 1 \quad 0 \quad \nu \quad 0 \rfloor^T$.

The quantities necessary for direct substitution into the algebraic equations of the weak principle Eq. (2.2–6) have now been defined.

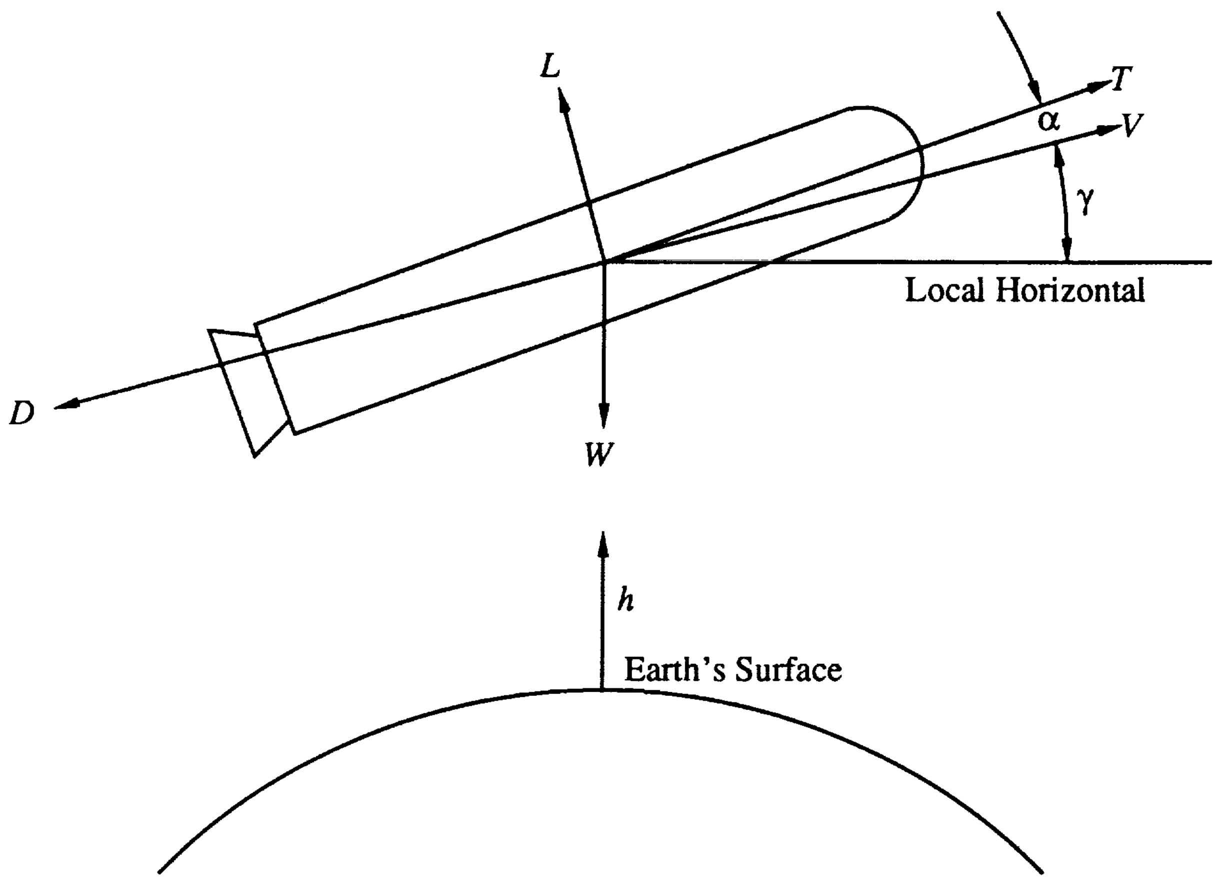

Fig. 3.1: Vertical plane dynamic model

Table 3.1: Aerodynamic coefficient $C_{N\alpha}$ versus Mach number

M	$C_{N\alpha}$
0.00	6.20
0.50	6.35
0.98*	7.70
1.00	7.70
1.02*	7.70
2.50	5.20
4.40*	4.70
5.00	5.50
6.00	6.00

(* denotes a common end point of two quadratic polynomial curves)

Table 3.2: Aerodynamic coefficient C_{D0} versus Mach number

M	C_{D0}
0.20	1.00
0.75	0.45
0.98*	0.80
1.00	0.80
1.02*	0.80
3.50	0.20
6.00	0.02

(* denotes a common end point of two quadratic polynomial curves)

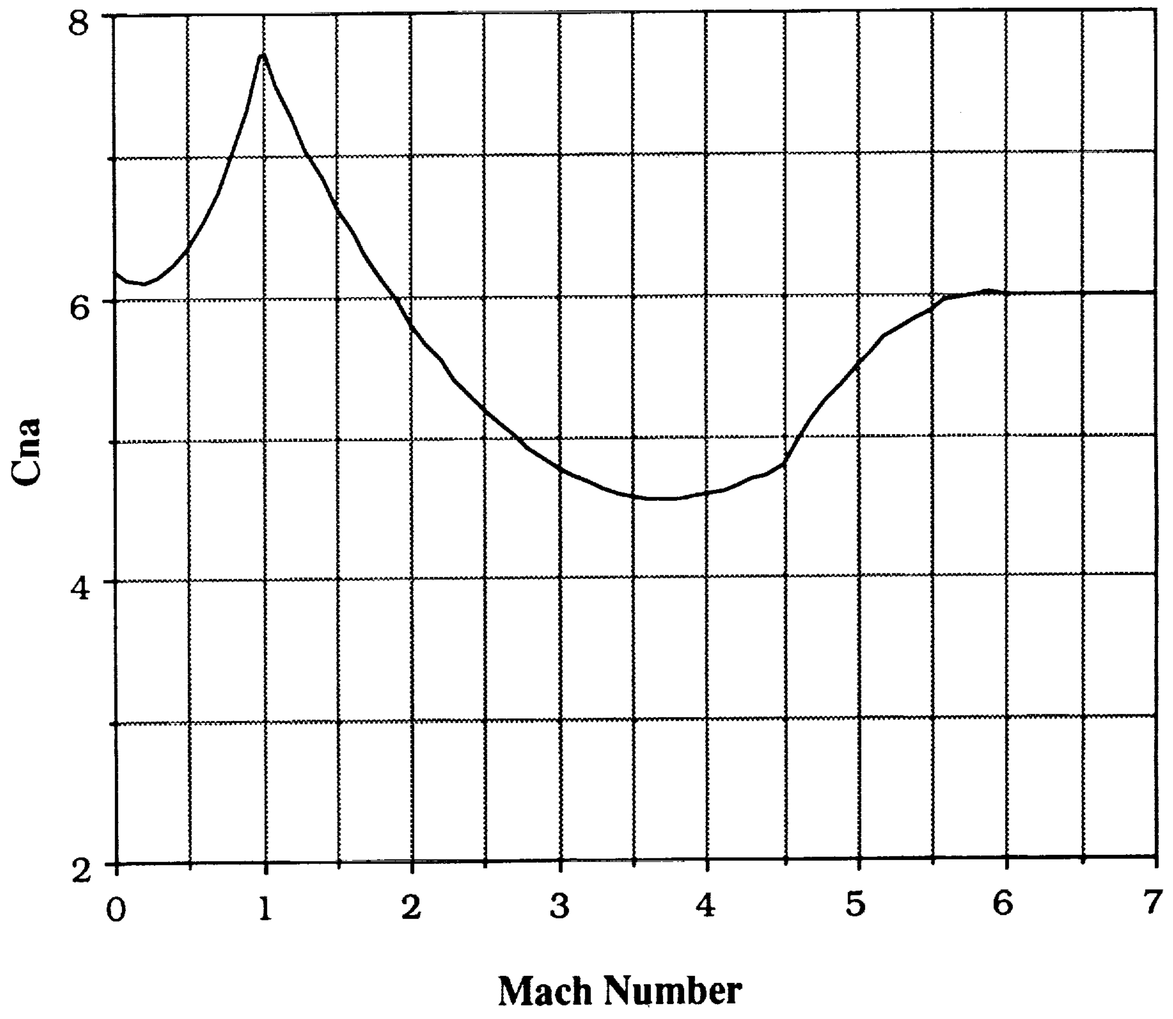

Fig. 3.2: $C_{N\alpha}$ versus Mach number

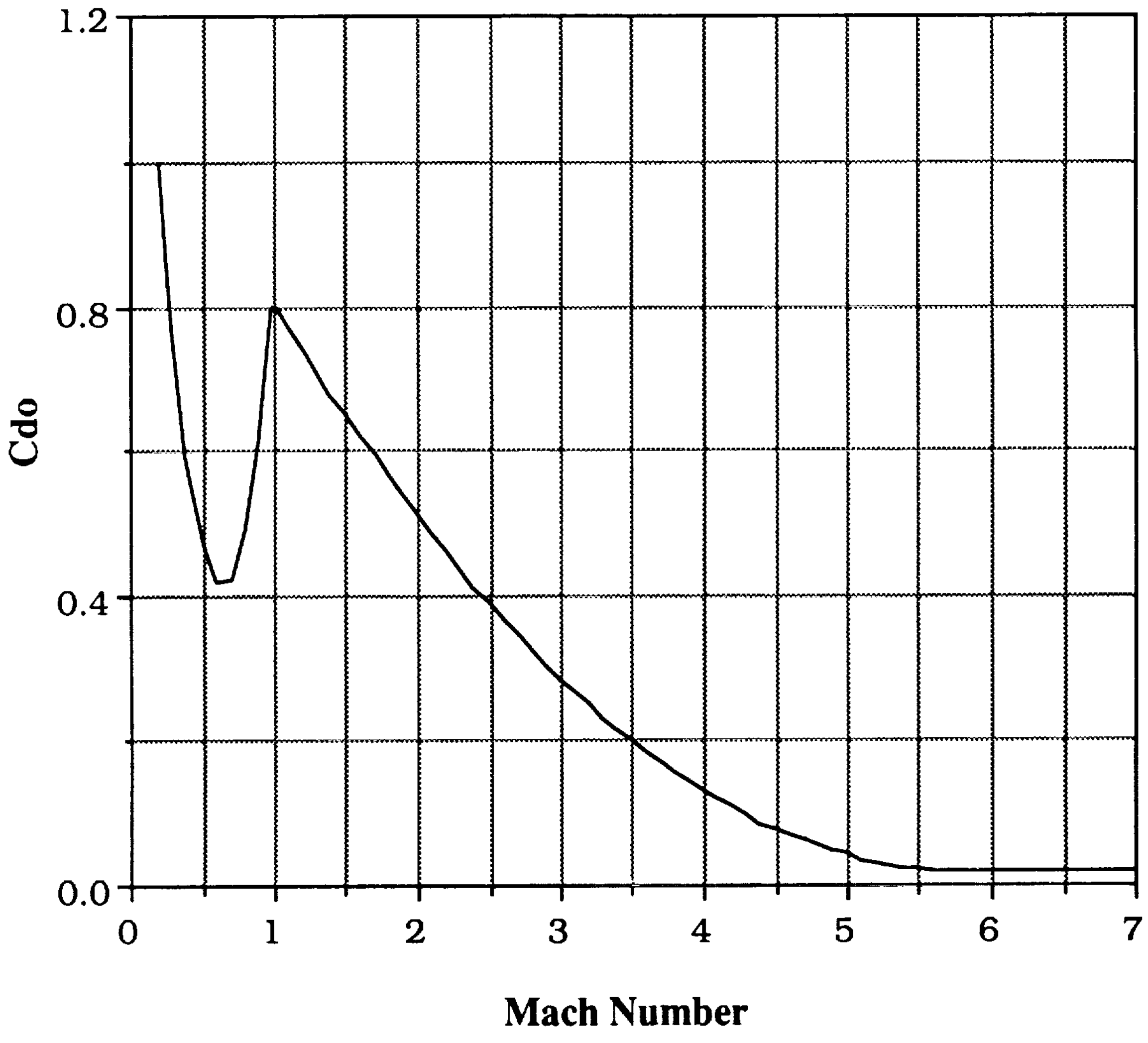

Fig. 3.3: C_{D0} versus Mach number

3.2. Results

Two codes have been developed that solve the resulting finite element equations. One uses the Newton-Raphson method, and the other uses the method of Levenberg-Marquardt as coded in the IMSL subroutine ZXSSQ [55]. With the former method, the initial conditions need to be reasonably accurate. However, running a case for a few elements generates a good approximation for larger numbers of elements as above. Also, with this method one can easily exploit sparsity; the computational savings of this will be investigated in later chapters. With the latter method, the initial guesses do not need to be very accurate, but the method is not nearly as computationally efficient as the former since it generates an approximate Jacobian from central differencing.

In Figs. 3.4 – 3.12, numerical results for the Saturn one-stage model are given. In these figures, the finite element results are shown as discrete symbols while the solid lines show results obtained from a multiple shooting code, an essentially exact solution [56]. In Figs. 3.4 – 3.7, the mass, altitude, specific energy, and flight-path angle profiles are shown. For the mass and specific energy, even $N = 4$ gives excellent results. For the altitude and flight-path angle, $N = 8$ gives good agreement with the multiple-shooting results. When higher final energies were tried, the rate of change of the energy became very steep, and the finite element results became more difficult to obtain. One must remember, however, that these results are not realistic because of the absence of state constraints and, particularly for the steep energy gradients, the use of single-stage modeling.

Figs. 3.8 – 3.11 show the costate profiles. The results for the mass costate and specific-energy costate are not quite as accurate as were the results for the states. Adding more elements would, however, increase the accuracy of the solution.

In Fig. 3.12 the angle of attack profile is shown. Here the $N = 4$ solution agrees well with the exact solution except near the beginning of the trajectory. Note the smooth and rapid convergence at $t = 0$ as N increases.

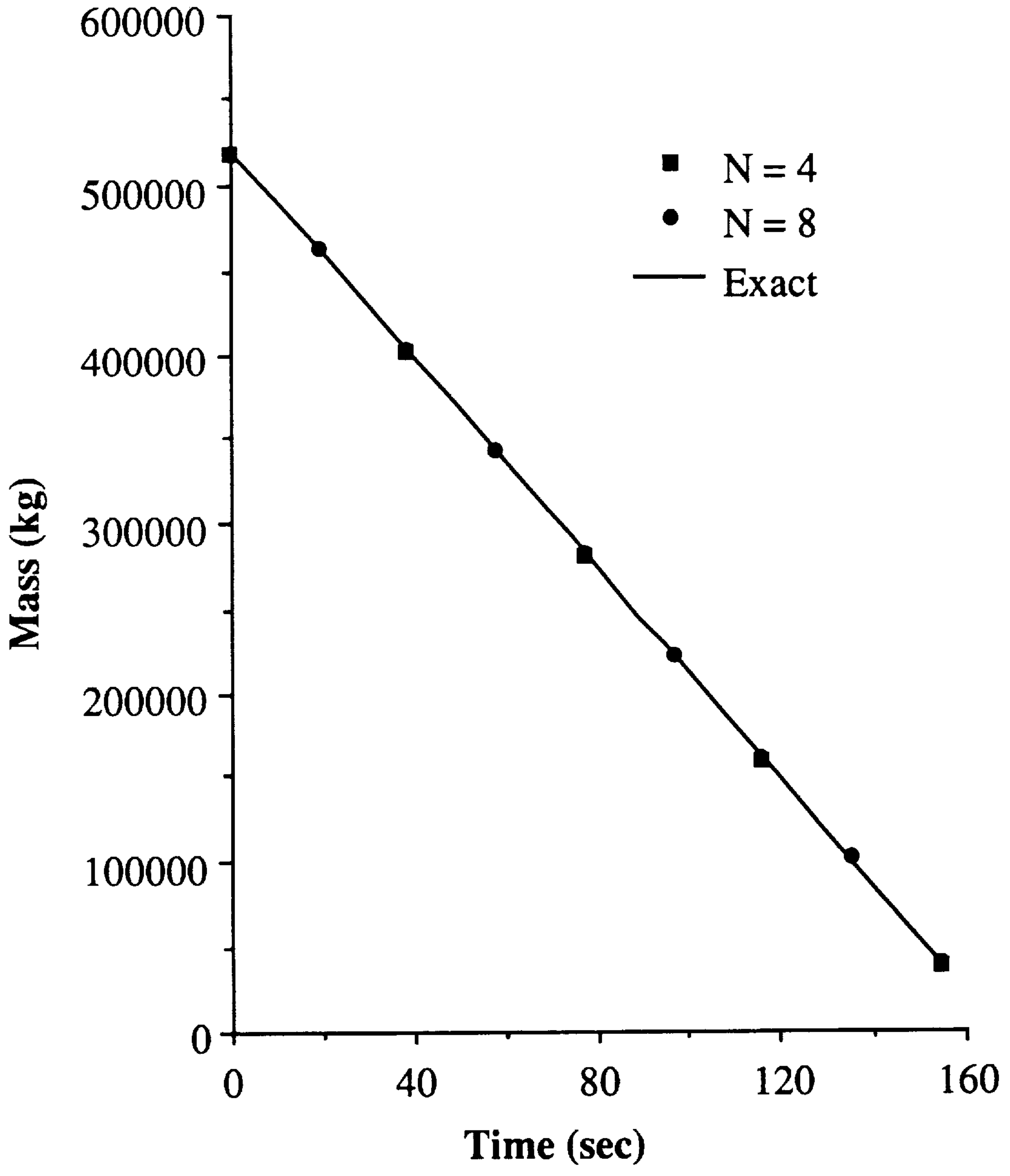

Fig. 3.4: Mass versus time
"exact" and finite element results

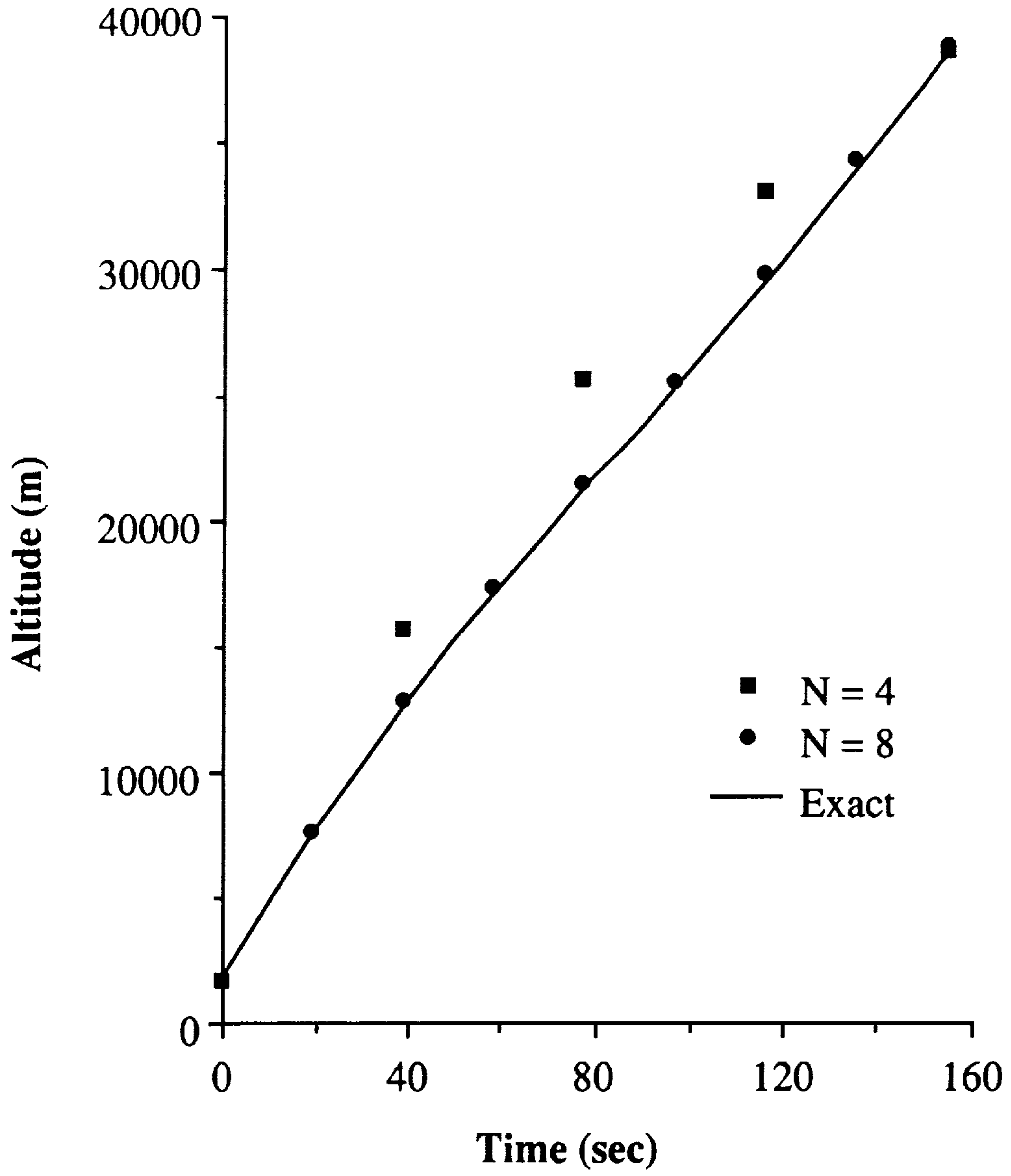

Fig. 3.5: Altitude versus time
"exact" and finite element results

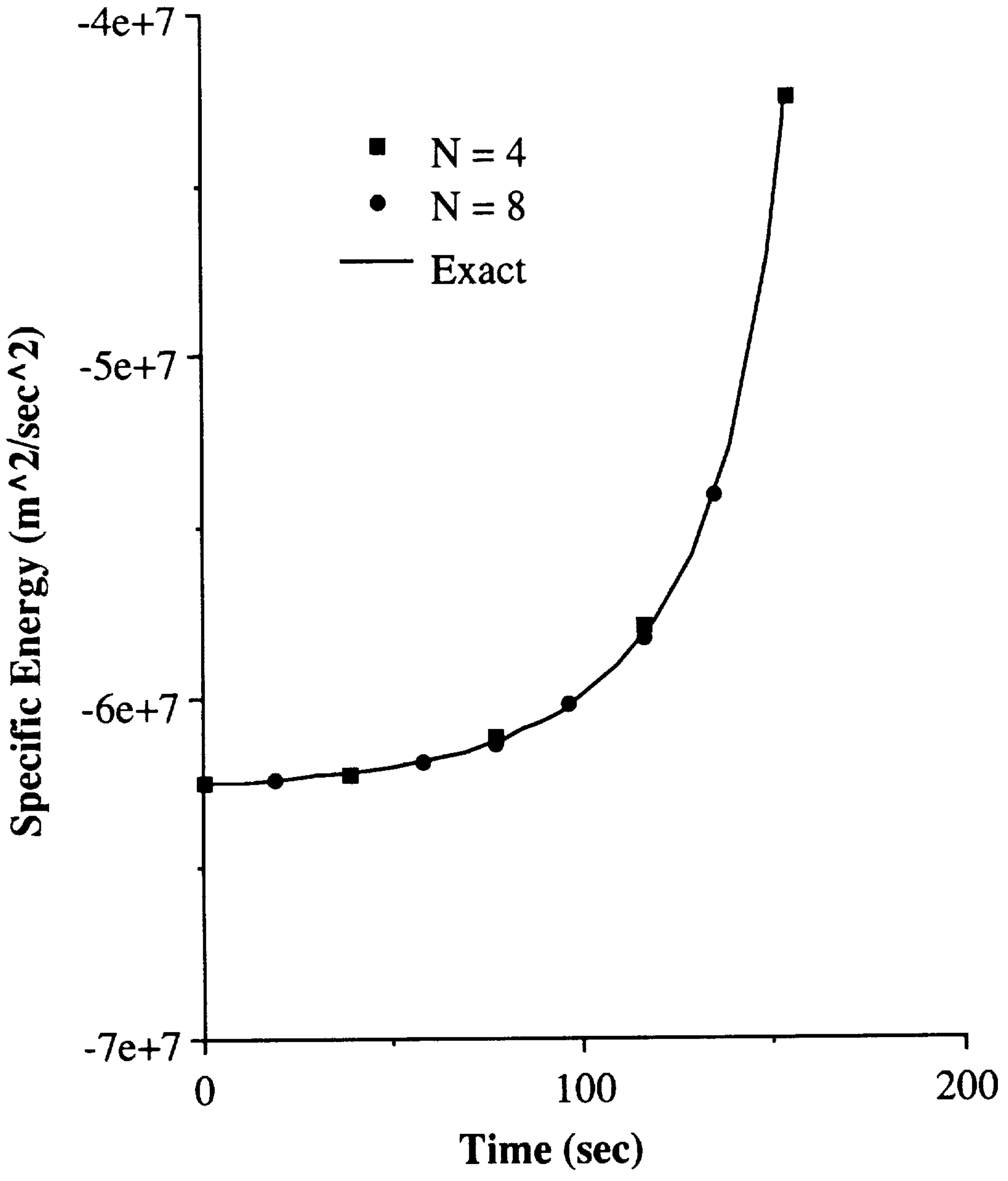

Fig. 3.6: Specific energy versus time
"exact" and finite element results

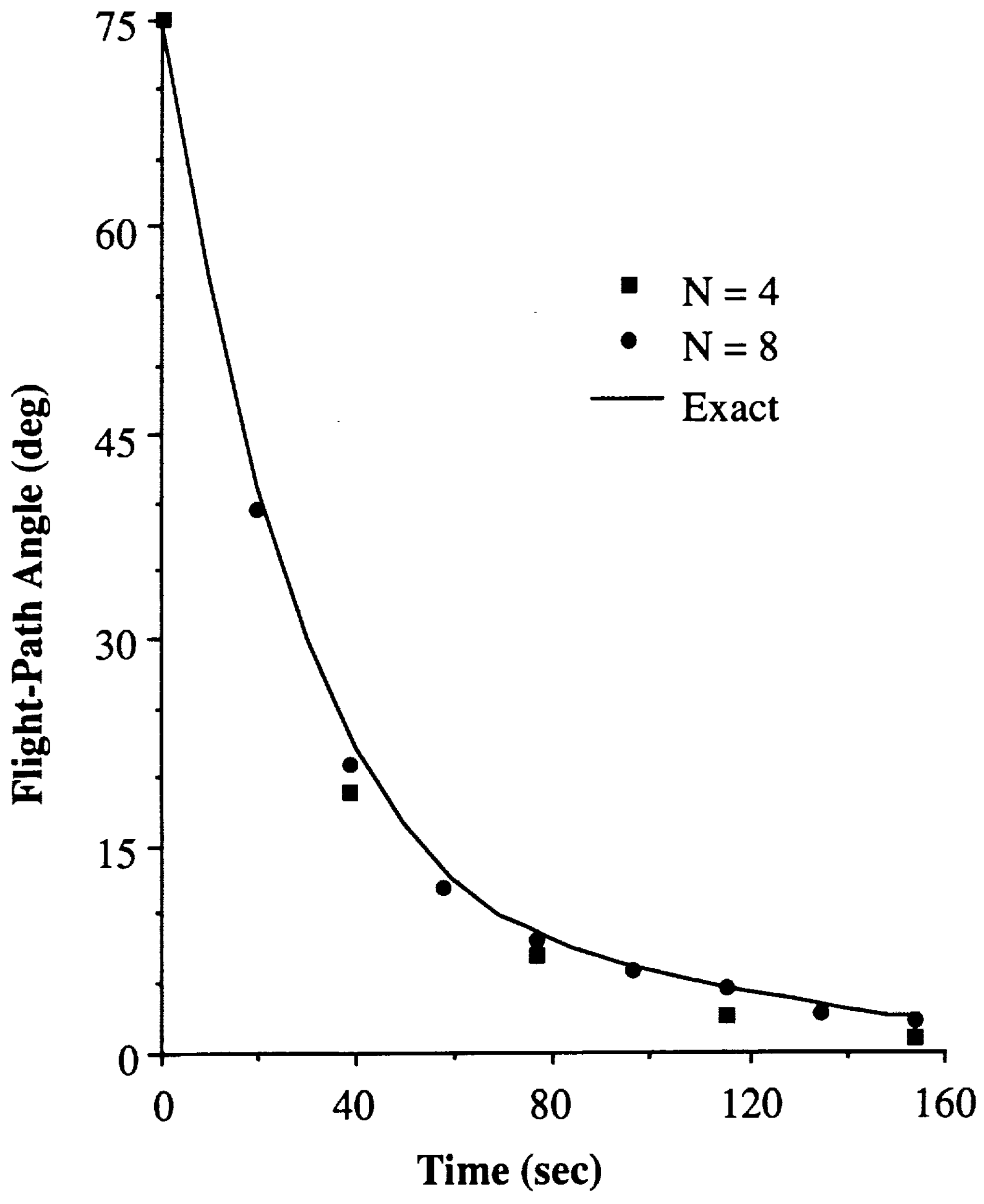

Fig. 3.7: Flight-path angle versus time
"exact" and finite element results

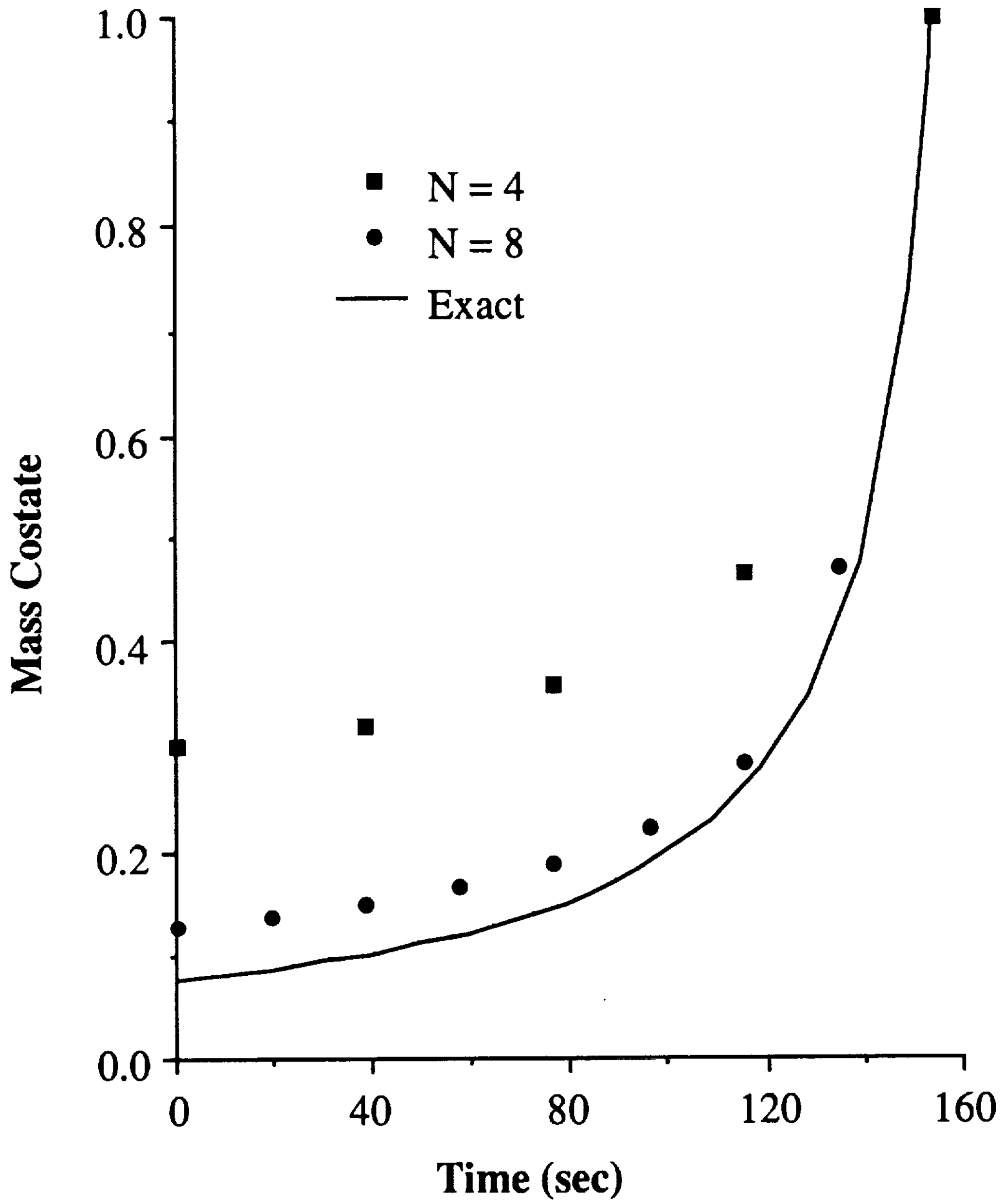

Fig. 3.8: Mass costate versus time
"exact" and finite element results

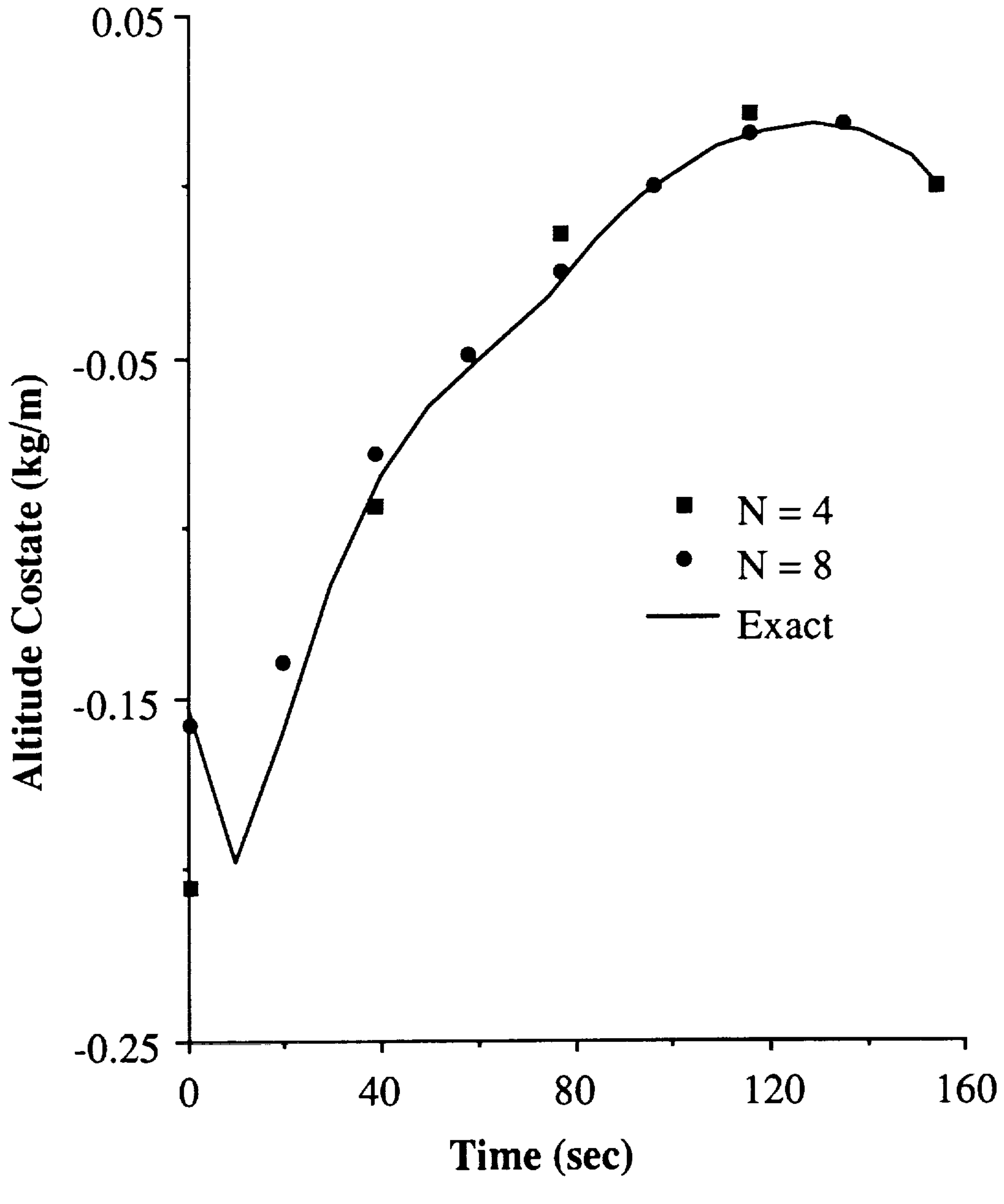

Fig. 3.9: Altitude costate versus time
"exact" and finite element results

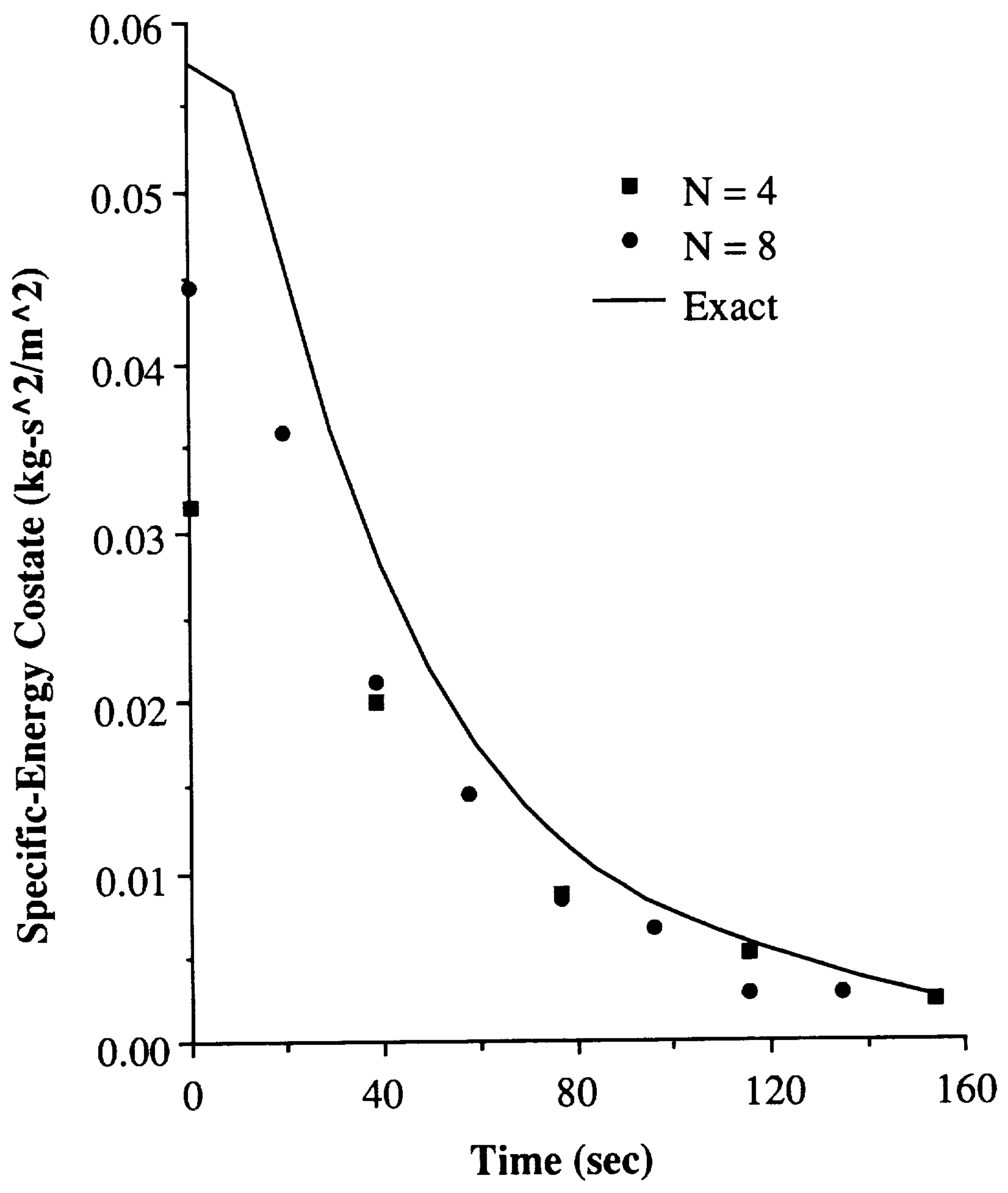

Fig. 3.10: Specific energy costate versus time "exact" and finite element results

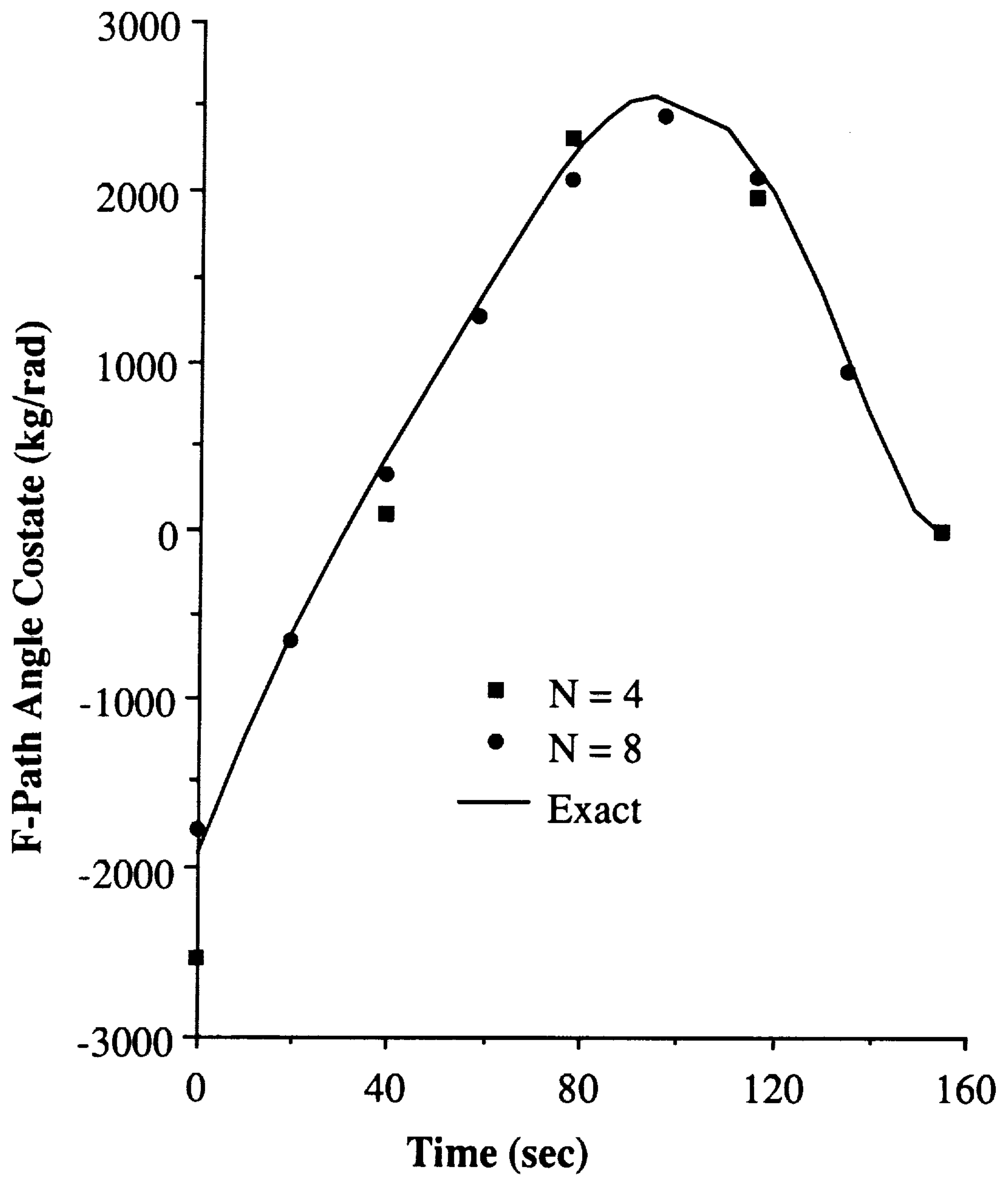

Fig. 3.11: Flight-path angle costate versus time
"exact" and finite element results

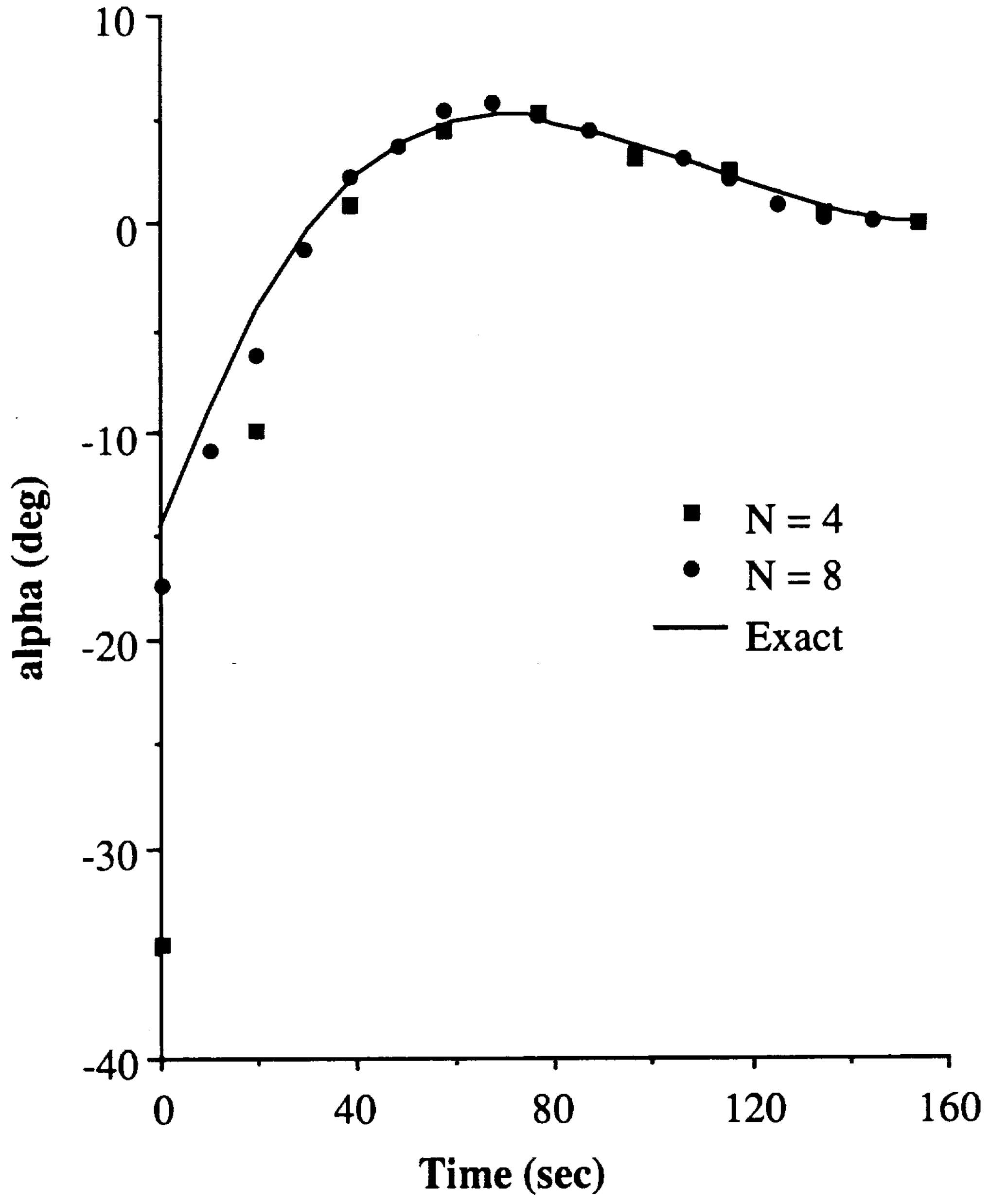

Fig. 3.12: Angle of attack (α) versus time
"exact" and finite element results

CHAPTER 4

DISCONTINUITIES IN THE SYSTEM EQUATIONS

Although many practical problems in optimal control theory can be solved by using the weak formulation derived in Chapter 2, there are problems that require more generality. Specifically, the weak formulation must be extended to allow for discontinuities in the states and/or discontinuities in the system equations. Such discontinuities might arise when finding the optimal trajectory of a multistage rocket. At the time the first stage is dropped, the mass state would suffer a discontinuity. Furthermore, the thrust of the rocket (one of the system parameters) would also change at this staging time, thereby creating a discontinuity in the system equations. These discontinuities produce jumps in the states, costates, and possibly the control variables.

The derivation of the weak formulation to include state discontinuities (or jumps) and discontinuities in the system equations is similar to the derivation in Chapter 2, (see also [57]); however, special care must be taken because of the unknown staging time. Therefore, the details of the derivation are presented below for a problem with one discontinuity. Of course, the extension of the formulation to problems with multiple discontinuities is possible and should not cause any confusion.

4.1. Theory for State Discontinuities

Consider a problem with one discontinuity where the time of discontinuity will be called the staging time and denoted by t_s. The formulation must be modified to accommodate the unknown staging time t_s, the constraint on the states at t_s (as opposed to a constraint on the states at the final time), the jump in the states at t_s, the jump in the costates at t_s, the change in state equations at t_s (due to the change in system parameters), and finally the transversality condition to find t_s. Furthermore, the control u may be discontinuous at the staging time.

Now, let a system be defined by a set of n states x and a set of m controls u. Furthermore, let the system be governed by a set of state equations of the form $\dot{x} = f_{\mathrm{I}}(x,u,t)$ prior to t_s and $\dot{x} = f_{\mathrm{II}}(x,u,t)$ after t_s. Elements of the performance index, J, may be denoted with integrands $L_{\mathrm{I}}(x,u,t)$ prior to t_s, $L_{\mathrm{II}}(x,u,t)$ after t_s, discrete functions of the states and time $\phi[x(t),t]$ defined at the initial and final times t_0 and t_f, and discrete functions of the states and time $\phi_s[x(t),t]$ defined just before and after the staging time signified by t_s^- and t_s^+ respectively. In addition, any constraints imposed on the states and time at t_0 and t_f may be placed in sets of functions $\psi[x(t),t]$, whereas constraints imposed at t_s^- and t_s^+ may be placed in $\psi_s[x(t),t]$. These constraints may be adjoined to the performance index by discrete Lagrange multipliers ν and ν_s respectively. Finally, the state equations may be adjoined to the performance index with a set of Lagrange multiplier functions $\lambda(t)$ which will be referred to as costates. For notational convenience, we define $\Phi = \phi[x(t),t] + \nu^T\psi[x(t),t]$ and $\Phi_s = \phi_s[x(t),t] + \nu_s^T\psi_s[x(t),t]$. Also, a constraint equation is adjoined to the performance index (as was done in Chapter 2) to transform the strong boundary conditions to weak ones. For variable t_s and t_f, this yields a performance index of the form

$$J = \int_{t_0}^{t_s^-} \left[L_{\mathrm{I}} + \lambda^T(f_{\mathrm{I}} - \dot{x})\right] \, dt + \int_{t_s^+}^{t_f} \left[L_{\mathrm{II}} + \lambda^T(f_{\mathrm{II}} - \dot{x})\right] \, dt$$
$$+ \Phi_s\Big|_{t_s^-}^{t_s^+} + \Phi\Big|_{t_0}^{t_f} + \alpha^T(x - \hat{x})\Big|_{t_0}^{t_f} \qquad (4.1\text{–}1)$$

Now, denote with $\delta()$ the variation of () when holding time fixed, and let $d()$ be the variation of () when time is allowed to vary. The fixed and free-time variations at $t = t_0$ and $t = t_f$ are related by (see [1] or [2])

$$\delta x(t_0) = dx(t_0) \quad \text{and} \quad \delta x(t_f) = dx(t_f) - \dot{x}|_{t_f} dt_f \qquad (4.1\text{–}2)$$

and similarly for λ. (Note that t_0 is considered to be a fixed time so that $dt_0 = 0$.) The free and fixed-time variations at $t = t_s^-$ and $t = t_s^+$ are related by

$$\delta x(t_s^-) = dx(t_s^-) - \dot{x}|_{t_s^-} dt_s \quad \text{and} \quad \delta x(t_s^+) = dx(t_s^+) - \dot{x}|_{t_s^+} dt_s \qquad (4.1\text{–}3)$$

and similarly for λ. It is noted that if a particular state (or costate) does not have a discontinuity at $t = t_s$, then the corresponding free-time variation is continuous at t_s, *i.e.*, $dx(t_s^-) = dx(t_s^+)$. Now, proceeding with the development of the weak form, the first variation of J is taken and the $\delta\dot{x}$ term appearing in both integrands is integrated by parts. The resulting equation is

$$
\begin{aligned}
dJ = &\int_{t_0}^{t_s^-} \left[\delta L_{\mathrm{I}}^T + \delta f_{\mathrm{I}}^T \lambda + \delta\lambda^T (f_{\mathrm{I}} - \dot{x}) + \delta x^T \dot{\lambda}\right] dt - \delta x^T \lambda \Big|_{t_0}^{t_s^-} \\
&+ \int_{t_s^+}^{t_f} \left[\delta L_{\mathrm{II}}^T + \delta f_{\mathrm{II}}^T \lambda + \delta\lambda^T (f_{\mathrm{II}} - \dot{x}) + \delta x^T \dot{\lambda}\right] dt - \delta x^T \lambda \Big|_{t_s^+}^{t_f} \\
&+ \delta\nu_s^T \psi_s \Big|_{t_s^-}^{t_s^+} + \delta\nu^T \psi \Big|_{t_0}^{t_f} + dx^T \left(\frac{\partial \Phi_s}{\partial x}\right)^T \Bigg|_{t_s^-}^{t_s^+} + dx^T \left(\frac{\partial \Phi}{\partial x}\right)^T \Bigg|_{t_0}^{t_f} \\
&+ \delta\alpha^T (x - \hat{x}) \Big|_{t_0}^{t_f} + \alpha^T (dx - d\hat{x}) \Big|_{t_0}^{t_f} + dt_f \left[L_{\mathrm{II}} + \lambda^T (f_{\mathrm{II}} - \dot{x}) + \frac{\partial \Phi}{\partial t}\right] \Bigg|_{t_f} \\
&+ dt_s \left\{ \left[L_{\mathrm{I}} + \lambda^T (f_{\mathrm{I}} - \dot{x})\right] \Big|_{t_s^-} - \left[L_{\mathrm{II}} + \lambda^T (f_{\mathrm{II}} - \dot{x})\right] \Big|_{t_s^+} + \frac{\partial \Phi_s}{\partial t_s} \right\}
\end{aligned}
\tag{4.1–4}
$$

A necessary condition for an extremal of J is that the first variation be zero. Also, the admissible variations of the states must be continuous at the initial and final times and therefore $(dx - d\hat{x})|_{t_0}^{t_f} = 0$. For notational convenience, define $H_{\mathrm{I}} = L_{\mathrm{I}} + \lambda^T f_{\mathrm{I}}$, $H_{\mathrm{II}} = L_{\mathrm{II}} + \lambda^T f_{\mathrm{II}}$, and

$$
\hat{\lambda}\Big|_{t_0} = \frac{\partial \Phi}{\partial x}\Big|_{t_0} \quad \text{and} \quad \hat{\lambda}\Big|_{t_f} = \frac{\partial \Phi}{\partial x}\Big|_{t_f}
\tag{4.1–5}
$$

Finally, by using Eqs. (4.1–2) and (4.1–3) to eliminate δx at t_0, t_s^-, t_s^+, and t_f, the above equation can be simplified to

$$
\begin{aligned}
&\int_{t_0}^{t_s^-} \left\{ \delta\lambda^T (f_{\mathrm{I}} - \dot{x}) + \delta x^T \left[\dot{\lambda} + \left(\frac{\partial H_{\mathrm{I}}}{\partial x} \right)^T \right] + \delta u^T \left(\frac{\partial H_{\mathrm{I}}}{\partial u} \right)^T \right\} dt \\
&+ \int_{t_s^+}^{t_f} \left\{ \delta\lambda^T (f_{\mathrm{II}} - \dot{x}) + \delta x^T \left[\dot{\lambda} + \left(\frac{\partial H_{\mathrm{II}}}{\partial x} \right)^T \right] + \delta u^T \left(\frac{\partial H_{\mathrm{II}}}{\partial u} \right)^T \right\} dt \\
&+ \delta\nu_s^T \psi_s \Big|_{t_s^-}^{t_s^+} + \delta\nu^T \psi \Big|_{t_0}^{t_f} + dx^T \left[\left(\frac{\partial \Phi_s}{\partial x} \right)^T + \lambda \right] \Bigg|_{t_s^-}^{t_s^+} \qquad (4.1\text{–}6) \\
&+ dx^T \left(\hat{\lambda} - \lambda \right) \Big|_{t_0}^{t_f} + \delta\alpha^T (x - \hat{x}) \Big|_{t_0}^{t_f} \\
&+ dt_f \left(H_{\mathrm{II}} + \frac{\partial \Phi}{\partial t} \right) \Big|_{t_f} + dt_s \left(H_{\mathrm{I}} \Big|_{t_s^-} - H_{\mathrm{II}} \Big|_{t_s^+} + \frac{\partial \Phi_s}{\partial t_s} \right) = 0
\end{aligned}
$$

It is easily verified that the necessary conditions for an extremal of J, as defined in [1], are contained in the coefficients of the virtual quantities [*i.e.*, the $\delta()$ and $d()$ terms] of Eq. (4.1–6). This is described in Chapter 2 and thus only the new boundary conditions which appear as the result of the staging are discussed here. Specifically, the requirement for the coefficient of $\delta\nu_s$ to vanish yields Eq. (3.7.3) of [1]. The requirement for the coefficient of $dx(t_s^-)$ to vanish yields Eq. (3.7.11), whereas the requirement for the coefficient of $dx(t_s^+)$ to vanish yields Eq. (3.7.12). Finally the coefficient of dt_s is the transversality condition in Eq. (3.7.13).

Having satisfied the requirement that none of the fundamental equations are altered, the weak formulation may now be derived. First, we choose $\delta\alpha = d\lambda$. Next, the $\dot{x}$ and $\dot{\lambda}$ terms in Eq. (4.1–6) are integrated by parts. Also, Eq. (4.1–2) is used to eliminate dx and $d\lambda$ at t_0 and t_f. The resulting equation is

$$\begin{aligned}
&\int_{t_0}^{t_s^-}\left[\delta\lambda^T f_{\mathrm{I}} + \delta\dot{\lambda}^T x - \delta\dot{x}^T\lambda + \delta x^T\left(\frac{\partial H_{\mathrm{I}}}{\partial x}\right)^T + \delta u^T\left(\frac{\partial H_{\mathrm{I}}}{\partial u}\right)^T\right]dt\\
&+\int_{t_s^+}^{t_f}\left[\delta\lambda^T f_{\mathrm{II}} + \delta\dot{\lambda}^T x - \delta\dot{x}^T\lambda + \delta x^T\left(\frac{\partial H_{\mathrm{II}}}{\partial x}\right)^T + \delta u^T\left(\frac{\partial H_{\mathrm{II}}}{\partial u}\right)^T\right]dt\\
&+\delta\nu_s^T\psi_s\Big|_{t_s^-}^{t_s^+} + \delta\nu^T\psi\Big|_{t_0}^{t_f} + \delta x^T\hat{\lambda}\Big|_{t_0}^{t_f} - \delta\lambda^T\hat{x}\Big|_{t_0}^{t_f}\\
&+dx^T\left[\left(\frac{\partial\Phi_s}{\partial x}\right)^T + \lambda\right]\Bigg|_{t_s^-}^{t_s^+} + \delta\lambda^T x\Big|_{t_s^-}^{t_s^+} - \delta x^T\lambda\Big|_{t_s^-}^{t_s^+}\\
&+dt_f\left(H_{\mathrm{II}} + \frac{\partial\Phi}{\partial t}\right)\Bigg|_{t_f} + dt_s\left(H_{\mathrm{I}}\Big|_{t_s^-} - H_{\mathrm{II}}\Big|_{t_s^+} + \frac{\partial\Phi_s}{\partial t_s}\right) = 0
\end{aligned} \tag{4.1-7}$$

Note that a $(\hat{\lambda}-\lambda)\dot{x}$ term and an $(\hat{x}-x)\dot{\lambda}$ term should appear in the dt_f equation but these terms are each *zero* in accordance with the natural boundary conditions (see the dx and $\delta\alpha$ coefficients in Eq. 4.1–6). Eq. (4.1–3) is now used to eliminate dx at t_s^- and t_s^+. Again, the natural boundary conditions in Eq. (4.1–6) must be used to avoid changing the dt_s coefficient. The weak principle is now given as

$$\begin{aligned}
&\int_{t_0}^{t_s^-}\left[\delta\lambda^T f_{\mathrm{I}} + \delta\dot{\lambda}^T x - \delta\dot{x}^T\lambda + \delta x^T\left(\frac{\partial H_{\mathrm{I}}}{\partial x}\right)^T + \delta u^T\left(\frac{\partial H_{\mathrm{I}}}{\partial u}\right)^T\right]dt\\
&+\int_{t_s^+}^{t_f}\left[\delta\lambda^T f_{\mathrm{II}} + \delta\dot{\lambda}^T x - \delta\dot{x}^T\lambda + \delta x^T\left(\frac{\partial H_{\mathrm{II}}}{\partial x}\right)^T + \delta u^T\left(\frac{\partial H_{\mathrm{II}}}{\partial u}\right)^T\right]dt\\
&+\delta x^T\hat{\lambda}\Big|_{t_0}^{t_f} - \delta\lambda^T\hat{x}\Big|_{t_0}^{t_f} + \delta x^T\left(\frac{\partial\Phi_s}{\partial x}\right)^T\Bigg|_{t_s^-}^{t_s^+} + \delta\lambda^T x\Big|_{t_s^-}^{t_s^+} + \delta\nu^T\psi\Big|_{t_0}^{t_f}\\
&+\delta\nu_s^T\psi_s\Big|_{t_s^-}^{t_s^+} + dt_s\left(H_{\mathrm{I}}\Big|_{t_s^-} - H_{\mathrm{II}}\Big|_{t_s^+} + \frac{\partial\Phi_s}{\partial t_s}\right) + dt_f\left(H_{\mathrm{II}} + \frac{\partial\Phi}{\partial t}\right)\Bigg|_{t_f} = 0
\end{aligned} \tag{4.1-8}$$

This is the governing equation for the weak Hamiltonian method for optimal control problems of the form specified. It will serve as the basis for the finite element discretization described below for constructing of candidate solutions (*i.e.*, solutions which satisfy all the necessary conditions).

4.2. Finite Element Discretization

Due to the staging, the finite element discretization must be handled somewhat differently than was done in Chapter 2. Therefore, for clarity, full details of the discretization will be given below. Let the time interval from t_0 to t_s^- be broken into N_1 elements and the time interval from t_s^+ to t_f be broken into N_2 elements. For notational convenience, define $N = N_1 + N_2$. The nodal values of time for these elements are t_i for $i = 1, 2, \ldots, N+1$ where $t_0 = t_1$, $t_s = t_{N_1+1}$, and $t_f = t_{N+1}$. A nondimensional elemental time τ is defined as

$$\tau = \frac{t - t_i}{t_{i+1} - t_i} = \frac{t - t_i}{\Delta t_i} \tag{4.2–1}$$

Since one derivative of δx and $\delta\lambda$ appears in Eq. (4.1–8), linear shape functions for δx and $\delta\lambda$ may be chosen. Since no derivatives of x or λ appear, then piecewise constant shape functions for x and λ are chosen. These shape functions are taken to be

$$\delta x = \delta x_i^+(1 - \tau) + \delta x_{i+1}^- \tau \tag{4.2–2}$$

and

$$x = \begin{cases} \hat{x}_i^+ & \text{if } \tau = 0; \\ \bar{x}_i & \text{if } 0 < \tau < 1; \\ \hat{x}_{i+1}^- & \text{if } \tau = 1 \end{cases} \tag{4.2–3}$$

and similarly for $\delta\lambda$ and λ. The superscripted "+" and "–" signs signify values just before and after the subscripted nodal value. For all nodes *except* the $N_1 + 1$ node which corresponds to t_s, the values for x and λ, as well as for δx and $\delta\lambda$, are equal on either side of the node. Furthermore, it is important to understand that

$\hat{x}_1^+ = \hat{x}_0 = x(t_0)$, $\hat{\lambda}_1^+ = \hat{\lambda}_0 = \lambda(t_0)$, $\hat{x}_{N+1}^- = \hat{x}_f = x(t_f)$, and $\hat{\lambda}_{N+1}^- = \hat{\lambda}_f = \lambda(t_f)$. We again choose $u = \bar{u}_i$ and $\delta u = \delta\bar{u}_i$ as shape functions for the control and its variation.

Plugging in the shape functions described for x, λ, and u, substituting $t = t_i + \tau\Delta t_i$, and carrying out the integration over τ from 0 to 1, we obtain a general algebraic form of our Hamiltonian weak form for optimal control problems of the form specified. The algebraic equations are

$$\begin{aligned}
&\sum_{i=1}^{N_1}\left\{\delta x_i^{+^T}\left[\bar{\lambda}_i + \frac{\Delta t_i}{2}\left(\frac{\partial \bar{H}_{\mathrm{I}}}{\partial \bar{x}}\right)_i^T\right] - \delta\lambda_i^{+^T}\left[\bar{x}_i - \frac{\Delta t_i}{2}\left(\bar{f}_{\mathrm{I}}\right)_i\right]\right.\\
&\qquad - \delta x_{i+1}^{-^T}\left[\bar{\lambda}_i - \frac{\Delta t_i}{2}\left(\frac{\partial \bar{H}_{\mathrm{I}}}{\partial \bar{x}}\right)_i^T\right] + \delta\lambda_{i+1}^{-^T}\left[\bar{x}_i + \frac{\Delta t_i}{2}\left(\bar{f}_{\mathrm{I}}\right)_i\right]\\
&\qquad \left.+ \delta\bar{u}_i^T\left[\Delta t_i\left(\frac{\partial \bar{H}_{\mathrm{I}}}{\partial \bar{u}}\right)_i^T\right]\right\} + dt_s\left(\hat{H}_{N_1+1}^- - \hat{H}_{N_1+1}^+ + \frac{\partial \Phi_s}{\partial t_s}\right)\\
&\qquad - \delta x_1^{+^T}\hat{\lambda}_1^+ + \delta\lambda_1^{+^T}\hat{x}_1^+ + \delta x_{N_1+1}^{+^T}\left(\frac{\partial \Phi_s}{\partial x}\right)^{+^T} + \delta\lambda_{N_1+1}^{+^T}\hat{x}_{N_1+1}^{+^T}\\
&\qquad - \delta x_{N_1+1}^{-^T}\left(\frac{\partial \Phi_s}{\partial x}\right)^{-^T} - \delta\lambda_{N_1+1}^{-^T}\hat{x}_{N_1+1}^{-^T} \qquad\qquad (4.2\text{–}4)\\
&+ \sum_{i=N_1+1}^{N}\left\{\delta x_i^{+^T}\left[\bar{\lambda}_i + \frac{\Delta t_i}{2}\left(\frac{\partial \bar{H}_{\mathrm{II}}}{\partial \bar{x}}\right)_i^T\right] - \delta\lambda_i^{+^T}\left[\bar{x}_i - \frac{\Delta t_i}{2}\left(\bar{f}_{\mathrm{II}}\right)_i\right]\right.\\
&\qquad - \delta x_{i+1}^{-^T}\left[\bar{\lambda}_i - \frac{\Delta t_i}{2}\left(\frac{\partial \bar{H}_{\mathrm{II}}}{\partial \bar{x}}\right)_i^T\right] + \delta\lambda_{i+1}^{-^T}\left[\bar{x}_i + \frac{\Delta t_i}{2}\left(\bar{f}_{\mathrm{II}}\right)_i\right]\\
&\qquad \left.+ \delta\bar{u}_i^T\left[\Delta t_i\left(\frac{\partial \bar{H}_{\mathrm{II}}}{\partial \bar{u}}\right)_i^T\right]\right\} + dt_f\left(\hat{H}_{N+1} + \frac{\partial \Phi}{\partial t_f}\right)\\
&\qquad - \delta\lambda_{N+1}^{-^T}\hat{x}_{N+1}^- + \delta x_{N+1}^{-^T}\hat{\lambda}_{N+1}^- + \delta\nu_s^T\psi_s\Big|_{t_s^-}^{t_s^+} + \delta\nu^T\psi = 0
\end{aligned}$$

where $\bar{H} = H(\bar{x}, \bar{u}, \bar{t})$ and $\hat{H} = H(\hat{x}, \hat{u}, \hat{t})$. Note that there are not only boundary conditions at t_0 and t_f, but also additional boundary conditions at the staging time

t_s, namely the coefficients of $\delta x^-_{N_1+1}$, $\delta x^+_{N_1+1}$, $\delta\lambda^-_{N_1+1}$, and $\delta\lambda^+_{N_1+1}$. This is where the jumps in the states and costates are allowed to occur. Thus, rather than the nodal values of the shape functions canceling one another at the staging time (as they do at all other internal nodes), the nodal values at t_s^- and t_s^+ will be distinct.

Eq. (4.2–4) is a system of nonlinear algebraic equations. The coefficient of each arbitrary virtual quantity ($\delta x, \delta\lambda, \delta u, dt_s, dt_f, \delta\nu_{\mathrm{I}}$, and $\delta\nu$) must be set equal to zero in order to satisfy Eq. (4.2–4). However, not all of the virtual quantities above are independent. As stated earlier, $\delta x_i^+ = \delta x_i^-$ for all i except $i = N_1 + 1$, which is the node number corresponding to the staging time. At this node, it was observed from the calculus of variations that the virtual quantities suffer a discontinuity. Now, the coefficients of $\delta x^-_{N_1+1}$ and $\delta x^+_{N_1+1}$ may be treated as separate equations; however, a different option has been chosen in an attempt to simplify the equations to be programmed. Define

$$\delta x^-_{N_1+1} = \delta x_{N_1+1} + \delta\eta_\lambda \tag{4.2–5}$$

and

$$\delta x^+_{N_1+1} = \delta x_{N_1+1} - \delta\eta_\lambda \tag{4.2–6}$$

When these values are substituted into Eq. (4.2–4), then two new arbitrary virtual quantities appear, namely δx_{N_1+1} and $\delta\eta_\lambda$. Fig. 4.1 helps clarify the assembly process of the virtual states. The figure shows three straight lines depicting linear shape functions over three elements. For the non-jump node N_1, the virtually quantities are equal at the node and replaced with δx_{N_1}. At the jump node, however, $\delta x^+_{N_1+1}$ and $\delta x^-_{N_1+1}$ are replaced with an average value δx_{N_1+1}. Another virtual quantity, $\delta\eta_\lambda$, also appears but is not shown in Fig. 4.1. The coefficient of δx_{N_1+1} is now of the same form as all the other δx terms and the $\delta\eta_\lambda$ coefficient contains an equation to extract the needed nodal value of λ at t_s, namely $\hat{\lambda}_{N_1+1}$.

To simplify matters further though, the coefficient of $\delta\eta_\lambda$ is replaced with a still simpler expression to extract the needed nodal value. This expression comes from the following recursive equation derived in Chapter 2.

$$\hat{z}_{i+1} = 2\bar{z}_i - \hat{z}_i \tag{4.2-7}$$

where z represents either the state x or the costate λ. The first nodal value $\hat{z}_1$ is equal to the initial values of the states and costates which are represented by $\hat{x}_0$ and $\hat{\lambda}_0$ in Eq. (4.2–4). The same process is done with the $\delta\lambda^-_{N_1+1}$ and $\delta\lambda^+_{N_1+1}$ terms so that a $\delta\lambda_{N_1+1}$ and $\delta\eta_x$ are introduced. As a final step, all superscripted "+" and "–" signs can now be dropped (except on $\hat{x}_{N_1+1}$ and $\hat{\lambda}_{N_1+1}$, because these values are still distinct) since they are equal at all nodes but the staging time node which has now been handled. The algebraic equations now take the form

$$
\begin{aligned}
&\sum_{i=1}^{N_1}\Bigg\{\delta x_i^T\left[\bar\lambda_i+\frac{\Delta t_i}{2}\left(\frac{\partial \bar H_{\mathrm{I}}}{\partial \bar x}\right)_i^T\right]-\delta\lambda_i^T\left[\bar x_i-\frac{\Delta t_i}{2}\left(\bar f_{\mathrm{I}}\right)_i\right]\\
&\quad-\delta x_{i+1}^T\left[\bar\lambda_i-\frac{\Delta t_i}{2}\left(\frac{\partial \bar H_{\mathrm{I}}}{\partial \bar x}\right)_i^T\right]+\delta\lambda_{i+1}^T\left[\bar x_i+\frac{\Delta t_i}{2}\left(\bar f_{\mathrm{I}}\right)_i\right]\\
&\quad+\delta\bar u_i^T\left[\Delta t_i\left(\frac{\partial \bar H_{\mathrm{I}}}{\partial \bar u}\right)_i^T\right]\Bigg\}+dt_s\left(\hat H_{N_1+1}^- - \hat H_{N_1+1}^+ + \frac{\partial \Phi_s}{\partial t_s}\right)\\
&\quad-\delta x_1^T\hat\lambda_1+\delta\lambda_1^T\hat x_1+\delta x_{N_1+1}^T\left[\left(\frac{\partial \Phi_s}{\partial x}\right)^{+^T}-\left(\frac{\partial \Phi_s}{\partial x}\right)^{-^T}\right]\\
&\quad+\delta\lambda_{N_1+1}^T\left(\hat x_{N_1+1}^+ - \hat x_{N_1+1}^-\right)\\
+&\sum_{i=N_1+1}^{N}\Bigg\{\delta x_i^T\left[\bar\lambda_i+\frac{\Delta t_i}{2}\left(\frac{\partial \bar H_{\mathrm{II}}}{\partial \bar x}\right)_i^T\right]-\delta\lambda_i^T\left[\bar x_i-\frac{\Delta t_i}{2}\left(\bar f_{\mathrm{II}}\right)_i\right]\\
&\quad-\delta x_{i+1}^T\left[\bar\lambda_i-\frac{\Delta t_i}{2}\left(\frac{\partial \bar H_{\mathrm{II}}}{\partial \bar x}\right)_i^T\right]+\delta\lambda_{i+1}^T\left[\bar x_i+\frac{\Delta t_i}{2}\left(\bar f_{\mathrm{II}}\right)_i\right]\\
&\quad+\delta\bar u_i^T\left[\Delta t_i\left(\frac{\partial \bar H_{\mathrm{II}}}{\partial \bar u}\right)_i^T\right]\Bigg\}+dt_f\left(\hat H_{N+1}+\frac{\partial \Phi}{\partial t_f}\right)\\
&\quad-\delta\lambda_{N+1}^T\hat x_{N+1}+\delta x_{N+1}^T\hat\lambda_{N+1}+\delta\nu_s^T\psi_s\big|_{t_s^-}^{t_s^+}+\delta\nu^T\psi\\
&\quad+\delta\eta_x\left\{\hat x_{N_1+1}^- -(-1)^{N_1}\left[\hat x_1+2\sum_{k=1}^{N_1}(-1)^k\bar x_k\right]\right\}\\
&\quad+\delta\eta_\lambda\left\{\hat\lambda_{N_1+1}^- -(-1)^{N_1}\left[\hat\lambda_1+2\sum_{k=1}^{N_1}(-1)^k\bar\lambda_k\right]\right\}=0
\end{aligned}
\tag{4.2-8}
$$

Eq. (4.2–8) may be used to solve optimal control problems of the form specified. Note that for this three-point boundary-value problem the elements must be assembled over the entire time interval. Only the nodal values (the hatted quantities) of the states and costates at t_0, t_s, and t_f appear in the algebraic equations.

Often in practical applications, these algebraic equations may be simplified slightly in order to minimize the number of unknown variables. Consider the case (as will be seen in Chapter 5) where ψ_s is a scalar function at t_s^- and t_s^+, *i.e.*, there is a *known* jump in *one* of the states.

More specifically, since the jump in the states expressed as $\hat{x}_{N_1+1}^- - \hat{x}_{N_1+1}^+$ is known, then the coefficient of $\delta\lambda_{N_1+1}$ will be replaced with an actual numerical value, such as the drop mass of a booster rocket stage. Also, Φ_s^- would be retained to define the unknown staging time, but Φ_s^+ would be set to zero since the jump has already been solved for. Then, the jump in the costates (from the natural boundary conditions) will be defined as

$$\hat{\lambda}_{N_1+1}^- - \hat{\lambda}_{N_1+1}^+ = \frac{\partial \Phi_s}{\partial x}^{-T} \tag{4.2–9}$$

To use Eq. (4.2–8), one would let $\hat{x}_{N_1+1}^-$ and $\hat{\lambda}_{N_1+1}^-$ be the unknown variables and then replace $\hat{x}_{N_1+1}^+$ (from the physical jump condition) and $\hat{\lambda}_{N_1+1}^+$ (from Eq. 4.2–9) in terms of other unknowns. Note that if there is no jump condition on a particular state or costate, then the nodal values just before and after staging are equal. Fig. 4.2 helps clarify this process. In Fig. 4.2 are shown three piecewise constant shape functions spanning three elements. The solid black circles in the top half of the figure represent the hatted or nodal values of the states at the beginning and end of the element. Note that only the jump node is labelled. After assembly, two conditions can occur. If there is no jump, then the nodal values from the left and right sides are equal and do not appear in the algebraic equations. These "disappearing" nodal values are represented by the hollow circles in the lower half of the figure. (There values are recoverable though as described earlier.) If there is a jump, then the nodal values $\hat{x}_{N_1+1}^+$ and $\hat{x}_{N_1+1}^-$ are *not* equal, but only $\hat{x}_{N_1+1}^-$ is treated as an unknown (represented by the solid black dot) and $\hat{x}_{N_1+1}^+$ is eliminated from the equations in terms of $\hat{x}_{N_1+1}^-$. Thus it is now one of the "disappearing" nodal values also and depicted by a hollow circle.

In addition, one must introduce the optimality condition ($\partial H/\partial u = 0$) at t_s^- and t_s^+ in order to solve for the values of the control u just before and after the staging time. Finally, the coefficient of the dt_s equation (*i.e.*, the continuity of the Hamiltonian) gives the extra equation to solve for the unknown staging time t_s.

Also, as was noted in Chapter 2, for most problems the initial conditions are given for all n states and thus, in accordance with Eq. (4.1–5), all the initial costates are unknown. Therefore, instead of treating elements of ν at $t = t_0$ as unknowns and replacing $\hat{\lambda}|_{t_0}$ with these unknowns, we will instead treat $\hat{\lambda}|_{t_0}$ as unknowns and eliminate the $\delta\nu|_{t_0}$ equations from the weak principle.

It may be instructive to count the number of equations and unknowns for a given problem. Consider a problem with n states, m controls, q_1 constraints on the states at t_s and q_2 constraints on the states at t_f. There will be N_1 elements in the first stage and N_2 elements in the second stage. The given boundary conditions will be for $\hat{x}_0$ and $\hat{\lambda}_f$. The number of unknowns is $2n$ (for $\hat{\lambda}_0$ and $\hat{x}_f$) $+ 2n(N_1 + N_2)$ (for $\bar{x}_i$ and $\bar{\lambda}_i$ in the first and second stages) $+ 2n$ (for $\hat{x}_{N_1+1}^-$ and $\hat{\lambda}_{N_1+1}^-$) $+ m(N_1 + N_2)$ (for $\bar{u}_i$ in the first and second stages) $+ 3m$ (for $\hat{u}$ at t_s^-, t_s^+, and t_f) $+ q_1$ (for ν_s) $+ q_2$ (for ν) $+ 2$ (for t_s and t_f). The number of equations is $2n(N_1 + N_2 + 1)$ (for $\delta\bar{x}_i$ and $\delta\bar{\lambda}_i$) $+ 2n$ (for $\delta\eta_x$ and $\delta\eta_\lambda$) $+ m(N_1 + N_2)$ (for $\delta\bar{u}_i$) $+ 3m$ (for $\partial H/\partial u = 0$ at t_s^-, t_s^+, and t_f) $+ q_1$ (for $\delta\nu_s$) $+ q_2$ (for $\delta\nu$) $+ 2$ (for dt_s and dt_f). Thus, the number of equations and unknowns is the same.

Eq. (4.2–8) is actually not much more complicated to program than are the equations presented in Chapter 2 [see Eq. (2.2–6)]. In fact, the one-stage rocket model was modified for the two-stage rocket model presented in the next chapter without any serious complications. One of the most tedious and time-consuming tasks was changing the program to account for the linearization of the new and unknown staging time, as this variable appears in about half of the equations. However, the general code described in Chapter 11 eliminates all the programming difficulties.

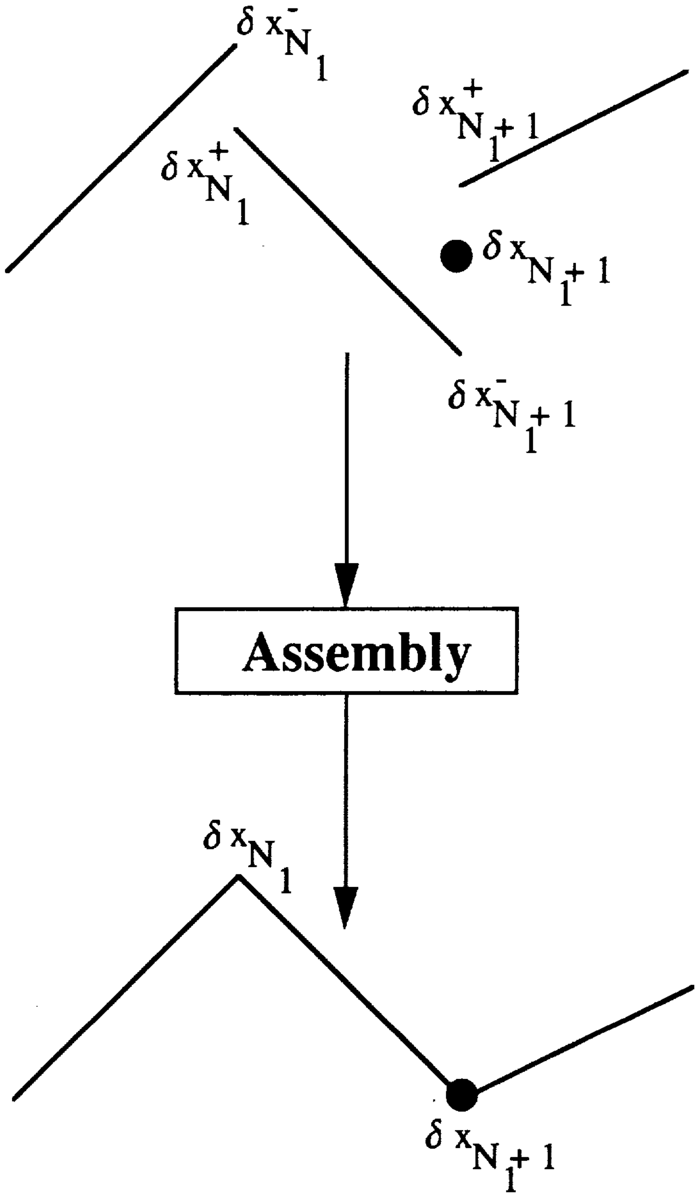

Fig. 4.1: Assembly of virtual quantities

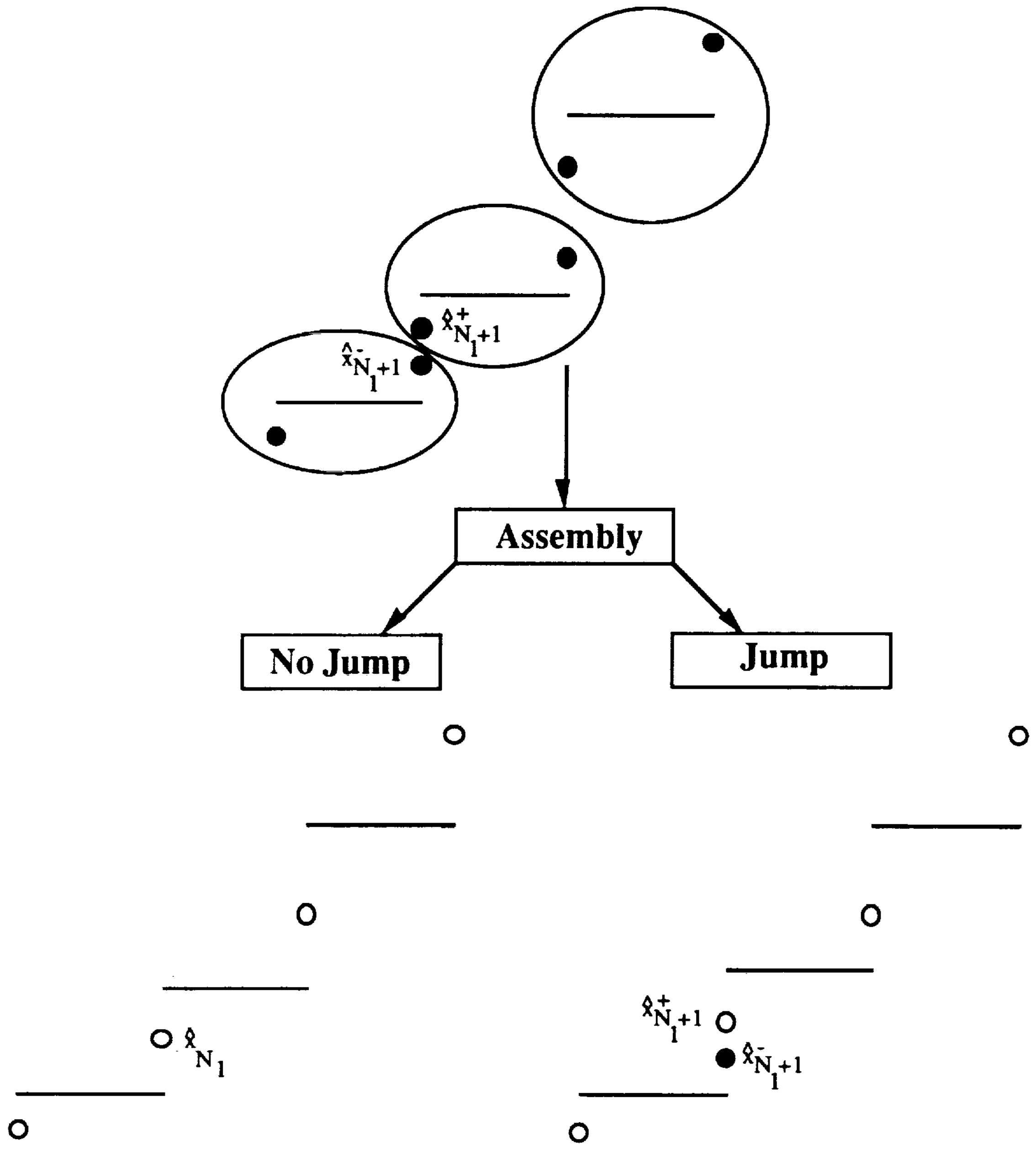

Fig. 4.2: Assembly of states

CHAPTER 5

SATURN TWO-STAGE ROCKET MODEL

In this chapter, a more complicated and realistic model than that of Chapter 3 is presented which is suitable for evaluating the potential usefulness of the weak Hamiltonian finite element approach for real-time guidance of a launch vehicle. A two-stage, four-state vehicle is considered that is simplified by not allowing for any inequality constraints. This model allows us to incorporate the theory developed in Chapters 2 and 4.

5.1. The Model

The same vehicle model from Fig. 3.1 is used. The dynamical equations are

$$\begin{aligned}
\dot{m} &= -\frac{T_{vac}}{9.81 I_{sp}} \\
\dot{h} &= V \sin\gamma \\
\dot{E} &= \left(\frac{T-D}{m}\right) V \\
\dot{\gamma} &= \left(\frac{T + qSC_{L\alpha}}{mV}\right)\alpha + \left(\frac{V}{r} - \frac{\mu}{r^2 V}\right)\cos\gamma
\end{aligned} \tag{5.1–1}$$

where T is the thrust, T_{vac} is the thrust in a vacuum, D is the drag, and V is the velocity. Here α, the angle of attack, has been adopted as a control variable.

The atmospheric, aerodynamic, and propulsion models are taken to be the same as in Eqs. (3.1–2) and (3.1–3). The vehicle parameters chosen for this model are based on a Saturn IB launch vehicle SA-217 [54] and are

$$\begin{aligned} I_{sp_{\rm I}} &= 263.4\,\mathrm{s}; \qquad I_{sp_{\rm II}} = 430.4\,\mathrm{s} \\ T_{vac_{\rm I}} &= 8155800\,\mathrm{N}; \quad T_{vac_{\rm II}} = 1186200\,\mathrm{N} \\ A_{e_{\rm I}} &= 8.47\,\mathrm{m}^2; \qquad A_{e_{\rm II}} = 5.29\,\mathrm{m}^2; \quad S = 33.468\,\mathrm{m}^2 \end{aligned} \tag{5.1-2}$$

where subscripts "I" and "II" refer to the first and second stages respectively.

The aerodynamic coefficient data $C_{N\alpha}$ and C_{D0} are also taken to be the same as the one-stage model. They are presented as functions of the Mach number M in Tables 3.1 and 3.2. The graphs are shown in Figs. 3.2 and 3.3. The physical constants used in the above model are the earth's gravitational constant $\mu = 3.9906 \times 10^{14}$ m^{3}s^{-2}, the earth's mean radius $R_e = 6.378 \times 10^6$ m, the sea-level atmospheric pressure $p_0 = 101320$ Nm^{-2}, the atmospheric pressure at 11 km $p_{11} = 22637$ Nm^{-2}, the sea-level density of air $\rho_0 = 1.225$ kg m^{-3}, and the sea-level speed of sound in air $a_0 = 340.3$ ms^{-1}.

The performance index is

$$J = \phi|_{t_f} = m|_{t_f} \tag{5.1-3}$$

and the final time t_f is open. The initial conditions specified are $m(0) = 5.2 \times 10^5$ kg, $h(0) = 1800$ m, $E(0) = -6.25 \times 10^7$ m^{2}s^{-2}, and $\gamma(0) = 75°$. The burnout mass of the first stage is 192000 kg ($\psi_s^- = \hat{m}(t_s^-) - 192000$) and the drop-mass of the booster is 51000 kg. The final energy is specified as $E(t_f) = -4.25 \times 10^7$ m^{2}s^{-2} ($\psi = \hat{E}(t_f) + 4.25 \times 10^7$).

From Eqs. (4.1–5) and (5.1–3) it is seen that $\hat{\lambda}_f$ is given by $\lfloor 1 \quad 0 \quad \nu_1 \quad 0 \rfloor^T$. Note that the only jumps are in the mass state and the mass costate, and these jumps are

$$\begin{aligned} \hat{m}(t_s^-) - \hat{m}(t_s^+) &= 51000 \\ \hat{\lambda}_m(t_s^-) - \hat{\lambda}_m(t_s^+) &= \nu_s \end{aligned} \tag{5.1-4}$$

5.2. Results

The finite element equations are solved using the method of Levenberg-Marquardt as coded in the IMSL subroutine ZXSSQ [55]. Running a case for a few elements generates a good approximation for larger numbers of elements. Initial guesses do not need to be very accurate, but the method is not nearly as computationally efficient as a Newton-Raphson procedure where sparsity in the Jacobian can be exploited.

Numerical results for the Saturn two-stage model are given in Figs. 5.1 – 5.10. Discrete points are given for 2, 4, and 8 elements in each time interval (denoted by $N = N_1 : N_2$ on the graphs). The converged results corresponding to $N_1 = 8$ and $N_2 = 16$ are shown as solid lines. Note that the number of elements in each interval is completely arbitrary.

Figs. 5.1 – 5.4 show the four states. In Fig. 5.1, notice how nicely the jump in the mass is allowed for by the discretization. Also, we point out that the awkward altitude profile of Fig. 5.2 (*i.e.*, the strange drop at the end of the trajectory) is a result of an unrealistic model. The model is unrealistic due to the absence of inequality constraints, and due to the large angles of attack (more than 30° at some points) even though small angles were assumed in the state equations. However, the model does suffice to illustrate the power of the method.

The four costates are shown in Figs. 5.5 – 5.8. Again the jump is allowed for very accurately by the discretization. Also, note that for the $N = 2 : 2$ case, the jump is actually in the *wrong* direction. Even though this is a very inaccurate result for the mass costate, the $N = 2 : 2$ case is still close enough to the real answer to allow us to interpolate and run higher numbers of elements. This gives some indication of the robustness of the method.

The control (α) is shown in Fig. 5.9. There is a jump in the control at the staging time due to the change in the thrust vector magnitude. The jump is solved for in the program by enforcing the optimality condition at t_s^- and t_s^+. Remember that the optimality condition is used at all nodes to get the control, and this leads to the extra data points on the control profiles.

As an indication of the accuracy of the method in a global sense, the Hamiltonian was observed to converge to zero (the exact answer) all along the trajectory as is seen in Fig. 5.10. The finite element results are converging to the exact solution as N increases.

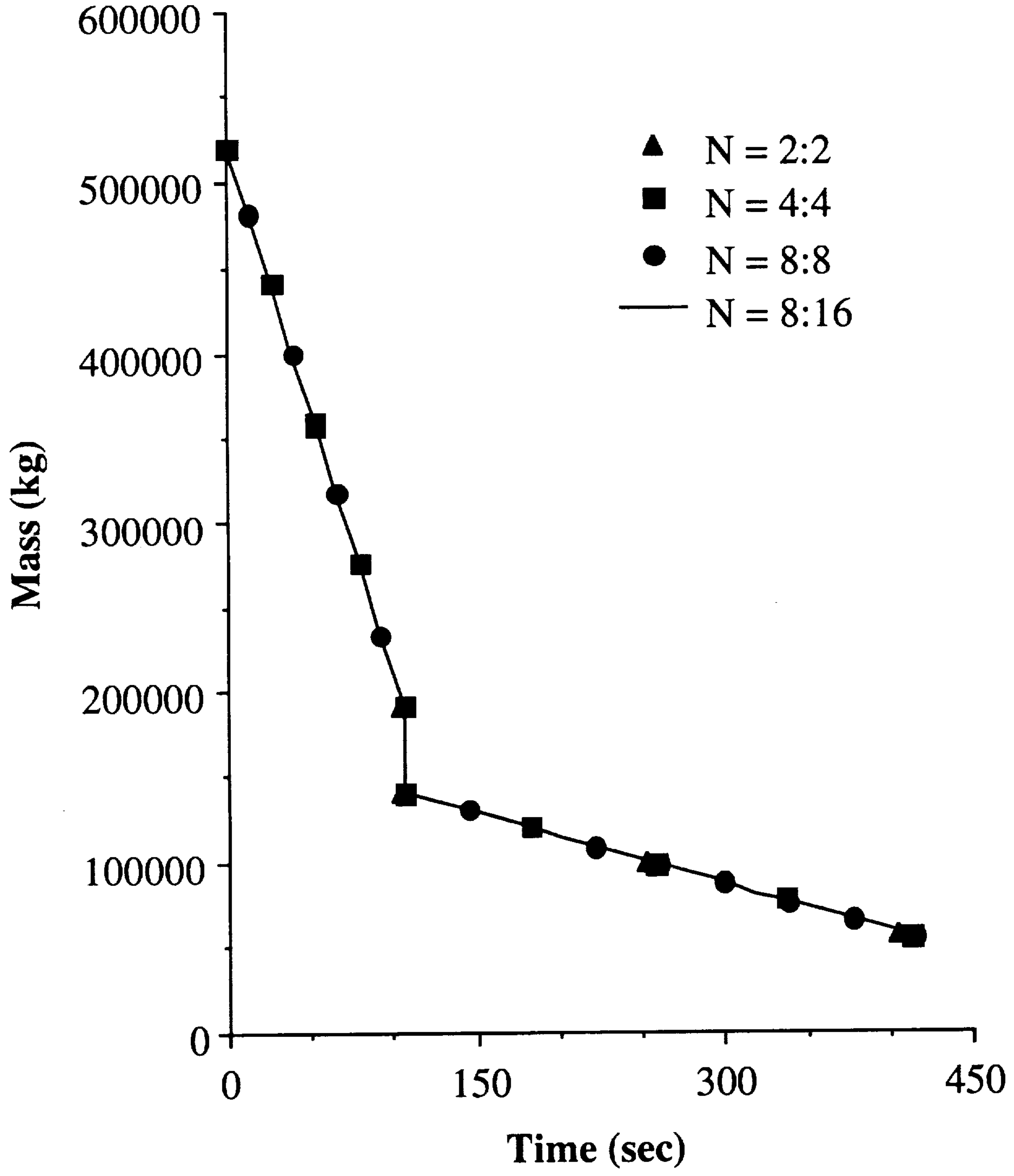

Fig. 5.1: Mass versus time

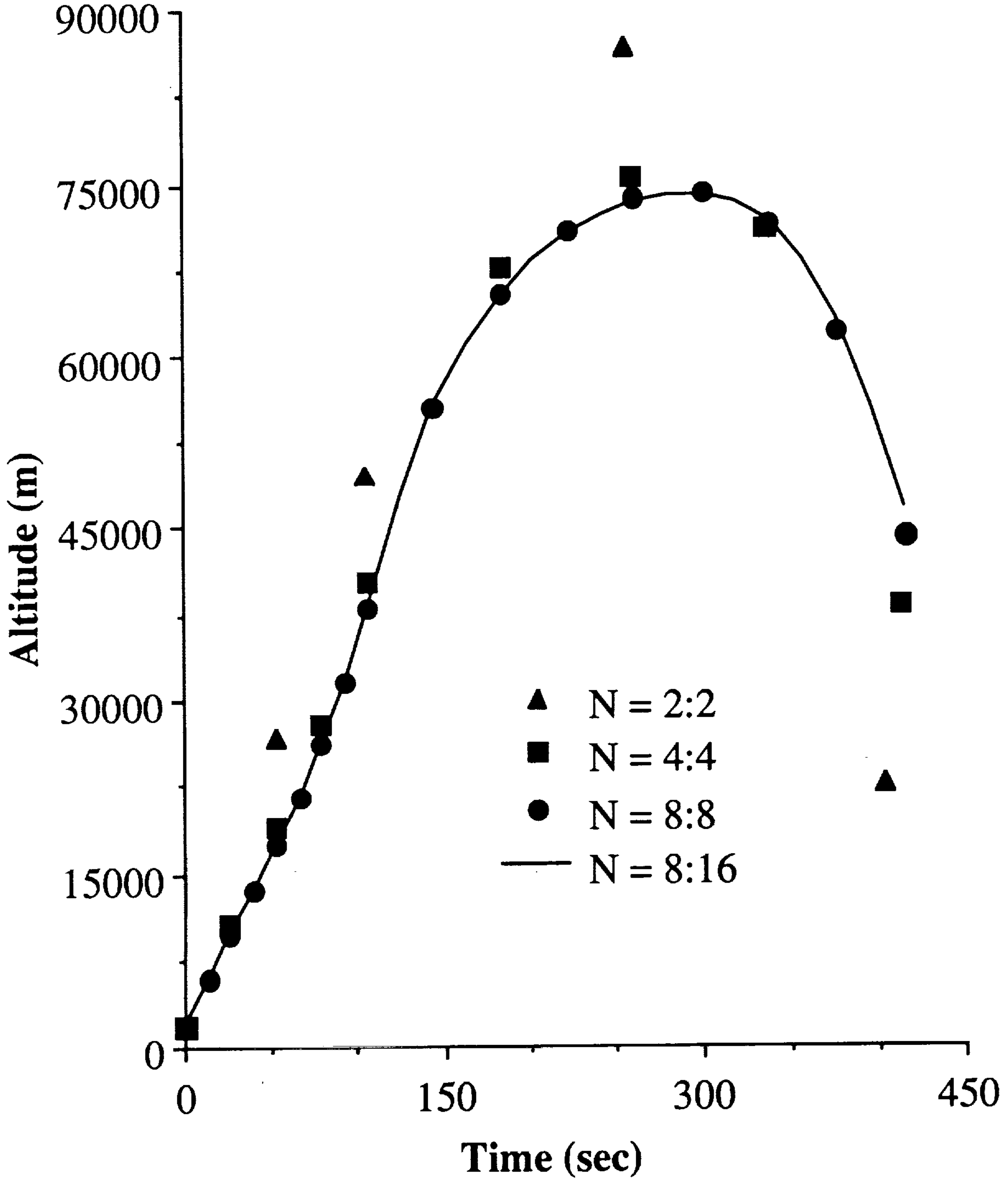

Fig. 5.2: Altitude versus time

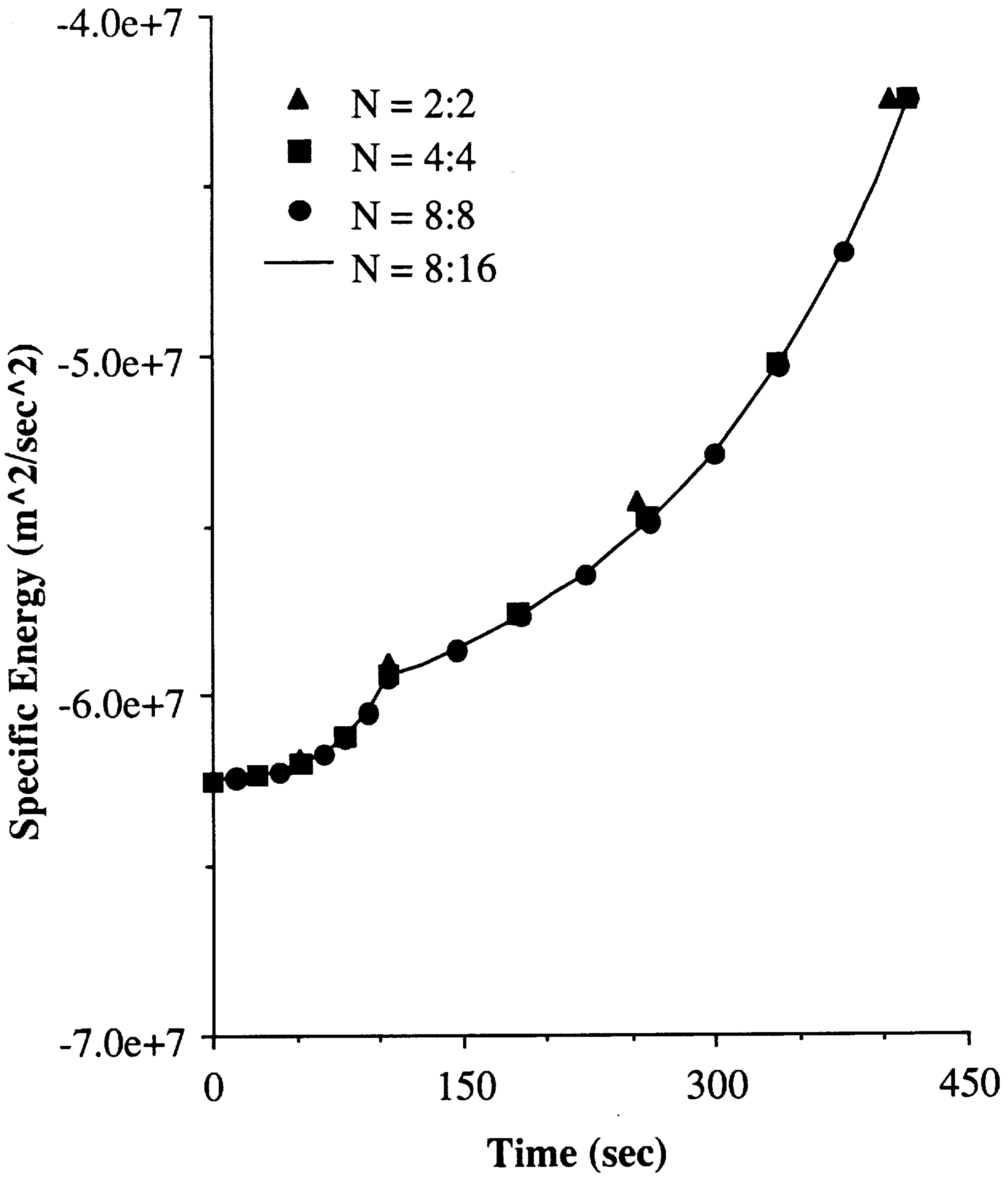

Fig. 5.3: Specific energy versus time

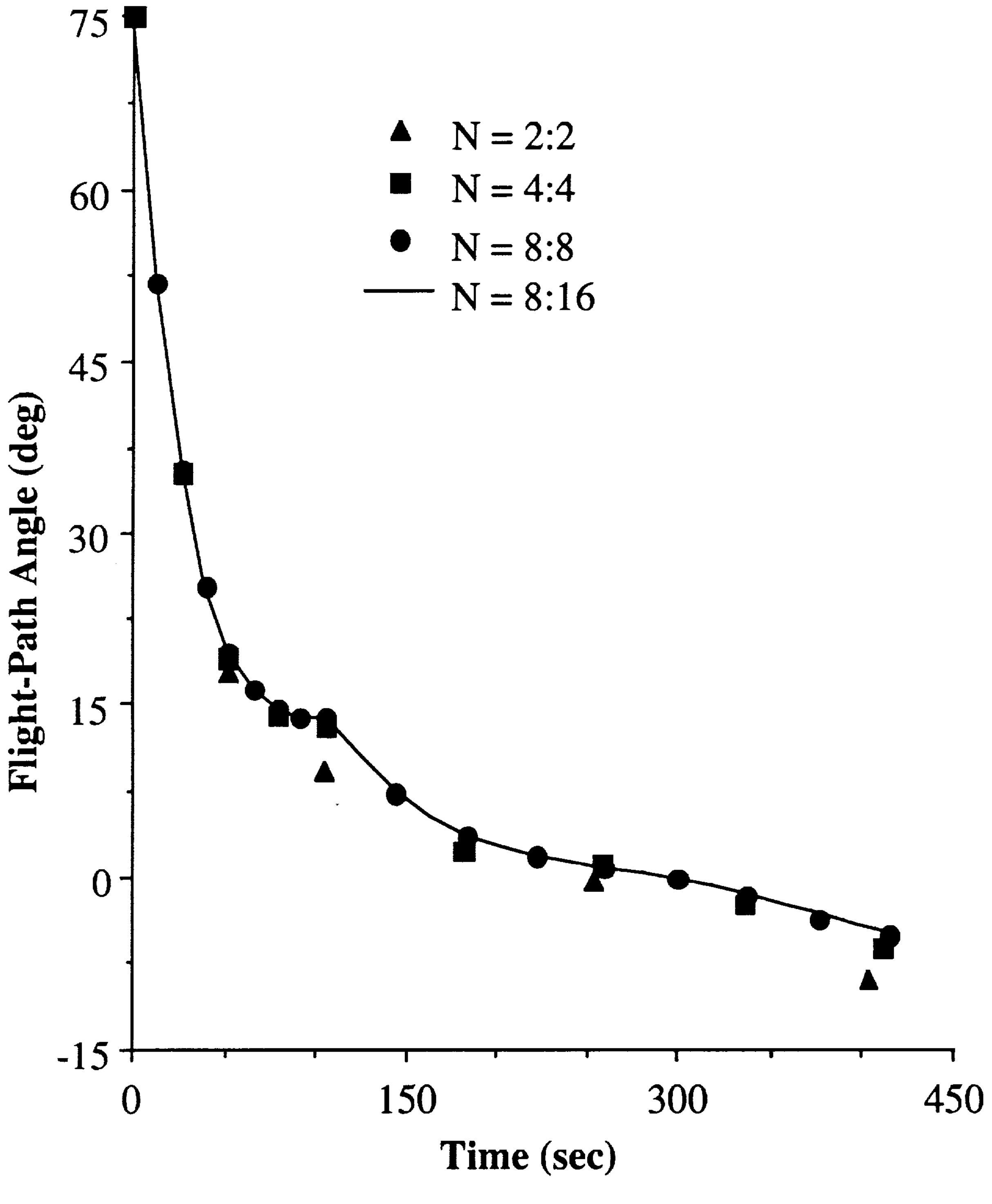

Fig. 5.4: Flight-path angle versus time

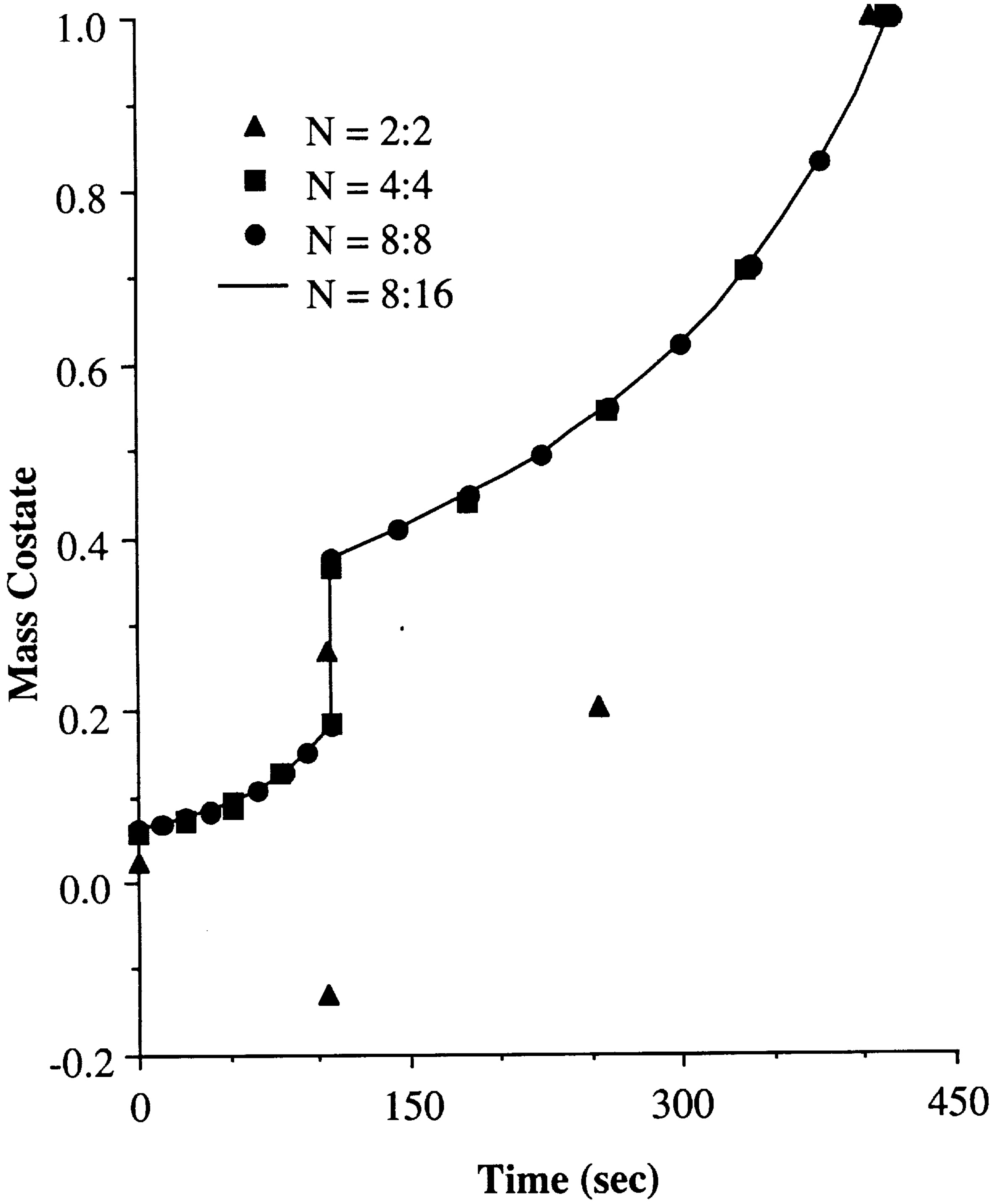

Fig. 5.5: Mass costate versus time

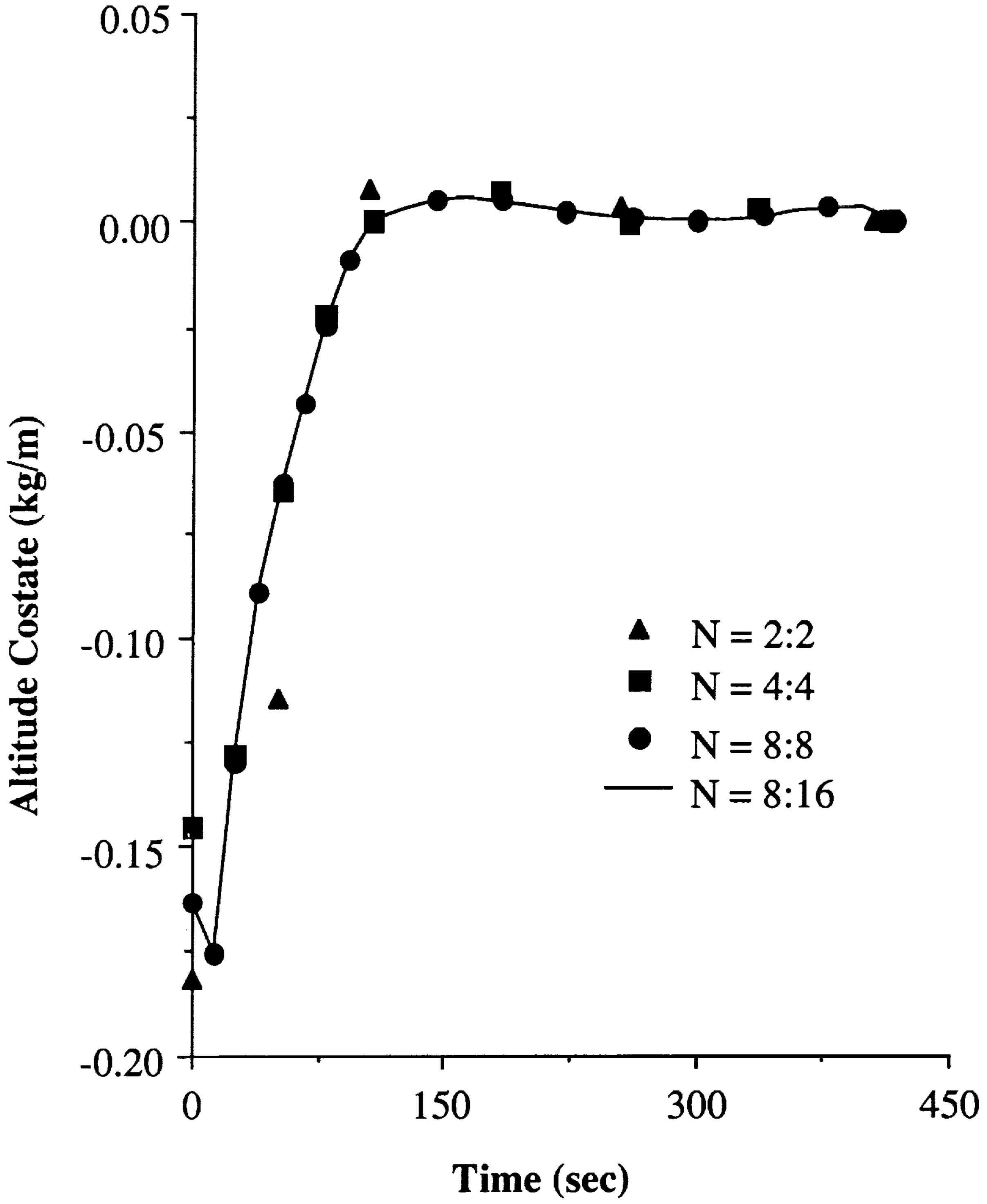

Fig. 5.6: Altitude costate versus time

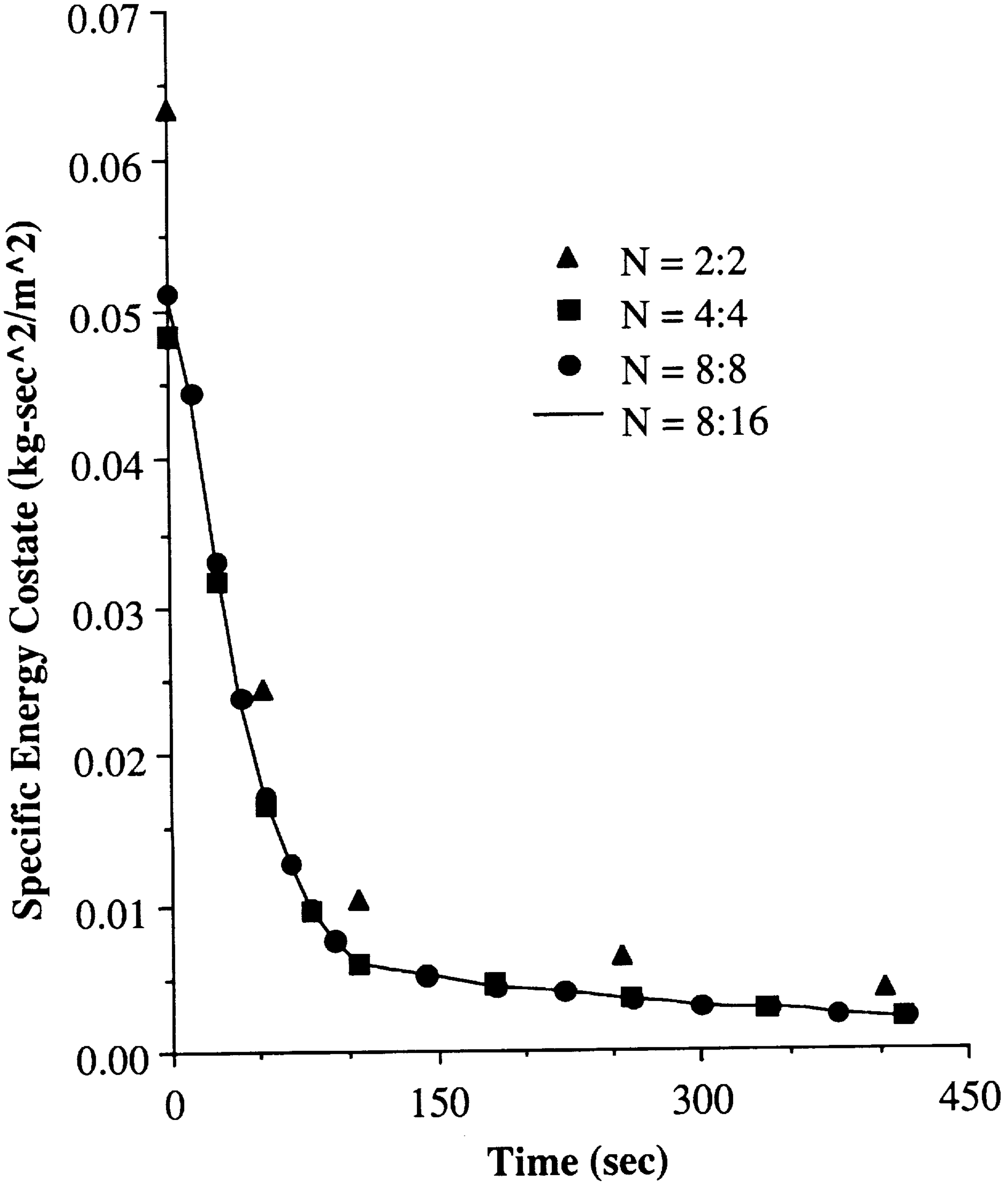

Fig. 5.7: Specific-energy costate versus time

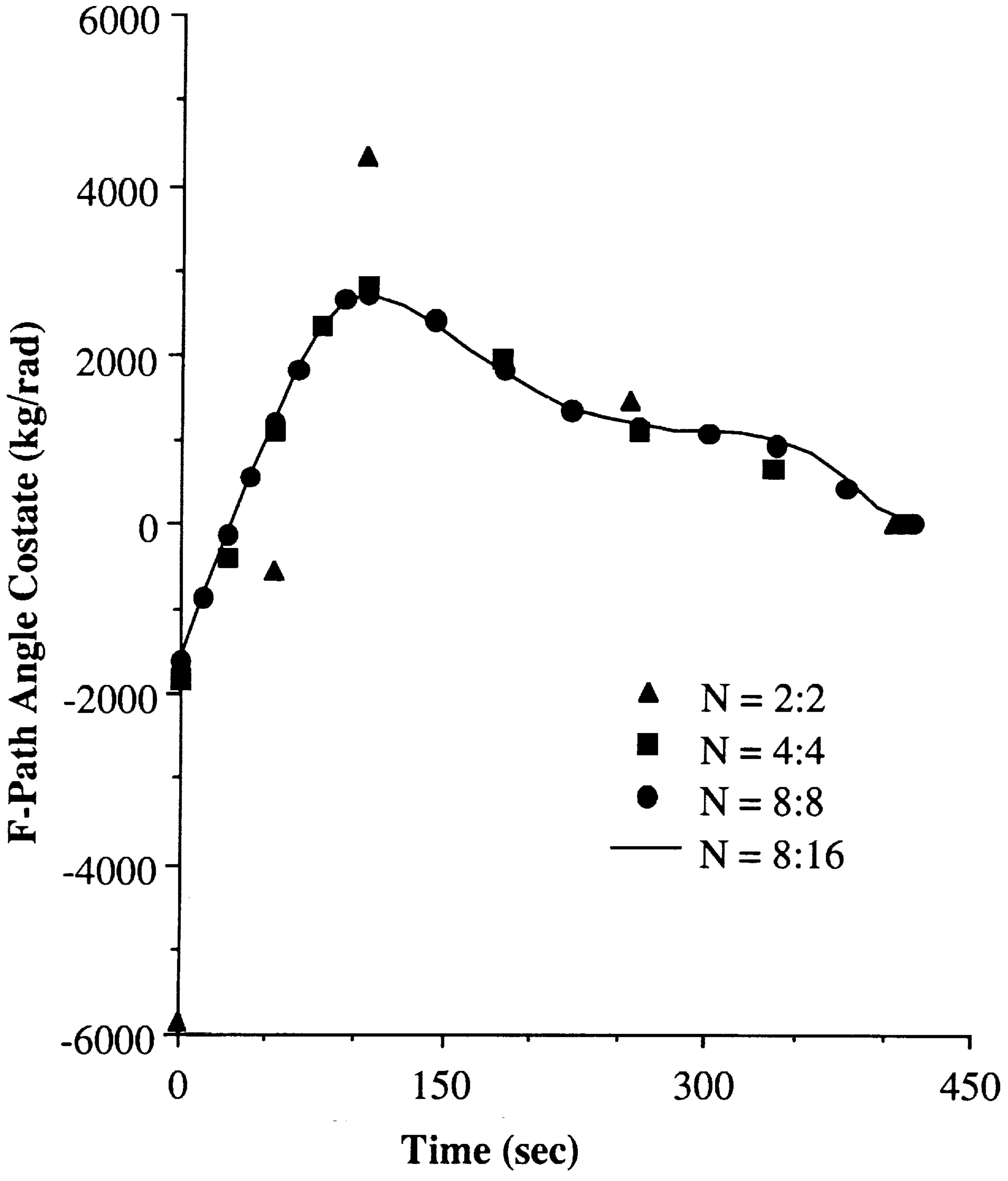

Fig. 5.8: Flight-path angle costate versus time

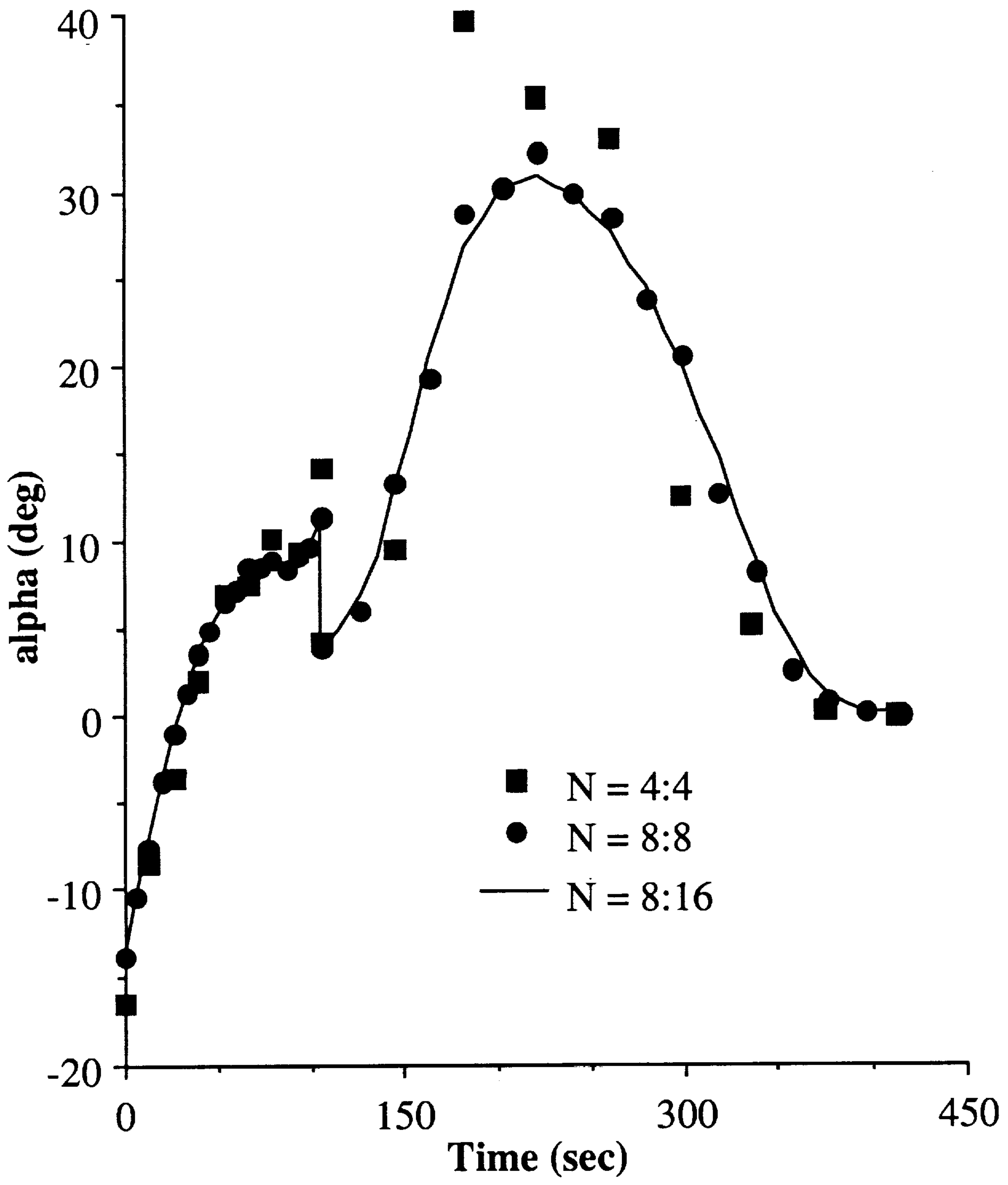

Fig. 5.9: Angle of attack (α) versus time

C-2

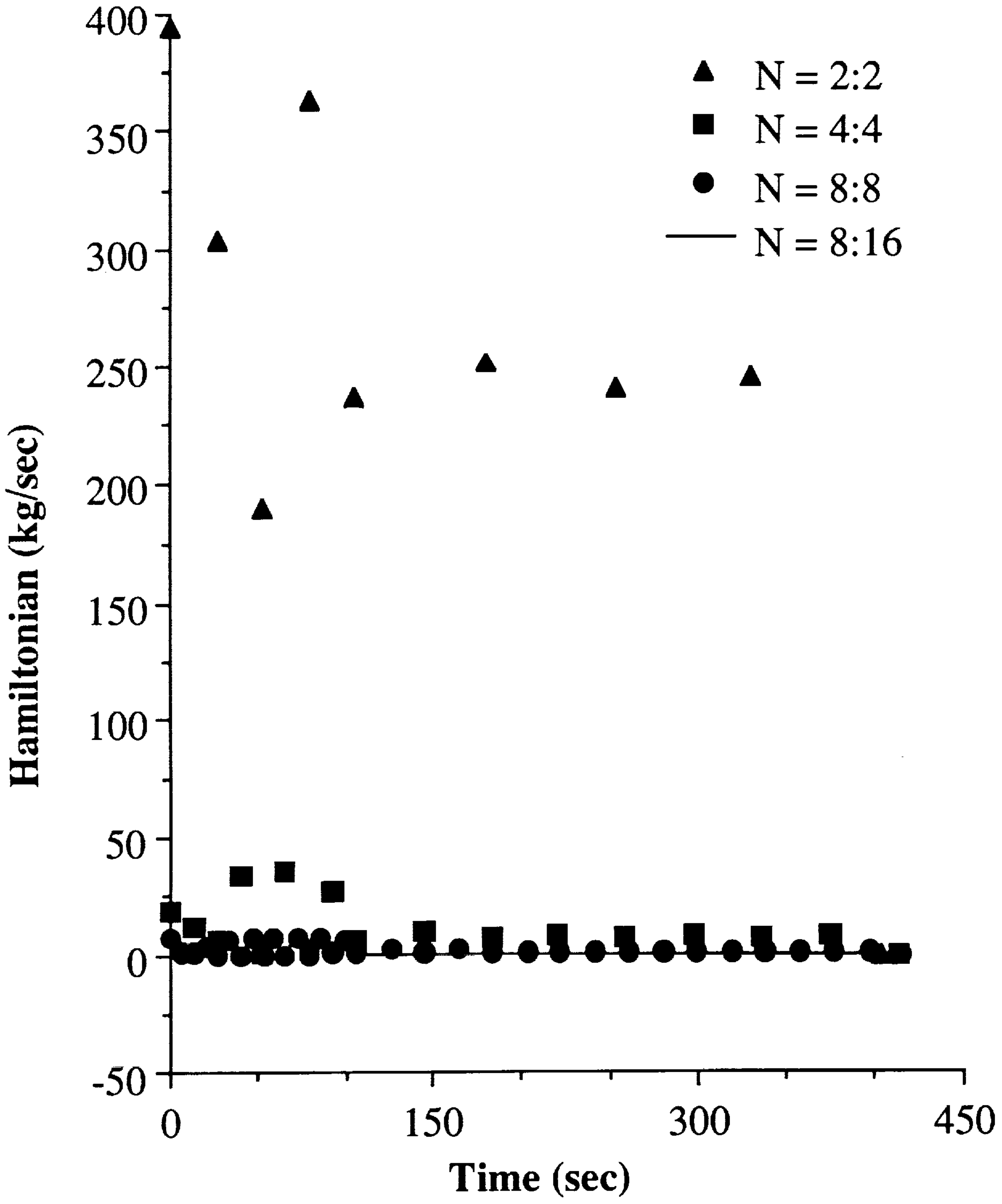

Fig. 5.10: Hamiltonian versus time

CHAPTER 6

CONTROL INEQUALITY CONSTRAINTS

Many practical optimal control problems have certain constraints imposed on the magnitude of the control variables. This is done for a variety of different reasons. For instance, if the control is to be produced by a power supply, then there may be constraints on the controls so that the power supply does not become saturated. Another reason for a control constraint would arise when studying flight vehicles where the structural integrity of the vehicle might be jeopardized by too large a control variable.

The weak principle will now be derived to include problems with control inequality constraints. After the derivation is given, a simple example problem is presented. The numerical results are compared to the exact solution. Of particular interest is the performance in terms of execution time and accuracy versus the number of elements used to represent the time span of the problem.

6.1. General Development

The same problem statement as was given in Chapter 2 is used for this chapter. Namely, consider a system where the states are continuous. Now, suppose that g is a $p \times 1$ column matrix of constraints on the controls of the form

$$g(x, u, t) \leq 0 \tag{6.1–1}$$

One way of handling inequality constraints is to use a "slack" variable [58]. The idea is that if $g \le 0$ then g plus some positive number (*i.e.*, the slack variable) is equal to zero. Thus denoting the slack variable by k^2, then the following $p \times 1$ column matrices for K and δK, the variation of K, may be defined.

$$\begin{aligned} K &= \lfloor k_1^2 \quad k_2^2 \quad \ldots \quad k_p^2 \rfloor^T \\ \delta K &= \lfloor 2k_1\delta k_1 \quad 2k_2\delta k_2 \quad \ldots \quad 2k_p\delta k_p \rfloor^T \end{aligned} \tag{6.1-2}$$

Now, from Eq. (6.1–1)

$$g(x,u,t) + K = 0 \tag{6.1-3}$$

Eq. (6.1–3) will also be adjoined to the performance index J by using p Lagrange multiplier functions

$$\mu(t) = \lfloor \mu_1 \quad \mu_2 \quad \ldots \quad \mu_p \rfloor^T \tag{6.1-4}$$

The performance index now takes the form:

$$J = \int_{t_0}^{t_f} \left[L(x,u,t) + \lambda^T(f - \dot{x}) + \mu^T(g + K) \right] \, dt + \Phi\big|_{t_0}^{t_f} + \alpha^T(x - \hat{x})\big|_{t_0}^{t_f} \tag{6.1-5}$$

To derive our weak principle, it is necessary to take the first variation of J and set it equal to zero. For simplicity, the derivation below is for the case of fixed final-time. The case for free final-time is discussed after this derivation. For notational convenience, the following variables are introduced.

$$\hat{\lambda}_0 = \left.\frac{\partial \Phi}{\partial x}\right|_{t_0} \quad \text{and} \quad \hat{\lambda}_f = \left.\frac{\partial \Phi}{\partial x}\right|_{t_f} \tag{6.1–6}$$

Also, as is shown in Section 2.1, the Lagrange multiplier α can be chosen so that $\delta\alpha = \delta\lambda$.

The first variation of J is

$$\begin{aligned}\delta J = \int_{t_0}^{t_f} \Bigg\{ &\delta\lambda^T(f - \dot{x}) - \delta\dot{x}^T\lambda + \delta x^T \left[\left(\frac{\partial L}{\partial x}\right)^T + \left(\frac{\partial f}{\partial x}\right)^T \lambda + \left(\frac{\partial g}{\partial x}\right)^T \mu \right] \\ &+ \delta u^T \left[\left(\frac{\partial L}{\partial u}\right)^T + \left(\frac{\partial f}{\partial u}\right)^T \lambda + \left(\frac{\partial g}{\partial u}\right)^T \mu \right] + \delta\mu^T(g + K) + \delta K^T \mu \Bigg\} dt \\ &+ \delta\nu^T\psi\Big|_{t_0}^{t_f} + \delta x^T\hat{\lambda}\Big|_{t_0}^{t_f} + \delta\lambda^T(x - \hat{x})\Big|_{t_0}^{t_f} + \alpha^T(\delta x - \delta\hat{x})\Big|_{t_0}^{t_f} = 0\end{aligned} \tag{6.1–7}$$

The admissible variations to the states must be continuous at the initial and final times and therefore $(\delta x - \delta\hat{x})|_{t_0}^{t_f} = 0$. Furthermore, it is noted that for most problems, the initial conditions are given for all n states and thus, in accordance with Eq. (6.1–6), all the initial costates are unknown. Therefore, instead of treating elements of ν at $t = t_0$ as unknowns and replacing $\hat{\lambda}|_{t_0}$ with these unknowns, $\hat{\lambda}|_{t_0}$ will be treated as unknowns and the $\delta\nu|_{t_0}$ equations will be eliminated from the weak principle. Finally, the weak principle is obtained by integrating the $\dot{x}$ term in Eq. (6.1–7) by parts. Denoting the variations of the variables at the initial and final times with subscripts 0 and f, then the resulting equation is

$$\int_{t_0}^{t_f} \left\{ -\delta \dot{x}^T \lambda + \delta \lambda^T f + \delta \dot{\lambda}^T x + \delta x^T \left[\left(\frac{\partial L}{\partial x}\right)^T + \left(\frac{\partial f}{\partial x}\right)^T \lambda + \left(\frac{\partial g}{\partial x}\right)^T \mu \right] \right.$$
$$\left. + \delta u^T \left[\left(\frac{\partial L}{\partial u}\right)^T + \left(\frac{\partial f}{\partial u}\right)^T \lambda + \left(\frac{\partial g}{\partial u}\right)^T \mu \right] + \delta \mu^T (g + K) - \delta K^T \mu \right\} dt \tag{6.1-8}$$
$$+ \delta \nu^T \psi \Big|_{t_f} + \delta x_f^T \hat{\lambda}_f - \delta x_0^T \hat{\lambda}_0 - \delta \lambda_f^T \hat{x}_f + \delta \lambda_0^T \hat{x}_0 = 0$$

This is the governing equation for the weak Hamiltonian method for fixed-time problems with control inequality constraints. It is easily shown by integrating the $\delta \dot{x}$ and $\delta \dot{\lambda}$ terms by parts in Eq. (6.1–8) that all the Euler-Lagrange equations are the same as in [1] and that all boundary conditions are now of the natural type. When the final time is allowed to vary, Eq. (6.1–8) remains unchanged except that one term is added, given by

$$\delta t_f \left(L + \lambda^T f + \frac{\partial \phi}{\partial t} + \nu^T \frac{\partial \psi}{\partial t} \right) \Bigg|_{t_f} \tag{6.1-9}$$

Note that in this term, values for u are required at t_f. To obtain u, one must also find the values of K and μ at t_f. These unknowns are found by setting the coefficients of δu_f, δK_f^T, and $\delta \mu_f^T$ equal to zero in the following

$$\begin{aligned} &\delta u_f^T \left[\left(\frac{\partial L}{\partial u}\right)^T + \left(\frac{\partial f}{\partial u}\right)^T \lambda + \left(\frac{\partial g}{\partial u}\right)^T \mu \right] \Bigg|_{t_f} \\ &\delta \mu_f^T (g + K) \Big|_{t_f} \\ &\delta K_f^T \mu \Big|_{t_f} \end{aligned} \tag{6.1-10}$$

Thus, the formulation has now been developed to handle fixed and free-time problems with inequality constraints on the control.

6.2. Finite Element Discretization

The same finite element discretization described in Chapter 2 is used for this problem. The only added step is to define shape functions for K and μ. Since the time derivatives of K, μ, δK, and $\delta\mu$ do not appear in the formulation,then

$$\begin{aligned} K &= \bar{K}_i & \mu &= \bar{\mu}_i \\ \delta K &= \delta\bar{K}_i & \delta\mu &= \delta\bar{\mu}_i \end{aligned} \tag{6.2–1}$$

By substituting Eq. (4.2–1) and the shape functions described above into Eq. (6.1–8), and carrying out the element quadrature over τ from 0 to 1, a general algebraic form of the Hamiltonian weak principle is obtained. Again, if the time t does not appear explicitly in the problem formulation then all integration is *exact* and can be done by inspection. If t does appear explicitly, then t may be approximated by a constant value over each element and the integration may still be done by inspection. The latter case occurs in the example problem presented shortly. For N elements, there are $2n(N+1)+mN+q+2Np$ equations and $2n(N+2)+mN+q+2Np$ unknowns. Therefore, $2n$ of the $4n$ endpoint values for the states and costates ($\hat{x}_0, \hat{\lambda}_0, \hat{x}_f$, and $\hat{\lambda}_f$) must be specified. In general, $\hat{x}_0$ (the initial conditions) is known in accordance with physical constraints. Also, $\hat{\lambda}_f$ can be specified in terms of other unknowns with the use of Eq. (6.1–6). Now there are the same number of equations as unknowns. These equations may be used for any optimal control problem of the form specified.

Although the shape functions for u, K, and μ only define a constant value within the element, values for these variables at additional points are also available. For instance, once the nodal values for the states and costates are found, then one may use the optimality condition ($\partial H/\partial u = 0$), the constraint equations ($g + K = 0$), and the condition that either k or μ be zero ($k\mu = 0$) to solve for u, K, and μ at a nodal point. This procedure is used in the following example problem.

6.3. Example

This example is taken from section 3.8 of [1]. The problem is to minimize

$$J = \frac{1}{2}x(T)^2 + \frac{1}{2}\int_0^T u^2\,dt \tag{6.3–1}$$

where $T = 10$, x and u are scalars, and the initial condition is $\hat{x}_0 = -19.945596$. The state equation is

$$\dot{x} = h(t)u \qquad \text{with} \qquad h(t) = 1 + t - \frac{3}{17}t^2 \tag{6.3–2}$$

The following two control inequality constraints are imposed.

$$\begin{aligned} g_1 &= u - 1 \le 0 \\ g_2 &= -(u+1) \le 0 \end{aligned} \tag{6.3–3}$$

The exact solution is found to be $x(T) = -17/39$ and for the control

$$u(t) = \begin{cases} -x(T)h(t) & \text{for} \quad 0 \le t \le 2 \\ 1 & \text{for} \quad 2 \le t \le 11/3 \\ -x(T)h(t) & \text{for} \quad 11/3 \le t \le 8 \\ -1 & \text{for} \quad 8 \le t \le 10 \end{cases} \tag{6.3–4}$$

The algebraic equations which come from the weak principle can be verified to be

$$\begin{aligned} \hat{\lambda}_0 - \bar{\lambda}^{(1)} &= 0 \\ \bar{\lambda}^{(i)} - \bar{\lambda}^{(i+1)} &= 0 \qquad \text{for} \qquad i = 1, 2, \ldots, N-1 \\ \bar{\lambda}^{(N)} - \hat{x}_f &= 0 \end{aligned} \tag{6.3–5}$$

for the δx coefficients,

$$\left.\begin{aligned} \bar{u}^{(i)} + \bar{\lambda}^{(i)} h\left[\bar{t}^{(i)}\right] + \mu_1^{(i)} - \mu_2^{(i)} &= 0 \\ k_1^{(i)}\mu_1^{(i)} &= 0 \\ k_2^{(i)}\mu_2^{(i)} &= 0 \\ \bar{u}^{(i)} - 1 + k_1^{(i)^2} &= 0 \\ -\bar{u}^{(i)} - 1 + k_2^{(i)^2} &= 0 \end{aligned}\right\} \qquad \text{for} \qquad i = 1, 2, \ldots, N \qquad (6.3\text{–}6)$$

for the δu, δK, and $\delta\mu$ coefficients, and

$$\begin{aligned} \bar{x}^{(1)} - \frac{\Delta t}{2} h\left[\bar{t}^{(1)}\right]\bar{u}^{(1)} &= \hat{x}_0 \\ \bar{x}^{(i+1)} - \bar{x}^{(i)} - \frac{\Delta t}{2}\left\{h\left[\bar{t}^{(i)}\right]\bar{u}^{(i)} + h\left[\bar{t}^{(i+1)}\right]\bar{u}^{(i+1)}\right\} &= 0 \quad \text{for} \quad i = 1, 2, \ldots, N-1 \\ -\bar{x}^{(N)} - \frac{\Delta t}{2} h\left[\bar{t}^{(N)}\right]\bar{u}^{(N)} + \hat{x}_f &= 0 \end{aligned} \qquad (6.3\text{–}7)$$

for the $\delta\lambda$ coefficients. Note that $\bar{t}^{(i)}$ is an average time value for the i^{th} element and (if $\Delta t_i = \Delta t = t_f/N$ for all i) can be expressed as

$$\bar{t}^{(i)} = \frac{2i-1}{2}\Delta t \qquad \text{for} \qquad i = 1, 2, \ldots, N \qquad (6.3\text{–}8)$$

Recall from [1] that one of the additional necessary conditions for problems with control constraints is that the multipliers be greater than or equal to zero for a minimizing problem. Therefore, in practice, the multipliers μ appearing in the first of Eqs. (6.3–6) are squared to ensure their positivity. Further recall that if the constraint is not violated, then $\mu = 0$. This condition is satisfied by the second and third equation in Eq. (6.3–6) which implies that either k or μ is zero for each element.

It is readily apparent from the equations shown in Eq. (6.3–5) that all the costate variables are equal to $\hat{x}_f$. Therefore, these equations were eliminated and all costates that occurred in the remaining equations were replaced with $\hat{x}_f$. The remaining $6N + 1$ algebraic equations were solved using a Newton-Raphson method and a FORTRAN code written on a SUN 3/260. The sparse, linearized equations are solved using subroutine MA28 from the Harwell subroutine library [59]. This subroutine takes advantage of sparsity which leads to great computational savings.

Table 6.1 shows the convergence rate of $\hat{x}_f = x(T)$, the elapsed computer time for the first 5 iterations, and the percentage of zeroes in the Jacobian (*i.e.*, the sparsity) versus the number of elements. The $x(T)$ column shows that the 32 element case has almost converged on the exact solution. Note further that the approximate $x(T)$ is not an upper bound of the exact value, which is common in mixed formulations. The third column of Table 6.1 gives the elapsed computer time for five iterations. It is easily seen that there is a modest increase in computer time with an increase in the number of elements. Note that in some cases a converged answer is found in five or fewer iterations. This is because the answers obtained from a small number of elements (say 2 or 4) may be interpolated to generate initial guesses for a higher number of elements. Thus, it is possible to solve a 16 or 32 element case in about 1.5 seconds. Finally, the extremely sparse structure of the Jacobian is demonstrated in the last column. This strongly encourages the use of a "smart" sparse matrix solver such as MA28. This subroutine leads to quicker solutions and tremendous savings in memory allocation since only the nonzeroes of the Jacobian need be stored.

Table 6.1: $x(T)$, elapsed computer time and percent sparsity of Jacobian versus the number of elements N

N	$x(T)$	Time (sec)	Sparsity (%)
1	−4.0632	0.42	65.3
2	−.82795	0.44	80.5
4	−.44065	0.66	89.6
8	−.43360	0.76	94.6
16	−.43928	1.03	97.3
32	−.43588	1.52	98.6
Exact	−.43590	−−	−−

Results for the control u are shown in Fig. 6.1 for 2, 4, and 8 elements and the exact solution. Note that although the 2 element case does not define the constraint boundaries very accurately, it is accurate enough to generate guesses for the 4 element case. Thus, in a problem with many constrained and unconstrained arcs, a small number of elements could still be used to generate guesses for a higher number of elements. Also, it is interesting to note that as few as 4 elements have essentially converged on the exact solution.

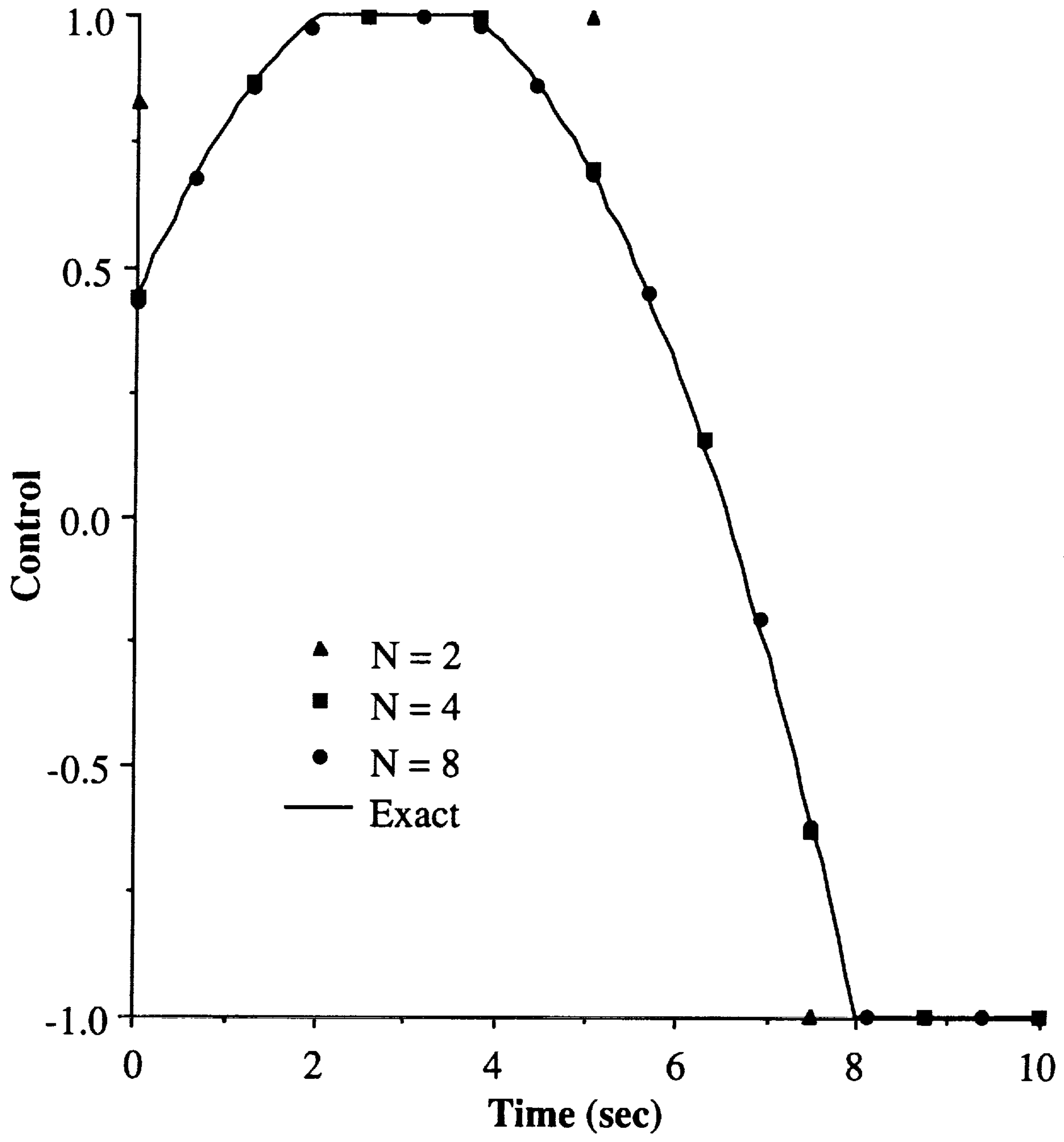

Fig. 6.1: Control versus time

CHAPTER 7

STATE INEQUALITY CONSTRAINTS

Optimal control problems with state inequality constraints are in general very difficult to solve. Although the problem does not seem more difficult conceptually than problems with control inequality constraints, there are additional necessary conditions to reckon with. These additional necessary conditions are a result of what are sometimes referred to as "tangency" conditions [1]. These tangency conditions arise from the following physical considerations.

If the constraints are of the form $S(x,t) \leq 0$, then successive total time derivatives of S are taken and $f(x,u,t)$ is substituted for $\dot{x}$ until an expression explicitly dependent on the control u is obtained. If p total time derivatives are required, then S is called a pth-order state variable inequality constraint. Now, since $S(x,t)$ can be controlled only by changing its pth time derivative, no finite control will keep the system on the constraint boundary if the path entering onto the constraint boundary does not meet the "tangency" conditions. These conditions are that S and all the time derivatives of S up to $p-1$ are zero. These conditions also apply to the path leaving the constraint boundary.

There have been several papers over the past 30 years presenting necessary conditions of optimality for optimal control problems with state-variable inequality constraints. In [60], the authors adjoined the pth derivative of the constraint to the performance index and allowed the tangency conditions stated above to form a set of interior boundary conditions. These boundary conditions require discontinuities in the costates at the junction points between constrained and unconstrained arcs.

However, one may arbitrarily pick the entry point as the place to satisfy these boundary conditions, and therefore the costates and Hamiltonian are discontinuous at the entry point and continuous at the exit point.

A new set of necessary conditions was given in [61]. Therein, the constraint was adjoined directly to the performance index. It was shown that unconstrained arcs had to satisfy tangency constraints at *both* ends of a constrained arc.

Solution of problems with state constraints may be handled in a variety of ways. For example, a penalty function approach was presented in [62]. A Valentine transformation technique (as was used in Chapter 6) was presented in [63]. This idea transformed a constrained problem into an equivalent unconstrained problem by adding additional state equations. Also, the control variable is transformed and the new control appears linearly in the formulation. The weak principle cannot handle linearly appearing controls, so this idea was not used. Finally, [64] demonstrates how a state constraint present in the full-order problem may be transformed to a control constraint in the reduced-order problem.

New necessary conditions for optimal control problems with state inequality constraints were developed in [65]. Therein, the authors tactfully say: "We do not imply that the necessary conditions obtained by previous workers are incorrect, but rather, that, inasmuch as they underspecify the conditions at the junction, there exists the possibility of non-stationary solutions satisfying these conditions ...." The authors generated an admittedly contrived example where the necessary conditions of [1], which are identical to those in [60], were satisfied by a non-extremal solution. These new necessary conditions are summarized below.

The new necessary conditions derived in [65] may be summarized as follows. Rather than take successive time derivatives of the constraint $S(x,t)$ until the control appears explicitly, the authors adjoin S directly to the Hamiltonian, as was done with control inequality constraints in Chapter 6. The Hamiltonian now takes the form

$$H = L + \lambda^T f + \eta S(x,t) \tag{7–1}$$

where η, S, and u are scalars.

As with control inequality constraints, the multiplier η is positive if $S = 0$ and zero if $S \leq 0$. At junction points t_i of boundary and interior arcs, the costates may be discontinuous. The boundary conditions are

$$\lambda(t_i^+) = \lambda(t_i^-) - \nu(t_i)\left(\frac{\partial S}{\partial x}\right)_{t_i} ; \qquad \nu(t_i) \geq 0 \tag{7–2}$$

and, in addition,

$$H(t_i^+) = H(t_i^-) - v\frac{\partial S}{\partial t_i} \tag{7–3}$$

An extremely interesting consequence of the new conditions is pointed out in the paper for problems which possess a Hamiltonian which is said to be *regular*. The Hamiltonian H is regular if along a given trajectory, H has a unique minimum. In this case, it was shown that the control u and its $(p-2)$ time derivatives are all continuous. Now, the interesting consequence of the new conditions is that for an odd-order constraint greater than two, the trajectory will, at most, only touch the boundary if the $(p-1)$th derivative of u is discontinuous at the junction point. Note that for $p = 1$, the control is continuous, so that boundary arcs *are* permitted for the first-order case.

7.1. General Development

Consider once again a system as defined in Chapter 2. Now, suppose that there is a pth order scalar constraint on the states and time defined by $S(x,t) \leq 0$. The first attempt to apply the present methodology to problems with state inequality constraints made use of the necessary conditions presented in [65]. These necessary conditions lead to successful and accurate solution strategies for states that only touch (*i.e.*, do not ride) the constraint boundary. As is derived in [65], for constraints of odd order greater than one, the solution can at most only touch the constraint boundary if the Hamiltonian is regular. However, for cases where the states ride the constraint boundaries for a nonzero length of time, the algebraic equations developed by the weak form are singular. Private discussions with Jason Speyer and Dan Moerder indicate that the cause is related to a reduced-dimensional manifold; however, we have not been able to develop a nonsingular weak form as of now.

Below are presented two very similar weak formulations using the necessary conditions of [65] for touch-point cases and [1] for ride cases. Fortunately, the necessary conditions presented in [1] are accurate for first and second order constraints where the solution often rides the constraint boundary and the conditions in [65] are accurate for the touch-point cases. It is noted that most practical applications will be third-order or less.

7.1.1. Touch-Point Cases

The weak formulation is now derived for touch-point cases. Assume that there is only one touch-point over the time interval of interest whose time will be denoted by t_{tp}. In this case, the state constraint is nothing more than an interior boundary point which creates a jump in the costate.

The performance index J now takes the form:

$$J = \int_{t_0}^{t_{tp}} \left[L(x,u,t) + \lambda^T(f - \dot{x})\right] \, dt + \int_{t_{tp}}^{t_f} \left[L(x,u,t) + \lambda^T(f - \dot{x})\right] \, dt \\ + \nu_1 S|_{t_{tp}} + \Phi|_{t_0}^{t_f} + \alpha^T(x - \hat{x})|_{t_0}^{t_f} \tag{7.1.1-1}$$

To derive the weak principle, it is necessary to take the first variation of J and set it equal to zero. This variation, and the entire development of the weak principle is almost identical to the derivation given in Chapter 4. The only difference is that the state equations are the same on either side of the constraint. As usual, we introduce

$$\hat{\lambda}_0 = \left.\frac{\partial \Phi}{\partial x}\right|_{t_0} \quad \text{and} \quad \hat{\lambda}_f = \left.\frac{\partial \Phi}{\partial x}\right|_{t_f} \tag{7.1.1-2}$$

Also, as is shown in Chapter 4, the Lagrange multiplier α can be chosen so that $\delta\alpha = d\lambda$. The final form of the weak principle is obtained after integrating by parts so that no derivatives of the states or costates appear. After defining the Hamiltonian $H = L + \lambda^T f$ and denoting the variations of the variables at the initial, touch-point, and final times with subscripts 0, 1, and f respectively, then the resulting equation is

$$\int_{t_0}^{t_{tp}} \left[-\delta\dot{x}^T\lambda + \delta\lambda^T f + \delta\dot{\lambda}^T x + \delta x^T \left(\frac{\partial H}{\partial x}\right)^T + \delta u^T \left(\frac{\partial H}{\partial u}\right)^T\right] dt \\ + \int_{t_{tp}}^{t_f} \left[-\delta\dot{x}^T\lambda + \delta\lambda^T f + \delta\dot{\lambda}^T x + \delta x^T \left(\frac{\partial H}{\partial x}\right)^T + \delta u^T \left(\frac{\partial H}{\partial u}\right)^T\right] dt \\ + \delta\nu_1^T S\Big|_{t_{tp}} + \delta\nu^T\psi\Big|_{t_f} + \delta x_1^T \left(\frac{\partial S}{\partial x}\right)^T \nu_1 \\ + \delta x_f^T\hat{\lambda}_f - \delta x_0^T\hat{\lambda}_0 - \delta\lambda_f^T\hat{x}_f + \delta\lambda_0^T\hat{x}_0 \\ + dt_{tp}\left[H(t_{tp}^-) - H(t_{tp}^+) + \nu_1\frac{\partial S}{\partial t}\right] + dt_f\left[H(t_f) + \frac{\partial \Phi}{\partial t_f}\right] = 0 \tag{7.1.1-3}$$

This is the governing equation for the weak Hamiltonian method for problems with touch-point state inequality constraints. It is easily shown by integrating the $\delta\dot{x}$ and $\delta\dot{\lambda}$ terms by parts in Eq. (7.1.1–3) that all the Euler-Lagrange equations are the same as in [65] and that all boundary conditions are now of the natural type.

One simplification may be made to Eq. (7.1.1–3). If the control is continuous across t_{tp} (as is guaranteed if the Hamiltonian is regular), then it is possible to simplify the dt_{tp} equation since then $f(t_{tp}^-) = f(t_{tp}^+) = f(t_{tp})$ and $L(t_{tp}^-) = L(t_{tp}^+) = L(t_{tp})$. From the necessary conditions that are found in [65] or from the ones that could be found from Eq. (7.1.1–3), it is seen that

$$\lambda^T(t_{tp}^-) - \lambda^T(t_{tp}^+) = \nu_1 \frac{\partial S}{\partial x} \tag{7.1.1–4}$$

Now, rewriting the coefficient of dt_{tp} as

$$\begin{aligned} H(t_{tp}^-) - H(t_{tp}^+) + \nu_1 \frac{\partial S}{\partial t} &= \left[\lambda^T(t_{tp}^-) - \lambda^T(t_{tp}^+)\right] f(t_{tp}) + \nu_1 \frac{\partial S}{\partial t} \\ &= \nu_1 \frac{\partial S}{\partial x}\dot{x} + \nu_1 \frac{\partial S}{\partial t} = \nu_1 \frac{dS}{dt} \end{aligned} \tag{7.1.1–5}$$

we see that the condition for continuity of the Hamiltonian reduces to the condition that the first total time derivative of the constraint be zero at t_{tp} if the control is continuous.

An example of a touch-point case is given in the next section.

7.1.2. Boundary Arc Case

For cases where there is a boundary arc (*i.e.*, the solution rides the constraint boundary for a nonzero length of time), then the weak formulation must be modified. For simplicity, consider the case where the solution has an unconstrained arc between t_0 and t_{en}, followed by a constrained arc between t_{en} and t_{ex}, and then another unconstrained arc between t_{ex} and t_f. Introducing a new Lagrange multiplier function η to adjoin the pth derivative of the constraint S to the performance index, then J becomes

$$\begin{aligned} J = &\int_{t_0}^{t_{en}} \left[L(x,u,t) + \lambda^T(f-\dot{x})\right] \, dt \\ &+ \int_{t_{en}}^{t_{ex}} \left[L(x,u,t) + \lambda^T(f-\dot{x}) + \eta \frac{d^pS}{dt^p}\right] \, dt \\ &+ \int_{t_{ex}}^{t_f} \left[L(x,u,t) + \lambda^T(f-\dot{x})\right] \, dt + \nu_1 N|_{t_{en}} + \Phi|_{t_0}^{t_f} + \alpha^T(x-\hat{x})|_{t_0}^{t_f} \end{aligned} \tag{7.1.2–1}$$

where N is a column matrix defined as

$$N^T = \begin{bmatrix} S & \frac{dS}{dt} & \dots & \frac{d^{p-1}S}{dt^{p-1}} \end{bmatrix} \tag{7.1.2–2}$$

Analogous steps to those described in Chapter 4 lead to a weak formulation for state constraint problems which ride the constraint boundary. There are only two minor differences. One is that the time line must be discretized between t_0 and t_{en}, from t_{en} to t_{ex}, and from t_{ex} to t_f. Also, there will be equations corresponding to the $\delta\eta$ coefficient over the interval from t_{en} to t_{ex}, just as there were with the control constraint of Chapter 6. These equations are that the pth derivative of S be zero between t_{en} and t_{ex}.

Also note that the dt_{en} and dt_{ex} equation will reduce to the condition that the pth total time derivative of S be zero. The proof is identical to that shown in Eq. (7.1.1–5) after noting that

$$\lambda^T(t_{en}^-) - \lambda^T(t_{en}^+) = \nu_1 \frac{\partial N}{\partial x} \tag{7.1.2-3}$$

Two examples of problems with a boundary arc are given in the next section.

7.2. Example: A First-Order Problem

Consider the classical brachistochrone problem in Section 3.11 of [1]. Let x and y define the horizontal and vertical (positive downward) position of the particle respectively. The governing state equations are

$$\begin{aligned} \dot{x} &= (2gy)^{1/2} \cos u \\ \dot{y} &= (2gy)^{1/2} \sin u \end{aligned} \tag{7.2-1}$$

where g is the acceleration due to gravity (a constant) and the control u is the angle that the tangent to the path makes with the horizontal. The problem is to minimize the time it takes for the particle to move from the origin to any point on the line $x = L$. The optimal path to the unconstrained problem was found over 200 years ago by several mathematicians to be a cycloid. The problem takes a new twist though when a state inequality constraint is added. Let $S = y - x \tan\theta - h \leq 0$ where θ and h are constants. The first total time derivative of S yields

$$\dot{S} = (2gy)^{1/2} \sin(u - \theta)/\cos\theta = 0 \tag{7.2-2}$$

or $u = \theta$ along the constraint boundary. Thus, this is a first order constraint.

By adjoining $\dot{S}$ to the performance index, the solution was readily found, although the programming was a little tricky because of the three unknown times to reckon with. Fig. 7.1 shows the trajectory of the particle and Fig. 7.2 shows the control history. In the figures, (2:2:2) designates that 2 elements were used on each of the unconstrained arcs and the constrained arc in between. The finite element solution for 2 and 4 elements in each phase are graphed versus the exact solution found in [1]. The answers are excellent for the states, but are a little off on the controls. When more elements are run, the answer is seen to converge on the exact answer. Fig. 7.3 shows the log of the error in the entry, exit and final times versus the number of elements. The lines are all approximately straight with a slope of about -1. This indicates that the times have an error proportional to about $1/N$. This is uncharacteristically inaccurate as compared to other results presented thus far.

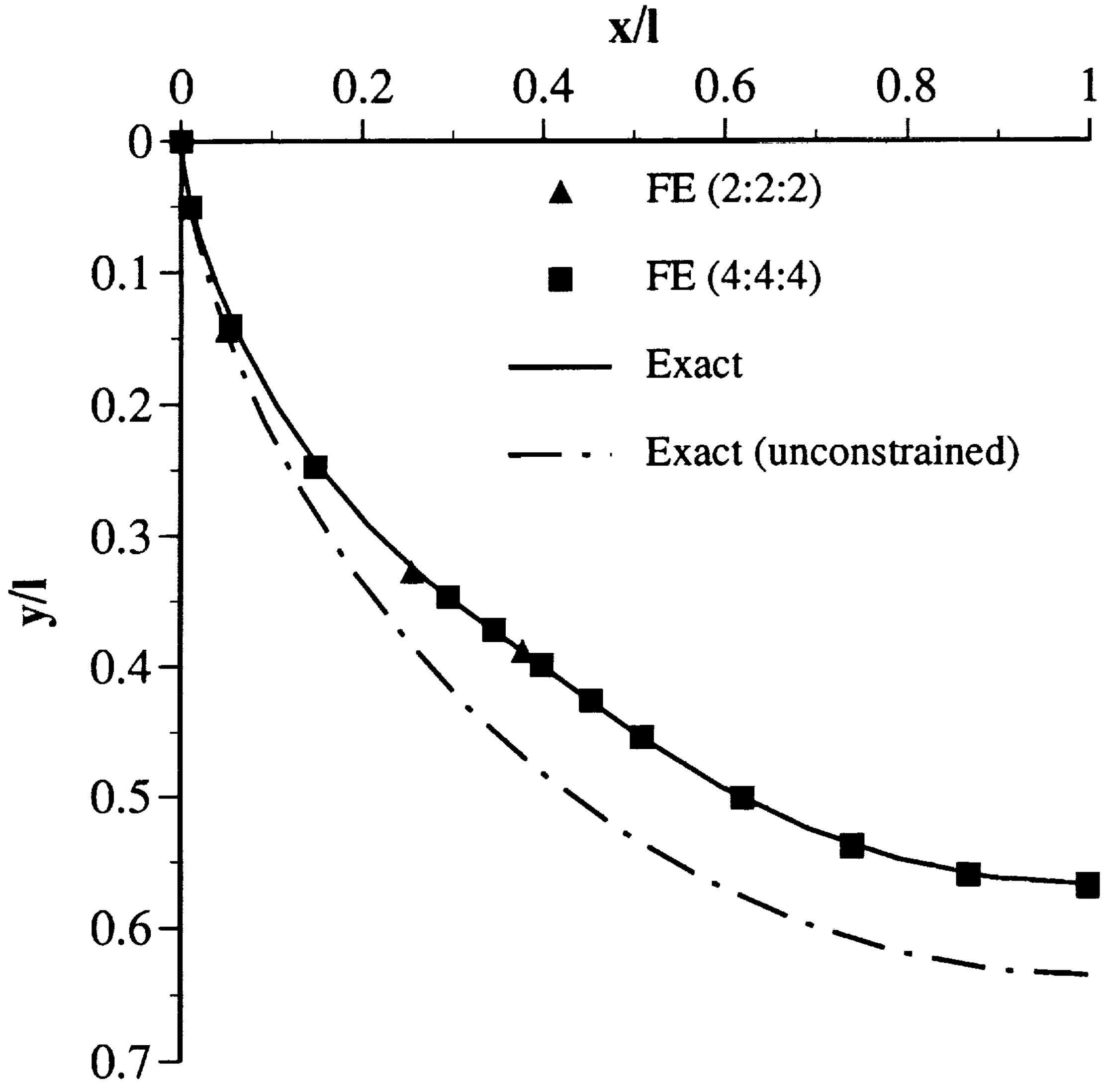

Fig. 7.1: x versus y

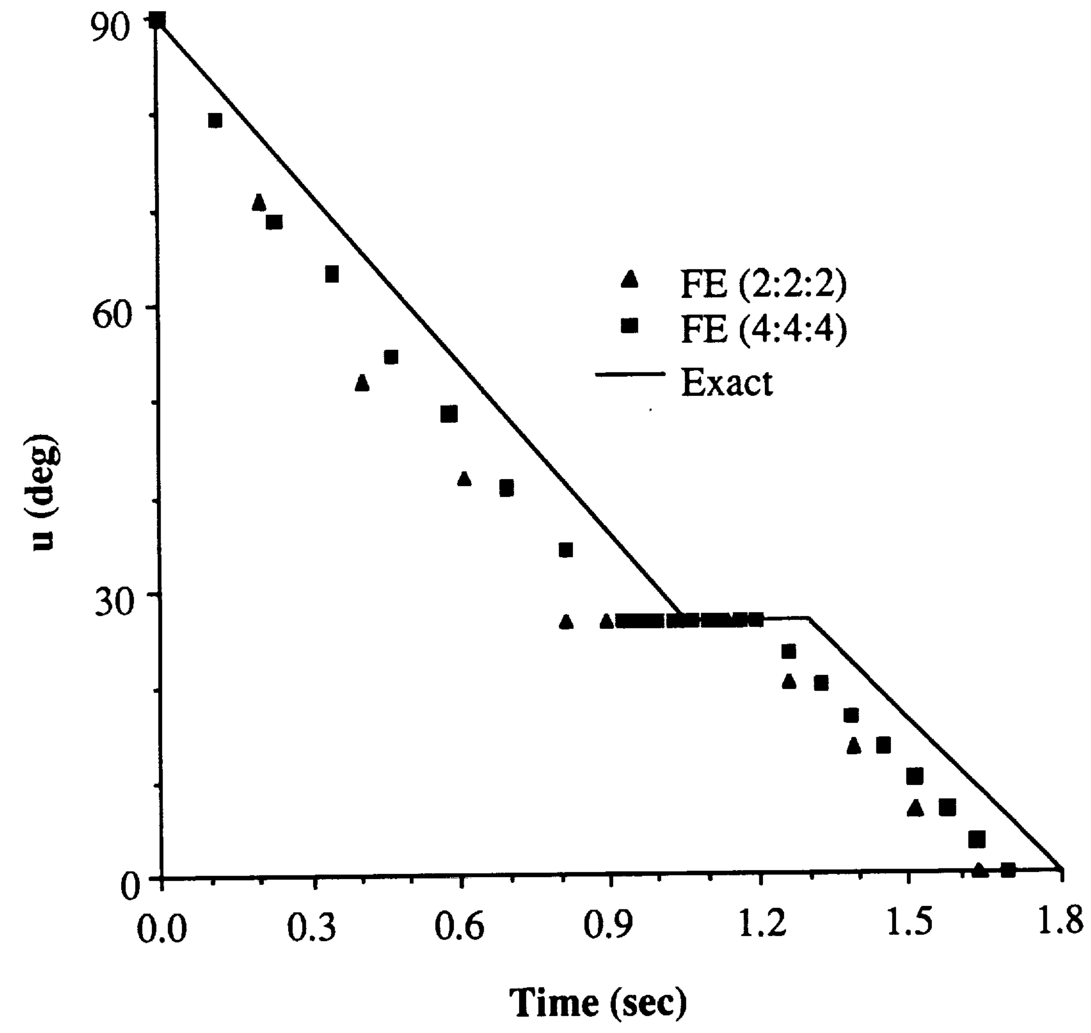

Fig. 7.2: u versus time

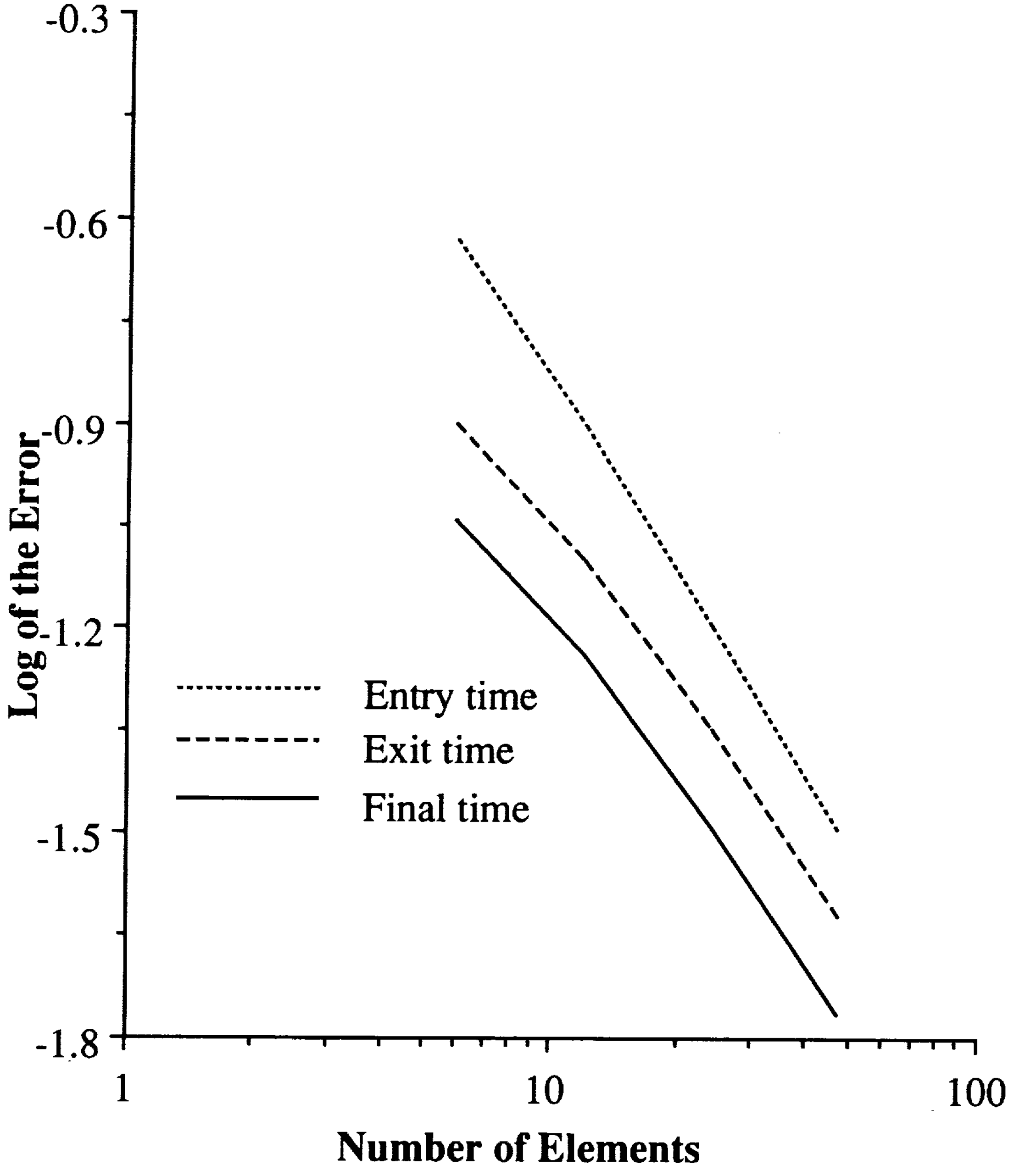

Fig. 7.3: Log of the error versus the number of elements

7.3. Example: A Second-Order Problem

This example is taken from section 3.11 of [1]. The problem is to minimize

$$J = \frac{1}{2}\int_0^1 u^2\,dt \tag{7.3-1}$$

The state equations are

$$\begin{aligned} \dot{x}_1 &= u \\ \dot{x}_2 &= x_1 \end{aligned} \tag{7.3-2}$$

The state inequality constraint $S(x,t) = x_2 - \ell \leq 0$ is to be imposed. For certain values of ℓ, the solution only touches the boundary, whereas for other values of ℓ the solution rides the boundary.

The algebraic equations for both examples were solved using a Newton-Raphson method and a FORTRAN code written on a SUN 3/260. The sparse, linearized equations are solved using subroutine MA28 from the Harwell subroutine library [62].

The state x_2 is shown in Fig. 7.4 for the single touch-point case. Results for 2, 4, and 8 elements on either side of the touch-point (denoted by 2:2, etc.) are compared to the exact solution. Note that even the 2:2 element case lies essentially on the exact solution. In Fig. 7.5, the state x_2 is shown for an example case where the state rides the boundary. Here, there are three time intervals and the number of elements in each interval is denoted by 2:2:2 etc. Again we see that the 2:2:2 case has essentially converged on the exact solution.

One drawback of the weak formulation is that two separate codes had to be written to solve this problem. Also, one must determine in advance if the solution will ride or just touch the constraint. However, with the general code described in Chapter 11, this is a simple and quick thing to do.

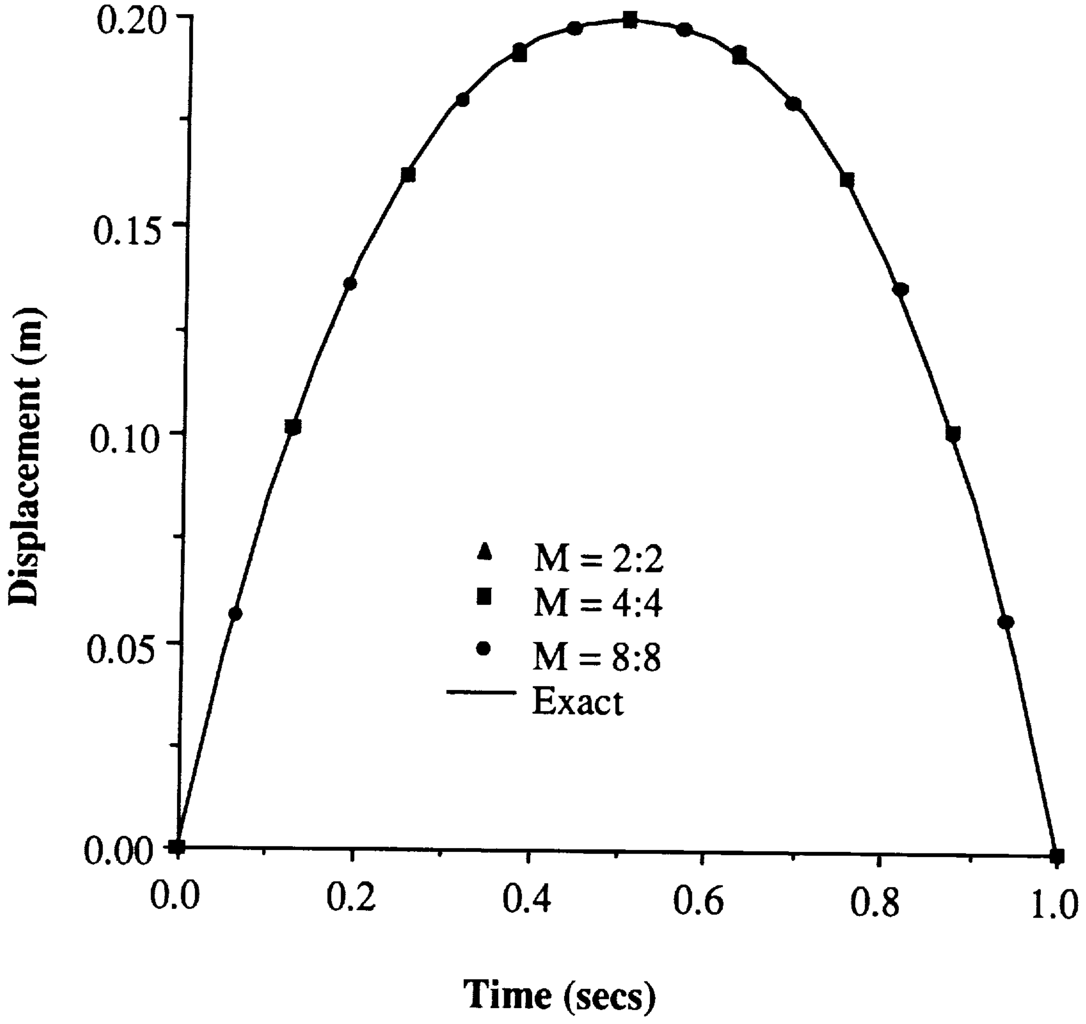

Fig. 7.4: x_2 versus time for a touch-point case of $\ell = 0.2$

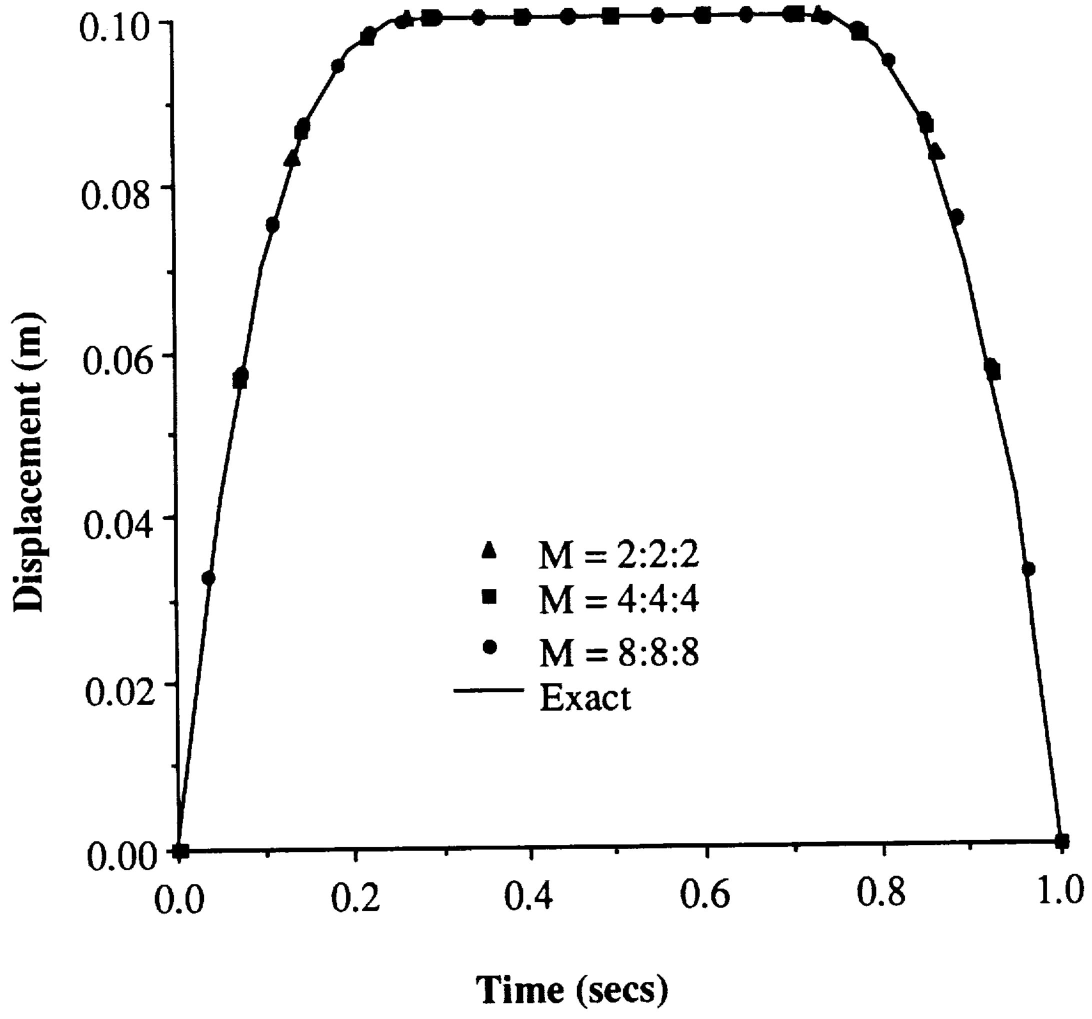

Fig. 7.5: x_2 versus time for a boundary arc case of $\ell = 0.1$

CHAPTER 8

AN ADVANCED LAUNCH VEHICLE

As stated in Chapter 1, future space transportation and deployment needs are critically dependent on the development of reliable and economical launch vehicles that will provide flexible, routine access to orbit. The objective of the Advanced Launch System program is to place large payloads – 100,000 to 150,000 pounds – into low Earth orbit at an order of magnitude lower cost per pound. The program also seeks to make the entire scope of space launch operations significantly more routine compared to present methods and procedures that are highly dependent on ground operations. The goals of the ALS program can only be met by development of reliable and efficient on-board algorithms that are capable of calculating real-time optimal trajectories.

In this chapter, a model of an advanced launch vehicle is presented [66]. This is a two-stage, four-state vehicle with control and state inequality constraints to be imposed. The results from the finite element algorithm are seen to compare favorably with multiple shooting results. Of great interest is the fact that the answers are obtained in a very short time interval as compared to the time span of the entire trajectory.

8.1. A Model for an Advanced Launch Vehicle

A two-stage, four-state vehicle is considered that has two control inequality constraints and one state constraint. (Only one of these constraints is violated and is therefore the only one included in the results.)

We confine our attention to vertical plane dynamics of a vehicle flying over a spherical, non-rotating earth (see Fig. 3.1). This results in the following model for the states m (mass), h (height), V (velocity), and γ (flight-path angle):

$$\begin{aligned}
\dot{m} &= -\frac{T_{vac}}{gI_{sp}} \\
\dot{h} &= V\sin\gamma \\
\dot{V} &= \frac{T\cos\alpha - D}{m} - \frac{\mu\sin\gamma}{r^2} \\
\dot{\gamma} &= \left(\frac{T\sin\alpha + L}{mV}\right) + \left(\frac{V}{r} - \frac{\mu}{r^2 V}\right)\cos\gamma
\end{aligned} \tag{8.1–1}$$

where T is the thrust, T_{vac} is the thrust in a vacuum, D is the drag, and L is the lift. Here α, the angle of attack, has been adopted as a control variable.

Note that now V is being used as a state instead of E. There are two reasons for this. First, using V as a state simplified the algebraic equations. Secondly, since E was two or three orders of magnitude larger than the other states in the Saturn models, convergence was easier to obtain with V as a state.

The aerodynamic and propulsion models are given by the following equations:

$$\begin{aligned}
T &= T_{vac} - A_e p(h); \quad M = \frac{V}{a(h)} \\
r &= R_e + h; \quad q = \frac{\rho V^2}{2} \\
D &= qSC_D(M,\alpha); \quad L = qSC_L(M,\alpha)
\end{aligned} \tag{8.1–2}$$

The atmospheric data for density, pressure, and speed of sound are obtained from the 1975 standard atmospheric data [67].

The vehicle parameters chosen for this model are

$$\begin{aligned}
S_{\text{I}} &= 131.34\,\text{m}^2; \qquad S_{\text{II}} = 65.67\,\text{m}^2 \\
T_{vac_{\text{I}}} &= 25813400\,\text{N}; \quad T_{vac_{\text{II}}} = 7744020\,\text{N} \\
A_{e_{\text{I}}} &= 37.515\,\text{m}^2; \qquad A_{e_{\text{II}}} = 11.254\,\text{m}^2 \\
I_{sp_{\text{I}}} &= I_{sp_{\text{II}}} = 430.0\,\text{s}
\end{aligned} \tag{8.1–3}$$

where subscripts "I" and "II" refer to the first and second stages respectively.

The aerodynamic coefficient data C_D and C_L are functions of the Mach number M and angle of attack α. The physical constants used in the above model are the earth's gravitational constant $\mu = 3.9906 \times 10^{14}$ m^{3}s^{-2}, the earth's mean radius $R_e = 6.378 \times 10^6$ m and the acceleration due to gravity $g = 9.81$ ms^{-2}.

The three constraints are

$$\begin{aligned}
g_1(x,u,t) &= \alpha q - 2925\,\text{rad-Pa} \le 0 \\
g_2(x,u,t) &= -(\alpha q + 2925)\,\text{rad-Pa} \le 0 \\
q(h,V) &- 40698.2\,\text{Pa} \le 0
\end{aligned} \tag{8.1–4}$$

where only the first constraint will be enforced since the other two are not violated.

The performance index is

$$J = \phi|_{t_f} = m|_{t_f} \tag{8.1–5}$$

and the final time t_f is open. The initial conditions specified are $m(0) = 1.52345 \times 10^6$ kg, $h(0) = 400$ m, $V(0) = 64.48941$ m/s, and $\gamma(0) = 89.5°$. The final conditions are $h(t_f) = 148160.0$ m, $V(t_f) = 7858.1995$ m/s, and $\gamma(t_f) = 0.0°$. The burnout mass of the first stage is 645500 kg and the drop-mass of the booster is 98880 kg.

8.2. Computational Aspects

The code for this model was written on a SUN 3/260. The code was written to be as efficient as possible so as to get a feel for the actual run-times one might see in an on-board computational setting.

An explicit Jacobian is formed within the code and a Newton-Raphson method was employed. The Jacobian was 85% to 95% sparse for the runs made. Great computational savings came from taking advantage of the Harwell sparse matrix solver MA28AD [59]. The code was run with double precision.

Initial guesses were, of course, necessary for the Newton-Raphson method. A Taylor series approach was taken that generated initial guesses for all variables. These guesses, although crude, were good enough to make a converged run with boundary conditions that differed somewhat from the specified conditions. We were then able to slowly "move" the boundary conditions out to the specified ones. An explanation of the Taylor series approach is given in Chapter 10.

8.3. Results

In Figs. 8.1 – 8.8, numerical results for the ALV model with no constraints enforced are given for 2, 4, and 8 elements per time interval, where the number of elements is denoted by $(N_1 : N_2)$ on the plots. These results are compared to a multiple-shooting code as a check on the accuracy of the method and of the program. The four states are shown in Figs. 8.1 – 8.4 and the costates are shown in Figs. 8.5 – 8.8. For all cases, the (8:8) run lies on the essentially exact curve corresponding to the multiple shooting (MS) code. In general, even the (4:4) run yields an excellent approximation to the solution.

The control is shown in Fig. 8.9. Although the (8:8) run is close to the exact curve, it has not converged on the answer. Due to the large slopes and sharp peaks in the control, the finite element method required 24 elements in the first stage to converge on the solution. However, it is important to note that we were still able to run only 8 elements in the second stage. Thus, it may be feasible to cluster the elements to refine the solution.

Fig. 8.10 shows a graph of αq. (The dynamic pressure q is not shown because it does not violate the constraint.) It can be seen clearly that only one control constraint, g_1, is violated. This is the only constraint that was added to the program to generate the next two graphs.

The (4:4) results above were obtained in 5.5 CPU seconds on a SUN 3/260, and five iterations were required with a Newton-Raphson method. Of course, the number of iterations depends on the quality of the initial guesses. In an on-board computational setting, the initial guesses should be pretty good since they would probably be determined from a previously obtained solution.

For Figs. 8.11 and 8.12, the control constraint g_1 was included into the computer model. Since the constraint is just barely violated, there was essentially no change in the data when the multiple shooting code was run; therefore, it is not seen in these two graphs. However, for illustrative purposes, the unconstrained case, the realistic constraint, and two unrealistic constraints are shown for the finite element case. Even for the lowest of the constraints, the states, costates, and dynamic pressure are virtually unchanged. Also, no significant extra computer time was expended.

Finally, as a feel for the global convergence of the method, the Hamiltonian of the unconstrained system is plotted in Fig. 8.13 for 2, 4, 8, and 16 elements per stage. There is a nice convergence toward the exact answer of zero with an increase in the number of elements.

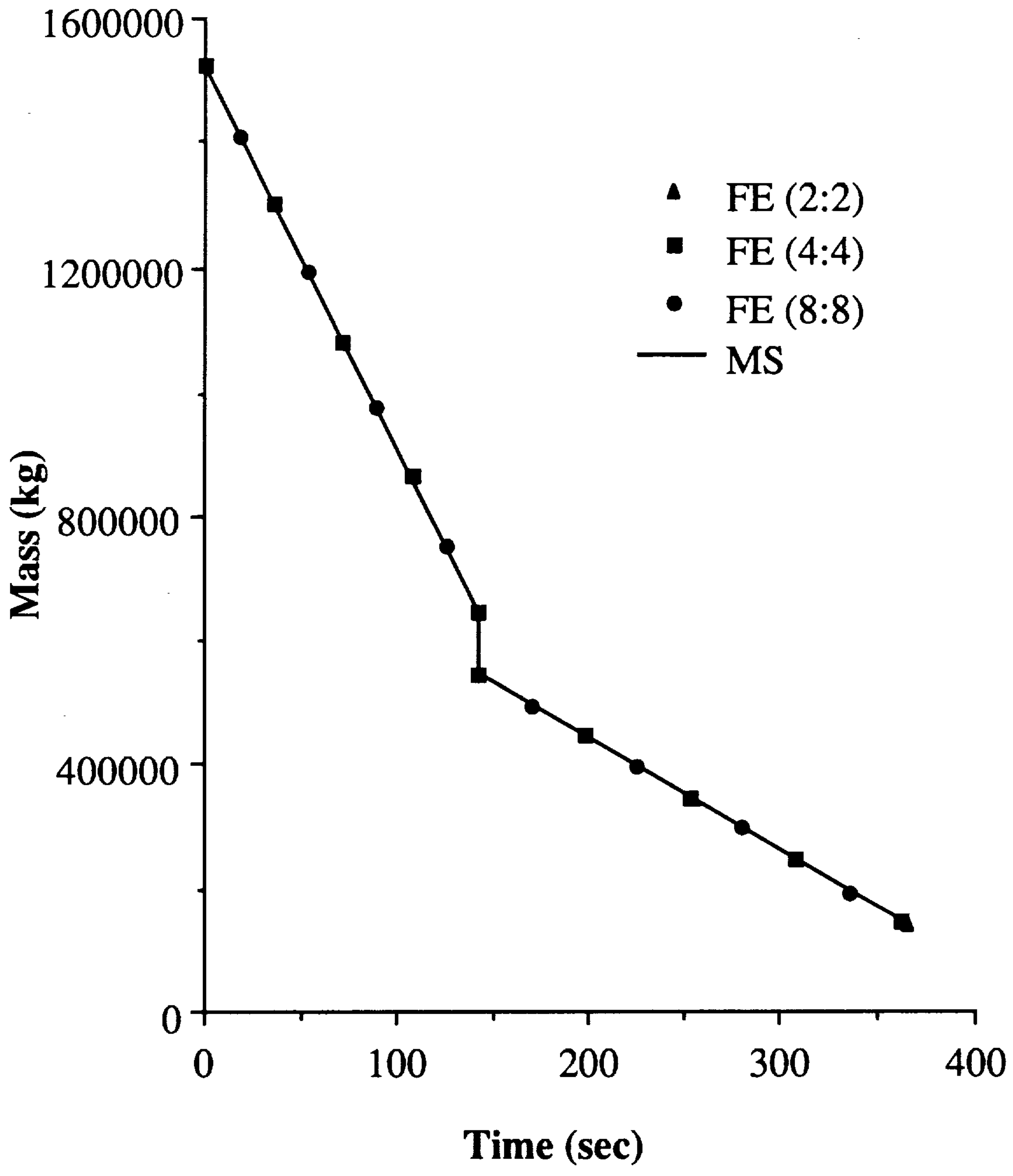

Fig. 8.1: Mass versus time

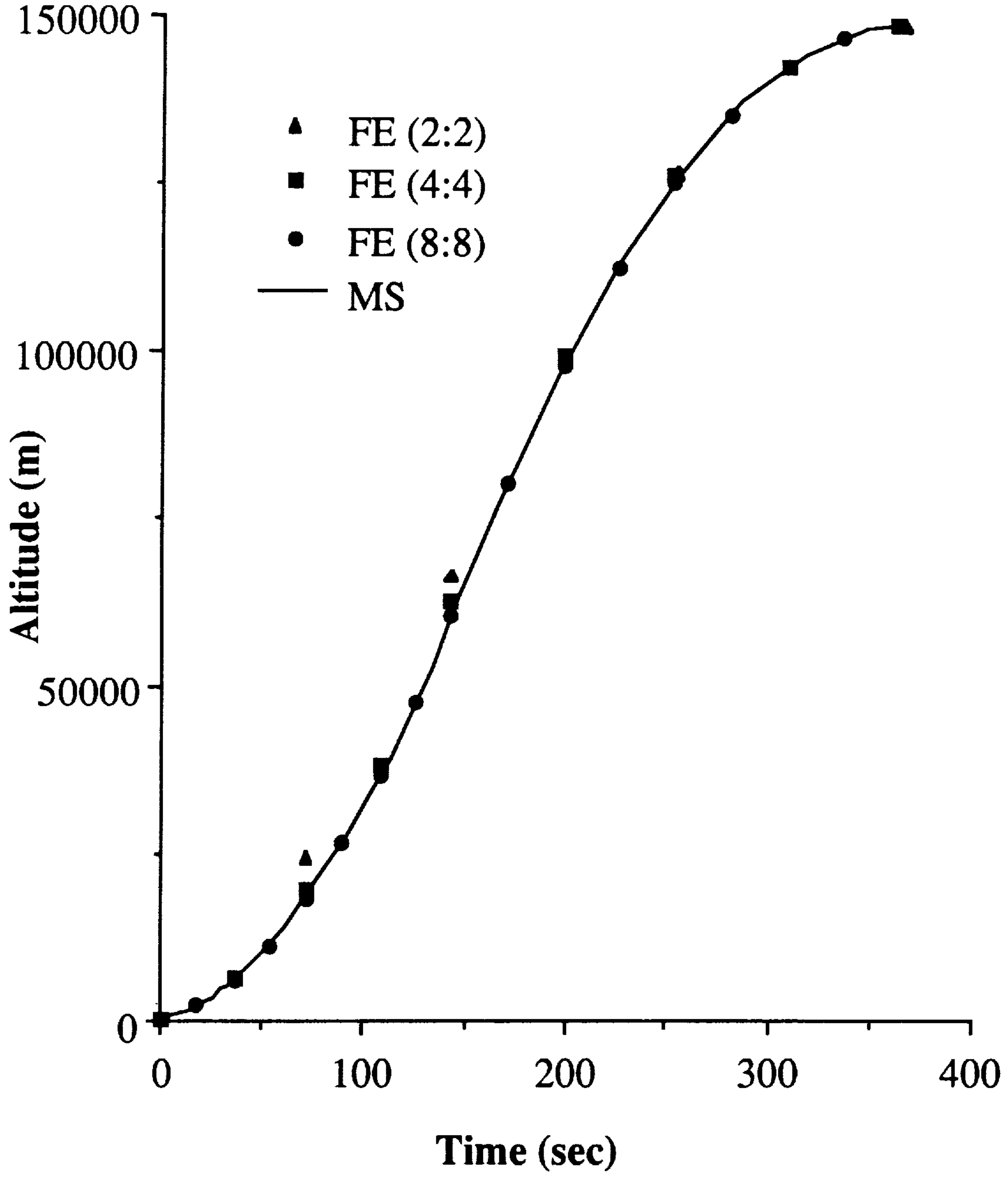

Fig. 8.2: Altitude versus time

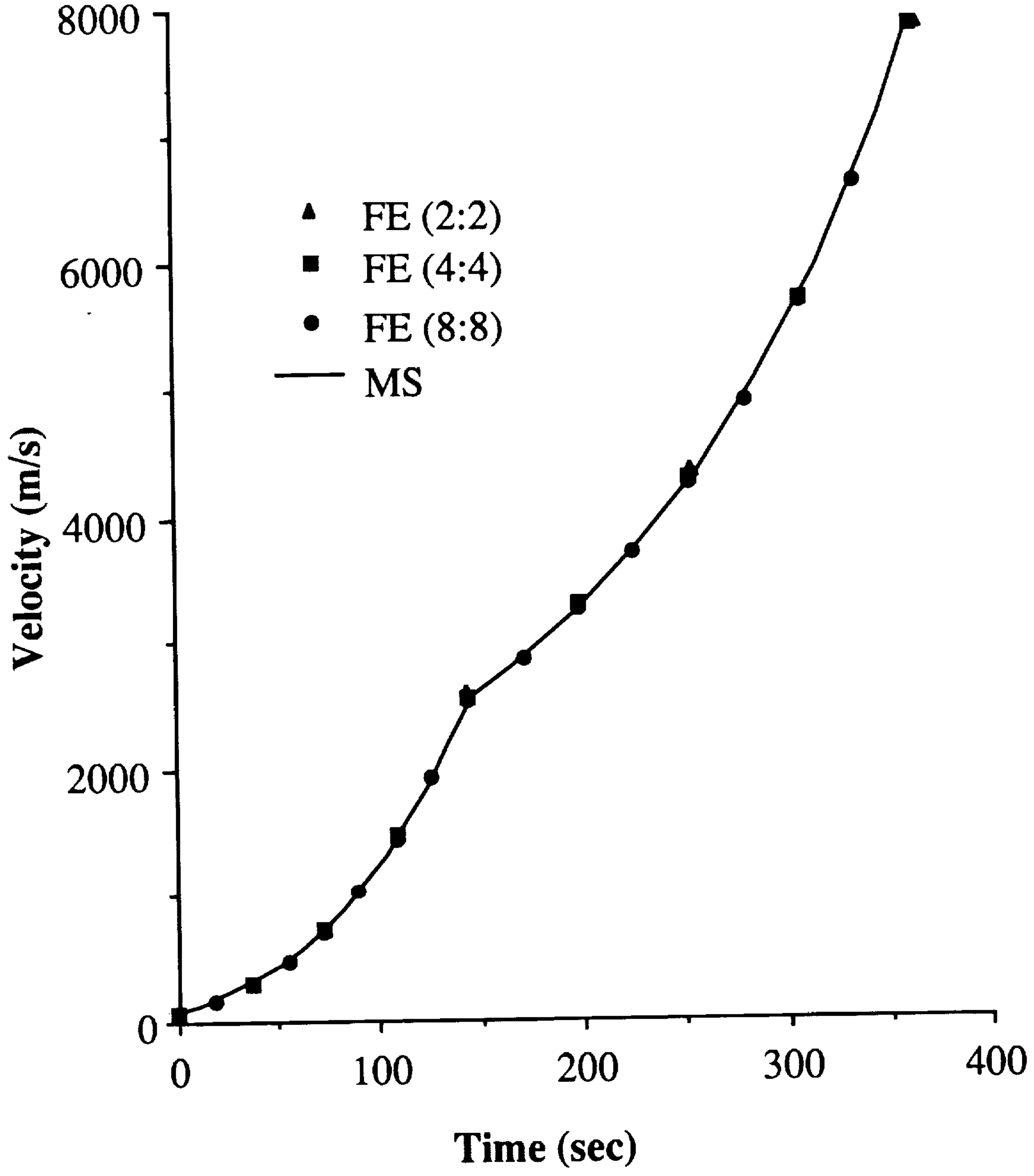

Fig. 8.3: Velocity versus time

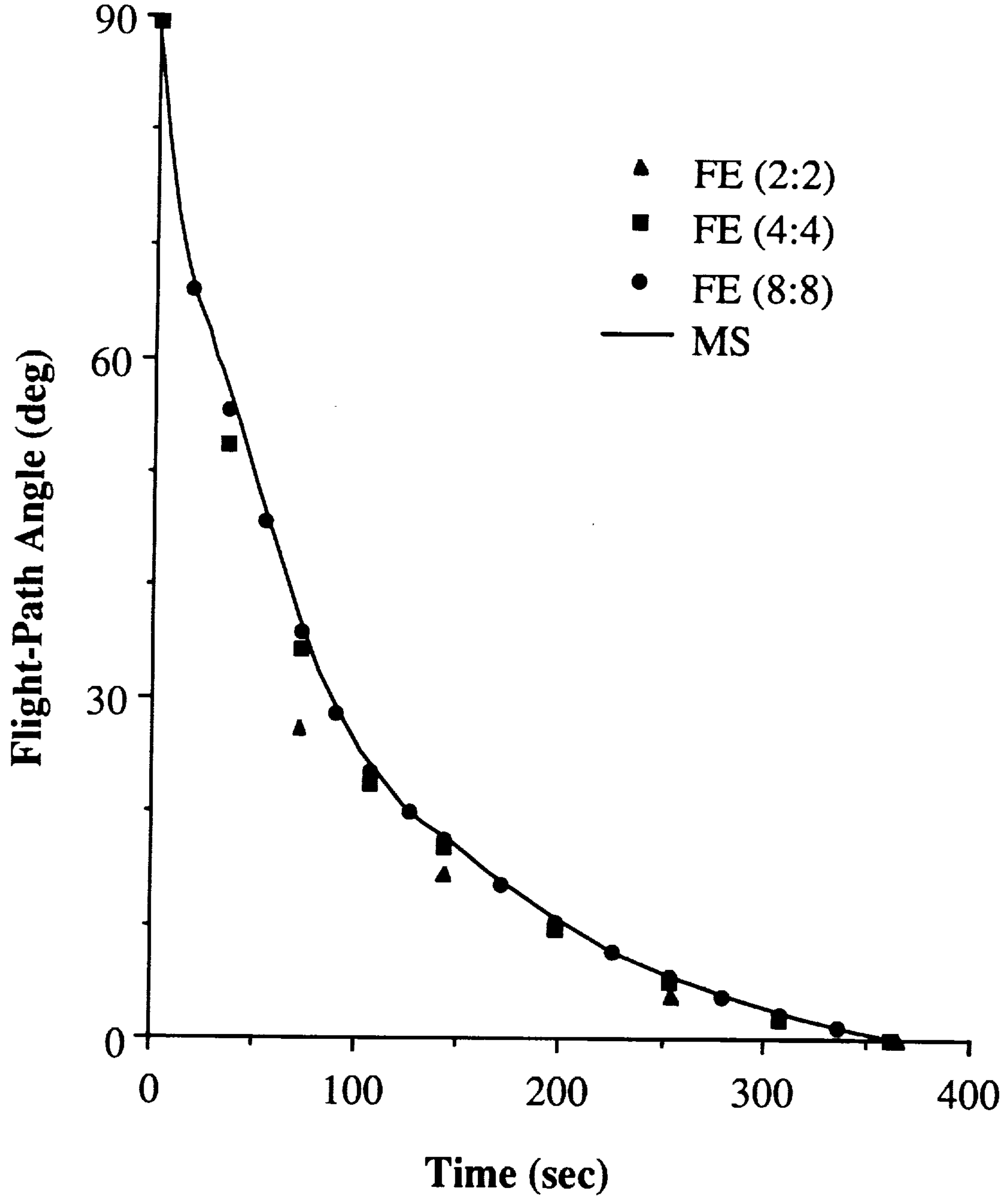

Fig. 8.4: Flight-path angle versus time

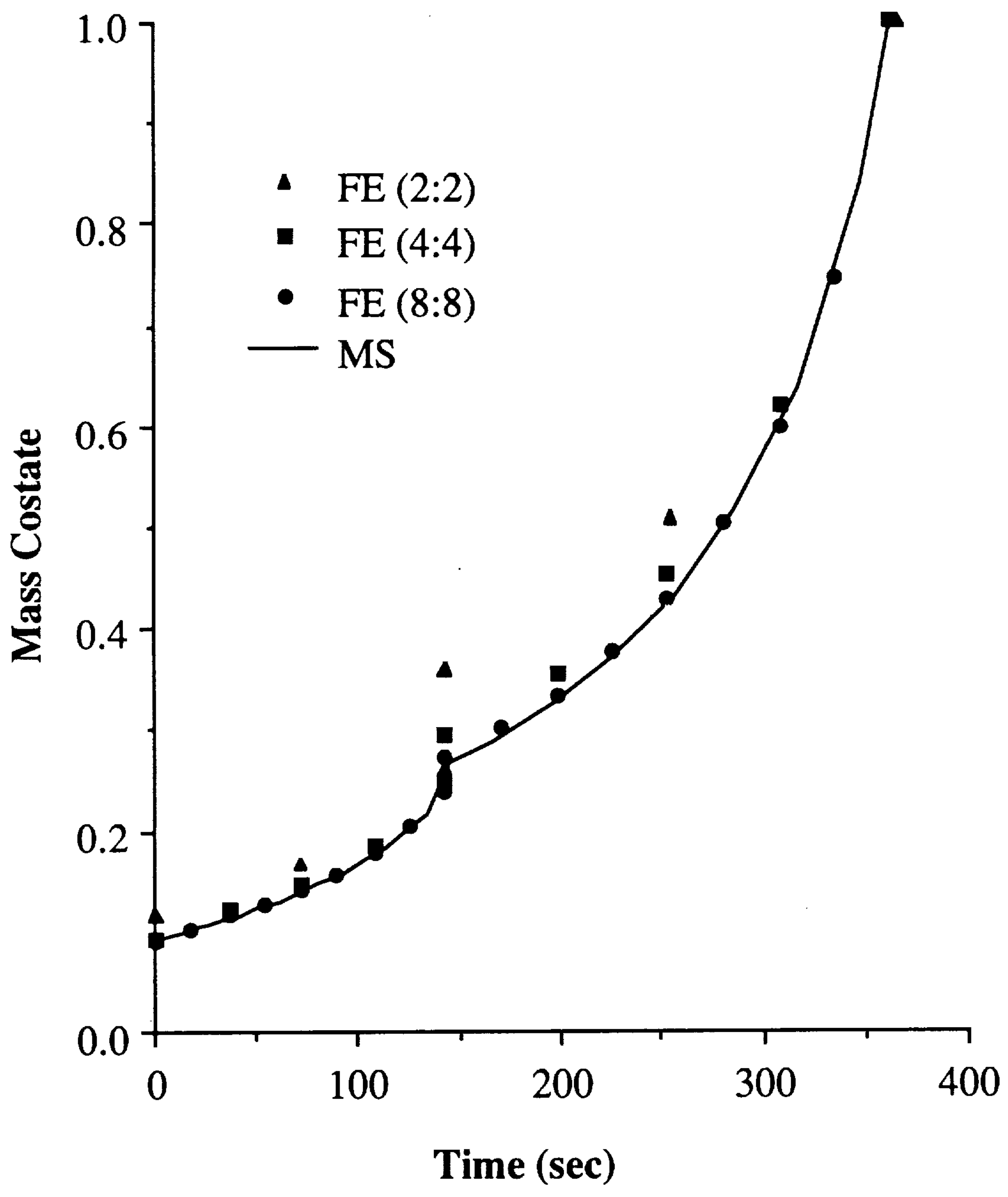

Fig. 8.5: Mass costate versus time

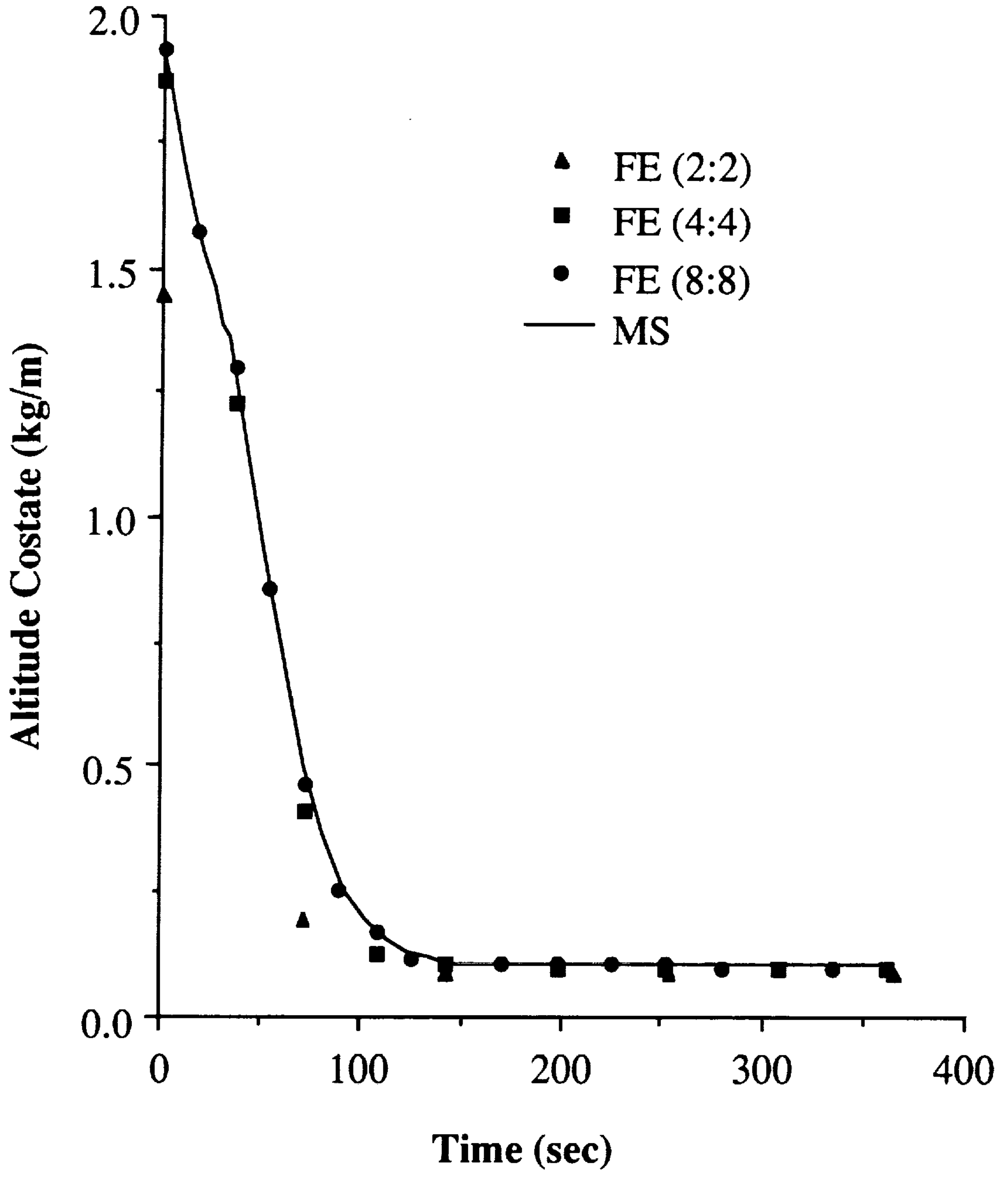

Fig. 8.6: Altitude costate versus time

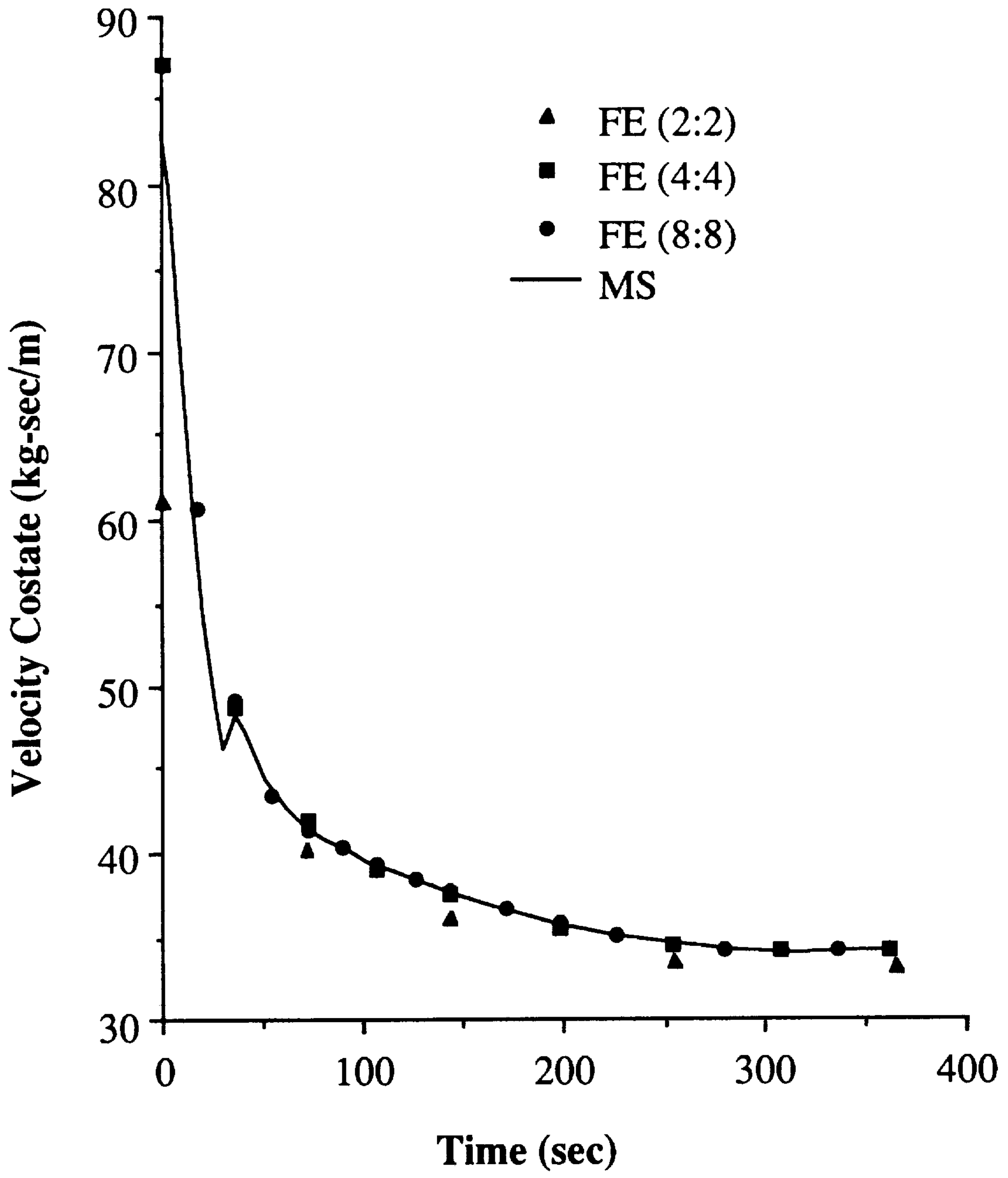

Fig. 8.7: Velocity costate versus time

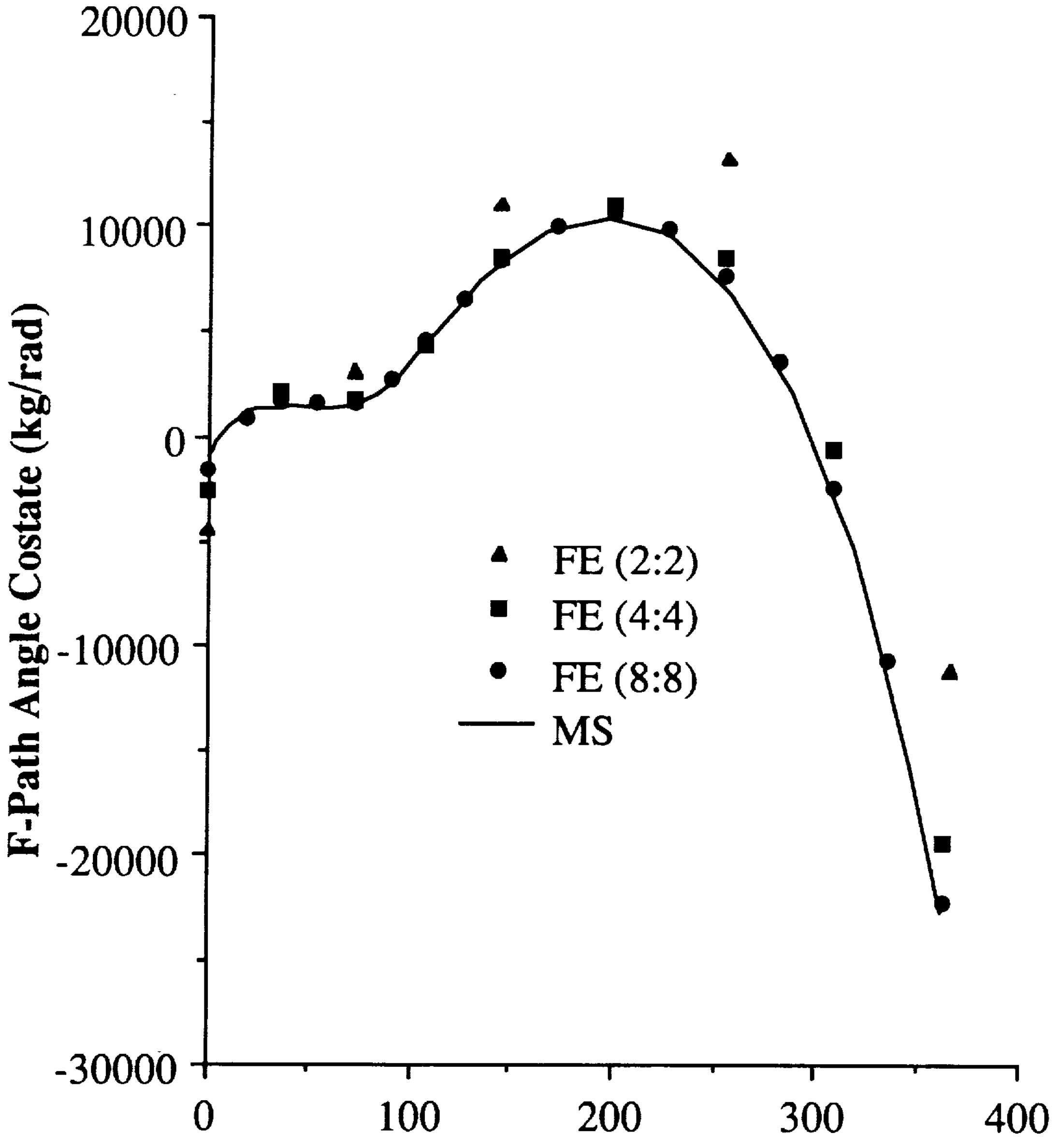

Fig. 8.8: Flight-path angle costate versus time

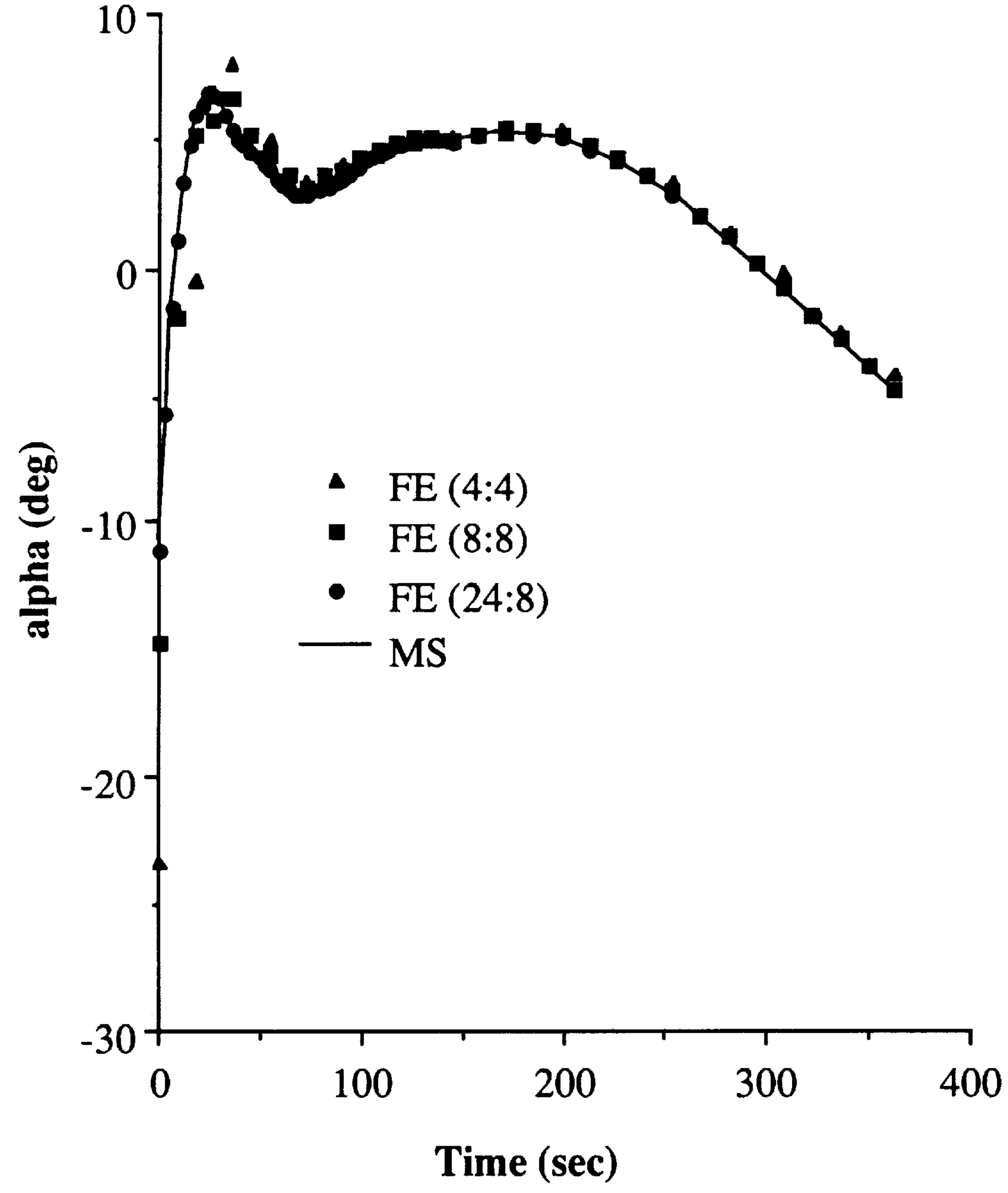

Fig. 8.9: Angle of attack (α) versus time

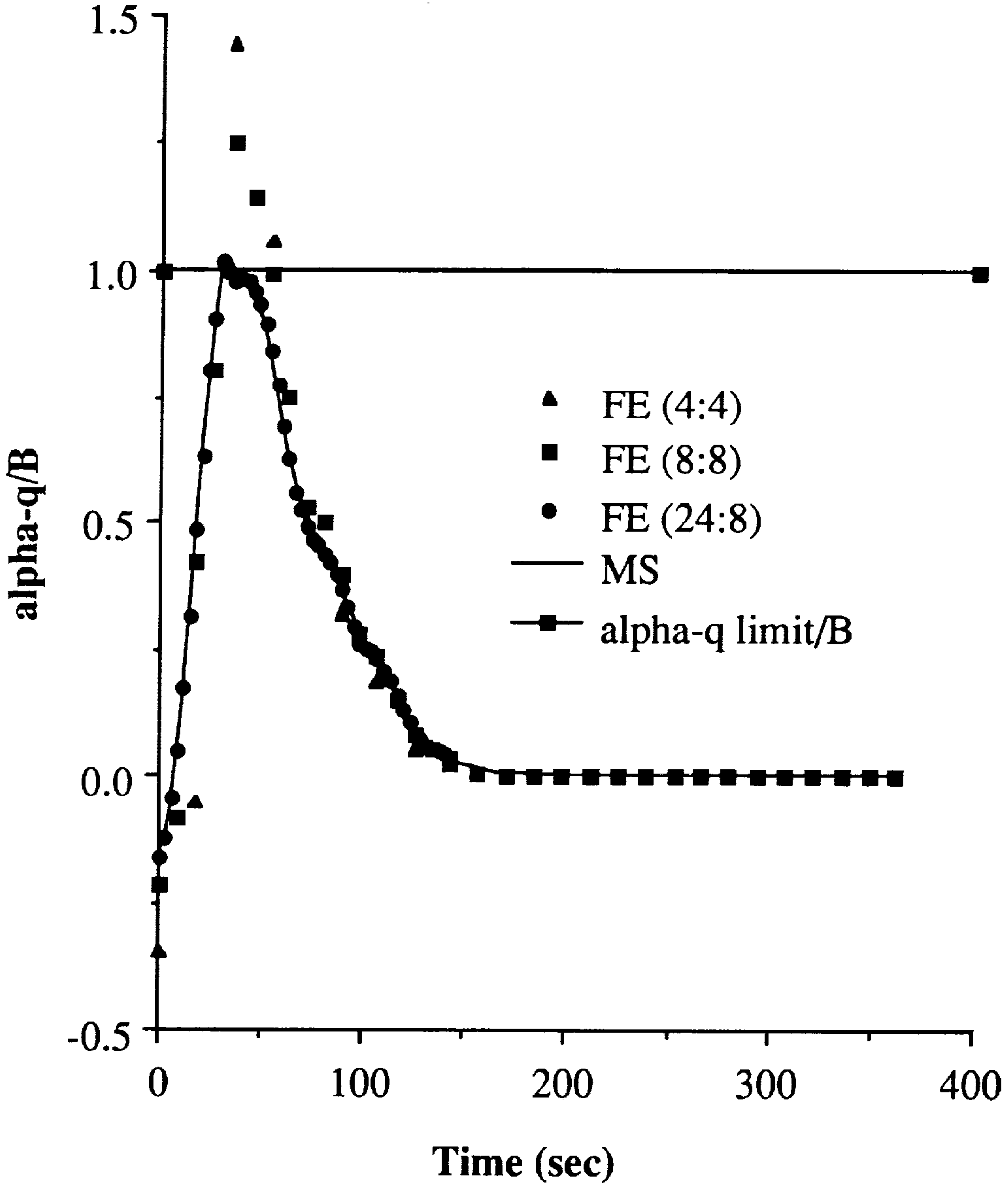

Fig. 8.10: αq versus time

One control constraint is violated

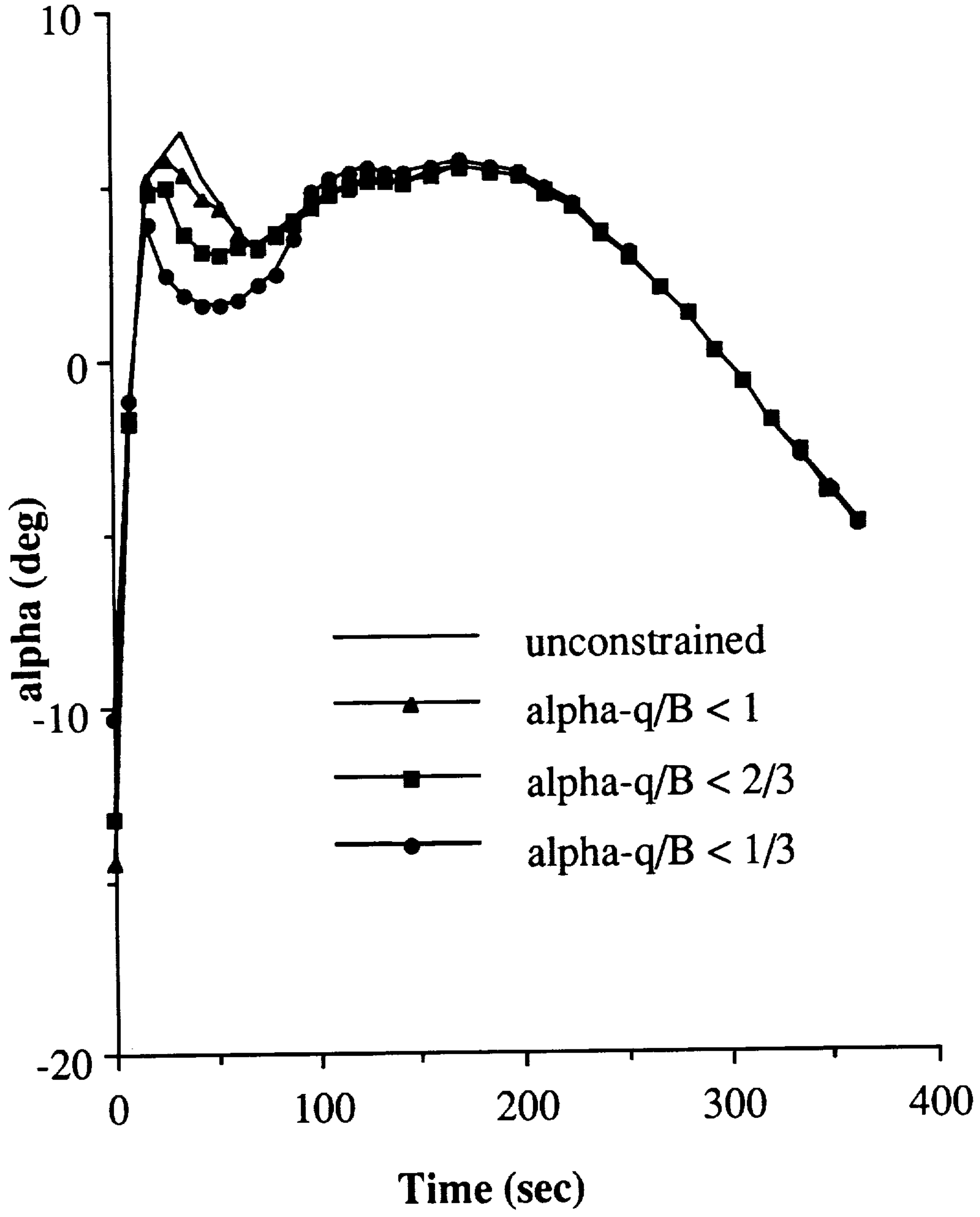

Fig. 8.11: Angle of attack (α) versus time
FE (8:8) run; B = 2925.0 rad-Pa

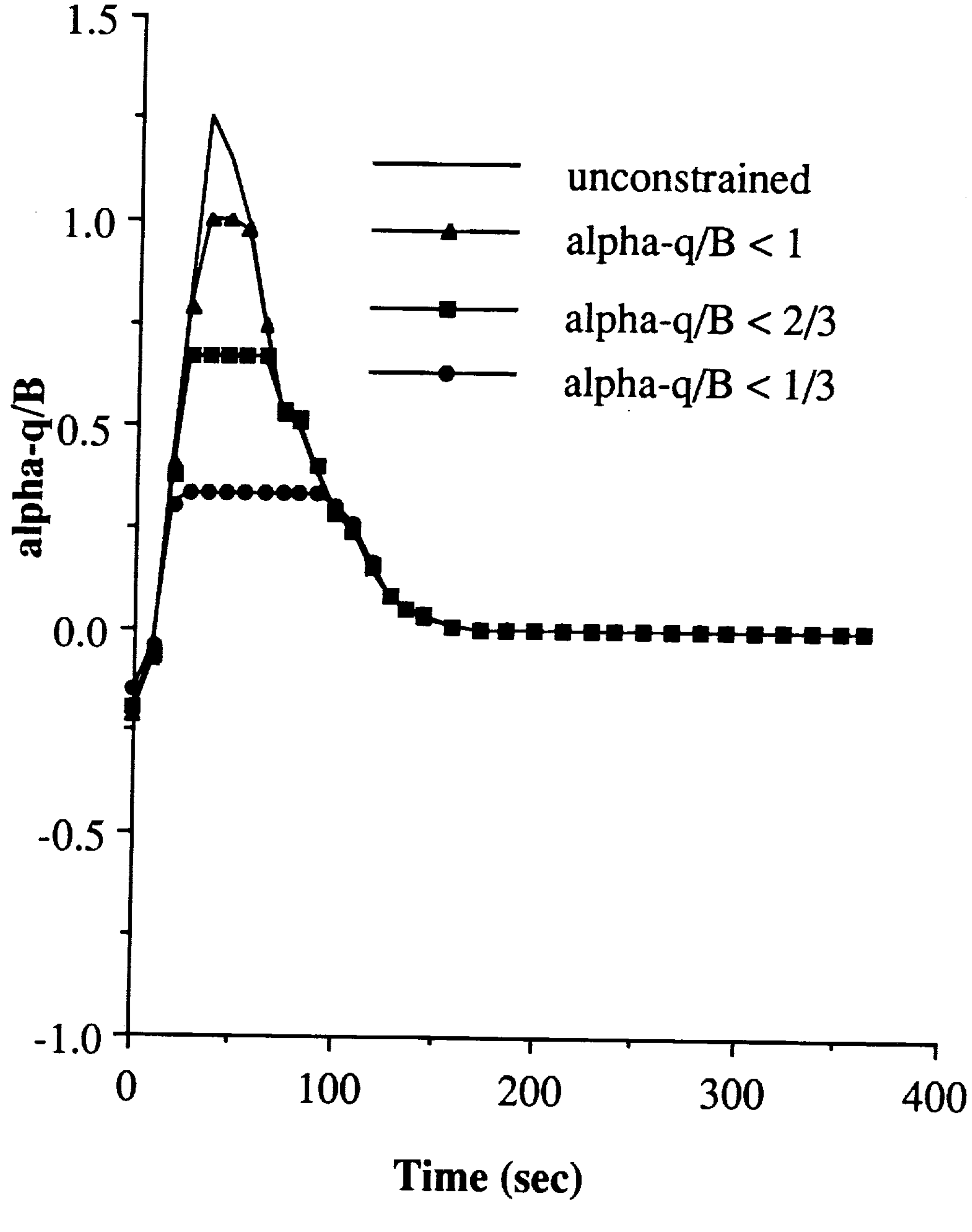

Fig. 8.12: αq versus time
FE (8:8) run; B = 2925.0 rad-Pa

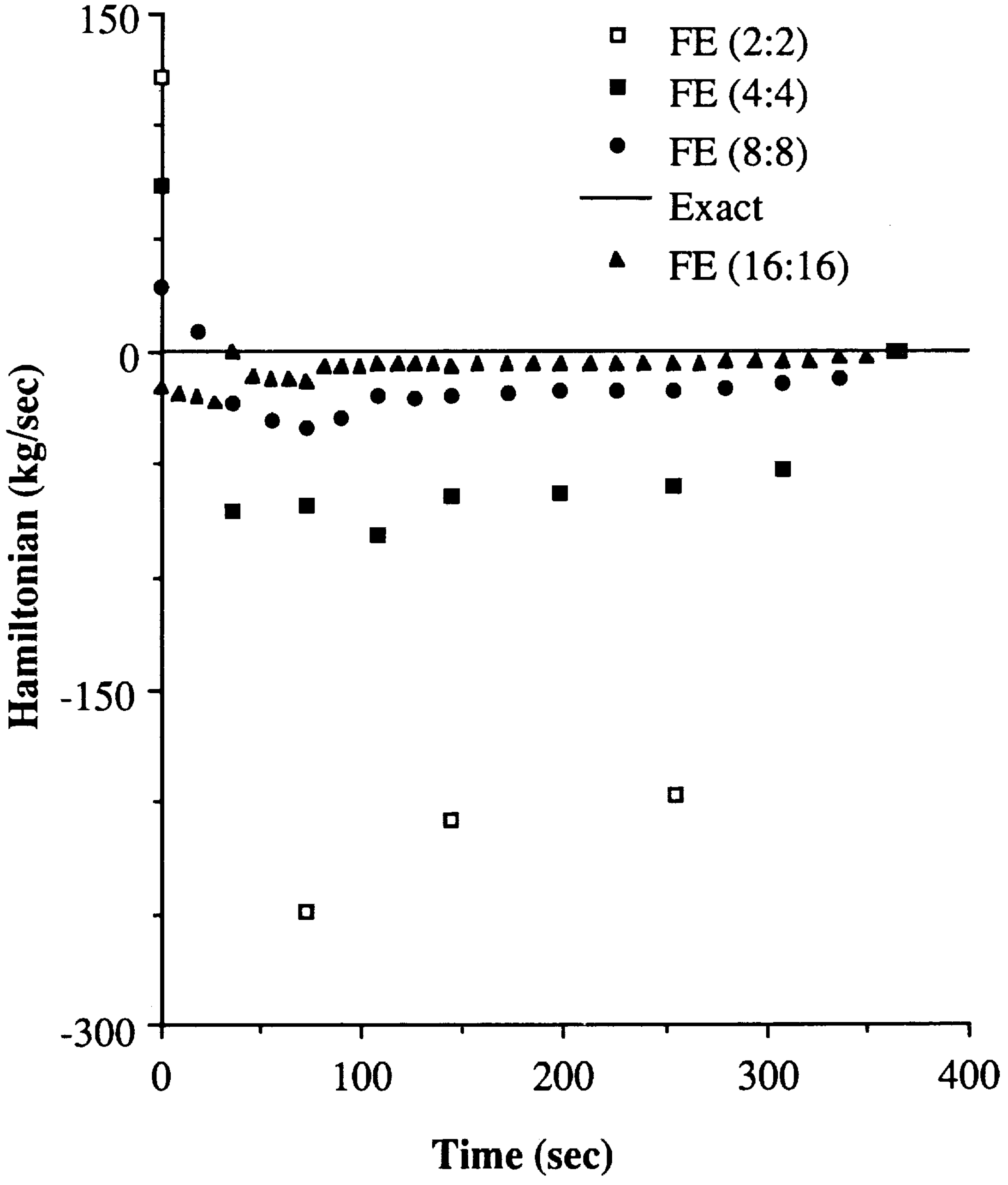

Fig. 8.13: Hamiltonian versus time

CHAPTER 9

ERROR ESTIMATES

Numerical examples have shown that the finite element method presented in this thesis yields very accurate solutions to initial-value ordinary differential equations, dynamics problems, and optimal control problems. It is the intent of this chapter to find a relationship between the step size Δt and the error of the integration performed [68]. Error estimates for dynamics have been performed by Hodges and Hou [69]. Therein, the authors find that the error for the linear oscillator problem to be $\frac{\Delta t^3}{12}$. It is noted that to obtain general error estimates for mixed methods is not state of the art.

The first part of this chapter will be concerned with comparing the local truncation error of Euler's method, a second-order Runge-Kutta method, and the finite element method for the initial-value ordinary differential equation $\dot{x} = f(x,t)$. The approximation to $x(t)$ will first be given for each of the three methods, and the Taylor Series expansion of each approximation will be derived. Then, the local truncation error of the three methods will be compared to the exact solution for four different $f(x,t)$'s. The finite element method will be seen to be proportional to Δt^3 for all examples.

The second part of the chapter will involve studying the error of the finite element method in optimal control problems. A control problem will be solved using the weak form developed in Chapter 2. It is possible to develop equations for all the unknown variables in terms of Δt. Consequently, convergence to the exact solution is proven and the error in each variable is shown to be proportional to Δt^2.

9.1. Initial-Value Problems

Consider the differential equation

$$\dot{x} = f(x,t) \tag{9.1–1}$$

where x is a scalar, t is the time, and $\hat{x}_0$ is the given initial condition. It is desired to find the local truncation error involved with the integration of this differential equation. Several approximation methods are explored. These are Euler's method (also known as a first-order Taylor Series method), a second-order Runge-Kutta method, and the finite element method which is developed in Appendix B. Following the summary of these methods, four example problems will be examined.

9.1.1. Methods of Solution

Let Δt be a small time step and let the value of x at $t = \Delta t$ be denoted by $\hat{x}$. To compare the error of an approximation method with the exact answer, $\hat{x}$ will be expanded in a Taylor Series in Δt about $\Delta t = 0$.

The value of $\hat{x}$ using Euler's method [70] will be denoted by $\hat{x}_{\text{EM}}$ and is defined as

$$\hat{x}_{\text{EM}} = \hat{x}_0 + f(\hat{x}_0, 0)\Delta t \tag{9.1.1–1}$$

This method is already in a Taylor Series form where only first-order terms in Δt are present.

The value of $\hat{x}$ using a second-order Runge-Kutta method [70] will be denoted by $\hat{x}_{\text{RK}}$. Defining $g = f(\hat{x}_0, 0)$, then $\hat{x}_{\text{RK}}$ is given as

$$\hat{x}_{\mathrm{RK}} = \hat{x}_0 + \Delta t f\left(\hat{x}_0 + \frac{\Delta t}{2} g, \frac{\Delta t}{2}\right) \tag{9.1.1–2}$$

For clarity, define

$$f_{\mathrm{RK}} = f\left(\hat{x}_0 + \frac{\Delta t}{2} g, \frac{\Delta t}{2}\right) \tag{9.1.1–3}$$

where it is noted that f_{RK} is *always* an explicit function of Δt. The Taylor Series expansion of Eq. (9.1.1–2) is

$$\hat{x}_{\mathrm{RK}} \approx \hat{x}_0 + f_{\mathrm{RK}}(\hat{x}_0, 0)\Delta t + 2\frac{\partial f_{\mathrm{RK}}(\hat{x}_0, 0)}{\partial \Delta t}\frac{\Delta t^2}{2} + 3\frac{\partial^2 f_{\mathrm{RK}}(\hat{x}_0, 0)}{\partial \Delta t^2}\frac{\Delta t^3}{6} \tag{9.1.1–4}$$

An expression for the approximation of $\hat{x}$ using the finite element method, $\hat{x}_{\mathrm{FE}}$, is not quite as easy to produce. This is because the finite element method is an *implicit* method. The equations derived in Appendix B (and Chapter 2 also, but for a different reason) are

$$\begin{gathered} \bar{x} - \frac{\Delta t}{2} f(\bar{x}, \bar{t}) = \hat{x}_0 \\ \hat{x}_{\mathrm{FE}} + \hat{x}_0 = 2\bar{x} \end{gathered} \tag{9.1.1–5}$$

Eq. (9.1.1–5) may be rewritten as one equation involving only nodal (*i.e.,* hatted) values. This equation is

$$\hat{x}_{\mathrm{FE}} - \Delta t f\left(\frac{\hat{x}_{\mathrm{FE}} + \hat{x}_0}{2}, \frac{\Delta t}{2}\right) = \hat{x}_0 \tag{9.1.1–6}$$

In order to find a Taylor Series approximation for $\hat{x}_{\mathrm{FE}}$ in the above equation, it is necessary to rewrite $\hat{x}_{\mathrm{FE}}$ as a polynomial in Δt with unknown coefficients. Thus, let

$$\hat{x}_{\mathrm{FE}} = \sum_{i=0}^{\infty} \hat{x}_{i+1} \Delta t^{i} \tag{9.1.1–7}$$

Eq. (9.1.1–7) is now substituted into Eq. (9.1.1–6) resulting in

$$G = \sum_{i=0}^{\infty} \hat{x}_{i+1} \Delta t^{i} - \Delta t f \left(\frac{\hat{x}_0 + \sum_{i=0}^{\infty} \hat{x}_{i+1} \Delta t^{i}}{2}, \frac{\Delta t}{2} \right) - \hat{x}_0 = 0 \tag{9.1.1–8}$$

Again for clarity, define

$$f_{\mathrm{FE}} = f \left(\frac{\hat{x}_0 + \sum_{i=0}^{\infty} \hat{x}_{i+1} \Delta t^{i}}{2}, \frac{\Delta t}{2} \right) \tag{9.1.1–9}$$

where f_{FE} is an explicit function of Δt. To find the unknown coefficients $\hat{x}_1$, $\hat{x}_2$, etc., the Taylor Series expansion of Eq. (9.1.1–8) will be taken about $\Delta t = 0$, and the coefficients of each Δt term will be set equal to zero. The required derivatives of G with respect to Δt, denoted by superscripted numbers, are

$$
\begin{aligned}
G &= \sum_{i=0}^{\infty} \hat{x}_{i+1}\Delta t^{i} - \Delta t f_{\text{FE}} - \hat{x}_0 \\
G^{(1)} &= \sum_{i=1}^{\infty} i\hat{x}_{i+1}\Delta t^{i-1} - f_{\text{FE}} - \Delta t f_{\text{FE}}^{(1)} \\
G^{(2)} &= \sum_{i=2}^{\infty} i(i-1)\hat{x}_{i+1}\Delta t^{i-2} - 2f_{\text{FE}}^{(1)} - \Delta t f_{\text{FE}}^{(2)} \\
&\vdots \\
G^{(n)} &= \sum_{i=n}^{\infty} n!\hat{x}_{i+1}\Delta t^{i-n} - n f_{\text{FE}}^{(n-1)} - \Delta t f_{\text{FE}}^{(n)}
\end{aligned}
\tag{9.1.1–10}
$$

Since G and the derivatives of G when evaluated at $\Delta t = 0$ are the coefficients in the Taylor Series approximation of G, then the unknown coefficients for $\hat{x}_{\text{FE}}$ are found by setting each equation above equal to zero. This results in

$$
\begin{aligned}
G\big|_{\Delta t=0} &= 0 = \hat{x}_1 - \hat{x}_0 \\
G^{(1)}\big|_{\Delta t=0} &= 0 = \hat{x}_2 - f_{\text{FE}}|_{\Delta t=0} \\
G^{(2)}\big|_{\Delta t=0} &= 0 = 2\hat{x}_3 - 2f_{\text{FE}}^{(1)}|_{\Delta t=0} \\
&\vdots \\
G^{(n)}\big|_{\Delta t=0} &= 0 = n!\hat{x}_{n+1} - (n-1)!f_{\text{FE}}^{(n-1)}|_{\Delta t=0}
\end{aligned}
\tag{9.1.1–11}
$$

Solving Eq. (9.1.1–11) for the unknown coefficients results in

$$
\begin{aligned}
\hat{x}_1 &= \hat{x}_0 \\
\hat{x}_2 &= f_{\text{FE}}|_{\Delta t=0} \\
&\vdots \\
\hat{x}_n &= \frac{1}{(n-2)!} f_{\text{FE}}^{(n-2)}|_{\Delta t=0}
\end{aligned}
\tag{9.1.1–12}
$$

So, finally, the approximation to $\hat{x}$ using the finite element method is given by Eq. (9.1.1–7) where the coefficients of the polynomial are given by Eq. (9.1.1–12).

The Taylor Series approximation of the Euler method, a second-order Runge-Kutta method, and the finite element method have now been derived. These equations are grouped together below for convenience. The coefficients in Eq. (9.1.1–12) have been substituted into Eq. (9.1.1–7).

$$\begin{aligned}
\hat{x}_{\mathrm{EM}} &= \hat{x}_0 + f(\hat{x}_0, 0)\Delta t \\
\hat{x}_{\mathrm{RK}} &\approx \hat{x}_0 + f_{\mathrm{RK}}|_{\Delta t=0}\Delta t + f_{\mathrm{RK}}^{(1)}|_{\Delta t=0}\Delta t^2 + \frac{1}{2} f_{\mathrm{RK}}^{(2)}|_{\Delta t=0}\Delta t^3 \\
\hat{x}_{\mathrm{FE}} &\approx \hat{x}_0 + f_{\mathrm{FE}}|_{\Delta t=0}\Delta t + f_{\mathrm{FE}}^{(1)}|_{\Delta t=0}\Delta t^2 + \frac{1}{2} f_{\mathrm{FE}}^{(2)}|_{\Delta t=0}\Delta t^3
\end{aligned} \tag{9.1.1–13}$$

where

$$f_{\mathrm{RK}} = f\left(\hat{x}_0 + \frac{\Delta t}{2} f(\hat{x}_0, 0), \frac{\Delta t}{2}\right) \tag{9.1.1–14}$$

and

$$f_{\mathrm{FE}} = f\left(\frac{\hat{x}_0 + \sum_{i=0}^{\infty} \hat{x}_{i+1}\Delta t^i}{2}, \frac{\Delta t}{2}\right) \tag{9.1.1–15}$$

9.1.2. Example Problems

Four example problems with exact solutions will now be examined to compare the local truncation error of the three methods described above. It will be shown that the finite element method is as good or better than the second-order Runge-Kutta method for all the example problems.

The first example will be a simple linear differential equation in x. Suppose $f(x,t) = x$ and $\hat{x}_0 = 1$ so that Eq. (9.1–1) becomes

$$\dot{x} = x \tag{9.1.2–1}$$

The exact answer is $x(t) = \exp(t)$ so that $\hat{x}_{\text{EX}}$, the exact value of x at $t = \Delta t$ is given as

$$\hat{x}_{\text{EX}} = \exp(\Delta t) \approx 1 + \Delta t + \frac{\Delta t^2}{2} + \frac{\Delta t^3}{6} \tag{9.1.2–2}$$

By using Eqs. (9.1.1–14) and (9.1.1–15), f_{RK} and f_{FE} are found to be

$$\begin{aligned} f_{\text{RK}} &= f\left(\hat{x}_0 + \frac{\Delta t}{2}\hat{x}_0, \frac{\Delta t}{2}\right) = 1 + \frac{\Delta t}{2} \\ f_{\text{FE}} &= f\left(\frac{\hat{x}_0 + \sum_{i=0}^{\infty} \hat{x}_{i+1}\Delta t^i}{2}, \frac{\Delta t}{2}\right) = \frac{1 + \sum_{i=0}^{\infty} \hat{x}_{i+1}\Delta t^i}{2} \end{aligned} \tag{9.1.2–3}$$

and the unknown coefficients above are found from Eq. (9.1.1–12). Now, Eq. (9.1.1–13) may be used to find the Taylor Series representation for each method. These expressions are

$$
\begin{aligned}
\hat{x}_{\mathrm{EM}} &= 1 + \Delta t \\
\hat{x}_{\mathrm{RK}} &= 1 + \Delta t + \frac{\Delta t^2}{2} \\
\hat{x}_{\mathrm{FE}} &\approx 1 + \Delta t + \frac{\Delta t^2}{2} + \frac{\Delta t^3}{4}
\end{aligned}
\tag{9.1.2–4}
$$

The local truncation error of the three integration schemes is found by comparing terms of the Taylor Series in Eq. (9.1.2–4) with the exact solution in Eq. (9.1.2–2). The errors, denoted by e, are found to be

$$
\begin{aligned}
e_{\mathrm{EM}} &= \frac{\Delta t^2}{2} && \text{for Euler's method} \\
e_{\mathrm{RK}} &= \frac{\Delta t^3}{6} && \text{for a Runge-Kutta method} \\
e_{\mathrm{FE}} &= \frac{\Delta t^3}{12} && \text{for the finite element method}
\end{aligned}
\tag{9.1.2–5}
$$

Thus, it is easily seen that the finite element method is a superior integration scheme to the Euler method or second-order Runge-Kutta method for this linear differential equation.

The second example involves a nonlinear differential equation in x only. Let

$$\dot{x} = 1 + \cos x \tag{9.1.2–6}$$

with $\hat{x}_0 = 0$. The exact solution to this equation is

$$x(t) = 2 \tan^{-1} t \tag{9.1.2–7}$$

Following the same procedure as in the first example of finding f_{RK}, f_{FE}, and the unknown $\hat{x}$'s, then Eq. (9.1.1–13) and the Taylor Series of Eq. (9.1.2–7) may be used to write down the following expressions.

$$\begin{aligned} \hat{x}_{\text{EM}} &= 2\Delta t \\ \hat{x}_{\text{RK}} &\approx 2\Delta t - \frac{1}{2}\Delta t^3 \\ \hat{x}_{\text{FE}} &\approx 2\Delta t - \frac{1}{2}\Delta t^3 \\ \hat{x}_{\text{EX}} &\approx 2\Delta t - \frac{2}{3}\Delta t^3 \end{aligned} \tag{9.1.2–8}$$

The errors are found to be

$$\begin{aligned} e_{\text{EM}} &= \frac{2\Delta t^3}{3} \\ e_{\text{RK}} &= \frac{\Delta t^3}{6} \\ e_{\text{FE}} &= \frac{\Delta t^3}{6} \end{aligned} \tag{9.1.2–9}$$

For this case, the finite element method and the second-order Runge-Kutta method have the same error, and the error is proportional to Δt^3.

The third example deals with a function of x and t. Consider

$$\dot{x} = x^2 \cos t \qquad \text{with} \qquad \hat{x}_0 = 1 \tag{9.1.2–10}$$

The exact solution for this problem is $x(t) = (1 - \sin t)^{-1}$. The Taylor Series approximations for the three approximation methods and the exact solution are

$$
\begin{aligned}
\hat{x}_{\mathrm{EM}} &= 1 + \Delta t \\
\hat{x}_{\mathrm{RK}} &\approx 1 + \Delta t + \Delta t^2 + \frac{\Delta t^3}{8} \\
\hat{x}_{\mathrm{FE}} &\approx 1 + \Delta t + \Delta t^2 + \frac{9\Delta t^3}{8} \\
\hat{x}_{\mathrm{EX}} &\approx 1 + \Delta t + \Delta t^2 + \frac{5\Delta t^3}{6}
\end{aligned}
\tag{9.1.2–11}
$$

and the errors are found to be

$$
\begin{aligned}
e_{\mathrm{EM}} &= \Delta t^2 \\
e_{\mathrm{RK}} &= \frac{17\Delta t^3}{24} \\
e_{\mathrm{FE}} &= \frac{7\Delta t^3}{24}
\end{aligned}
\tag{9.1.2–12}
$$

Once again, the finite element method is more accurate than the second-order Runge-Kutta method. The local truncation error is on the order of Δt^3.

For the fourth and final example, consider the special case of Eq. (9.1–1) where f is an integrable function of t only. Then, Eq. (9.1–1) becomes

$$
\dot{x} = f(t) \tag{9.1.2–13}
$$

Eq. (9.1.1–6) for the finite element method becomes

$$
\hat{x}_{\mathrm{FE}} = \hat{x}_0 + \Delta t f(\frac{\Delta t}{2}) \tag{9.1.2–14}
$$

Note that this is now an *explicit* equation for $\hat{x}_{\mathrm{FE}}$ and the Taylor Series expansion may be found without specifying a particular $f(t)$. The Taylor Series expansion for

Euler's method, the Runge-Kutta method, and the finite element method are found from Eqs. (9.1.1–13) and (9.1.2–14). Note that $f_{\mathrm{RK}} = f_{\mathrm{FE}} = f(\Delta t/2)$ so that

$$f^{(n)}_{\mathrm{RK}} = f^{(n)}_{\mathrm{FE}} = \frac{1}{2^n} f^{(n)}(\frac{\Delta t}{2}) \qquad (9.1.2\text{–}15)$$

where again the superscripts refer to derivatives with respect to Δt.

These expressions, along with the expansion of the exact solution, and the local truncation error associated with each method are given in Table 9.1. A summary of the errors for each example problem studied in this section is given in Table 9.2.

Table 9.1: Taylor Series and error versus exact for $\dot{x} = f(t)$

Method	Taylor Series Expansion	Error
Euler	$\hat{x}_0 + f(0)\Delta t$	$\frac{\partial f(0)}{\partial \Delta t}\frac{\Delta t^2}{2}$
Runge-Kutta	$\hat{x}_0 + f(0)\Delta t + \frac{\partial f(0)}{\partial \Delta t}\frac{\Delta t^2}{2} + \frac{\partial^2 f(0)}{\partial \Delta t^2}\frac{\Delta t^3}{8}$	$\frac{\partial^2 f(0)}{\partial \Delta t^2}\frac{\Delta t^3}{24}$
finite element	$\hat{x}_0 + f(0)\Delta t + \frac{\partial f(0)}{\partial \Delta t}\frac{\Delta t^2}{2} + \frac{\partial^2 f(0)}{\partial \Delta t^2}\frac{\Delta t^3}{8}$	$\frac{\partial^2 f(0)}{\partial \Delta t^2}\frac{\Delta t^3}{24}$
Exact	$\hat{x}_0 + f(0)\Delta t + \frac{\partial f(0)}{\partial \Delta t}\frac{\Delta t^2}{2} + \frac{\partial^2 f(0)}{\partial \Delta t^2}\frac{\Delta t^3}{6}$	$--$

Table 9.2: Summary of errors

$f(x,t)$	e_{EM}	e_{RK}	e_{FE}
x	$\Delta t^2/2$	$\Delta t^3/6$	$\Delta t^3/12$
$1+\cos x$	$2\Delta t^3/3$	$\Delta t^3/6$	$\Delta t^3/6$
$x^2 \cos t$	Δt^2	$17\Delta t^3/24$	$7\Delta t^3/24$
$f(t)$	$f^{(1)}(0)\Delta t^2/2$	$f^{(2)}(0)\Delta t^3/24$	$f^{(2)}(0)\Delta t^3/24$

9.2. Optimal Control Problems

The weak Hamiltonian finite element formulation is a mixed method, which is to say that there is more than one field variable. Developments of mathematical proofs of convergence and expressions for error bounds are not state-of-the-art for mixed methods; however, some feel for the accuracy of the weak Hamiltonian formulation may be obtained by studying a simple example problem. Afterwards, an error plot from the fixed-time problem of Chapter 2 will be examined.

Consider the following optimal control problem where x and u are scalars.

$$\begin{aligned} J &= \frac{1}{2}x(1)^2 + \int_0^1 \frac{1}{2}u^2\,dt \\ \dot{x} &= tu \qquad \text{with} \qquad x(0) = 4 \end{aligned} \tag{9.2–1}$$

The exact solution is readily found to be

$$\begin{aligned} \lambda(t) &= 3 \\ x(t) &= 4 - t^3 \\ u(t) &= -3t \end{aligned} \tag{9.2–2}$$

It is desired to find an error estimate of each unknown variable using the finite element formulation. Since the error estimate will be a function of the element size Δt, then the number of elements $N = 1/\Delta t$ will be left as a free parameter.

The equations to be solved are easily verified to be

$$\begin{aligned} \hat{\lambda}_0 - \bar{\lambda}^{(1)} &= 0 \\ \bar{\lambda}^{(i)} - \bar{\lambda}^{(i+1)} &= 0 \qquad \text{for} \qquad i = 1, 2, \ldots, N-1 \\ \bar{\lambda}^{(N)} - \hat{x}_f &= 0 \end{aligned} \tag{9.2–3}$$

for the costates,

$$\bar{u}^{(i)} + \bar{\lambda}^{(i)}\bar{t}^{(i)} = 0 \qquad \text{for} \qquad i = 1, 2, \ldots, N \tag{9.2–4}$$

for the controls, and

$$\begin{aligned} \bar{x}^{(1)} - \frac{\Delta t}{2}\bar{t}^{(1)}\bar{u}^{(1)} &= 4 \\ \bar{x}^{(i+1)} - \bar{x}^{(i)} - \frac{\Delta t}{2}\left[\bar{t}^{(i)}\bar{u}^{(i)} + \bar{t}^{(i+1)}\bar{u}^{(i+1)}\right] &= 0 \qquad \text{for} \qquad i = 1, 2, \ldots, N-1 \\ -\bar{x}^{(N)} - \frac{\Delta t}{2}\bar{t}^{(N)}\bar{u}^{(N)} + \hat{x}_f &= 0 \end{aligned} \tag{9.2–5}$$

for the states. Note that $\bar{t}^{(i)}$ is an average time value for the i^{th} element and can be expressed as

$$\bar{t}^{(i)} = \frac{2i-1}{2}\Delta t \qquad \text{for} \qquad i = 1, 2, \ldots, N \tag{9.2–6}$$

There are $3N + 2$ equations with $3N + 2$ unknowns which are $\bar{x}^{(i)}, \bar{\lambda}^{(i)}$, and $\bar{u}^{(i)}$ for $i = 1, 2, \ldots, N$, and $\hat{x}_f$ and $\hat{\lambda}_0$. As will be shown, it is possible to find an expression for $\hat{x}_f$ solely as a function of Δt. Then, all the unknowns can be expressed as a function of Δt and the errors of each unknown can be found as compared to the exact solution.

To start with, the $N + 1$ equations in Eq. (9.2–3) will be solved for the costates yielding

$$\hat{\lambda}_0 = \bar{\lambda}^{(i)} = \cdots = \bar{\lambda}^{(N)} = \hat{x}_f \tag{9.2–7}$$

Next, the control variables are found from Eq. (9.2–4) to be

$$\bar{u}^{(i)} = -\bar{\lambda}^{(i)}\bar{t}^{(i)} = -\hat{x}_f \bar{t}^{(i)} \qquad \text{for} \qquad i = 1, 2, \ldots, N \tag{9.2–8}$$

After substituting Eq. (9.2–8) for $\bar{u}^{(i)}$ and Eq. (9.2–6) for $\bar{t}^{(i)}$ into Eq. (9.2–5), then what remains are $N+1$ equations for the $N+1$ unknown states. These equations are

$$\begin{aligned}
\bar{x}^{(1)} + \hat{x}_f \left(\frac{\Delta t}{2}\right)^3 &= 4 \\
\bar{x}^{(i+1)} - \bar{x}^{(i)} + \hat{x}_f \left(\frac{\Delta t}{2}\right)^3 (8i^2 + 2) &= 0 \qquad \text{for} \qquad i = 1, 2, \ldots, N-1 \\
-\bar{x}^{(N)} + \hat{x}_f \left(\frac{\Delta t}{2}\right)^3 (2N-1)^2 + \hat{x}_f &= 0
\end{aligned} \tag{9.2–9}$$

Solving the middle of the above equations for $\bar{x}^{(i)}$ in terms of $\hat{x}_f$ results in

$$\bar{x}^{(i)} = \bar{x}^{(N)} + \hat{x}_f \left(\frac{\Delta t}{2}\right)^3 \sum_{j=i}^{N-1} (8j^2 + 2) = 0 \qquad \text{for} \qquad i = 1, 2, \ldots, N-1 \tag{9.2–10}$$

Using Eq. (9.2–10) to find an expression for $\bar{x}^{(1)}$, it is now possible to solve the first and last of Eq. (9.2–9) for $\bar{x}^{(N)}$ and $\hat{x}_f$. The equation for $\hat{x}_f$ is

$$\hat{x}_f = \frac{4}{\left(\frac{\Delta t}{2}\right)^3 \left[(2N-1)^2 + 1 + \sum_{i=1}^{N-1}(8i^2 + 2)\right] + 1} \tag{9.2–11}$$

This messy expression is of more value than it may seem at first. By a series of simplifications, it will be possible to prove that $\hat{x}_f$ converges to the exact solution as the number of elements N approaches infinity. Furthermore, it will be possible to find an estimate of the error for each of the unknowns.

The simplification will begin with the denominator of Eq. (9.2–11). Noting that $\Delta t = 1/N$, the denominator becomes

$$\left(\frac{1}{2N}\right)^3 \left[(2N-1)^2 + 1 + \sum_{i=1}^{N-1}(8i^2+2)\right] + 1 \tag{9.2–12}$$

For the case of $N > 1$, Eq. (9.2–12) becomes

$$\left(\frac{1}{8N^3}\right)\left(4N^2 - 4N + 2 + 2(N-1) + 8\sum_{i=1}^{N-1} i^2\right) + 1 \tag{9.2–13}$$

Making use of the following identity

$$\sum_{i=1}^{N-1} i^2 = \frac{(N-1)N[2(N-1)+1]}{6} \tag{9.2–14}$$

in Eq. (9.2–13), then simplification results in

$$\frac{1}{8N^3}\left(\frac{8}{3}N^3 - \frac{2}{3}N\right) + 1 = \frac{4}{3} - \frac{1}{12N^2} \tag{9.2–15}$$

Now, after replacing $N = 1/\Delta t$ in the above equation, then Eq. (9.2–11) becomes

$$\hat{x}_f = \frac{4}{\frac{4}{3} - \frac{\Delta t^2}{12}} \tag{9.2–16}$$

Note that as $N \to \infty$, $\Delta t \to 0$ so $\hat{x}_f = 3$ which is the exact answer. Thus, at least for this problem, convergence is guaranteed as more and more elements are taken.

To study the error in $\hat{x}_f$, the Taylor Series expansion of Eq. (9.2–16) in Δt will be taken about $\Delta t = 0$. The resulting equation is

$$\hat{x}_f \approx 3 + \frac{3}{8}\frac{\Delta t^2}{2} - \frac{9}{16}\frac{\Delta t^3}{6} \tag{9.2–17}$$

The exact value of $\hat{x}_f$ is 3, so the error is $\frac{3}{16}\Delta t^2$. The final value of the state is the least accurate of all the state variables.

Since all the costate variables are the same and equal to $\hat{x}_f$ (see Eq. 9.2–7), then the error in the costates for this problem is equal to $3\Delta t^2/16$. Finally, consider the error in the control variable at the point $t = 0.5$. This is a non-moving node (for N even) and allows for an error estimate of the control. From Eqs. (9.2–8) and (9.2–17),

$$\hat{u} = -\hat{x}_f/2 = -1.5 - \frac{3}{32}\Delta t^2 + \frac{9}{192}\Delta t^3 \tag{9.2–18}$$

The exact control is $\hat{u}_{\text{EX}} = -1.5$, so the error is

$$\frac{3\Delta t^2}{32} \tag{9.2–19}$$

In summary, convergence of this simple control problem has been proven as the number of elements increases. In addition, it has been shown that the error in the costates and the final value of the state is proportional to Δt^2. The error for the control is also proportional to Δt^2.

It would be difficult, if not impossible, to use this procedure on a more complicated problem; however, from observed numerical results, it is believed that the finite element method is second-order (Δt^2) accurate for almost all problems. To help support this statement, consider once again the fixed-time trajectory problem presented in Chapter 2. Fig. 2.8 (repeated in Fig. 9.1) shows a plot of the relative error of the performance index $J = \hat{x}_{f,(3)}$ and the endpoint multiplier ν_1 versus the number of elements. The relative error of the final control value is also included. In Chapter 2, it was noted that the slope of the line is about -2 which indicates that the error varies inversely with the square of N, or that the error is proportional to Δt^2. This is similar to *a-posteriori* error bounds as formulated in usual finite-element applications [53].

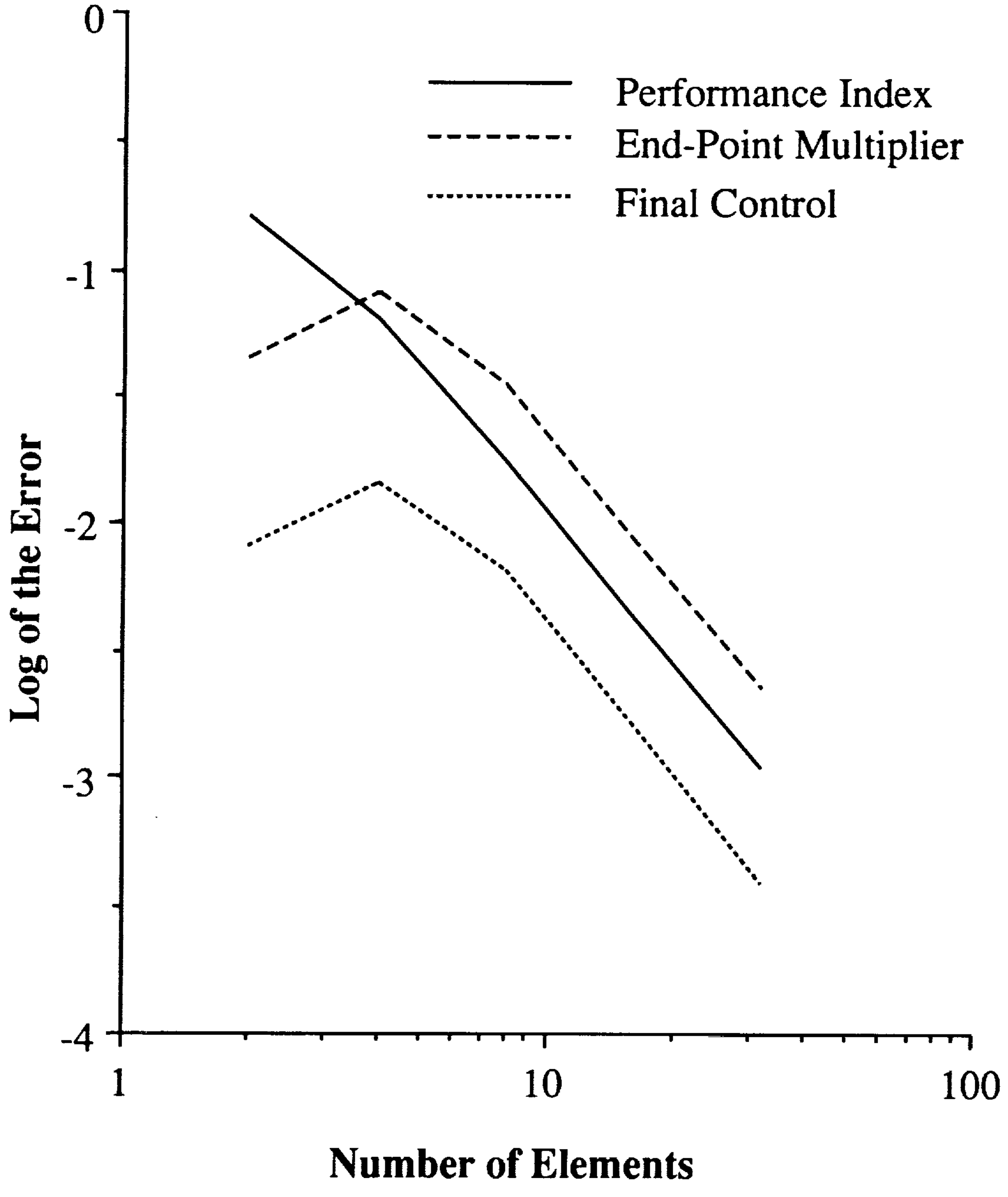

Fig. 9.1: Relative error of the performance index, ν_1, and final control vs. N

CHAPTER 10

INITIAL GUESSES

Almost all of the problems solved to date have used a Newton-Raphson procedure to solve the nonlinear algebraic equations. This procedure, of course, requires initial guesses. And not all initial guesses will converge to the solution.

Several different ideas have been used throughout the course of this research. For example, in the two examples presented in Chapter 2, initial guesses were chosen that were not too different from the boundary conditions given in the problem. This method worked well for these simple problems and converged solutions were obtained without much difficulty. Of course, this idea is nothing more than a trial and error method.

As the problems studied became more difficult, initial guesses became harder to obtain. This was due in part to the fact that the costates have little physical meaning and hence their range of possible values are generally unknown. For the Saturn one-stage model, the solution was obtained by using the method of Levenberg-Marquardt [55] which did not require very good guesses. The two-stage solution was obtained by "slowly" perturbing the vehicle parameters and boundary conditions and resolving the problem at every perturbation. Solutions for the actual specified conditions were obtained fairly rapidly using this procedure.

10.1. Taylor Series Approach

With the attempt to automate the program to handle different vehicle parameters and boundary conditions more easily, a new initial guess procedure was developed. This approach is based on the Taylor series expansion of one of the states. The method is outlined below.

Recall from Eq. (3.1–1) of the Saturn one-stage model the following equations for m (mass), h (altitude), E (specific energy), and γ (flight-path angle):

$$\begin{aligned} \dot{m} &= f_1(h) \\ \dot{h} &= f_2(h, E, \gamma) \\ \dot{E} &= f_3(m, h, E, \gamma, u) \\ \dot{\gamma} &= f_4(m, h, E, \gamma, u) \end{aligned} \tag{10.1–1}$$

The first element discretized equations for the states are

$$\begin{aligned} \bar{m}^{(1)} - \frac{\Delta t}{2}\bar{f}_1^{(1)} &= \hat{m}_0 \\ \bar{h}^{(1)} - \frac{\Delta t}{2}\bar{f}_2^{(1)} &= \hat{h}_0 \\ \bar{E}^{(1)} - \frac{\Delta t}{2}\bar{f}_3^{(1)} &= \hat{E}_0 \\ \bar{\gamma}^{(1)} - \frac{\Delta t}{2}\bar{f}_4^{(1)} &= \hat{\gamma}_0 \end{aligned} \tag{10.1–2}$$

where $\bar{x}^{(1)}$ denotes the midpoint value of the state x in the first element, Δt is an arbitrary time step, $\bar{f}^{(1)}$ is the value of f evaluated at the midpoint of the first element, and $\hat{x}_0$ is the given initial condition of the state x.

Using a Taylor series expansion for $\bar{h}^{(1)}$ results in

$$\bar{h}^{(1)} \approx \hat{h}_0 + \frac{\Delta t}{2} f_2(\hat{h}_0, \hat{E}_0, \hat{\gamma}_0) \tag{10.1–3}$$

Now an actual value for $\bar{h}^{(1)}$ has been obtained. If Eq. (10.1–3) is substituted into the second equation of Eq. (10.1–2), then the resulting equation after simplification is

$$\bar{f}_2^{(1)}(\bar{h}^{(1)}, \bar{E}^{(1)}, \bar{\gamma}^{(1)}) = f_2(\hat{h}_0, \hat{E}_0, \bar{\gamma}_0) \qquad (10.1\text{–}4)$$

The first, third and fourth equations from Eq. (10.1–2), along with the above equation are four equations with the four unknowns ($\bar{m}^{(1)}, \bar{E}^{(1)}, \bar{\gamma}^{(1)}$, and $\bar{u}^{(1)}$). Once the midpoint values are found, the nodal values are extracted and the process is repeated. We thus time march until the boundary conditions are approximately satisfied. Initial guesses on the costates are then obtained by time marching backwards from the final boundary conditions. The only unknown boundary condition for the costates was found by evaluating the Hamiltonian at the final time.

The above procedure did generate guesses for all the unknowns and the guesses did indeed converge on the solution.

Unfortunately, there are several serious drawbacks to this idea. First, there is no way to control the boundary conditions that are to be satisfied. In other words, even if the final conditions on the states were changed, the same initial guesses would still be generated. Second, the optimality condition is not satisfied. The control solved for is not optimal and is, in fact, nothing more than a parameter to satisfy the dynamical equations. Third, and most important, there is no guarantee whatsoever that the initial guesses will converge. Since a fully automated code is desired, a different method to generate initial guesses must be found.

10.2. Homotopy and Continuation Methods

An algorithm for computing Brouwer fixed points wherein the homotopy procedure has guaranteed convergence is proposed in [71]. The homotopy method was used to generate a numerical algorithm in [72]. Therein, the author develops "beautifully simple" analytical steps that allow one to find roots of equations.

The method involves following the zero curve of the homotopy map

$$\rho_a(\lambda, x) = \lambda\,[x - f(x)] + (1 - \lambda)(x - a)$$

starting from $(0,a)$. The zero curve is parameterized by arc length s and it can be shown that the zero curve of ρ_a emanating from $(0,a)$ is the solution of an initial value problem. When the solution of the initial value problem reaches $\lambda = 1$, the corresponding x is a fixed point of f. Virtually any a will lead to a root of f. The art of this procedure lies in following the zero curve accurately, but not too closely as this leads to increased computational expense.

The homotopy method has been applied to nonlinear two-point boundary value problems [73,74]. Also, [75] proposes a homotopy algorithm for sparse systems of nonlinear equations, which is precisely what the weak principle yields. The algorithm leads to substantial computational savings.

Homotopy methods are rather involved algorithms. When dealing with well-behaved equations that have unique solutions, simpler methods may often be used to find fixed points of nonlinear algebraic equations. Therefore, a continuation method [76] has been adopted to generate initial guesses. The method is now described.

The most general set of n simultaneous algebraic equations in n unknowns $x_1, \ldots, x_n$ can be written as

$$f_i(x_1, \ldots, x_n) = 0 \qquad \text{for} \qquad i = 1, \ldots, n \tag{10.2–1}$$

Let $y_1(\tau), \ldots, y_n(\tau)$ be a set of n functions of a variable τ, with $0 \leq \tau \leq 1$ and take

$$y_i(0) = k_i \qquad \text{for} \qquad i = 1, \ldots, n \tag{10.2–2}$$

where k_i is a constant and selected arbitrarily. Now, require that $y_1(\tau), \ldots, y_n(\tau)$ satisfy the equations

$$f_i(y_1, \ldots, y_n) = f_i(k_1, \ldots, k_n)(1 - \tau) \qquad \text{for} \qquad i = 1, \ldots, n \tag{10.2–3}$$

Then, as the right-hand side of Eq. (10.2–3) vanishes at $\tau = 1$, the functions $y_1(\tau), \ldots, y_n(\tau)$ satisfy, at $\tau = 1$, precisely the same equations as do $x_1, \ldots, x_n$ [see Eq. (10.2–1)]. Now, $y_1(1), \ldots, y_n(1)$, and, hence $x_1, \ldots, x_n$, may be found as follows: Differentiate Eq. (10.2–3) with respect to τ, thus obtaining the set of first-order differential equations

$$\begin{aligned} \frac{\partial f_1}{\partial y_1}\frac{dy_1}{d\tau} + \cdots + \frac{\partial f_1}{\partial y_n}\frac{dy_n}{d\tau} &= -f_1(k_1, \ldots, k_n) \\ &\vdots \\ \frac{\partial f_n}{\partial y_1}\frac{dy_1}{d\tau} + \cdots + \frac{\partial f_n}{\partial y_n}\frac{dy_n}{d\tau} &= -f_n(k_1, \ldots, k_n) \end{aligned} \tag{10.2–4}$$

and perform a numerical integration of these equations using Eq. (10.2–2) as initial conditions and terminating the integration at $\tau = 1$.

As was stated previously, $k_1, \ldots, k_n$ may be assigned any values whatsoever. However, it can occur that, for certain choices of $k_1, \ldots, k_n$, some of $y_1, \ldots, y_n$ do not possess real values for some values of τ in the interval $0 \leq \tau \leq 1$, in which event the numerical integration of the differential equations cannot be carried to completion. When this happens, one simply changes one or more of $k_1, \ldots, k_n$. In general, results are obtained most expeditiously when $k_1, \ldots, k_n$ are good approximations to $x_1, \ldots, x_n$, respectively. Fortunately, in connection with physical problems, one can often make good guesses regarding $x_1, \ldots, x_n$, and hence assign suitable values to $k_1, \ldots, k_n$. Finally, it is worth noting that many distinct sets of values of $k_1, \ldots, k_n$ can lead to the same values of $x_1, \ldots, x_n$.

Let's look at a simple example of the use of the above procedure. Consider the scalar equation

$$f(x) = x - \cos x = 0 \tag{10.2–5}$$

Now, let $y(0) = k = 0$ so that Eq. (10.2–3) takes the form

$$f(y) = y - \cos y = f(k)(1 - \tau) = \tau - 1 \tag{10.2–6}$$

The equation corresponding to Eq. (10.2–4) is

$$(1 + \sin y)\frac{dy}{dt} = 1 \tag{10.2–7}$$

so that

$$\frac{dy}{dt} = (1 + \sin y)^{-1} \tag{10.2–8}$$

A second-order Runge-Kutta method is used to numerically integrate Eq. (10.2–8). A time step of 0.02 was used. When the integration was completed at $\tau = 1$, then $y(1)$, and thus x, was found to have a value of 0.7390654, which is correct to the fifth decimal place.

This continuation method is used in the general code described in the next chapter. In practice, the integration is performed and then one or two Newton-Raphson iterations are required to obtain the final answers. This could be avoided by using a more accurate integration scheme, but this tends to lead to increased computational effort.

CHAPTER 11

AN ALGORITHM FOR OPTIMAL CONTROL PROBLEMS

The weak principle for optimal control problems has been fully developed in Chapters 2, 4, 6, and 7. The formulation is capable of solving optimal control problems that have continuous states, costates, and controls, and problems with discontinuities arising from staging (*i.e.*, discontinuities in the system equations), control inequality constraints and state inequality constraints. The algebraic equations which come from the weak formulation may be derived prior to specifying the problem to be solved. It is this feature in particular that allows for a general problem-solving environment to be created.

The main goal of the general code is to reliably solve a large class of optimal control problems with a *minimum* of user interaction. Specifically, it is desired to create an environment where the user does not have to write subroutines. To this end, a general code has been developed on a SUN 3/260 workstation and requires a FORTRAN 77 compiler, MACSYMA [77], and the Harwell subroutine library [59]. The general procedure can be broken into three parts that must interface together. The first part is the FORTRAN code. This code contains all the subroutines necessary to solve any of the optimal control problems described above. However, if certain problems require table look-up routines (such as aerodynamic data for a rocket model), then these subroutines must be given by the user and interfaced to the rest of the general code. Thus, there may be a need for some user programming for certain problems. The second part of the general procedure is the use of MACSYMA. The user must supply an input file specifying the problem. This input file is in symbolic form and will be loaded into MACSYMA. MACSYMA will then evaluate all the necessary expressions and automatically generate the FORTRAN code. This code is spliced into a template file and becomes one of the subroutines. The third and final part of the general procedure will consists of subroutines to generate initial guesses that will reliably converge. The continuation method described in Chapter 10 is being used. This method converts the algebraic equations to initial-value ordinary differential equations. A second-order Runge-Kutta method is used to integrate the equations and obtain initial guesses for a Newton-Raphson method. This method has worked on all the problems tested to date.

Every example problem in this thesis has been solved using the general code. Although this makes the code useful in and of its self, we wish to emphasize that the *setup* time required to solve these problems is the important factor. Setup time is the time required for a user to write all the proper input files and subroutines in order to run the program. The setup time to use any of the existing codes (see Chapter 1 for a discussion of these) can range from several hours to weeks. However, the setup time using the general code is only about 10 minutes for any problem (assuming that the subroutines for table look-up data are already available). This is because only a symbolic input file is required. MACSYMA and the FORTRAN subroutines do all the rest.

As mentioned above, the code can handle a wide range of problems. There are some limitations though. One limitation is that the code can currently only handle a problem with one or two stages. Also, the code can only handle a scalar state constraint. These restrictions are present because of the extra programming involved to remove them coupled with the fact that there is no way of testing the results analytically. It is felt that there are possibly better ways to handle state constraints by using a canonical form as is done in [39]. This is a good problem for future research (see Chapter 12).

Several example input files are now given to demonstrate the use of the general code. Afterwards, a sample output is given for a third-order state constraint problem. The chapter concludes with a chart of the times taken to solve the example problems.

11.1. Example Input Files

The last four pages of this section contain four example input files. These files contain four distinct cases to be discussed now.

Consider the free time trajectory optimization problem presented in Section 2.4. The first input file is used to solve this problem. The user is required to supply the number of states NS, the number of control constraints NP (zero in this example), the number of phases NPH (to be described shortly), the number of controls M, and the number of constraints on the states at the final time Q. The next series of lines from F[1] to F[4] define the system equations as given in Eqs. (2.3–1). After the equations are formed, the user supplies the performance index L and PHI. The variable S contains the state constraint, which is zero for this example since there are no constraints to be imposed. Then the Q constraints are given in PSI and the initial conditions are given in IC. Next the user supplies the final time TF and a guess at the value of the final time TFGUES. Since the final time is unknown, TF is set to zero and the user gives a guess at the final time. This guess only needs to be within an order of magnitude generally. Also, guesses for the states at the midpoint of the trajectory and the final point are given in XGUES. These guesses may be *very* crude and can even be zero for many problems. Since the final value of three of the states were known for this problem, crude guesses were easily and obviously obtained. Finally, the number of elements to be run is given in NE.

Regardless of the value of NE, the code automatically starts with the two element case and uses the continuation method of [76] and the Newton-Raphson method to solve the problem. The code then interpolates the solution to this case and runs a four element case using only the Newton-Raphson method. The code continues in this manner until NE is met. If the Newton-Raphson fails to converge for the four or higher element case (which is rare) then the program will start that case over and try the continuation method to solve the four element case.

The output was verified to be identical to the solution previously obtained. Again, we emphasize that not only is the method accurate as has been observed throughout this work, but now the general code can produce these results with a simple 30 line input file. This file was created in about five minutes and, as will be seen in the last section, answers for 2, 4, and 8 elements were obtained about 6.25 minutes thereafter.

The second input file is used to solve the control constraint example of Chapter 6. Most of the input is the same as in the preceding example. There are, however, a couple of important changes to make. One is that now the number of control constraints, NP, is 2. Also, these constraints are put in the F array at the end of the state equations. Thus, we see that F[2] and F[3] contain the two control constraints. Also note that this problem has an explicit dependence on time which is handled by the code. Finally, since this problem is a fixed time problem, TF is set to the known value and TFGUES is also set to this value. These values must be equal for the code to know it is a fixed time problem.

The third input file introduces another new problem that the code can handle. This file will solve the advanced launch vehicle problem presented in Chapter 8. For *simplicity only* in describing this file, the atmospheric effects have been neglected since there are no analytical expressions for atmospheric and aerodynamic conditions. This example, although somewhat unrealistic due to the absence of atmospheric effects, does demonstrate the power and versatility of the general code. Several new variables are introduced that need to be explained. First, IST appears in the sixth line and is set to a value of 1 if there is a second stage involved. The next group of lines (7 – 16) are values defined only to simplify the writing of the state equations. F[1] – F[4] define the state equations in the first stage and F1[1] – F1[4] define the state equations in the second stage. The staging time is determined by PSITS, which is a constraint on the states at the staging time. Also, the known jump in the state is given in JUMP. The index number of JUMP tells the code which state is experiencing the discontinuity. Notice again the very crude guesses given for the states. It was essential that the code work with a minimum of physical insight into the problem.

As a final example file, a third-order state constraint problem is set up. This example is found in [65] and contains an analytical solution. A finite element code had not been previously written to solve this problem. Fortunately, the general code was able to solve the problem in a matter of minutes.

A couple of new features are present in this input file. One is that the number of phases NPH is something other than 1 for a change. The number of phases is used to indicate how often the trajectory enters or leaves a state constraint boundary. For this problem, the solution touches the boundary once, so NPH equals 2. Also new is an expression for S. S contains the state constraint equation to be enforced by the program. It is also found (from results not shown) that a lower (or stricter) state constraint causes two touch-points, in which case the user should choose NPH = 3. The user is given the task of determining the number of phases in a problem. If the incorrect number of phases are chosen that either the constraint will be violated or the multipliers which are supposed to be positive will be negative. It is hoped that this may be avoided eventually as the general code is further developed. The output of this example is given in the next section.

```
NS:4;
NP:0;
NPH:1;
M:1;
Q:3;
F[1]:1.12397*COS(U(1));
F[2]:1.12397*SIN(U(1));
F[3]:X(1);
F[4]:X(2);
L:1.0;
S:0;
PHI:0.0;
PSI[1]:X(1)-12.2129;
PSI[2]:X(2);
PSI[3]:X(4)-100.0;
IC[1]:0.0;
IC[2]:0.0;
IC[3]:0.0;
IC[4]:0.0;
TF:0.0;
TFGUES:10.0;
XGUES[1,1]:6.0;
XGUES[1,2]:12.2129;
XGUES[2,1]:1.0;
XGUES[2,2]:0.0;
XGUES[3,1]:50.0;
XGUES[3,2]:100.0;
XGUES[4,1]:50.0;
XGUES[4,2]:100.0;
NE:8;
```

```
NS:1;
NP:2;
NPH:1;
M:1;
Q:0;
F[1]:(1+T-3*T**2/17)*U(1);
F[2]:U(1)-1;
F[3]:-U(1)-1;
S:0.0;
L:0.5*U(1)**2;
PHI:0.5*X(1)**2;
IC[1]:-19.945596;
TF:10.0;
TFGUES:10.0;
XGUES[1,1]:-10.0;
XGUES[1,2]:-1.0;
NE:8;
```

```
NS:4;
NP:0;
M:1;
Q:3;
NPH:1;
IST:1;
TVAC1:2.58134E07;
TVAC2:7.74402E06;
AE1:37.515;
AE2:11.254;
S1:131.34;
S2:65.67;
ISP:430.0;
GRAV:9.81;
MU:3.9906E14;
RE:6378000.0;
H(X):= RE+X(2);
F[1]:-TVAC1/(GRAV*ISP);
F[2]:X(3)*SIN(X(4));
F[3]:TVAC1*COS(U(1))/X(1)  - MU*SIN(X(4))/H(X)^2;
F[4]:TVAC1*SIN(U(1))/(X(1)*X(3))
     + (X(3)/H(X)-MU/(X(3)*H(X)**2))*COS(X(4));
F1[1]:-TVAC2/(GRAV*ISP);
F1[2]:X(3)*SIN(X(4));
F1[3]:TVAC2*COS(U(1))/X(1)  - MU*SIN(X(4))/H(X)^2;
F1[4]:TVAC2*SIN(U(1))/(X(1)*X(3))
      + (X(3)/H(X)-MU/(X(3)*H(X)**2))*COS(X(4));
L:0.0;
PHI:X(1);
S:0.0;
PSITS:X(1)-645500.0;
JUMP[1]:98880.0;
PSI[1]:X(2)-148160;
PSI[2]:X(3)-7858.1995;
PSI[3]:X(4);
IC[1]:1.52345E06;
IC[2]:400.0;
IC[3]:160.0;
IC[4]:1.4;
TF:0.0;
TFGUES:300.0;
XGUES[1,1]:0.7E06;
XGUES[1,2]:10000.0;
XGUES[2,1]:70000.0;
XGUES[2,2]:148160.0;
XGUES[3,1]:3000.0;
XGUES[3,2]:7858.1995;
XGUES[4,1]:0.5;
XGUES[4,2]:.0;
NE:8;
```

```
NS:3;
NP:0;
NPH:2;
M:1;
Q:3;
F[1]:X(2);
F[2]:X(3);
F[3]:U(1);
S:X(1)-0.3;
L:0.5*U(1)**2;
PHI:0.0;
PSI[1]:X(1);
PSI[2]:X(2)+1;
PSI[3]:X(3)-2;
IC[1]:0.0;
IC[2]:1.0;
IC[3]:2.0;
TF:1.0;
TFGUES:1.0;
XGUES[1,1]:.3;
XGUES[1,2]:0.0;
XGUES[2,1]:0.0;
XGUES[2,2]:-1.0;
XGUES[3,1]:2.0;
XGUES[3,2]:2.0;
NE:8;
```

11.2. Sample output file

The output (given on the following pages) of the state constraint example consists of the solutions for the states, costates, controls, and Hamiltonian for 2, 4, and 8 elements *per phase.* At the top of each page is the total elapsed computer time from the start of the program. On the two element case sheets is 10.40 secs. This is the time the code took to run the continuation method and the Newton-Raphson method for this case. This is a rather small number given the complexity of the problem and the fact that an accurate second-order Runge-Kutta method was used to solve the problem. The time at the top of the four element case is 11.94 which tells us that only 11.94 − 10.40 = 1.54 seconds was required to run the four-element case given the solution to the two element case. Finally, the desired eight element case solution was obtained in a *total* of 15.08 secs and only 3.14 secs from the four element case. Note that this time includes the extraction of nodal values and the production of the data files. This is a nonnegligible part of the total time.

In summary, a third order state constraint problem which might have taken several days or weeks to program from scratch was solved in about 10 or 15 minutes with the general code. The simple input file is typed in a few minutes and a few minutes are required by MACSYMA to create the FORTRAN subroutines. After that, the program runs in a matter of seconds.

```
                    NODAL VALUES FOR THE STATES

    NUMBER OF ELEMENTS =  2     TOTAL ELAPSED TIME =  10.40

     X1            X2            X3            X4           TIME

 0.00000E+00   0.10000E+01   0.20000E+01   0.00000E+00   0.00000E+00
 0.21250E+00   0.70000E+00   -.44000E+01   0.00000E+00   0.25000E+00
 0.30000E+00   0.00000E+00   -.12000E+01   0.00000E+00   0.50000E+00
 0.30000E+00   0.00000E+00   -.12000E+01   0.00000E+00   0.50000E+00
 0.21250E+00   -.70000E+00   -.44000E+01   0.00000E+00   0.75000E+00
 0.55511E-16   -.10000E+01   0.20000E+01   0.00000E+00   0.10000E+01
```

```
                    NODAL VALUES FOR THE STATES

      NUMBER OF ELEMENTS =   4      TOTAL ELAPSED TIME =   11.94

     X1             X2             X3             X4            TIME

0.00000E+00    0.10000E+01    0.20000E+01    0.00000E+00    0.00000E+00
0.12385E+00    0.98158E+00    -.22947E+01    0.00000E+00    0.12500E+00
0.22506E+00    0.63780E+00    -.32057E+01    0.00000E+00    0.25000E+00
0.28246E+00    0.28062E+00    -.25091E+01    0.00000E+00    0.37500E+00
0.30000E+00    0.27756E-15    -.19809E+01    0.00000E+00    0.50000E+00
0.30000E+00    0.27756E-15    -.19809E+01    0.00000E+00    0.50000E+00
0.28246E+00    -.28062E+00    -.25091E+01    0.00000E+00    0.62500E+00
0.22506E+00    -.63780E+00    -.32057E+01    0.00000E+00    0.75000E+00
0.12385E+00    -.98158E+00    -.22947E+01    0.00000E+00    0.87500E+00
-.13878E-15    -.10000E+01    0.20000E+01    0.00000E+00    0.10000E+01
```

```
                    NODAL VALUES FOR THE STATES

      NUMBER OF ELEMENTS =  8      TOTAL ELAPSED TIME =  15.08

     X1             X2             X3             X4            TIME

 0.00000E+00    0.10000E+01    0.20000E+01    0.00000E+00    0.00000E+00
 0.63949E-01    0.10464E+01    -.51643E+00    0.00000E+00    0.62500E-01
 0.12682E+00    0.96565E+00    -.20665E+01    0.00000E+00    0.12500E+00
 0.18238E+00    0.81215E+00    -.28453E+01    0.00000E+00    0.18750E+00
 0.22738E+00    0.62799E+00    -.30479E+01    0.00000E+00    0.25000E+00
 0.26086E+00    0.44308E+00    -.28695E+01    0.00000E+00    0.31250E+00
 0.28330E+00    0.27512E+00    -.25051E+01    0.00000E+00    0.37500E+00
 0.29595E+00    0.12965E+00    -.21499E+01    0.00000E+00    0.43750E+00
 0.30000E+00    -.55511E-15    -.19990E+01    0.00000E+00    0.50000E+00
 0.30000E+00    -.55511E-15    -.19990E+01    0.00000E+00    0.50000E+00
 0.29595E+00    -.12965E+00    -.21499E+01    0.00000E+00    0.56250E+00
 0.28330E+00    -.27512E+00    -.25051E+01    0.00000E+00    0.62500E+00
 0.26086E+00    -.44308E+00    -.28695E+01    0.00000E+00    0.68750E+00
 0.22738E+00    -.62799E+00    -.30479E+01    0.00000E+00    0.75000E+00
 0.18238E+00    -.81215E+00    -.28453E+01    0.00000E+00    0.81250E+00
 0.12682E+00    -.96565E+00    -.20665E+01    0.00000E+00    0.87500E+00
 0.63949E-01    -.10464E+01    -.51643E+00    0.00000E+00    0.93750E+00
 -.15266E-15    -.10000E+01    0.20000E+01    0.00000E+00    0.10000E+01
```

```
                    NODAL VALUES FOR THE COSTATES

      NUMBER OF ELEMENTS =  2     TOTAL ELAPSED TIME =  10.40

     L1             L2             L3             L4            TIME

0.20480E+04    0.66560E+03    0.76800E+02    0.00000E+00    0.00000E+00
0.20480E+04    0.15360E+03    -.25600E+02    0.00000E+00    0.25000E+00
0.20480E+04    -.35840E+03    -.71054E-14    0.00000E+00    0.50000E+00
-.20480E+04    -.35840E+03    -.71054E-14    0.00000E+00    0.50000E+00
-.20480E+04    0.15360E+03    0.25600E+02    0.00000E+00    0.75000E+00
-.20480E+04    0.66560E+03    -.76800E+02    0.00000E+00    0.10000E+01
```

```
                    NODAL VALUES FOR THE COSTATES

      NUMBER OF ELEMENTS =  4     TOTAL ELAPSED TIME =  11.94

     L1            L2            L3            L4            TIME

 0.90935E+03   0.33023E+03   0.51445E+02   0.00000E+00   0.00000E+00
 0.90935E+03   0.21656E+03   0.17271E+02   0.00000E+00   0.12500E+00
 0.90935E+03   0.10289E+03   -.26947E+01   0.00000E+00   0.25000E+00
 0.90935E+03   -.10779E+02   -.84517E+01   0.00000E+00   0.37500E+00
 0.90935E+03   -.12445E+03   0.46185E-13   0.00000E+00   0.50000E+00
 -.90935E+03   -.12445E+03   0.46185E-13   0.00000E+00   0.50000E+00
 -.90935E+03   -.10779E+02   0.84517E+01   0.00000E+00   0.62500E+00
 -.90935E+03   0.10289E+03   0.26947E+01   0.00000E+00   0.75000E+00
 -.90935E+03   0.21656E+03   -.17271E+02   0.00000E+00   0.87500E+00
 -.90935E+03   0.33023E+03   -.51445E+02   0.00000E+00   0.10000E+01
```

```
                    NODAL VALUES FOR THE COSTATES

        NUMBER OF ELEMENTS =  8     TOTAL ELAPSED TIME =  15.08

     L1            L2            L3            L4            TIME

 0.79917E+03   0.29734E+03   0.48774E+02   0.00000E+00   0.00000E+00
 0.79917E+03   0.24739E+03   0.31751E+02   0.00000E+00   0.62500E-01
 0.79917E+03   0.19745E+03   0.17850E+02   0.00000E+00   0.12500E+00
 0.79917E+03   0.14750E+03   0.70708E+01   0.00000E+00   0.18750E+00
 0.79917E+03   0.97549E+02   -.58689E+00   0.00000E+00   0.25000E+00
 0.79917E+03   0.47601E+02   -.51228E+01   0.00000E+00   0.31250E+00
 0.79917E+03   -.23476E+01   -.65370E+01   0.00000E+00   0.37500E+00
 0.79917E+03   -.52296E+02   -.48294E+01   0.00000E+00   0.43750E+00
 0.79917E+03   -.10224E+03   -.26645E-14   0.00000E+00   0.50000E+00
 -.79917E+03   -.10224E+03   -.26645E-14   0.00000E+00   0.50000E+00
 -.79917E+03   -.52296E+02   0.48294E+01   0.00000E+00   0.56250E+00
 -.79917E+03   -.23476E+01   0.65370E+01   0.00000E+00   0.62500E+00
 -.79917E+03   0.47601E+02   0.51228E+01   0.00000E+00   0.68750E+00
 -.79917E+03   0.97549E+02   0.58689E+00   0.00000E+00   0.75000E+00
 -.79917E+03   0.14750E+03   -.70708E+01   0.00000E+00   0.81250E+00
 -.79917E+03   0.19745E+03   -.17850E+02   0.00000E+00   0.87500E+00
 -.79917E+03   0.24739E+03   -.31751E+02   0.00000E+00   0.93750E+00
 -.79917E+03   0.29734E+03   -.48774E+02   0.00000E+00   0.10000E+01
```

```
                    ALL VALUES FOR CONTROL AND HAMILTONIAN

          NUMBER OF ELEMENTS =  2     TOTAL ELAPSED TIME =  10.40

     U1            U2            U3            HAMIL          TIME

 -.76800E+02   0.00000E+00   0.00000E+00   0.43008E+03   0.00000E+00
 -.25600E+02   0.00000E+00   0.00000E+00   0.92160E+03   0.12500E+00
 0.25600E+02   0.00000E+00   0.00000E+00   0.43008E+03   0.25000E+00
 0.12800E+02   0.00000E+00   0.00000E+00   0.92160E+03   0.37500E+00
 0.15843E-13   0.00000E+00   0.00000E+00   0.43008E+03   0.50000E+00
 0.71054E-14   0.00000E+00   0.00000E+00   0.43008E+03   0.50000E+00
 -.12800E+02   0.00000E+00   0.00000E+00   0.92160E+03   0.62500E+00
 -.25600E+02   0.00000E+00   0.00000E+00   0.43008E+03   0.75000E+00
 0.25600E+02   0.00000E+00   0.00000E+00   0.92160E+03   0.87500E+00
 0.76800E+02   0.00000E+00   0.00000E+00   0.43008E+03   0.10000E+01
```

```
                    ALL VALUES FOR CONTROL AND HAMILTONIAN

         NUMBER OF ELEMENTS =   4     TOTAL ELAPSED TIME =   11.94

      U1              U2              U3             HAMIL           TIME

 -.51445E+02    0.00000E+00    0.00000E+00    0.24651E+03    0.00000E+00
 -.34358E+02    0.00000E+00    0.00000E+00    0.27045E+03    0.62500E-01
 -.17271E+02    0.00000E+00    0.00000E+00    0.24651E+03    0.12500E+00
 -.72880E+01    0.00000E+00    0.00000E+00    0.27045E+03    0.18750E+00
 0.26947E+01    0.00000E+00    0.00000E+00    0.24651E+03    0.25000E+00
 0.55732E+01    0.00000E+00    0.00000E+00    0.27045E+03    0.31250E+00
 0.84517E+01    0.00000E+00    0.00000E+00    0.24651E+03    0.37500E+00
 0.42258E+01    0.00000E+00    0.00000E+00    0.27045E+03    0.43750E+00
 -.58764E-13    0.00000E+00    0.00000E+00    0.24651E+03    0.50000E+00
 -.46185E-13    0.00000E+00    0.00000E+00    0.24651E+03    0.50000E+00
 -.42258E+01    0.00000E+00    0.00000E+00    0.27045E+03    0.56250E+00
 -.84517E+01    0.00000E+00    0.00000E+00    0.24651E+03    0.62500E+00
 -.55732E+01    0.00000E+00    0.00000E+00    0.27045E+03    0.68750E+00
 -.26947E+01    0.00000E+00    0.00000E+00    0.24651E+03    0.75000E+00
 0.72880E+01    0.00000E+00    0.00000E+00    0.27045E+03    0.81250E+00
 0.17271E+02    0.00000E+00    0.00000E+00    0.24651E+03    0.87500E+00
 0.34358E+02    0.00000E+00    0.00000E+00    0.27045E+03    0.93750E+00
 0.51445E+02    0.00000E+00    0.00000E+00    0.24651E+03    0.10000E+01
```

```
                    ALL VALUES FOR CONTROL AND HAMILTONIAN

            NUMBER OF ELEMENTS =  8     TOTAL ELAPSED TIME =  15.08

     U1             U2             U3            HAMIL           TIME

-.48774E+02   0.00000E+00   0.00000E+00   0.20438E+03   0.00000E+00
-.40263E+02   0.00000E+00   0.00000E+00   0.20918E+03   0.31250E-01
-.31751E+02   0.00000E+00   0.00000E+00   0.20438E+03   0.62500E-01
-.24801E+02   0.00000E+00   0.00000E+00   0.20918E+03   0.93750E-01
-.17850E+02   0.00000E+00   0.00000E+00   0.20438E+03   0.12500E+00
-.12461E+02   0.00000E+00   0.00000E+00   0.20918E+03   0.15625E+00
-.70708E+01   0.00000E+00   0.00000E+00   0.20438E+03   0.18750E+00
-.32419E+01   0.00000E+00   0.00000E+00   0.20918E+03   0.21875E+00
0.58689E+00   0.00000E+00   0.00000E+00   0.20438E+03   0.25000E+00
0.28549E+01   0.00000E+00   0.00000E+00   0.20918E+03   0.28125E+00
0.51228E+01   0.00000E+00   0.00000E+00   0.20438E+03   0.31250E+00
0.58299E+01   0.00000E+00   0.00000E+00   0.20918E+03   0.34375E+00
0.65370E+01   0.00000E+00   0.00000E+00   0.20438E+03   0.37500E+00
0.56832E+01   0.00000E+00   0.00000E+00   0.20918E+03   0.40625E+00
0.48294E+01   0.00000E+00   0.00000E+00   0.20438E+03   0.43750E+00
0.24147E+01   0.00000E+00   0.00000E+00   0.20918E+03   0.46875E+00
0.44409E-15   0.00000E+00   0.00000E+00   0.20438E+03   0.50000E+00
0.26645E-14   0.00000E+00   0.00000E+00   0.20438E+03   0.50000E+00
-.24147E+01   0.00000E+00   0.00000E+00   0.20918E+03   0.53125E+00
-.48294E+01   0.00000E+00   0.00000E+00   0.20438E+03   0.56250E+00
-.56832E+01   0.00000E+00   0.00000E+00   0.20918E+03   0.59375E+00
-.65370E+01   0.00000E+00   0.00000E+00   0.20438E+03   0.62500E+00
-.58299E+01   0.00000E+00   0.00000E+00   0.20918E+03   0.65625E+00
-.51228E+01   0.00000E+00   0.00000E+00   0.20438E+03   0.68750E+00
-.28549E+01   0.00000E+00   0.00000E+00   0.20918E+03   0.71875E+00
-.58689E+00   0.00000E+00   0.00000E+00   0.20438E+03   0.75000E+00
0.32419E+01   0.00000E+00   0.00000E+00   0.20918E+03   0.78125E+00
0.70708E+01   0.00000E+00   0.00000E+00   0.20438E+03   0.81250E+00
0.12461E+02   0.00000E+00   0.00000E+00   0.20918E+03   0.84375E+00
0.17850E+02   0.00000E+00   0.00000E+00   0.20438E+03   0.87500E+00
0.24801E+02   0.00000E+00   0.00000E+00   0.20918E+03   0.90625E+00
0.31751E+02   0.00000E+00   0.00000E+00   0.20438E+03   0.93750E+00
0.40263E+02   0.00000E+00   0.00000E+00   0.20918E+03   0.96875E+00
0.48774E+02   0.00000E+00   0.00000E+00   0.20438E+03   0.10000E+01
```

We conclude this chapter with a chart showing the actual clock time required to solve the above four problems. The four problems (to read the chart) are I, the free time trajectory optimization problem of Chapter 2, II, the control constraint problem of Chapter 6, III, the advanced launch vehicle of Chapter 8, and IV, the state constraint example problem presented above.

The time to type in the different input files is approximately the same for all problems, around 5 to 10 minutes. The first column in Table 11.1 gives the clock time (in minutes) it took for MACSYMA to read in the input file and produce the two necessary FORTRAN subroutines to be used by the general code. These subroutines contain analytical first and second mixed partial derivatives for all the state equations. The second column gives the total elapsed time (in minutes) until the program generated all the requested results. This includes compilation and linking of the new subroutines created by MACSYMA. The third column gives the time required (in seconds) to generate each set of results for 2, 4, and 8 elements. These are the times given at the top of each output page as described above. It is seen by studying Table 11.1 that even the complicated advanced launch vehicle problem was solved in just 8 minutes and 20 seconds, from start to finish. Also, the majority of the time is used by MACSYMA to generate the subroutines.

Table 11.1: Setup and run times for the general code

Problem	Setup Time (min:sec)	Total Time (min:sec)	Run Time (sec) 2/4/8
I	5 : 05	6 : 15	7.80/9.28/11.14
II	2 : 45	4 : 00	3.68/4.80/6.12
III	6 : 10	8 : 20	21.76/25.60/33.38
IV	3 : 30	4 : 40	10.40/11.94/15.08

CHAPTER 12

CONCLUSIONS AND FUTURE RESEARCH

This thesis was concerned with a method for the solution of optimal control problems. The method, called the weak principle for optimal control, is based on time-domain finite elements. The goal of the weak principle was to develop an efficient and accurate algorithm in the hopes that it might be capable of producing optimal trajectory solutions in a real-time environment. This would lead to tremendous savings in terms of time and money for many space related projects.

Some of the characteristics and features of the weak principle are summarized below.

(1) The necessary conditions of optimality are satisfied and all strong boundary conditions are transformed into weak boundary conditions. The weak principle finds candidate extremal solutions, *i.e.*, ones that satisfy all the necessary conditions.

(2) Because all strong boundary conditions are cast in the form of natural or weak boundary conditions, then the same shape functions may be chosen for every optimal control problem.

(3) The choice of shape functions allowed by the weak principle permits all integration to be done by inspection, regardless of the degree of nonlinearity in the problem. Thus, no errors are introduced as the result of numerical quadrature.

(4) One tremendous advantage of the integration being done by inspection is that algebraic equations may be derived prior to specifying the problem to be solved.

Because the form of the algebraic equations is known, it was possible to create a general algorithm for the solution of optimal control problems.

(5) The algebraic equations which come from the weak principle possess a *very* sparse Jacobian. When this sparsity is exploited by way of a smart sparse matrix solver, a very efficient algorithm may be produced.

(6) Symmetry in the solution is manifested by the weak principle. Problems, such as those presented in Chapter 2, that possess a state, costate, or control that is analytically symmetric, will also have symmetric approximated solutions obtained by the weak principle.

These features of the weak principle make it a powerful and versatile tool for solving optimal control problems. Particular attention was given to the application of the method to advanced launch vehicle guidance. A real-time algorithm was to be developed which requires that the optimal trajectory be computed in a small time interval as compared to the total time interval. This thesis presented a solution to this optimization problem which was obtained in about 5.5 secs as compared to the 360 second time span of the entire trajectory. This fact, coupled with the accuracy demonstrated by the weak principle, make real-time trajectory optimization a possibility.

A general code for the solution of optimal control problems was developed based on the weak principle. It was possible to create a general purpose, robust, and efficient algorithm because the algebraic equations are known before the problem is specified. The most promising feature of the general code is the reduction in setup time for any given problem over any other existing formulation. Problems that might take weeks to program and solve can be solved in a matter of minutes with the general code. This can lead to tremendous savings in terms of time, money, and human resources.

There is still work that can be done for developing the weak principle further. Below are a few ideas.

(1) The algebraic equations are derived in terms of the midpoint values of the shape functions. For problems with staging or state constraints, certain nodal values appear and must be recovered around the unknown time(s). It may be easier if all the algebraic equations were rewritten in terms of the nodal values. This may also be more efficient since the nodal values need to be recovered anyway. However, the resulting Jacobian may not be as sparse as with the midpoint values.

(2) It was noted that the algebraic equations were all linear in terms of the costates. It may be possible to use MACSYMA to symbolically solve for the costates in terms of the other unknowns. The advantages of this would be two-fold. One is that the number of equations would be cut almost in half and there would be no associated decrease in the percentage of zeros in the Jacobian. Second, there would be no need to obtain initial guesses for the costates. This is helpful because there is often times little or no physical insight into the magnitude of the costate variables.

(3) It may be possible to generate a canonical form for constraints that will allow for easier programming of multiple constraints. A canonical form, which is successfully used in [39], could have several advantages, including that it might allow for multiple constraints and that there may also be a way of eliminating the users need for estimating *a priori* and iterating on the number of phases present in the solution.

(4) Work can be continued on the weak principle to remove the singularity associated with state constraint problems when using the necessary conditions of Ref. 65. This would also allow more uniformity in the programming.

(5) The weak principle could be extended to include singular control problems (see [1]). Singular control problems are characterized by the control appearing linearly in the formulation. Since the optimality condition does not determine

the control, successive time derivatives are taken until the control does appear. It should be possible to incorporate this type of problem into the weak principle.

(6) Finally, the general code can be continually updated. Along with the suggestions above that can be incorporated into the general code, there are other potential savings to be explored. One is the generation of the subroutines by MACSYMA. Perhaps there are more efficient ways of producing the needed code, or perhaps Mathematica (another symbolic manipulator) would be better. In addition, the general code would be very useful if it were available for desktop computers. One drawback of existing codes is that many of them only run on large, and sometimes inaccessible, machines. A fast and efficient code to run on desktop computers would be of great value to industry and academia.

APPENDIX A

HAMILTON'S WEAK PRINCIPLE FOR DYNAMICS

The potential of obtaining a direct solution in the time domain is very much analogous to obtaining the solution of a beam deflection problem with the beam axial coordinate broken into several segments or finite elements. In the present case, however, it is the time interval which is broken into segments; thus, the phrase "finite elements in time" has been adopted by several investigators.

Only recently has a mixed formulation of Hamilton's Weak Principle (HWP) been investigated as a computational tool for finite elements in time [47]. In this section, the mixed form of HWP is derived and its application to dynamics problems is illustrated.

A.1. General Development

To this aim, let us consider an arbitrary holonomic mechanical system. The configuration is completely defined by a set of generalized coordinates q. Further, let us denote with $L(q, \dot{q}, t)$ the Lagrangean of the system, Q the set of nonconservative generalized forces applied to the system, and $p = \partial L / \partial \dot{q}$ the set of generalized momenta. The generalized coordinates q should be piecewise differentiable and the generalized momenta p will have discrete values at t_0 and t_f. (For a more mathematically rigorous discussion, see [53].) Then the following variational equation, known as HWP [45], describes the real motion of the system between the two *known* times t_0 and t_f:

$$\int_{t_0}^{t_f} \delta L \, dt + \int_{t_0}^{t_f} \delta q^T Q \, dt = \delta q_f^T \hat{p}_f - \delta q_0^T \hat{p}_0 \qquad (\text{A.1}-1)$$

where δq, the variation of q, should be of the same class of functions as is q, $\delta q_f = \delta q(t = t_f)$ and $\delta q_0 = \delta q(t = t_0)$. Although HWP contains p in the form of discrete values at the end points denoted by the hatted quantities, this particular variational equation is said to be in displacement form because it only involves the variation of q. Keeping $\hat{p}$ distinct from $\partial L / \partial \dot{q}$ allows for accurate extraction of the momenta without differentiation of q. Although this formulation has been shown to be of practical use in dynamics (see [45] and [46]), an even more useful formulation may be derived if independent variations in both displacements and momenta are allowed, resulting in a mixed formulation.

In order to derive the mixed formulation, let the Hamiltonian be defined as

$$H(q, p, t) \equiv p^T \dot{q} - L(q, \dot{q}, t) \qquad (\text{A.1}-2)$$

Taking the variation of Eq. (A.1–2) and substituting for δL in Eq. (A.1–1) results in

$$\int_{t_0}^{t_f} \left(\delta p^T \dot{q} + \delta \dot{q}^T p - \delta H + \delta q^T Q \right) \, dt = \delta q_f^T \hat{p}_f - \delta q_0^T \hat{p}_0 \qquad (\text{A.1}-3)$$

Now, introducing

$$q|_{t_0} \triangleq \lim_{t \to t_0^+} q(t) \quad \text{and} \quad q|_{t_f} \triangleq \lim_{t \to t_f^-} q(t) \qquad (12\text{–}1)$$

and

$$\hat{q}|_{t_0} \triangleq q(t_0) \quad \text{and} \quad \hat{q}|_{t_f} \triangleq q(t_f) \qquad (12\text{–}2)$$

then continuity between the values of q and p on the interior and $\hat{q}$ and $\hat{p}$ on the boundary is weakly enforced by adjoining $\delta\alpha^T(q-\hat{q})|_{t_0}^{t_f}$ to Eq. (A.1–3) where $\delta\alpha$ is a set of discrete unknown Lagrange multipliers defined only at t_0 and t_f. The resulting equation is

$$\int_{t_0}^{t_f} \left(\delta p^T\dot{q} + \delta\dot{q}^T p - \delta H + \delta q^T Q\right) dt = \delta q_f^T \hat{p}_f - \delta q_0^T \hat{p}_0 + \delta\alpha^T(q-\hat{q})|_{t_0}^{t_f} \quad (\text{A.1}-4)$$

It is now possible to choose $\delta\alpha = \delta\lambda$ without changing any necessary conditions. Also, to finish the development, the first term in Eq. (A.1–4) is integrated by parts yielding

$$\int_{t_0}^{t_f} \left(\delta\dot{q}^T p - \delta\dot{p}^T q - \delta H + \delta q^T Q\right) dt = \delta q_f^T \hat{p}_f - \delta q_0^T \hat{p}_0 - \delta p_f^T \hat{q}_f + \delta p_0^T \hat{q}_0 \quad (\text{A.1}-5)$$

This is called a mixed formulation because it contains independent variations of q and p. It is also in the "weakest" possible form in the sense that all boundary conditions are of the natural type, enforced by the variational equation for unconstrained variations. Note that now p *and* q should have discrete values at t_0 and t_f, p and q should be piecewise continuous, and δp and δq should be piecewise differentiable (C^0).

There are two main advantages of the mixed formulation over the displacement formulation. The first advantage is that the mixed formulation generally provides a more accurate solution for a given level of computational effort than does the displacement formulation. The second advantage is that a simpler choice of shape functions is allowed. Note in Eq. (A.1–5) that time derivatives of δq and δp are present. However, no time derivatives of q and p exist. Therefore, it is possible to implement linear shape functions for δq and δp and constant shape functions for q and p within each element.

A.2. Finite Elements in Time

Let the time interval from t_0 to t_f be broken into N equally spaced elements. The nodal values of these elements are t_i for $i = 1, \dots, N+1$ where $t_0 = t_1$ and $t_f = t_{N+1}$. A nondimensional elemental time τ is defined as

$$\tau = \frac{t - t_i}{t_{i+1} - t_i} = \frac{t - t_i}{\Delta t_i} \tag{A.2 – 1}$$

The linear shape functions for the virtual coordinates and momenta are

$$\begin{aligned} \delta q &= \delta q_i(1-\tau) + \delta q_{i+1}\tau \\ \delta p &= \delta p_i(1-\tau) + \delta p_{i+1}\tau \end{aligned} \tag{A.2 – 2}$$

For the generalized coordinates and momenta

$$q = \begin{cases} \hat{q}_i & \text{if } \tau = 0; \\ \bar{q}_i & \text{if } 0 < \tau < 1; \\ \hat{q}_{i+1} & \text{if } \tau = 1 \end{cases} \tag{A.2 – 3}$$

and

$$p = \begin{cases} \hat{p}_i & \text{if } \tau = 0; \\ \bar{p}_i & \text{if } 0 < \tau < 1; \\ \hat{p}_{i+1} & \text{if } \tau = 1 \end{cases} \tag{A.2 – 4}$$

It is important to understand that $\hat{q}_1 = q(t_0), \hat{p}_1 = p(t_0), \hat{q}_{N+1} = q(t_f)$, and $\hat{p}_{N+1} = p(t_f)$. In other words, the hatted values of q and p at the beginning and end of our time marching scheme *are* the discrete values of q and p that are needed in the mixed formulation. When these shape functions are substituted into Eq. (A.1–5), one can either generate an implicit time-marching procedure for nonlinear problems or apply standard finite element assembly procedures to solve periodic or two-point boundary value problems [48]. When this formulation is applied to the linear oscillator, a time-marching algorithm emerges that is unconditionally stable [45]. Higher-order (so-called p-version) elements could be developed [69], and they would certainly be attractive for linear problems or for nonlinear problems with nonlinearities of low order. For nonlinear problems in general, use of the crude shape functions allowable with the mixed method would seem to be more efficient than use of higher-order shape functions in a p-version. The reason for this is that, with the exception of the term involving Q, which may contain time explicitly, all element quadrature can be done by inspection regardless of the order of the nonlinearities.

A.3. Example: A Nonlinear Initial-Value Problem

Applying the shape functions of Eqs. (A.2–2 – A.2–4) to Eq. (A.1–5) for an initial value problem, a recursive set of nonlinear algebraic equations is obtained of the form

$$f_j\left(\bar{q}_i, \hat{q}_{i+1}, \bar{p}_i, \hat{p}_{i+1}\right) = 0 \qquad j = 1, 2, \ldots, n \qquad \text{(A.3 – 1)}$$

where n is four times the number of degrees of freedom of the system. Eq. (A.3–1) can be solved by a Newton-Raphson method yielding an implicit time-marching procedure. The key advantage of using finite elements and a weak variational approach over numerical integration is that the solution (for linear problems) is stable for *all* time steps. In other words, no matter how large a time step is used, a finite approximation of the solution will be obtained. This unconditional stability is obtained without *ad hoc* procedures such as selective or reduced element quadrature which are necessary in displacement formulations.

Also noteworthy of the finite element discretization is that the midpoint values of q and p are just the average values of the adjoining nodal values, or

$$\begin{aligned} 2\bar{q}_i &= \hat{q}_i + \hat{q}_{i+1} \\ 2\bar{p}_i &= \hat{p}_i + \hat{p}_{i+1} \end{aligned} \qquad \text{(A.3 – 2)}$$

Thus, it is possible to cut the number of equations and unknowns in half. This can be very useful for a multi-degree of freedom problem in terms of computer savings.

Consider a simple pendulum composed of a lumped mass m and a weightless bar of length ℓ (see Fig. A.1). The single generalized coordinate q is the angular displacement of the bar from the vertical. Denoting the kinetic energy of the system with K and the potential energy with V, then we may define the following:

$$\begin{aligned} K &= \frac{1}{2} m\ell^2 \dot{q}^2 \\ V &= mg\ell(1-\cos q) \\ L &= K - V \\ p &= \frac{\partial L}{\partial \dot{q}} = m\ell^2 \dot{q} \\ H &= \frac{p^2}{2m\ell^2} + mg\ell(1-\cos q) \end{aligned} \tag{A.3 - 3}$$

There are no nonconservative forces Q applied to this system.

Substituting $t = t_i + \tau\Delta t_i$ from Eq. (A.2–1), along with Eq. (A.3–3), and substituting the shape functions defined in Eqs. (A.2–2 – A.2–4) into Eq. (A.1–5) we obtain (for $i = 1, 2, \ldots, N$)

$$\begin{aligned} \Delta t_i \int_0^1 \Bigg\{ & \left(\frac{\delta q_{i+1} - \delta q_i}{\Delta t_i}\right) \bar{p}_i - mg\ell \sin \bar{q}_i \left[\delta q_i(1-\tau) + \delta q_{i+1}\tau\right] \\ & - \left(\frac{\delta p_{i+1} - \delta p_i}{\Delta t_i}\right) \bar{q}_i - \left(\frac{\bar{p}_i}{m\ell^2}\right) \left[\delta p_i(1-\tau) + \delta p_{i+1}\tau\right] \Bigg\} d\tau \\ & - \delta q_{i+1}\hat{p}_{i+1} + \delta p_{i+1}\hat{q}_{i+1} + \delta q_i \hat{p}_i - \delta p_i \hat{q}_i = 0 \end{aligned} \tag{A.3 - 4}$$

Carrying out the integration by inspection and setting the coefficient of each virtual quantity ($\delta q_i, \delta p_i, \delta q_{i+1}$, and δp_{i+1}) to zero, the following four independent equations for each value of i are obtained.

$$\begin{aligned}\hat{p}_i - \bar{p}_i - \frac{mg\ell\Delta t_i \sin\bar{q}_i}{2} &= 0\\ \bar{p}_i - \hat{p}_{i+1} - \frac{mg\ell\Delta t_i \sin\bar{q}_i}{2} &= 0\\ \bar{q}_i - \hat{q}_i - \frac{\bar{p}_i\Delta t_i}{2m\ell^2} &= 0\\ \hat{q}_{i+1} - \bar{q}_i - \frac{\bar{p}_i\Delta t_i}{2m\ell^2} &= 0\end{aligned} \qquad (A.3-5)$$

There are six unknowns; however, for an initial-value problem, we will specify $\hat{q}_i$ and $\hat{p}_i$ and solve for the remaining unknowns as outlined below. Thus, Eq. (A.3–5) is of the form of Eq. (A.3–1).

Recall that i ranges from 1 to N. To start with, $i = 1$ and $\hat{q}_1$ and $\hat{p}_1$ (i.e. the initial conditions) are specified. Now, solve for $\bar{q}_1, \bar{p}_1, \hat{q}_2$, and $\hat{p}_2$. Next, let $i = 2$, use the known $\hat{q}_2$ and $\hat{p}_2$ (we just found those values), and solve for the new four unknowns. This process is repeated until $i = N$.

For this simple pendulum example, the variables will be nondimensionalized as follows. If we define $\omega^2 = g/\ell$, then a dimensionless time step $\Delta\bar{t}$ may be defined that does not vary with i so that $\Delta\bar{t} = \omega\Delta t_i$. Also, instead of solving directly for p, the dimensionless $p/m\ell^2\omega$ will be solved for.

The initial conditions of the pendulum are $\hat{q}_1 = 60°$ and $\hat{p}_1 = 0.0$. The equations will be solved for $\Delta\bar{t} = 0.4$, 0.8, and 1.6. Graphs of the solutions are shown in Fig. A.2 and Fig. A.3 and compared to the exact elliptic integral solution [78]. (Since the element midpoint values are simply the average of the element nodal values, only the nodal values are given for these results.) From Figs. A.2 and A.3, it is easily seen that $\Delta\bar{t} = 0.4$ gives acceptable results for both displacement and angular momentum. Also, note that even the large 1.6 time step yields a finite approximation of the exact solution.

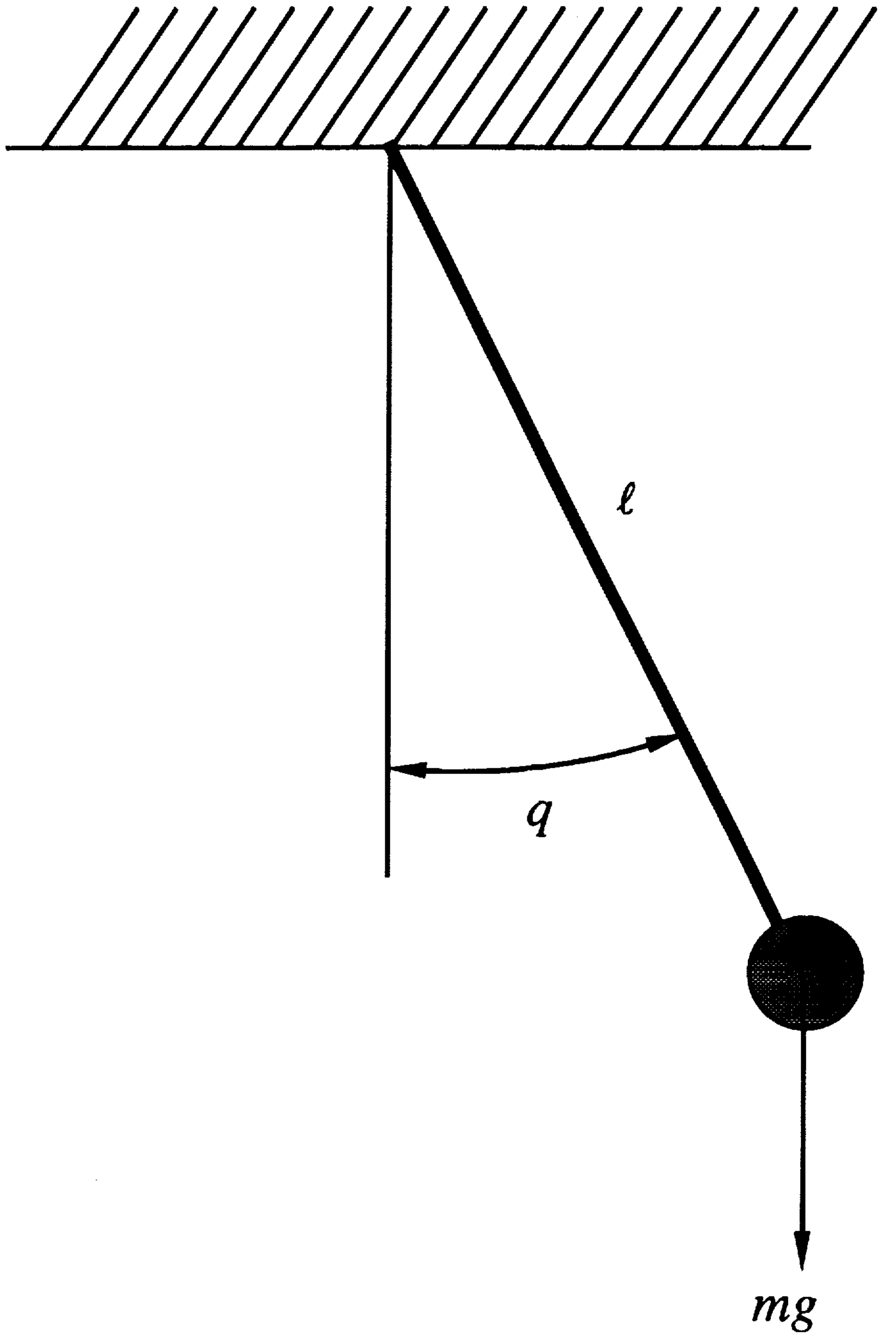

Fig. A.1: Nomenclature for example

A simple pendulum composed of a lumped mass m and a weightless bar of length ℓ

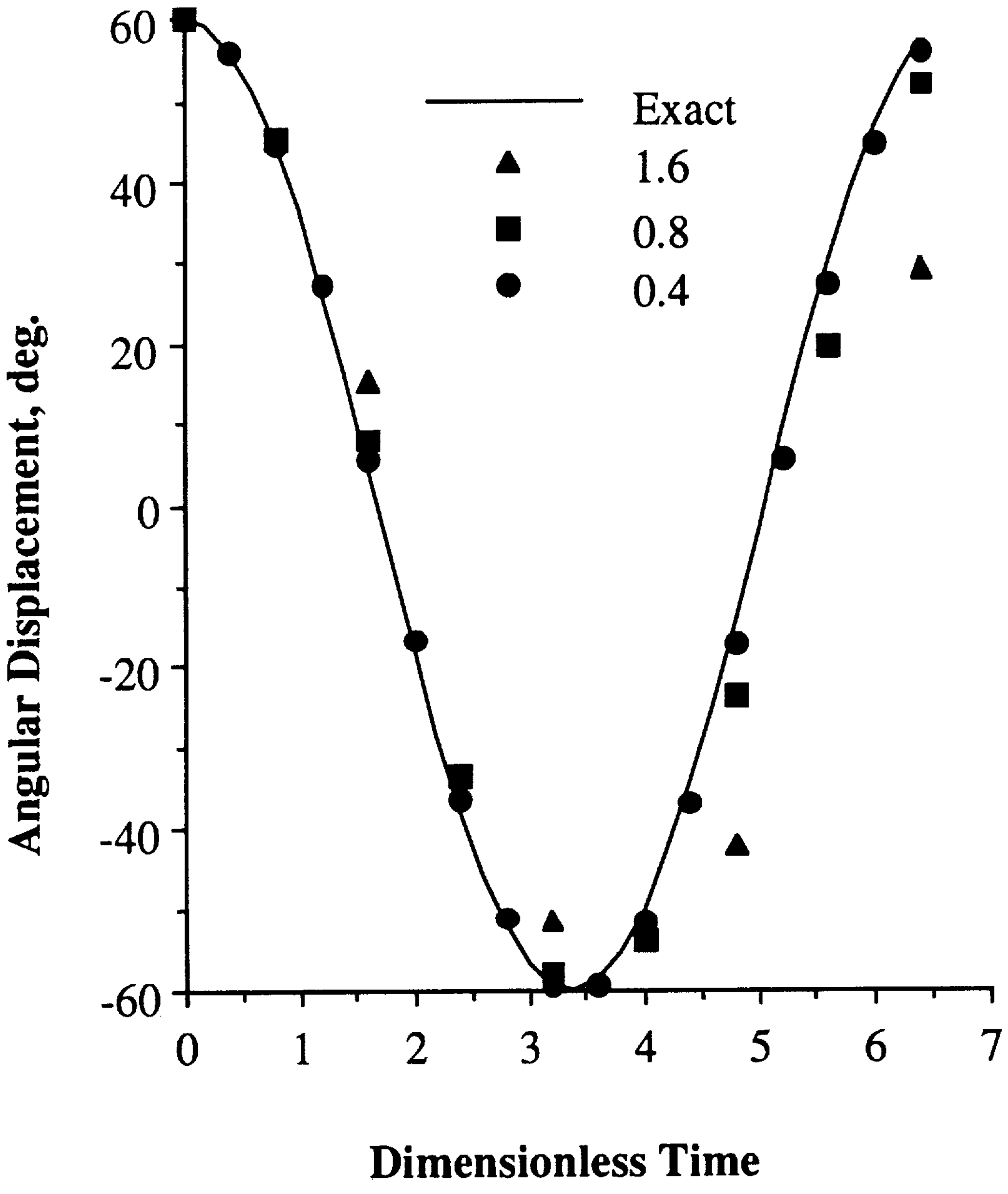

Fig. A.2: Angular displacement q versus dimensionless time

Results for three values of the time step $\Delta\bar{t}$ and the exact elliptic integral solution

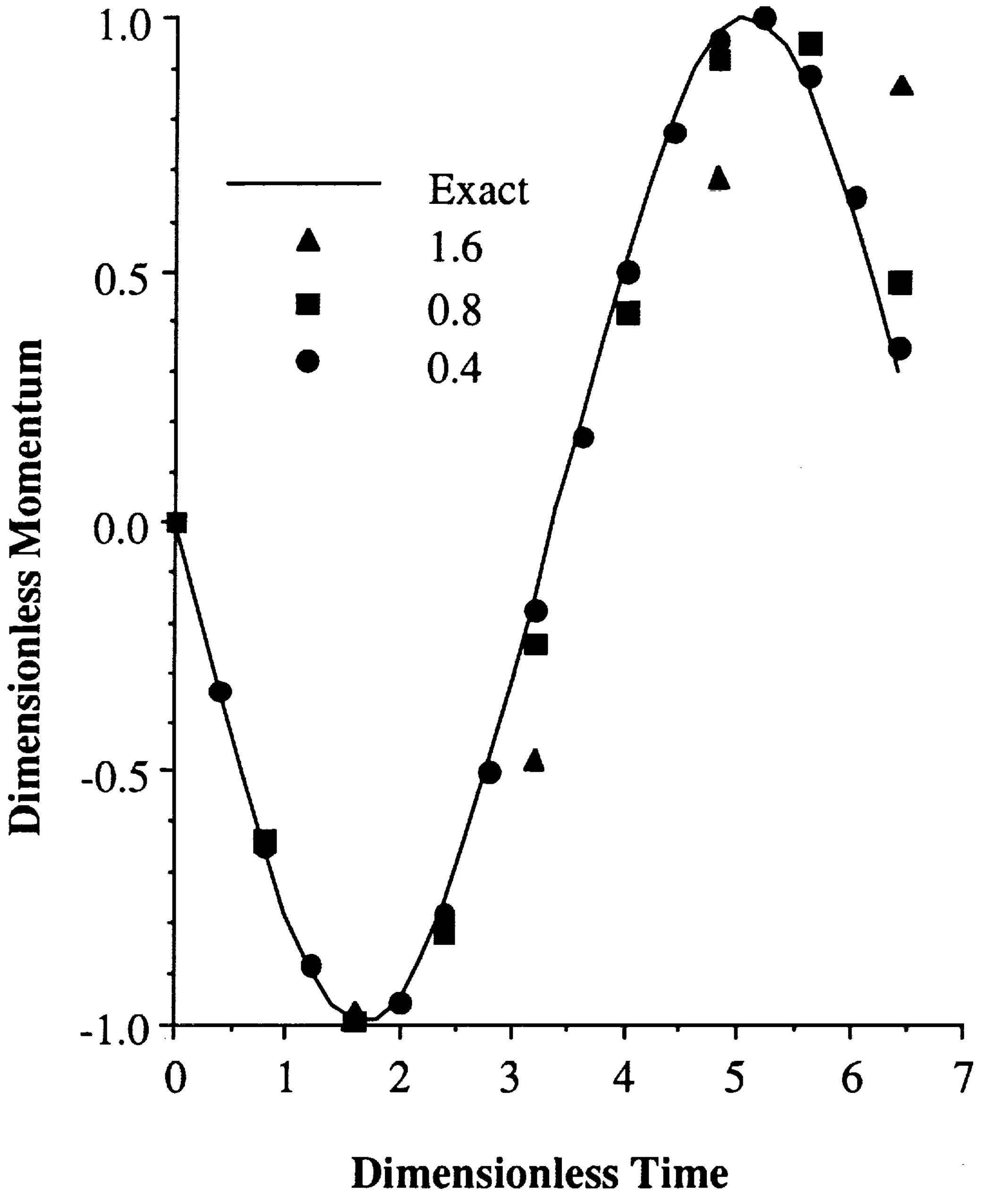

Fig. A.3: Dimensionless momentum $p/m\ell^2\omega$ versus dimensionless time
Results for three values of the time step $\Delta\bar{t}$ and the exact elliptic integral solution

APPENDIX B

INITIAL-VALUE ODE's

A very useful idea came from the work described in Chapter 2 and Appendix A concerning the solution of first-order ordinary differential equations. Of course, initial-value solvers are ubiquitous, so the goal of this work was to produce an easier and more efficient method to solve the differential equations. Furthermore, it was desired to employ a symbolic environment (such as MACSYMA) to perform all the necessary computations and write certain FORTRAN subroutines. This is similar to what is done by the general code described in Chapter 11, only this program is much simpler. This aspect of the work fully automates the procedure of solving initial-value problems and minimizes the amount of user interaction.

B.1. General Development

The problem may be stated as follows: Given a set of n first-order ordinary differential equations of the form

$$\dot{x} = f(x,t) \qquad t_1 \le t \le t_2 \tag{B.1 - 1}$$

with $x(t_0)$ specified, find $x(t)$.

To begin, a weighted residual method is used and Eq. (B.1–1) is integrated over the time interval of interest.

$$\int_{t_1}^{t_2} \delta\lambda^T \left[\dot{x} - f(x,t)\right] dt = 0 \tag{B.1 - 2}$$

Now, as was done in Chapters 2 and Appendix A, the strong boundary conditions are transformed to natural boundary conditions by weakly enforcing continuity of the states at the initial and final times. This is done via a discrete Lagrange multiplier which we identify as $\delta\lambda$. Eq. (B.1–2) now becomes

$$\int_{t_1}^{t_2} \delta\lambda^T \left[\dot{x} - f(x,t)\right] \, dt + \delta\lambda^T(\hat{x} - x)\Big|_{t_1}^{t_2} = 0 \tag{B.1 - 3}$$

Integrating the above equation by parts results in

$$\int_{t_1}^{t_2} \left(\delta\dot{\lambda}^T x + \delta\lambda^T f\right) \, dt - \delta\lambda^T \hat{x}\Big|_{t_1}^{t_2} = 0 \tag{B.1 - 4}$$

Once again introducing a nondimensional time τ as

$$\tau = \frac{t - t_1}{t_2 - t_1} = \frac{t - t_1}{\Delta t} \tag{B.1 - 5}$$

a linear shape function for $\delta\lambda$ of the form

$$\delta\lambda = \delta\lambda_i(1 - \tau) + \delta\lambda_{i+1}\tau \tag{B.1 - 6}$$

and a piecewise-constant shape function for x of the form

$$x = \begin{cases} \hat{x}_i & \text{if } \tau = 0; \\ \bar{x}_i & \text{if } 0 < \tau < 1; \\ \hat{x}_{i+1} & \text{if } \tau = 1 \end{cases} \tag{B.1 - 7}$$

then these values are substituted into Eq. (B.1–4). Carrying out the integration, the following set of algebraic equations is obtained

$$\begin{aligned} \bar{x}_i - \frac{\Delta t}{2}\bar{f}_i &= \hat{x}_i \\ \bar{x}_i + \frac{\Delta t}{2}\bar{f}_i &= \hat{x}_{i+1} \end{aligned} \tag{B.1 - 8}$$

where $\bar{f}_i = f(x = \bar{x}_i, t = \bar{t}_i)$.

Eq. (B.1–8) yields a time-marching algorithm since the value of Δt_i is specified at each time step. In practice, one solves the first of Eq. (B.1–8) for $\bar{x}_i$, and then obtains the nodal value $\hat{x}_{i+1}$ from

$$\hat{x}_{i+1} = 2\bar{x}_i - \hat{x}_i \qquad (B.1-9)$$

This process then repeats until the final desired value of x is reached.

B.2. MACSYMA: A Symbolic Manipulator

Eq. (B.1–8) is a set of very simple nonlinear equations. These equations can be solved by a Newton-Raphson method. Since we are time-marching, the previous known nodal values serve as initial guesses. To date, the roots of the equations have *always* been found using these initial guesses.

Using the above method, it was necessary to write a new code (or a large part of it) incorporating the new equations for each initial-value problem. Also, it was necessary to take explicit derivatives of the f's and code those values in for the linearization process. Therefore, it was desirable to combine FORTRAN code and MACSYMA code to automate our initial-value solver.

MACSYMA is a large symbolic manipulation program developed at the MIT Laboratory for Computer Science [77]. This program has many capabilities which include taking an analytical expression, finding an analytical derivative, writing FORTRAN code for the analytical expression, and splicing the FORTRAN code into a subroutine template file. This is exactly what we needed MACSYMA to do.

The general procedure to solve initial-value ordinary differential equations consists of three batch files, two subroutines, and a main program. Each of the above six components of the procedure is briefly described below. The whole procedure is very compact and runs very efficiently.

One batch file, "run.bat," is the top-level command file that controls the whole procedure. Its commands include putting the user into an input file in order to input the f's for MACSYMA to read. Next, run.bat starts up MACSYMA. At this point other batch files take over. Upon return to run.bat, the commands include compiling the MACSYMA written subroutine, adding the subroutine to an archive library, compiling the main program with the updated archive library, and finally running the program.

The MACSYMA input file, named "input.macsyma," prompts the user to enter the order of the system, n, and then the f's of the system in MACSYMA form. For example, if one wishes to solve the scalar system $\dot{x} = x$, then "input.macsyma" would contain the two lines "N:1;" and "G[1]:x(1);".

Once MACSYMA is started by "run.bat," the user must type "batch(run);". This batch file calls another batch file which loads the user supplied equations. Then MACSYMA is asked to evaluate the expressions in the template file and splice in the FORTRAN code. Finally, MACSYMA is terminated and control is returned to "run.bat."

The template file is very short and simple. It is nothing more than a FORTRAN subroutine (IVMAC) with MACSYMA expressions put in some places. After MACSYMA is called, all MACSYMA expressions in the template file are replaced with legal FORTRAN statements. This interaction between FORTRAN and MACSYMA was the key to automating the procedure and minimizing user interaction.

The largest file, containing only 83 lines of code, is the subroutine SOLVER. This routine linearizes the discretized algebraic equations, calls IVMAC for values of f and derivatives of f, calls some Harwell subroutines to solve the equations, and writes the data to an output file.

Finally, the main program, IVODE, asks the user for initial conditions, the time interval, and the time step to take. IVODE calls SOLVER and computes the elapsed computer time.

The above work has been a key part of the research effort. Although the initial-value problem is much easier than the optimal-control problem, the work indicated that a general procedure to solve optimal-control problems is realizable. The above outlined procedure must simply be broadened to handle the more complicated equations that come from optimal-control problems.

APPENDIX C

APPLICATION TO BEAM THEORY

The weak principle for optimal control developed in this thesis and, in particular, the general code described in Chapter 11 may be used to solve virtually any problem that can be cast in the proper form. Specifically, if one can identify a performance index and state equations, then the general code could be used to solve these problems. Problems from areas such as chemical engineering, robotics, electrical engineering and elasticity could be solved. This appendix examines how simple beam problems can be solved with the general code of Chapter 11.

C.1. Transformation of Beam Problem

Consider a simple cantilever beam of length L with a distributed load q, an end load $\hat{F}_L$, and an end moment $\hat{M}_L$. The deflection and slope of the neutral axis of the beam are signified by v and β respectively; and the curvature is denoted by κ.

The first thing to do is identify elements of the beam problem with elements of the optimal control problem. From simple elasticity, we know that the first derivative of deflection with respect to the beam axis yields the slope, and the second derivative gives the curvature. These two equations can be written as two first-order equations as

$$v' = \beta \qquad \text{and} \qquad \beta' = \kappa \tag{C.1 - 1}$$

It is easy to identify v and β as the two states and κ as the control variable. What remains is to identify a suitable performance index. It is well known in elasticity that the variation of the strain energy equals the variation of the work produced by external forces. Therefore, a performance index may be written as

$$J = \int_0^L \left(\frac{EI\kappa^2}{2} - qv \right) dx - \hat{F}_L v_L - \hat{M}_L \beta_L \qquad \text{(C.1 − 2)}$$

where EI is the bending stiffness of the beam and x measures the distance along the axis of the beam. The Hamiltonian can now be identified as

$$H = \frac{EI\kappa^2}{2} - qv + \lambda_v \beta + \lambda_\beta \kappa \qquad \text{(C.1 − 3)}$$

The costate equations yield additional insight into the problem. The equations are

$$\lambda_v' = -\frac{\partial H}{\partial v} = q \quad \text{and} \quad \lambda_\beta' = -\frac{\partial H}{\partial \beta} = -\lambda_v \qquad \text{(C.1 − 4)}$$

From these differential equations, we identify $\lambda_v = -F$ and $\lambda_\beta = -M$, or in words, the costates yield the shear force and moment distribution along the beam.

The beam problem has now been transformed to the equivalent optimal control problem and is ready for solution.

C.2. Example

A simple example problem is now considered. Consider a cantilevered beam with a tip load. Let $EI = 10^6$ psi, $L = 100$ in, and the tip load $\hat{F}_L = 30$ lb. The states will have the following constraints at the final time

$$\psi_1 = v - \frac{\hat{F}_L L^3}{3EI} \quad \text{and} \quad \psi_2 = \beta - \frac{\hat{F}_L L^2}{3EI} \qquad \text{(C.2 − 1)}$$

These constraints impose that the beam have a deflection equal to that which it would have with no constraint, however, the slope of the beam at the end is changed so that is lies along a line joining the origin and the point of deflection at the end of the beam.

The equations and boundary conditions were put into the general code (as described in Chapter 11) and the solution was readily found. Figs. C.1 – C.4 show the deflection, slope, shear force, and moment distributions for 2 and 4 elements and the exact answer. An 8 element case is also given for the shear force in Fig. C.3. In all the graphs, the accuracy of the finite element method is once again seen.

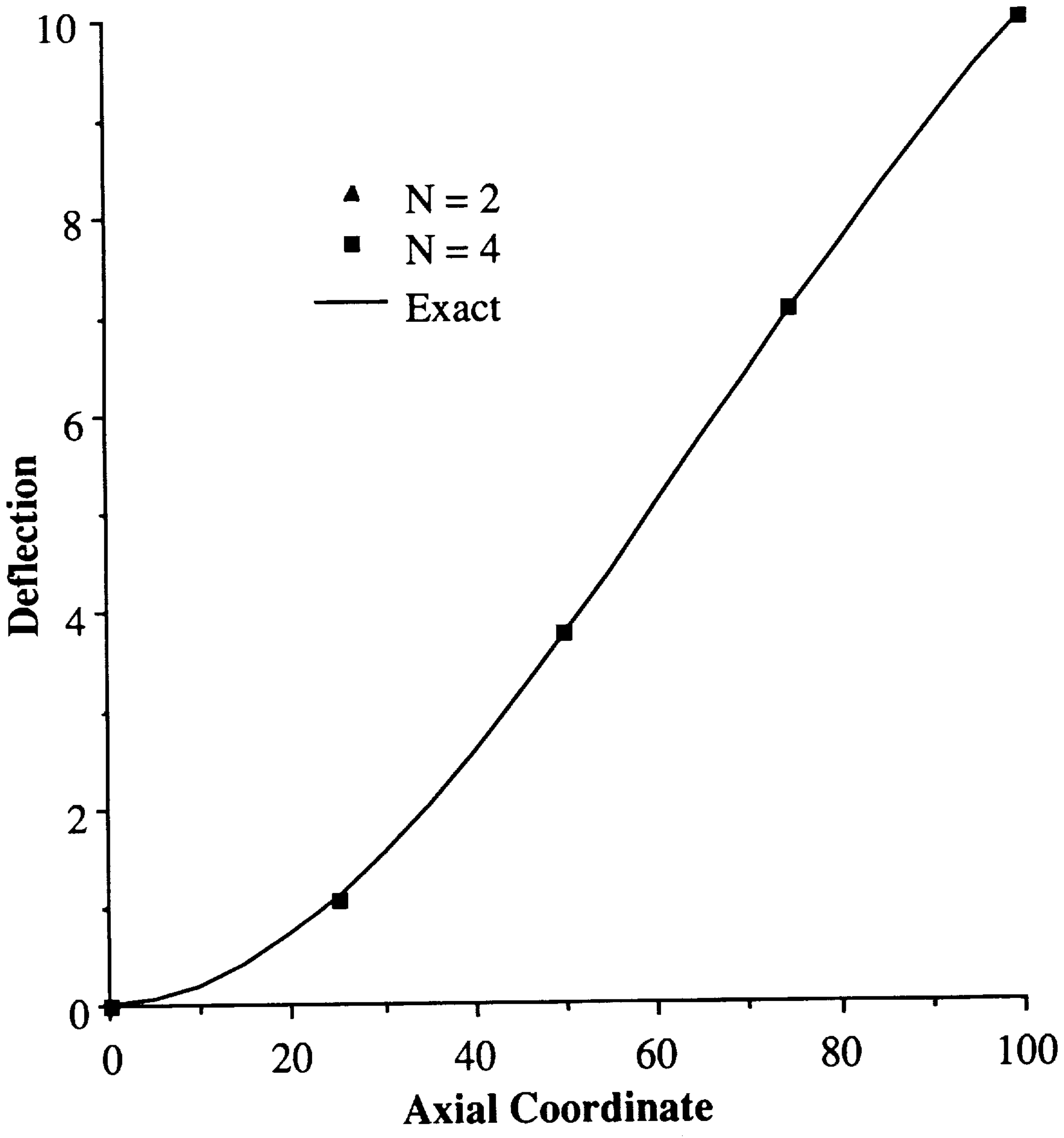

Fig. C.1: Deflection versus axial coordinate

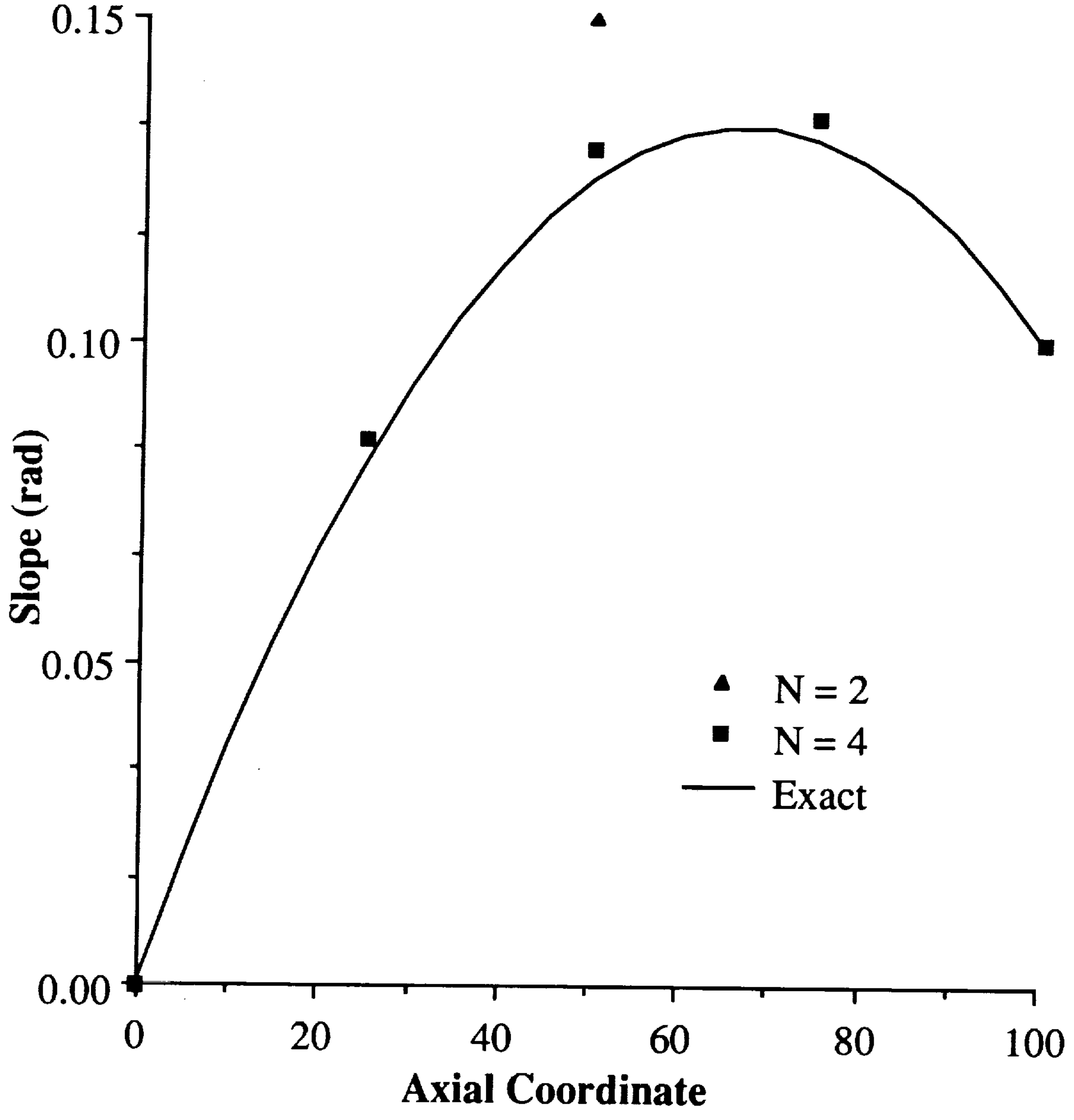

Fig. C.2: Slope versus axial coordinate

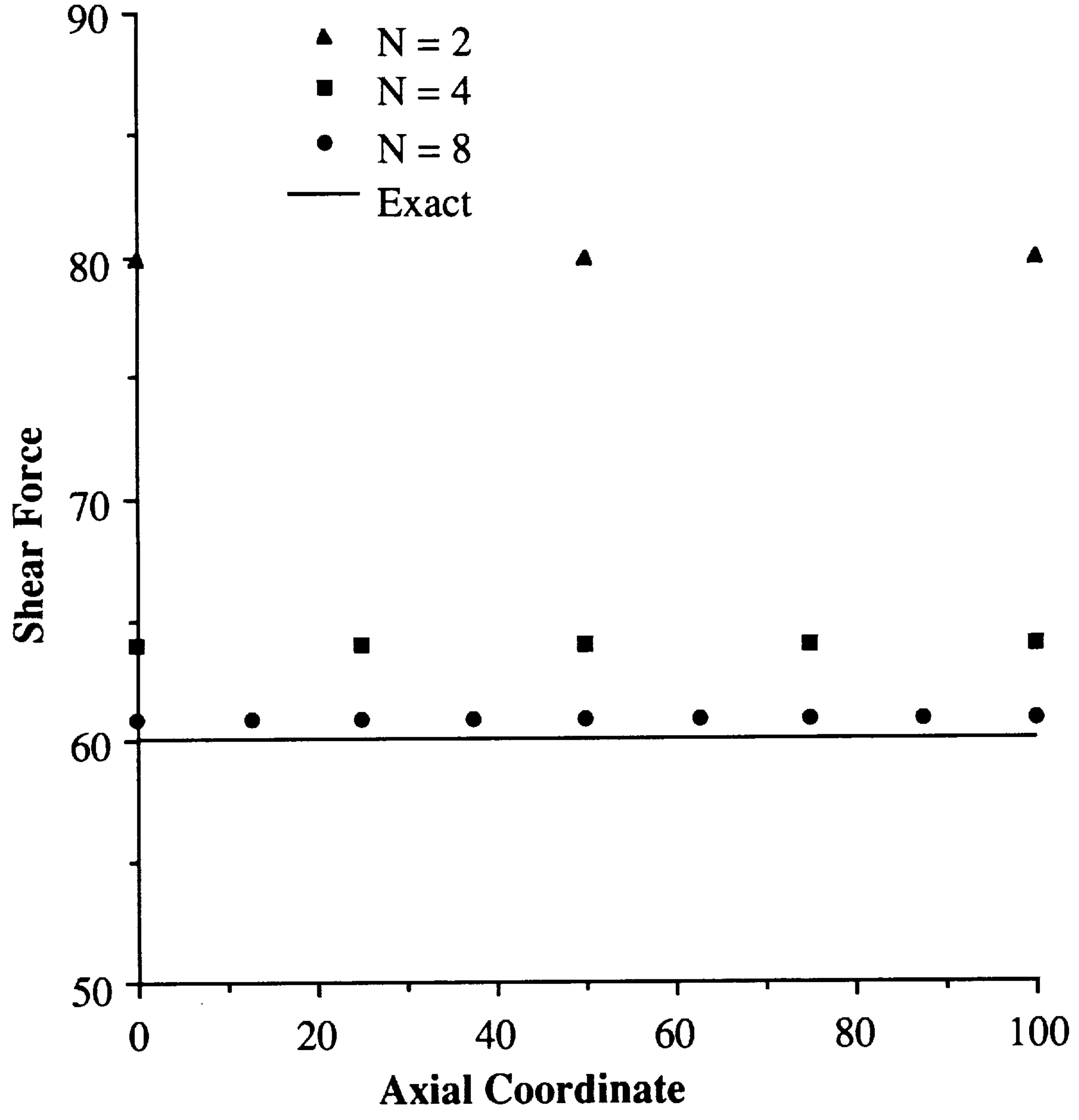

Fig. C.3: Shear force versus axial coordinate

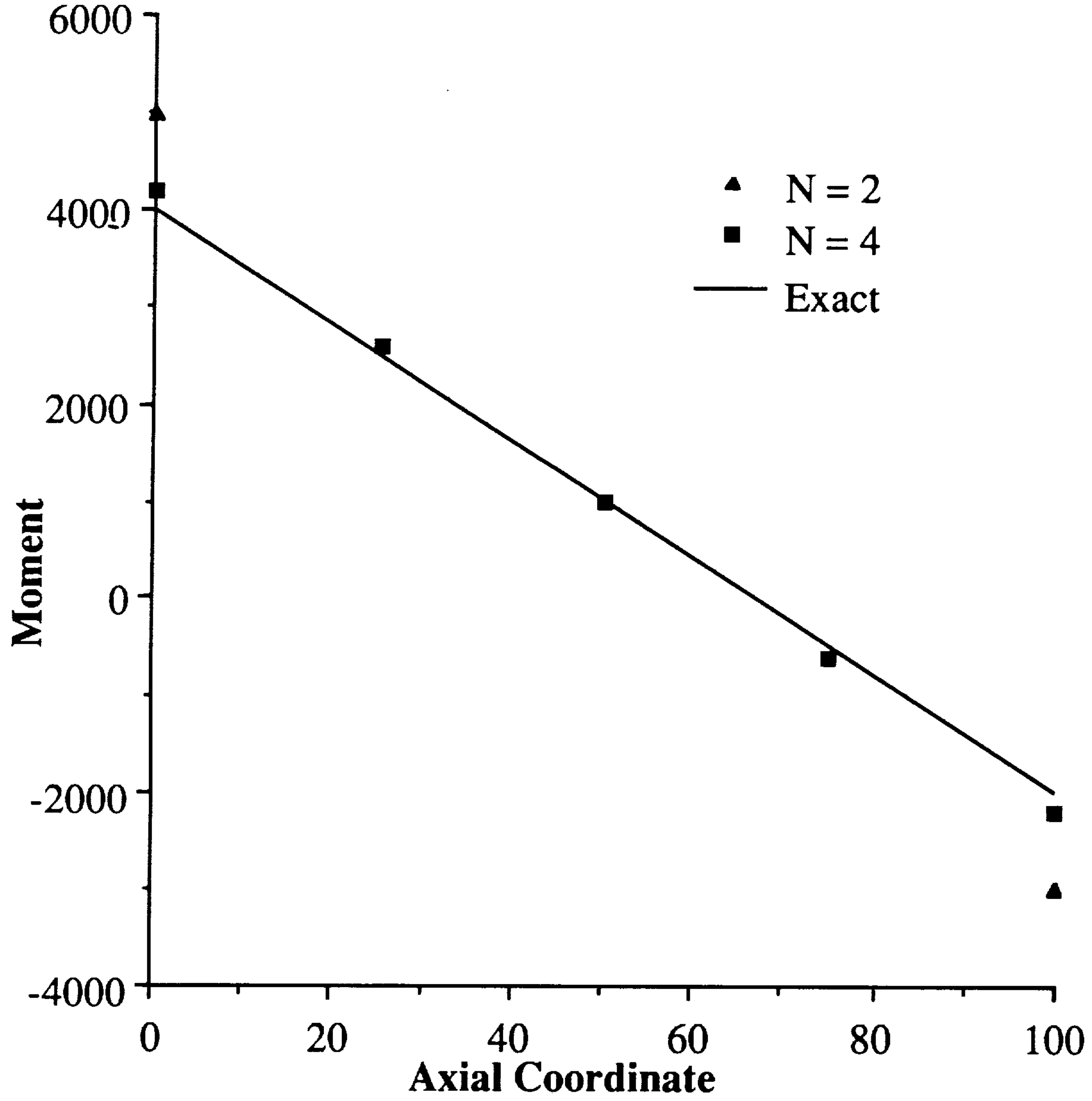

Fig. C.4: Moment versus axial coordinate

REFERENCES

1. Bryson, Arthur E. Jr., and Ho, Yu-Chi, *Applied Optimal Control*, Blaisdell Publishing Company, Waltham, Massachusetts, 1969.

2. Gelfand, I. M., and Fomin, S. V., *Calculus of Variations*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1963.

3. Petrov, Iu. P., *Variational Methods in Applied Optimum Control Theory*, Translated by Friedman, M. D., with the assistance of H. J. ten Zeldman, Academic Press, New York, New York, 1968.

4. Corban, J. Eric, "Real-time Guidance and Propulsion Control for Single-stage-to-orbit Airbreathing Vehicles," Ph.D. thesis, Georgia Institute of Technology, 1990.

5. "USAF Seeks Industry Responses for Advanced Launch Systems," *Aviation Week & Space Technology*, p. 26, May 11, 1987.

6. Hardtla, J. W., Piehler, M. J., and Bradt, J. E., "Guidance Requirements for Future Launch Vehicles," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, Paper No. 87-2462, August 1987.

7. Chen, C. F., and Hsiao, C. H., "Walsh Series Analysis in Optimal Control," *International Journal of Control*, Vol. 21, No. 6, 1975, pp. 881 - 897.

8. Chen, Wen-Liang, and Shih, Yen-Ping, "Analysis and Optimal Control of Time-Varying Linear Systems via Walsh Functions," *International Journal of Control*, Vol. 27, No. 6, 1978, pp. 917 - 932.

9. Rao, V. Purnachandra, and Rao, K. Ranganatha, "Optimal Feedback Control Via Block-Pulse Functions," *IEEE Transactions on Automatic Control*, Vol. AC-24, No. 2, 1979, pp. 372 - 374.

10. Hsu, Ning-Show, and Cheng, Bing, "Analysis and Optimal Control of Time-Varying Linear Systems Via Block-Pulse Functions," *International Journal of Control*, Vol. 33, No. 6, 1981, pp. 1107 – 1122.

11. Paraskevopoulos, P. N., "Chebychev Series Approach to System Identification, Analysis and Optimal Control," *Journal of the Franklin Institute*, Vol. 316, No. 2, 1983, pp. 135 – 157.

12. Kekkeris, G. Th., and Paraskevopoulos, P. N., "Hermite Series Approach to Optimal Control," *International Journal of Control*, Vol. 47, No. 2, 1988, pp. 557 – 567.

13. Yang, Ching-Yu, and Chen, Cha'o-Kuang, "Analysis and Optimal Control of Time-Varying Systems Via Polynomial Series," *International Journal of Systems Science*, Vol. 19, No. 2, 1988, pp. 289 – 310.

14. Razzaghi, Mohsen, and Arabshahi, Abdollah, "Optimal Control of Linear Distributed-Parameter Systems via Polynomial Series," *International Journal of System Science*, Vol. 20, No. 7, 1989, pp. 1141 – 1148.

15. Shyu, Kuo-Kai, and Hwang, Chyi, "Optimal Control of Linear Time-Varying Discrete Systems via Discrete Legendre Orthogonal Polynomials," *Journal of the Franklin Institute*, Vol. 235, No. 4, 1988, pp. 509 – 525.

16. Agrawal, O. P., "General Formulation for the Numerical Solution of Optimal Control Problems," *International Journal of Control*, Vol. 50, No. 2, 1989, pp. 627 – 638.

17. Hargraves, C. R., and Paris, S. W., "Direct Trajectory Optimization Using Nonlinear Programming and Collocation," *Journal of Guidance*, Vol. 10, No. 4, 1987, pp. 338 – 342.

18. Hargraves, Charles, Johnson, Forrester, Paris, Stephen, and Rettie, Ian, "Numerical Computation of Optimal Atmospheric Trajectories," *Journal of Guidance and Control*, Vol. 4, No. 4, 1981, pp. 406 – 414.

19. Vlassenbroeck, Jacques, "A Chebychev Polynomial Method for Optimal Control with State Constraints," *Automatica*, Vol. 24, No. 4, 1988, pp. 499 - 506.

20. Zhu, Jian-Min, and Lu, Yong-Zai, "Hierarchical Strategy for Non-linear Optimal Control Systems Via STWS Approach," *International Journal of Control*, Vol. 47, No. 6, 1988, pp. 1837 - 1848.

21. Prenter, P. M., *Splines and Variational Methods*, John-Wiley & Sons, New York, New York, 1975.

22. Kelley, C. T., and Sachs, E. W., "Quasi-Newton Methods and Unconstrained Optimal Control Problems," *SIAM Journal on Control and Optimization*, Vol. 25, No. 6, 1987, pp. 1503 - 1516.

23. Kuo, Chung-Feng, and Kuo, Chen-Yuan, "Improved Gradient-Type Algorithms for Zero Terminal Gradient Optimal Control Problems," *Journal of Dynamic Systems, Measurement, and Control*, Vol. 109, 1987, pp. 355 - 362.

24. Gruver, W. A., and Sachs, E., *Algorithmic Methods in Optimal Control*, Pitman Publishing, London, 1980.

25. Rader, James E., and Hull, David J., "Computation of Optimal Aircraft Trajectories Using Parameter Optimization Methods," *Journal of Aircraft*, Vol. 12, No. 11, 1975, pp. 864 - 866.

26. Hewett, Marle D., and Kirk, Donald E., "Trajectory Optimization Using a Second-Order Epsilon Method," *Journal of Aircraft*, Vol. 13, No. 3, 1976, pp. 174 - 179.

27. Oberle, H. J., "Numerical Treatment for Minimax Optimal Control Problems with Application to the Reentry Flight Path Problem," *The Journal of the Astronautical Sciences*, Vol. 36, Nos. 1/2, 1988, pp. 159 - 178.

28. Pesch, Hans Josef, "Real-Time Computation of Feedback Controls for Constrained Optimal Control Problems. Part I: Neighbouring Extremals," *Optimal Control Applications and Methods*, Vol. 10, 1989, pp. 129 - 145.

29. Pesch, Hans Josef, "Real-Time Conputation of Feedback Controls for Constrained Optimal Control Problems. Part II: A Correction Method Based on Multiple Shooting," *Optimal Control Applications and Methods*, Vol. 10, 1989, pp. 147 – 171.

30. Menon, P. K. A., and Lehman, L. L., "A Parallel Quasi-Linearization Algorithm for Air Vehicle Trajectory Optimization," *Journal of Guidance*, Vol. 9, No. 1, 1986, pp. 119 – 121.

31. Roberts, S. M., and Shipman, J. S., "Multipoint Solution of Two-Point Boundary-Value Problems," *Journal of Optimization Theory and Applications*, Vol. 7, No. 4, 1971, pp. 301 – 318.

32. Roberts, Sanford M., and Shipman, Jerome S., *Two-Point Boundary Value Problems: Shooting Methods*, American Elsevier Publishing, New York, New York, 1972.

33. Subrahmanyam, M. B., "A Computational Method for the Solution of Time-Optimal Control Problems by Newton's Method," *International Journal of Control*, Vol. 44, No. 5, 1986, pp. 1233 – 1243.

34. Patten, W. N., "Near Optimal Feedback Control for Nonlinear Aerodynamic Systems with an Application to the High-Angle-of-Attack Wing Rock Problem," AIAA Paper 88-4052-CP, 1988.

35. Bosarge, W. E. Jr., and Johnson, O. G., "Error Bounds of High Order Accuracy for the State Regulator Problem Via Piecewise Polynomial Approximations," *SIAM Journal on Control*, Vol. 9, No. 1, 1971, pp. 15 – 28.

36. Chen, Goong, and Mills, Wendell H. Jr., "Finite Elements and Terminal Penalization for Quadratic Cost Optimal Control Problems Governed by Ordinary Differential Equations," *SIAM Journal on Control and Optimization*, Vol. 19, No. 6, 1981, pp. 744 –764.

37. Mathis, F. G., and Reddien, G. W., "Ritz-Trefftz Approximations in Optimal Control," *SIAM Journal on Control and Optimization*, Vol. 17, No. 2, 1979, pp. 307 – 310.

38. Asselmeyer, Bernhard, "A Ritz-Type Optimization Method for Optimal Control Problems and Its Application to Hierarchical Final-Value Controllers," *Control and Dynamic Systems: Advances in Theory and Application*, Vol. 23, 1986, pp. 295 – 318.

39. Goh, C. J., and Teo, K. L., "MISER: A FORTRAN Program for Solving Optimal Control Problems," *Advanced Engineering Software*, Vol. 10, No. 2, 1988, pp. 90 – 99.

40. Brauer, G. L., *et al*, "Final Report: Program To Optimize Simulated Trajectories (POST)," Volume I – Formulation Manual, NASA CR—132689, 1975.

41. Bailey, C. D., "Application of Hamilton's Law of Varying Action," *AIAA Journal*, Vol. 13, No. 9, 1975, pp. 1154 – 1157.

42. Simkins, T. E., "Unconstrained Variational Statements for Initial and Boundary Value Problems," *AIAA Journal*, Vol. 16, 1978, pp. 559 – 563.

43. Hitzl, D. L., and Levinson, D. A., "Application of Hamilton's Law of Varying Action to the Restricted Three-Body Problem," *Celestial Mechanics*, Vol. 22, 1980, pp. 255 – 266.

44. Baruch, M., and Riff, R., "Hamilton's Principle, Hamilton's Law, 6^n Correct Formulations," *AIAA Journal*, Vol. 20, 1982, pp. 687 – 692.

45. Borri, M. *et al.*, "Dynamic Response of Mechanical Systems by a Weak Hamiltonian Formulation," *Computers and Structures*, Vol. 20, No. 1 – 3, 1985, pp. 495 – 508.

46. Peters, David A., and Izadpanah, Amir, "*hp*-Version Finite Elements for the Space-Time Domain," *Computational Mechanics*, Vol. 3, 1988, pp. 73 – 88.

47. Borri, M., *et al.*, "Primal and Mixed Forms of Hamilton's Principle for Constrained and Flexible Dynamical Systems: Numerical Studies," ARO/AFOSR Workshop on Nonlinear Dynamics, Virginia Polytechnic Institute and State University, Blacksburg, Virginia, June 1 – 3, 1988.

48. Borri, M., "Helicopter Rotor Dynamics by Finite Element Time Approximation," *Computers and Mathematics with Applications*, Vol. 12A, No. 1, 1986, pp. 149 – 160.

49. Hodges, Dewey H., and Bless, Robert R., "Weak Hamiltonian Finite Element Method for Optimal Control Problems," *Journal of Guidance, Control, and Dynamics*, Vol. 14, No. 1, 1991, pp. 148 – 156.

50. Hodges, Dewey H., Calise, Anthony J., Bless, Robert R., and Leung, Martin, "A Weak Hamiltonian Finite Element Method for Optimal Guidance of an Advanced Launch Vehicle," *Proceedings of the American Control Conference*, Pittsburgh, PA, Vol. 3, June 1989, pp. 2037 – 2043.

51. Shames, Irving H., and Dym, Clive L., *Energy and Finite Element Methods in Structural Mechanics*, Hemisphere Publishing Corporation, New York, 1985.

52. Wu, J. J., and Simkins, T. E., "A Numerical Comparison Between Two Unconstrained Variational Formulations," *Journal of Sound and Vibration*, Vol. 72, 1980, pp. 491 – 505.

53. Becker, E. B., Carey, G. F., and Oden, J. T., *Finite Elements: An Introduction, Volume 1*, Prentice-Hall, Inc., 1981.

54. Anon., "Saturn IB SA-217 Reference Launch Vehicle," NASA TM X-53686, 1968.

55. Anon., The International Mathematical and Statistical Library, IMSL Inc., Houston, Texas, Ch. Z, 1984.

56. Bulirsch, R., "The Multiple Shooting Method for Numerical Solution of Nonlinear Boundary Value Problems and Optimal Control Problems (in German)," Carl-Cranz-Gesellschaft, Tech. Rpt., Heidelberg, 1971.

57. Bless, Robert R., Hodges, Dewey H., and Calise, Anthony J., "A Finite Element Based Method for Solution of Optimal Control Problems," *Aerospace Computational Control Conference*, Oxnard, CA, August 1989.

58. Valentine, F. A., "The Problem of Lagrange with Differential Inequalities as Added Side Conditions," *Contributions to the Calculus of Variations*, Chicago, IL, Chicago University Press, 1937, pp. 407 – 448.

59. Duff, I. S., Harwell Subroutine Library, Computer Science and Systems Division, Harwell Laboratory, Oxfordshire, England, February 1988, Chapter M.

60. Bryson, A. E. Jr., Denham, W. F., and Dreyfus, S. E., "Optimal Programming Problems with Inequality Constraints, I: Necessary Conditions for Extremal Solutions," *AIAA Journal*, 1963, pp. 2544 – 2550.

61. Speyer, J. L., and Bryson, A. E. Jr., "Optimal Programming Problems with a Bounded State Space," *AIAA Journal*, 1968, pp. 1488 – 1491.

62. Xing, A. Q., *et. al.*, "Exact Penalty Function Approach to Constrained Optimal Control Problems," *Optimal Control Applications and Methods*, Vol. 10, 1989, pp. 173 – 180.

63. Jacobson, David H., and Lele, Milind M., "A Transformation Technique for Optimal Control Problems with a State-Variable Inequality Constraint," *IEEE Transactions on Automatic Control*, Vol. AC-14, No. 5, 1969, pp. 457 – 464.

64. Calise, A. J., and Corban, J. E., "Optimal Control of Singularly Perturbed Nonlinear Systems with State-Variable Inequality Constraints," *AIAA Guidance, Navigation, and Control Conference*, Portland, Oregon, 1990.

65. Jacobson, D. H., Lele, M. M., and Speyer, J. L., "New Necessary Conditions of Optimality for Control Problems with State-Variable Inequality Constraints,"

Journal of Mathematical Analysis and Applications, Vol. 35, 1971, pp. 255 – 284.

66. Bless, Robert R., and Hodges, Dewey H., "Finite Element Solution of Optimal Control Problems with Inequality Constraints," *Proceedings of the American Control Conference*, San Diego, CA, May 1990.

67. Minzner, R. A., *et al.*, "Defining Constants, Equations, and Abbreviated Tables of the 1975 U.S. Standard Atmosphere," NASA TR R–459, 1976.

68. Segerlind, Larry J., "Weighted Residual Solutions in the Time-Domain," *International Journal for Numerical Methods in Engineering*, Vol. 28, 1989, pp. 679 – 685.

69. Hodges, Dewey H., and Hou, Lin-Jun, "Shape Functions for Mixed p-Version Finite Elements in the Time Domain," *Journal of Sound and Vibration*, Vol. 145, No. 2, 1991, pp. 169 – 178.

70. Stroud, A. H., *Numerical Quadrature and Solution of Ordinary Differential Equations*, Applied Mathematical Sciences: Volume 10, Springer-Verlag New York, 1974.

71. Chow, Shui-Nee, Mallet-Paret, John, and Yorke, James A., "Finding Zeroes of Maps: Homotopy Methods that are Constructive With Probability One," *Mathematics of Computation*, Vol. 32, No. 143, 1978, pp. 887 – 899.

72. Watson, Layne T., "A Globally Convergent Algorithm for Computing Fixed Points of C^2 Maps," *Applied Mathematics and Computation*, Vol. 5, 1979, pp. 297 – 311.

73. Watson, Layne T., "An Algorithm that is Globally Convergent with Probability One for a Class of Nonlinear Two-Point Boundary Value Problems," *SIAM Journal on Numerical Analysis*, Vol. 16, No. 3, 1979, pp. 394 – 401.

74. Watson, Layne T., "Solving Finite Difference Approximations to Nonlinear Two-Point Boundary Value Problems by a Homotopy Method," *SIAM Journal Sci. Stat. Comput.*, Vol. 1, No. 4, 1980, pp. 467 – 480.

75. Todd, Michael J., "Piecewise-Linear Homotopy Algorithms for Sparse Systems of Nonlinear Equations," *SIAM Journal on Control and Optimization*, Vol. 21, No. 2, 1983, pp. 204 – 214.

76. Kane, T. R., and Levinson, D. A., *Dynamics: Theory and Applications*, McGraw-Hill, 1985.

77. Symbolics (1988). MACSYMA Reference Manual. Symbolics, Inc., Burlington, Massachusetts.

78. Milne-Thomson, L. M., *Jacobian Elliptic Function Tables*, Dover Publications, 1950, p. 38.

National Aeronautics and Space Administration

Report Documentation Page

1. Report No. NASA CR-4376	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle: Time-Domain Finite Elements in Optimal Control With Application to Launch-Vehicle Guidance		5. Report Date: May 1991
		6. Performing Organization Code
7. Author(s): Robert R. Bless		8. Performing Organization Report No.
		10. Work Unit No. 946-01-00-78
9. Performing Organization Name and Address: Georgia Institute of Technology, School of Aerospace Engineering, Atlanta, GA 30332		11. Contract or Grant No. NAG1-939
12. Sponsoring Agency Name and Address: National Aeronautics and Space Administration, Langley Research Center, Hampton, VA 23665-5225		13. Type of Report and Period Covered: Contractor Report Progress
		14. Sponsoring Agency Code

15. Supplementary Notes

The information presented in this report was offered as a thesis in partial fullfillment of the requirements for the degree of Doctor of Philosophy in Aerospace Engineering at the Georgia Institute of Technology in Atlanta, Georgia in January 1991.

Langley Technical Monitor: Daniel D. Moerder

16. Abstract

A time-domain finite element method is developed for optimal control problems. The theory derived is general enough to handle a large class of problems including optimal control problems that are continuous in the states and controls, problems with discontinuities in the states and/or system equations, problems with control inequality constraints, problems with state inequality constraints, or problems involving any combination of the above. The theory is developed in such a way that no numerical quadrature is necessary regardless of the degree of nonlinearity in equations. Also, the same shape functions may be employed for every problem because all strong boundary conditions are transformed into natural or weak boundary conditions. In addition, the resulting nonlinear algebraic equations are very sparse. Use of sparse matrix solvers allows for the rapid and accurate solution of very difficult optimization problems. The formulation is applied to launch-vehicle trajectory optimization problems, and results show that real-time optimal guidance is realizable with this method. Finally, a general problem solving environment is created for solving a large class of optimal control problems. The algorithm uses both FORTRAN and a symbolic computation program to solve problems with a minimum of user interaction. The use of symbolic computation eliminates the need for user-written subroutines which greatly reduces the setup time for solving problems.

17. Key Words (Suggested by Author(s)): Finite Elements, Optimal Control, Numerical Method, Trajectory Optimization		18. Distribution Statement: Unclassified-Unlimited, Subject Category 15	
19. Security Classif. (of this report): Unclassified	20. Security Classif. (of this page): Unclassified	21. No. of pages: 213	22. Price: A10

NASA FORM 1626 OCT 86 For sale by the National Technical Information Service, Springfield, VA 22161-2171

NASA-Langley, 1991