

THE UNIVERSITY OF CALGARY

**Assessing Approximation Methods for Modelling
Non-Linear Dynamic Behaviour of Structures**

by

Tracy Teh

A THESIS

**SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE**

DEPARTMENT OF CIVIL ENGINEERING

CALGARY, ALBERTA

APRIL, 1995

© Tracy Teh 1995

THE UNIVERSITY OF CALGARY
FACULTY OF GRADUATE STUDIES

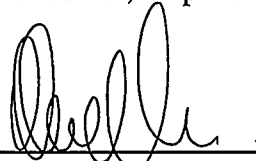
The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies for acceptance, a thesis entitled "Assessing Approximation Methods for Modelling Non-Linear Dynamic Behaviour of Structures" submitted by Tracy Teh in partial fulfilment of the requirements for the degree of Master of Science.



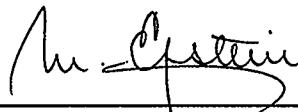
Supervisor, Dr.T.G.Brown, Dept. of Civil Engineering



Dr.A.Ghali, Dept. of Civil Engineering



Dr.M.A.Maes, Dept. of Civil Engineering



Dr.M.Epstein, Dept. of Mechanical Engineering

April 1995.

Date

ABSTRACT

The objective of this thesis is to review approximation methods that solve linear structural dynamic problems. Performance of these methods for both linear and non-linear models using various time step sizes will be examined.

The thesis is split into two main components, the first looks at a selection of common approximation methods. The general "Single-Step" scheme, which models a variety of methods, was examined in detail. Procedures for assessing the approximation methods for linear problems were investigated; a program was developed and used to evaluate some of these procedures.

The second stage of the thesis looks at non-linear problems, the concept of iterative procedures modelling material non-linearity is introduced. The finite element program Nonsap was modified to include the general "Single-Step" scheme. Nonsap was used to investigate the accuracy of displacements and computational efficiency of the approximation methods and iterative procedures for non-linear problems.

ACKNOWLEDGEMENTS

I would like to thank Dr. Thomas Brown for his supervision, he was very helpful in directing me and providing valuable ideas during the course of this thesis. I would also like to thank him for selecting me for this research as it gave me the opportunity to study a very interesting topic. I would like to thank Dr. Nenard Bicanic and Mr. Maosong Huang, Swansea University, for their invaluable references. Finally my thanks goes to my husband for his support and encouragement.

DEDICATION

*I would like to dedicate this thesis to my parents in appreciation
for their love and support.*

TABLE OF CONTENTS

APPROVAL PAGE ii
ABSTRACT iii
ACKNOWLEDGEMENTS iv
DEDICATION v
TABLE OF CONTENTS vi
LIST OF TABLES ix
LIST OF FIGURES x
LIST OF NOTATION xiv
CHAPTER ONE: INTRODUCTION 1
1.1 WILSON θ METHOD 4
1.2 THESIS OUTLINE 8
CHAPTER TWO: DYNAMIC SOLUTION SCHEMES 10
2.1 MULTI-STEP SCHEMES 10
2.1.1 TRUNCATED TAYLOR SERIES 10
COLLOCATION	
2.1.2 A WEIGHTED RESIDUAL APPROACH 12
2.2 SINGLE-STEP SCHEMES 17
2.2.1 GENERALIZED NEWMARK SCHEMES 17
2.2.2 SINGLE-STEP TIME MARCHING 20
SCHEMES	
CHAPTER THREE: STABILITY AND ACCURACY OF METHODS 25
3.1 STABILITY AND ACCURACY 29
3.2 PROGRAMMING THEORY 42
3.3 EXAMPLES 46

CHAPTER FOUR: SOLUTION OF NON-LINEAR PROBLEMS 61
USING ITERATIVE PROCEDURES	
4.1 ITERATIVE METHODS 61
4.1.1 MODIFIED NEWTON ITERATION SCHEME 62
4.1.2 QUASI-NEWTON METHOD 65
4.2 NONSAP, A FINITE ELEMENT PROGRAM 67
4.2.1 A LINEAR FORMULATION 70
4.2.2 A NON-LINEAR FORMULATION 72
CHAPTER FIVE: EXAMPLES USING NONSAP 78
5.1 PRELIMINARY LINEAR STATIC AND DYNAMIC 78
STUDY OF NONSAP	
5.1.1 STUDY OF TIME INCREMENT SIZE 83
5.1.2 STUDY OF PARAMETER VALUES 85
5.2 NON-LINEAR DYNAMIC STUDY 90
5.2.1 VERIFICATION OF NEW SSpj SCHEMES 94
5.2.2 STUDY OF TIME INCREMENT SIZE 97
5.2.3 STUDY OF SOLUTION SCHEMES AND 101
PARAMETER VALUES	
CHAPTER SIX: CONCLUSIONS AND RECOMMENDATIONS 107
6.1 CONCLUSIONS 107
6.2 RECOMMENDATIONS 111
6.3 COMMENTS 112
BIBLIOGRAPHY 113
APPENDIX A 116
A.1 TRUNCATED TAYLOR SERIES COLLOCATION 116
A.2 WEIGHTED RESIDUAL 118

APPENDIX B	120
B.1 DYNAMIC.C	120
B.1.1 INTRODUCTION	120
B.1.2 ASSUMPTIONS AND LIMITATIONS	120
B.1.3 INPUT DESCRIPTION	121
B.1.4 OUTPUT DESCRIPTION	123
B.1.5 RUNNING INSTRUCTIONS	123
B.2 EIGEN.C	126
B.2.1 INTRODUCTION	126
B.2.2 ASSUMPTIONS AND LIMITATIONS	126
B.2.3 INPUT AND OUTPUT DESCRIPTION	126
B.2.4 RUNNING INSTRUCTIONS	126
B.3 DYNAMIC.C LISTING	128
B.4 EIGEN.C LISTING	140
APPENDIX C		
GRAPHS FROM NON-LINEAR STUDY USING	142
NONSAP		

LIST OF TABLES

2.1	Parameter Values for Common Solution Schemes Using the Weighted Residual Cubic Equation. 16
2.2	Parameter Values for Common Solution Schemes Using Either the GNPj or the SSPj General Equation. 24
3.1	Coefficients for the Multi-Step Equivalent of the Single-Step Equations. 39
3.2	Coefficients to Calculate Truncation Error and Stability for the Single-Step Equations. 40
3.3	Sample Single-Step Schemes and Their Corresponding Truncation Errors and Stability Requirements. 47
4.1	The Solution Schemes a _i Parameter Values. 71
5.1	Frequency Data for Different Mass Matrices. 82
5.2	Corresponding Time Increment Size and Periods. 83
5.3	Frequency and Period Data for the Non-Linear Cantilever Problem Using a Lumped Mass Matrix. 93
5.4	Time Increment Size and Period Data. 93
B.1	Possible Solution Schemes in dynamic.c. 122

LIST OF FIGURES

1.1	Generalized Single Degree of Freedom Dynamic System.	3
1.2	Linear Acceleration for a Discrete Time Period.	3
2.1	Shape Functions for a Quadratic and a Cubic Equation.	13
2.2	Approximation of x for a Quadratic Algorithm.	22
3.1	Flow Chart of Dynamic.c.	26
3.2	Flow Chart of Eigen.c.	28
3.3	Mapping Unit Circle on to the Imaginary Axis $\text{Re } z$ Equal to Zero.	34
3.4	Measuring Program Accuracy Using the Trapezium Solution.	48
3.5	SS22 Solution Schemes - a. Displacement Plot	50
	b. Spectral Radius		
3.6	SS22 Solution Schemes - a. Amplitude Decay	51
	b. Period Elongation		
3.7	SS32 Solution Schemes - a. Displacement Plot	52
	b. Spectral Radius		
3.8	SS32 Solution Schemes - a. Amplitude Decay	53
	b. Period Elongation		
3.9	Comparison of Schemes With Differing Parameter Values -	57
	a. Displacement Plot		
	b. Spectral Radius		
3.10	Comparison of Schemes With Differing Parameter Values -	58
	a. Amplitude Decay		
	b. Period Elongation		
3.11	The Bossak and Hilber, Hughes and Taylor Schemes -	59
	a. Displacement Plot		
	b. Spectral Radius		

3.12	The Bossak and Hilber, Hughes and Taylor Schemes - 60
	a. Amplitude Decay	
	b. Period Elongation	
4.1	The Modified Newton Iteration Method. 64
4.2	The Secant Iteration Method. 64
4.3	Flow Chart of Nonsap's Methodology. 68
4.4	The Wilson θ and Newmark Iterative Method. 77
4.5	The SSpj Iterative Method. 77
5.1	Example Cantilever. 79
5.2	Graph for the Linear Static Cantilever Model. 79
5.3	Dynamic Loading Function. 81
5.4	Significance of Number of Elements in Cantilever Model. 81
5.5	Significance of the Mass Matrix Formulation for the Cantilever Model. 82
5.6	Sensitivity of Time Increment Size for the Wilson θ Scheme - 84
	a. Displacement Plot	
	b. Close-Up of Displacement Plot	
5.7	Percentage Period Elongation Results Published by Bathe and Wilson (1973). 86
5.8	Percentage Amplitude Decay Results Published by Bathe and Wilson (1973). 87
5.9	Sensitivity of Time Increment on the Newmark Scheme - 88
	a. Displacement Plot	
	b. Close-Up of Displacement Plot	
5.10	Comparison of Solution Schemes with Different Parameter Values - 89
	a. Displacement Plot	
	b. Close-Up of Displacement Plot	
5.11	a. Von Mises Yield Model. 91
	b. Stress-Strain Relationship.	
5.12	Non-Linear Cantilever Model. 92

5.13	Forcing Function. 92
5.14	Comparison Plots of the Wilson θ and Newmark Schemes with the New SSpj Schemes - 95
	a. Convergence Tolerance Equal To 0.005	
	b. Convergence Tolerance Equal To 0.001	
5.15	Comparison Plots of the SSpj Schemes with Different Iterative Tolerances. 96
5.16	Comparison Plots of the Wilson θ Scheme with Different Time Step Sizes - 98
	a. Displacement Plot	
	b. Solution Time	
5.17	Comparison Stress Plot of the SSpj Schemes with Different Time Step Sizes. 99
5.18	Comparison Plots of Different Solution Schemes with Time Step Equal to 5.0E-4 Seconds - 103
	a. Displacement Plot	
	b. Solution Time	
5.19	Comparison Stress Plot of Different Solution Schemes with Time Step Equal to 5.0E-4 Seconds. 104
5.20	Comparison Plots of Different Solution Schemes with Time Step Equal to 1.25E-4 Seconds - 105
	a. Displacement Plot	
	b. Solution Time	
5.21	Comparison Plot of Different Parameter Values for Wilson θ and Newmark Schemes, Δt 5.0E-4 Seconds - 106
	a. Displacement Plot	
	b. Solution Time	
B.1	Example Input File 122
B.2	Example Output File " <i>filename</i> ".res 124

B.3	a. Example Output File " <i>filename</i> ".dat for Quadratic Solution Scheme 125
	b. Example Output File " <i>filename</i> ".dat for Cubic Solution Scheme	
B.4	Example Output File ap_ " <i>filename</i> " 127
B.5	Example Output File " <i>filename</i> ".dat2 127
C.1	Comparison Plots of the Trapezium Scheme with Different Time Step Sizes - 143
	a. Displacement Plot	
	b. Solution Time	
C.2	Comparison Plots of the Bossak Case 2 Scheme with Different Time Step Sizes - 144
	a. Displacement Plot	
	b. Solution Time	
C.3	Comparison Plots of the Hilber, Hughes and Taylor Scheme with Different Time Step Sizes - 145
	a. Displacement Plot	
	b. Solution Time	
C.4	Comparison Plots of the Houbolt Scheme with Different Time Step Sizes - 146
	a. Displacement Plot	
	b. Solution Time	

LIST OF NOTATION

A	=	Amplification matrix
A_i	=	Invariants of amplification matrix
a_i	=	Nonsap parameter values
b_i	=	Routh-Herwitz equations
C	=	Damping matrix
C_i	=	Consistency equation
c	=	SDOF damping
E	=	Young's modulus
E_t	=	Strain hardening
F_{NL}	=	Non-linear internal forces
$f(t)$	=	External forcing function
f_D, f_S, f_I	=	Damping, stiffness and internal forces
I	=	Identity matrix
K	=	Stiffness matrix
K_L, K_{NL}	=	Linear and non-linear stiffness contributions
k	=	SDOF stiffness
L	=	Load operator
M	=	Mass matrix
m	=	SDOF mass
N_i	=	Shape function
n	=	Time step
$P(x)$	=	Internal force vector
$P_n(x_i)$	=	Divided difference polynomial
R	=	Nonsap load
RE	=	Effective residual equation
T	=	Period
t	=	Time
W	=	Weighting function

x	=	Displacement
Δx	=	Increment in displacement
α	=	Approximation equation parameter
$\alpha_n^{(p)}$	=	Average of pth derivative
$\Delta \alpha_n^{(p)}$	=	Incremental change in average p derivative
β	=	Approximation equation parameter
β_i	=	GNpj parameter
γ	=	Approximation equation parameter
δ	=	Approximation equation parameter
Δt	=	Time increment
ε	=	Strain
θ_w	=	Wilson θ parameter
θ_i	=	SSpj parameter
λ	=	Eigenvalue for amplification matrix
ρ	=	Density
σ	=	Stress
$\sigma(\zeta)$	=	Characteristic polynomial
σ_y	=	Yield stress
τ	=	Variable increase in time
ν	=	Poissons ratio
$\phi(\zeta)$	=	Characteristic polynomial
ω	=	Frequency
\mathcal{L}	=	Linear difference operator
\dot{x}	=	Velocity
\ddot{x}	=	Acceleration
$\frac{p}{x}$	=	p derivative of x
\hat{K}	=	Effective stiffness matrix
\hat{R}	=	Effective load vector

CHAPTER ONE

INTRODUCTION

The objective of this thesis is to review the approximation methods which model linear dynamic equations and then apply these same methods to non-linear problems. Ultimately this research will be used to model the non-linear dynamic equations governing structural vibrations due to ice impact, however this is not within the scope of this thesis.

The dynamic system with n-degrees of freedom is represented by non-linear differential equations. Geometric and material variations cause the non-linearity in the structural stiffness. Figure 1.1 represents the generalized single degree of freedom dynamic system. The forces existing in the system are shown in the force equilibrium diagram. The terms f_d , f_s and f_i denote the damping, stiffness and inertial forces respectively. The equilibrium equation of forces acting on the mass may be written:

$$f_d(t) + f_s(t) + f_i(t) = f(t) \quad (1.1)$$

Below is the corresponding general multi-degree of freedom dynamic equation formed in terms of the systems kinematics.

$$M\ddot{x} + C\dot{x} + Kx = f(t) \quad (1.2)$$

Where M , C and K are the mass, damping and stiffness matrices, x the displacement, the dots represent time derivatives, and $f(t)$ is the applied external force.

Direct time integration of the differential equations can require a large number of very small time steps depending on the degree of non-linearity; this is because it is assumed that the physical properties of the system remain linear or constant for the increments of time. The time stepping is generally based on finite difference methods which simplify the numerical integration process. Difference equations move forward in time for a finite

number of discrete time intervals, Δt . Assumptions on the variation of displacement, velocity and acceleration are made for each time step, the form of these assumptions affects the accuracy, stability and eventual cost of the solution procedure.

The Newmark and Houbolt schemes (Bathe and Wilson (1973)), two of the earliest one-step formulations, were conceived in the 1950's. E.L.Wilson formed a direct integration scheme, called the Wilson θ method, at the University of California in 1968 (Bathe and Wilson (1973) and Clough and Penzien (1993)). The Wilson θ method is a self starting, one-step integration scheme, which satisfies the dynamic equations of motion outside the time interval at the θ , or collocation point. Since the development of these three well known and indeed well utilized schemes, researchers have continued to try and find the ultimate approximation method that is highly sensitive to stability and accuracy considerations. O.C.Zienkiewicz, one of the pioneers in the development of the dynamic approximation methods, looked at the formulation of a general solution scheme first by examining the Taylor Series Collocation and then the Weighted Residual approach. The earliest work formed multi-step finite difference formulas; later it was found that one-step methods were more advantageous and hence the development of the "Single-Step Time Marching" and "Generalized Newmark" procedures (Zienkiewicz and Taylor (1991)).

Most of the one-step schemes used to solve dynamic equations use a similar methodology as the Wilson θ method. To begin the path of research, an examination of this method will be conducted. This will then provide a platform from which to work.

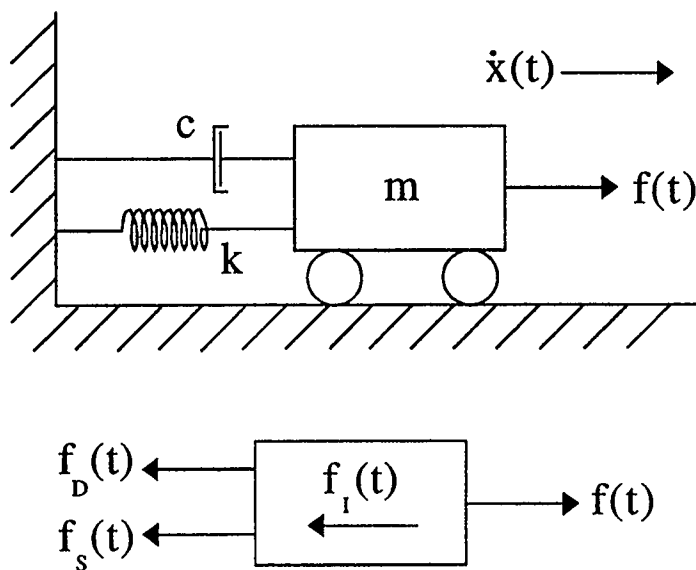


FIGURE 1.1 GENERALIZED SINGLE DEGREE OF FREEDOM DYNAMIC SYSTEM

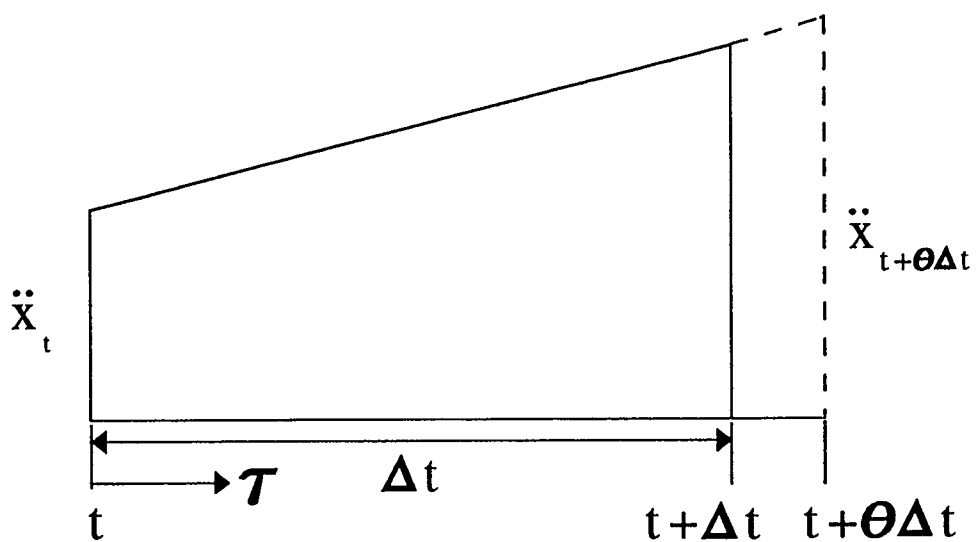


FIGURE 1.2 LINEAR ACCELERATION FOR A DISCRETE TIME PERIOD

1.1 WILSON θ METHOD

The Wilson θ method models the dynamic equation by splitting a time domain into a series of discrete time steps, Δt . Acceleration is assumed to vary linearly over an extended time interval, from t to $t+\theta\Delta t$, see Figure 1.2. Using the known initial conditions at the start of the time step, the conditions at the end of the time increment are calculated explicitly.

τ represents a variable increase in time, its value ranging from $t \leq \tau \leq t+\theta\Delta t$. From Figure 1.2 it is seen that equation (1.3) can be formulated by comparing the accelerations from time t to $t+\Delta t$ and t to $t+\tau$.

$$\frac{\ddot{x}_{t+\tau} - \ddot{x}_t}{\ddot{x}_{t+\Delta t} - \ddot{x}_t} = \frac{\tau}{\Delta t} \quad (1.3)$$

Equation (1.3) is rearranged in terms of the acceleration at any point in the time interval.

$$\ddot{x}_{t+\tau} = \ddot{x}_t + \frac{\tau}{\Delta t} (\ddot{x}_{t+\Delta t} - \ddot{x}_t) \quad (1.4)$$

To calculate the velocity and displacement at τ , the acceleration equation (1.4) is integrated with respect to time.

$$\dot{x}_{t+\tau} = \dot{x}_t + \ddot{x}_t \tau + \frac{\tau^2}{2\Delta t} (\ddot{x}_{t+\Delta t} - \ddot{x}_t) \quad (1.5)$$

$$x_{t+\tau} = x_t + \dot{x}_t \tau + \frac{\ddot{x}_t \tau^2}{2} + \frac{\tau^3}{6\Delta t} (\ddot{x}_{t+\Delta t} - \ddot{x}_t) \quad (1.6)$$

If τ is made equal to the collocation point, that is the end of the extended time step $\theta\Delta t$, then $t+\tau$ becomes $t+\theta\Delta t$. Equations (1.4), (1.5) and (1.6) are therefore reduced to the

following:

$$\ddot{x}_{t+\theta\Delta t} = \ddot{x}_t + \theta(\ddot{x}_{t+\Delta t} - \ddot{x}_t) \quad (1.7)$$

$$\dot{x}_{t+\theta\Delta t} = \dot{x}_t + \ddot{x}_t\theta\Delta t + \frac{\theta^2\Delta t}{2}(\ddot{x}_{t+\Delta t} - \ddot{x}_t) \quad (1.8)$$

$$x_{t+\theta\Delta t} = x_t + \dot{x}_t\theta\Delta t + \frac{\ddot{x}_t\theta^2\Delta t^2}{2} + \frac{\theta^3\Delta t^2}{6}(\ddot{x}_{t+\Delta t} - \ddot{x}_t) \quad (1.9)$$

To define an approximation solution the dynamic equation (1.2) is first written in terms of the kinematics at the end of the extended time period.

$$M\ddot{x}_{t+\theta\Delta t} + C\dot{x}_{t+\theta\Delta t} + Kx_{t+\theta\Delta t} = f_{t+\theta\Delta t} \quad (1.10)$$

Equations (1.7), (1.8) and (1.9) are substituted into the new dynamic equation (1.10). All prescribed terms are collected together on the right hand side and the unknown acceleration at $t + \Delta t$ terms on the left hand side of the equation.

$$\begin{aligned} \ddot{x}_{t+\Delta t} \left\{ M\theta + \frac{C\theta^2\Delta t}{2} + \frac{K\theta^3\Delta t^2}{6} \right\} &= f_{t+\theta\Delta t} - M\ddot{x}_t(1-\theta) \\ -C \left\{ \dot{x}_t + \ddot{x}_t \left(\theta\Delta t - \frac{\theta^2\Delta t}{2} \right) \right\} - K \left\{ x_t + \dot{x}_t\theta\Delta t + \frac{\ddot{x}_t(\theta\Delta t)^2}{2} \left(1 - \frac{\theta}{3} \right) \right\} & \end{aligned} \quad (1.11)$$

If a new symbol κ is defined, equation (1.11) can be simplified further:

$$\kappa = \left\{ M\theta + \frac{C\theta^2\Delta t}{2} + \frac{K\theta^3\Delta t^2}{6} \right\}^{-1} \quad (1.12)$$

$$\begin{aligned} \ddot{x}_{t+\Delta t} = & f_{t+\theta\Delta t}\kappa - M\kappa\ddot{x}_t(1-\theta) - C\kappa\left\{\dot{x}_t + \ddot{x}_t\left(\theta\Delta t - \frac{\theta^2\Delta t}{2}\right)\right\} \\ & - K\kappa\left\{x_t + \dot{x}_t\theta\Delta t + \frac{\ddot{x}_t(\theta\Delta t)^2}{2}\left(1 - \frac{\theta}{3}\right)\right\} \end{aligned} \quad (1.13)$$

Equations (1.5) and (1.6) are recalculated for τ equal to the end of the discrete time period Δt , giving (1.14) and (1.15).

$$\dot{x}_{t+\Delta t} = \dot{x}_t + \ddot{x}_t\Delta t + \frac{\Delta t}{2}(\ddot{x}_{t+\Delta t} - \ddot{x}_t) \quad (1.14)$$

$$x_{t+\Delta t} = x_t + \dot{x}_t\Delta t + \frac{\ddot{x}_t\Delta t^2}{2} + \frac{\Delta t^2}{6}(\ddot{x}_{t+\Delta t} - \ddot{x}_t) \quad (1.15)$$

The next step is to find the velocity and displacement at the end of the discrete time step, this is achieved by substituting the newly derived acceleration expression into equations (1.14) and (1.15). Now that equations for the kinematics at $t+\Delta t$ are formulated they can be combined and simplified into the final approximation difference equation with the following structure:

$$\begin{Bmatrix} \ddot{x}_{t+\Delta t} \\ \dot{x}_{t+\Delta t} \\ x_{t+\Delta t} \end{Bmatrix} = A \begin{Bmatrix} \ddot{x}_t \\ \dot{x}_t \\ x_t \end{Bmatrix} + L f_{t+\theta\Delta t} \quad (1.16)$$

Where A is the amplification matrix and L the load operator, and they have the following formation:

$$A = \begin{bmatrix} \kappa \varepsilon & -\kappa(C + K\theta\Delta t) & -\kappa K \\ \frac{\Delta t}{2}(1 + \kappa \varepsilon) & 1 - \frac{\Delta t}{2}\kappa(C + K\theta\Delta t) & -\frac{\kappa K \Delta t}{2} \\ \frac{\Delta t^2}{3}(1 + \frac{\kappa \varepsilon}{2}) & \Delta t \left(1 - \frac{\Delta t}{6}\kappa(C + K\theta\Delta t)\right) & 1 - \frac{\kappa K \Delta t^2}{6} \end{bmatrix} \quad (1.17)$$

$$L = \begin{bmatrix} \kappa \\ \frac{\Delta t}{2}\kappa \\ \frac{\Delta t^2}{6}\kappa \end{bmatrix} \quad (1.18)$$

Where:

$$\varepsilon = M(\theta - 1) + C\theta\Delta t \left(\frac{\theta}{2} - 1\right) + K \frac{(\theta\Delta t)^2}{2} \left(\frac{\theta}{3} - 1\right) \quad (1.19)$$

The Wilson θ solution scheme can also be applied using the following matrix equation where τ is equal to $\theta\Delta t$:

$$\hat{K}x_{t+\tau} = \hat{R}_{t+\tau} \quad (1.20)$$

The effective stiffness matrix and effective load vector are calculated in Bathe and Wilson (1976), for reference they are listed here:

$$\hat{K} = K + \frac{6}{(\theta\Delta t)^2}M + \frac{3}{\theta\Delta t}C \quad (1.21)$$

$$\begin{aligned} \hat{R}_{t+\theta\Delta t} = f_t + \theta(f_{t+\Delta t} - f_t) + M \left(\frac{6}{(\theta\Delta t)^2} x_t + \frac{6}{\theta\Delta t} \dot{x}_t + 2\ddot{x}_t \right) \\ C \left(\frac{3}{\theta\Delta t} x_t + 2\dot{x}_t + \frac{\theta\Delta t}{2} \ddot{x}_t \right) \end{aligned} \quad (1.22)$$

This method is adopted in Nonsap, a computer program which will be examined further in Chapter Four.

The displacement calculated at the collocation point $\theta\Delta t$ is then used to solve the following expressions for displacement, velocity and acceleration at the end of the time step.

$$\begin{aligned} \ddot{x}_{t+\Delta t} = \frac{6}{\theta^3\Delta t^2}(x_{t+\theta\Delta t} - x_t) - \frac{6}{\theta^2\Delta t}\dot{x}_t + \ddot{x}_t \left(1 - \frac{3}{\theta} \right) \\ \dot{x}_{t+\Delta t} = \dot{x}_t + \frac{\Delta t}{2}(\ddot{x}_{t+\Delta t} + \ddot{x}_t) \end{aligned} \quad (1.23)$$

$$x_{t+\Delta t} = x_t + \Delta t\dot{x}_t + \frac{\Delta t^2}{6}(\ddot{x}_{t+\Delta t} + 2\ddot{x}_t)$$

1.2 THESIS OUTLINE

The next two chapters will review a selection of common approximation methods under two classifications, multi-step and single-step schemes. The general "Single-Step Time Marching" solution scheme, which can model a variety of other methods, will be examined in detail. Procedures for assessing the approximation methods for linear

problems will be investigated. Computer program dynamic.c, developed by the writer to study a selection of common solution schemes including the "Single-Step Time Marching" scheme, will be used to apply some of these procedures.

Chapters four and five of the thesis will look at non-linear problems, the idea of iterative procedures to model material non-linearity will be introduced. The finite element program Nonsap, developed by Bathe, Wilson and Iding, was modified to include the general "Single-Step Time Marching" scheme. Nonsap will be used to explore the accuracy of the displacements output by the approximation methods for non-linear problems. Time step sizes will be studied to determine their effect on the solution time for both the approximation method and the non-linear equilibrium iterative procedure.

The final chapter will draw on the conclusions made during the analysis of the approximation methods for both linear and non-linear problems. Some recommendations will also be suggested.

CHAPTER TWO

DYNAMIC SOLUTION SCHEMES

Since the development of the Wilson θ scheme there has been a concerted effort to try and formulate the ideal numerical method approximation. In the 1960's and 1970's many of these schemes took the form of multi-step finite difference approximations. These algorithms have now been superseded by improved single-step methods, development of which mostly began in the 1980's and they remain a topic of research today. This chapter will examine both multi- and single-step methods.

2.1 MULTI-STEP SCHEMES

Multi-step algorithms are finite difference approximations of the dynamic equation. Finite difference approximations require knowledge of the displacement and force history over a number of time steps to determine new displacements. The name multi-step is derived from the notion that the kinematic history is prescribed for a number of discrete time increments.

2.1.1 TRUNCATED TAYLOR SERIES COLLOCATION

Finite difference expressions for velocity and displacement at the end of a time step, $n+1$, are initially derived using Taylor series expansions:

$$\dot{x}_{n+1} = \dot{x}_n + \ddot{x}_n \Delta t + \frac{\ddot{\ddot{x}}_n \Delta t^2}{2} + \dots \quad (2.1)$$

$$x_{n+1} = x_n + \dot{x}_n \Delta t + \frac{\ddot{x}_n \Delta t^2}{2} + \frac{\ddot{\ddot{x}}_n \Delta t^3}{3!} + \dots \quad (2.2)$$

The velocity expression is truncated after the acceleration term. Acceleration at n is represented by an interpolation function from n to $n+1$. Constant parameter γ , which has

limits $0 \leq \gamma \leq 1$, expresses the variation of acceleration within the time interval in terms of initial and final acceleration. If, for example, γ equals a half then constant acceleration is applied.

$$\dot{x}_{n+1} = \dot{x}_n + (1-\gamma)\ddot{x}_n\Delta t + \gamma\ddot{x}_{n+1}\Delta t \quad (2.3)$$

The displacement equation is also truncated after the acceleration term which is again represented by an interpolation function. Constant parameter 2β controls the acceleration behaviour and has limits $0 \leq 2\beta \leq 1$.

$$x_{n+1} = x_n + \dot{x}_n\Delta t + \Delta t^2(\frac{1}{2}-\beta)\ddot{x}_n + \Delta t^2\beta\ddot{x}_{n+1} \quad (2.4)$$

If equations (2.3) and (2.4) are rearranged in terms of the new acceleration they become:

$$\ddot{x}_{n+1} = \frac{1}{\gamma\Delta t} (\dot{x}_{n+1} - \dot{x}_n - (1-\gamma)\ddot{x}_n\Delta t) \quad (2.5)$$

$$\ddot{x}_{n+1} = \frac{1}{\beta\Delta t^2} (x_{n+1} - x_n - \dot{x}_n\Delta t - (\frac{1}{2}-\beta)\ddot{x}_n\Delta t^2) \quad (2.6)$$

Chan *et. al.* (1969) formulated a general expression for the special case of the dynamic equation where γ equals a half. A similar procedure can be applied to produce a general expression for variable γ , see solution in Appendix A.1. The method utilizes equation (1.2) at time t_{n+1} , t_n and t_{n-1} . Substitutions for displacement, velocity and acceleration at the end of the time steps using equations (2.3)-(2.6) are made. The final simplified form of the difference equation is:

$$\begin{aligned}
& M [x_{n+1} - 2x_n + x_{n-1}] + C\Delta t [\gamma x_{n+1} + (1-2\gamma)x_n + (\gamma-1)x_{n-1}] \\
& + K\Delta t^2 \left[\beta x_{n+1} + \left(\frac{1}{2} + \gamma - 2\beta\right)x_n + \left(\frac{1}{2} - \gamma + \beta\right)x_{n-1} \right] \\
& - \Delta t^2 \left[\beta f_{n+1} + \left(\frac{1}{2} + \gamma - 2\beta\right)f_n + \left(\frac{1}{2} - \gamma + \beta\right)f_{n-1} \right] = 0
\end{aligned} \tag{2.7}$$

The displacement and velocity expansions can also be used to derive a matrix equation in terms of displacement, velocity and acceleration at a single time step as shown in Chapter One with the Wilson θ method, Hilber and Hughes (1978) and Hilber *et. al.* (1977). A general single-step solution scheme which encompasses the above method will be discussed later in section 2.2.1.

2.1.2 A WEIGHTED RESIDUAL APPROACH

An alternative method of deriving equation (2.7) is to use the finite element weighted-residual process documented by Zienkiewicz (1977). The domain of the problem is set from $-\Delta t$ to $+\Delta t$, and the unknown displacement term x , is expanded in terms of shape functions N_i :

$$x = \sum N_i x_i \quad i = n-1, n, n+1 \tag{2.8}$$

The resultant shape functions at time $n+1$, n and $n-1$ are seen in Figure 2.1 a-c, they are written mathematically as follows: $(-1 \leq \xi = t/\Delta t \leq +1)$

$$\begin{aligned}
N_{n-1} &= -\frac{\xi}{2}(1-\xi) \\
N_n &= (1-\xi)(1+\xi) \\
N_{n+1} &= \frac{\xi}{2}(1+\xi)
\end{aligned} \tag{2.9}$$

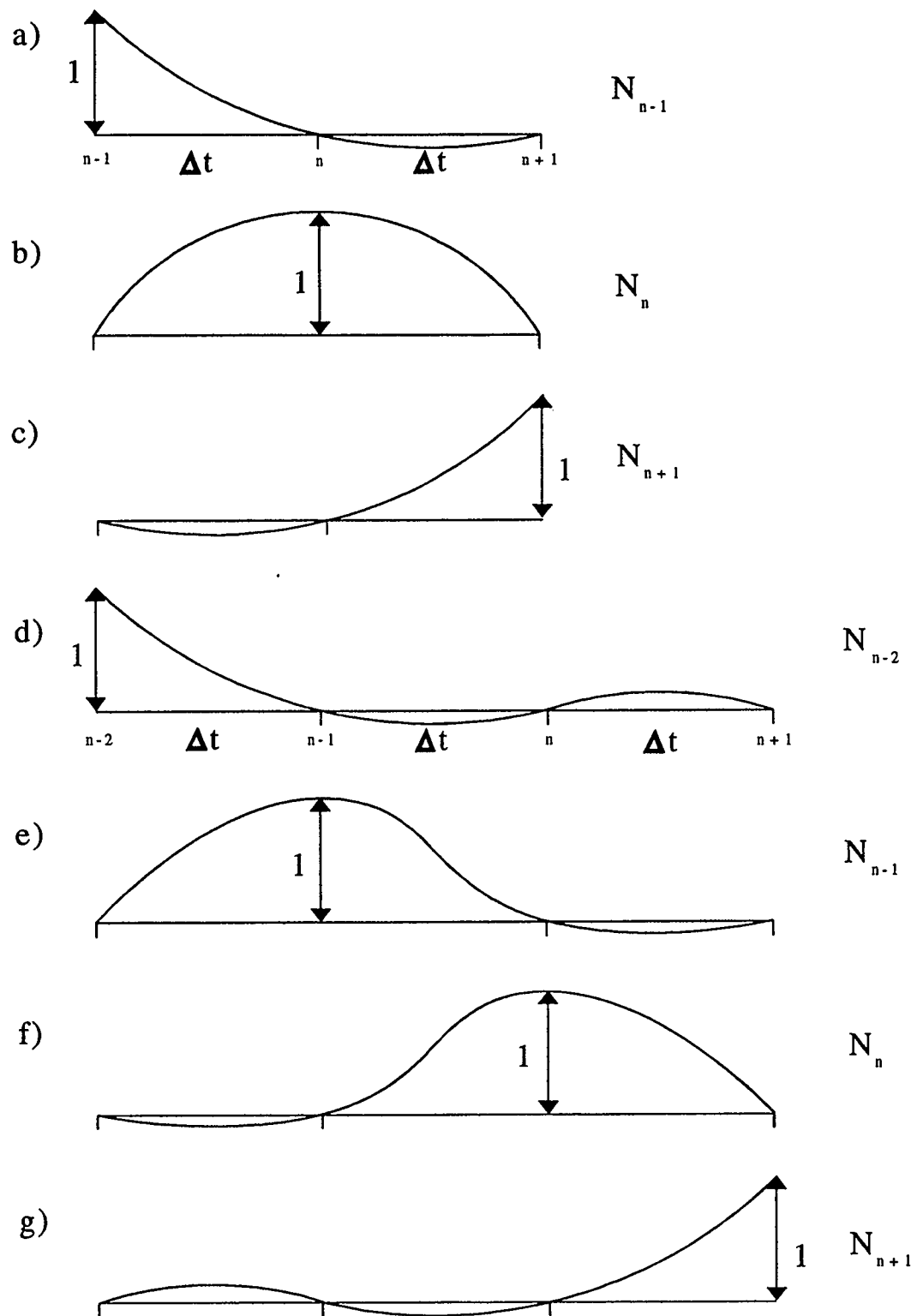


FIGURE 2.1 SHAPE FUNCTIONS FOR A QUADRATIC AND A CUBIC EQUATION

A weighted-residual equation for the set domain is calculated using (2.10):

$$\int_{-\Delta t}^{\Delta t} W \left[M \ddot{x} + C \dot{x} + Kx - f \right] dt = 0 \quad (2.10)$$

$$\int_{-\Delta t}^{\Delta t} W \left[M \sum \ddot{N}_i x_i + C \sum \dot{N}_i x_i + K \sum N_i x_i - \sum f_i N_i \right] dt = 0$$

The above shape functions are differentiated with respect to time (ξ is replaced by $t/\Delta t$) and then substituted into equation (2.10). It is assumed that f is interpolated in the same way as x in equation (2.8) using the same shape functions. On integration the following expression is formulated if W is set equal to one:

$$M \left[x_{n+1} - 2x_n + x_{n-1} \right] + \frac{C\Delta t}{2} \left[x_{n+1} - x_{n-1} \right] \quad (2.11)$$

$$+ \frac{K\Delta t^2}{3} \left[\frac{x_{n+1}}{2} + 2x_n + \frac{x_{n-1}}{2} \right] - \frac{\Delta t^2}{3} \left[\frac{f_{n+1}}{2} + 2f_n + \frac{f_{n-1}}{2} \right] = 0$$

The full derivation of this equation is in Appendix A.2. Equation (2.11) is identical to (2.7) when parameters $\gamma=1/2$ and $\beta=1/6$ are substituted. Values for the parameters γ and β are determined for the known weighting function W . The standard equations (2.12) and (2.13) for γ and β are:

$$\gamma = \left[\int_{-1}^1 W \left(\xi + \frac{1}{2} \right) d\xi / \int_{-1}^1 W d\xi \right] \quad (2.12)$$

$$\beta = \frac{1}{2} \left[\int_{-1}^1 W \xi (1 + \xi) d\xi / \int_{-1}^1 W d\xi \right] \quad (2.13)$$

These parameters are solved using a weighting function W equal to one which yields our values of $\gamma=1/2$ and $\beta=1/6$. An improved form of this equation is achieved by increasing the order from a quadratic to a cubic using shape functions for four time stations $n+1$, n , $n-1$, $n-2$, with ξ in the domain $0 \leq \xi \leq 3$, see Figure 2.1 d-g.

$$\begin{aligned}
 N_{n+1} &= \frac{\xi}{6} (\xi - 1)(\xi - 2) \\
 N_n &= -\frac{\xi}{2} (\xi - 1)(\xi - 3) \\
 N_{n-1} &= \frac{\xi}{2} (\xi - 2)(\xi - 3) \\
 N_{n-2} &= -\frac{\xi}{6} (\xi - 1)(\xi - 2)(\xi - 3)
 \end{aligned} \tag{2.14}$$

To study the accuracy and stability of the general cubic equation a particular dynamic case is examined where damping and force effects are eliminated. ($\omega^2=k/m$)

$$\ddot{x} = -\omega^2 x = g(x,t) \tag{2.15}$$

The cubic equation derivation follows a similar methodology to the quadratic equation, the following is the weighted-residual solution:

$$\begin{aligned}
 &\left[(\gamma-1) + \left(\frac{\alpha}{6} - \frac{\beta}{2} + \frac{\gamma}{3} \right) (\omega \Delta t)^2 \right] x_{n+1} + \left[(-3\gamma+4) + \left(-\frac{\alpha}{2} + 2\beta - \frac{3}{2}\gamma \right) (\omega \Delta t)^2 \right] x_n \\
 &\left[(3\gamma-5) + \left(\frac{\alpha}{2} - \frac{5}{2}\beta + 3\gamma \right) (\omega \Delta t)^2 \right] x_{n-1} + \left[(-\gamma+2) + \left(-\frac{\alpha}{6} + \beta - \frac{11}{6}\gamma + 1 \right) (\omega \Delta t)^2 \right] x_{n-2} = 0
 \end{aligned} \tag{2.16}$$

Because the order of the general equation has been increased by one there are three parameters to consider α , β and γ .

$$\begin{aligned}
\alpha &= \int_0^3 W \xi^3 d\xi / \int_0^3 W d\xi \\
\beta &= \int_0^3 W \xi^2 d\xi / \int_0^3 W d\xi \\
\gamma &= \int_0^3 W \xi d\xi / \int_0^3 W d\xi
\end{aligned}
\tag{2.17}$$

The Wilson θ family parameters are derived using equivalence equations introduced by Wood, see footnote in Zienkiewicz (1977).

$$\begin{aligned}
\alpha &= 2 + 4\theta + 3\theta^2 + \theta^3 \\
\beta &= \frac{4}{3} + 2\theta + \theta^2 \\
\gamma &= 1 + \theta
\end{aligned}
\tag{2.18}$$

Table 2.1 lists the parameter values for some of the common solution schemes to be used in the general cubic equation.

Method	Controlling Factor	α	β	γ
Houbolt	$W=1$ @ $t=3$	27	9	3
Wilson θ	$\theta = 1.4$	16.224	6.0933	2.4
Wilson θ	$\theta = 2.0$	30	28/3	3
Newmark	$\theta = 0.0$	2	4/3	1

TABLE 2.1 PARAMETER VALUES FOR COMMON SOLUTION SCHEMES USING THE WEIGHTED RESIDUAL CUBIC EQUATION

Equation (2.16) can be reduced to a difference equation when the $(\omega\Delta t)^2$ terms are moved to the right hand side and like displacement terms are collected together.

$$\sum_{j=0}^3 v_j x_{n+j-2} = \Delta t^2 \sum_{j=0}^3 \eta_j g_{n+j-2} \quad (2.19)$$

The variables v_j and η_j are as follows:

$$\begin{aligned} v_0 &= \frac{2-\gamma}{\gamma-1} & \eta_0 &= \frac{-\frac{1}{6}\alpha + \beta - \frac{11}{6}\gamma + 1}{\gamma-1} \\ v_1 &= \frac{3\gamma-5}{\gamma-1} & \eta_1 &= \frac{\frac{1}{2}\alpha - \frac{5}{2}\beta + 3\gamma}{\gamma-1} \\ v_2 &= \frac{4-3\gamma}{\gamma-1} & \eta_2 &= \frac{-\frac{1}{2}\alpha + 2\beta - \frac{3}{2}\gamma}{\gamma-1} \\ v_3 &= 1 & \eta_3 &= \frac{\frac{1}{6}\alpha - \frac{1}{2}\beta + \frac{1}{3}\gamma}{\gamma-1} \end{aligned} \quad (2.20)$$

Wood (1977), used the above difference equation to analyze the stability and accuracy of the solution schemes listed in Table 2.1, a discussion of his work is in chapter three.

2.2 SINGLE-STEP SCHEMES

The single-step algorithms approximate the dynamic equation using known displacement, velocity and force values from the previous step only and hence the name single-step.

2.2.1 GENERALIZED NEWMARK SCHEMES

In 1984 Katona began developing the beta-m method which was later to be published in a paper by Katona and Zienkiewicz (1985). The beta-m method is a single-step formulation of the Truncated Taylor Series Collocation family. Its development was a

response to the need for a variable time increment which could yield more accurate results. The single-step schemes were also an advantage as only initial conditions from the previous time step were required to initiate the solution process, and also they were viable methods for non-linear dynamic problems. Since the beta-m method, the Generalized Newmark scheme, which is essentially the same but simpler to apply, was derived. This scheme is also known as the GNpj method (Zienkiewicz and Taylor (1991)), the p representing the order of approximation polynomial; for example a linear, quadratic or cubic solution. The j stands for the order of the problem equation which in our case is dynamic and therefore j equals two. The following three equations are Taylor expansion approximations at the collocation point for displacement, velocity and acceleration. Note the notation:

$$\frac{p}{x} = \frac{d^p x}{dt^p}$$

$$x_{n+1} = x_n + \Delta t \dot{x}_n + \dots + \frac{\Delta t^p}{p!} \frac{p}{x_n} + \beta_p \frac{\Delta t^p}{p!} \left(\frac{p}{x_{n+1}} - \frac{p}{x_n} \right) \quad (2.21)$$

$$\dot{x}_{n+1} = \dot{x}_n + \Delta t \ddot{x}_n + \dots + \frac{\Delta t^{p-1}}{(p-1)!} \frac{p}{x_n} + \beta_{p-1} \frac{\Delta t^{p-1}}{(p-1)!} \left(\frac{p}{x_{n+1}} - \frac{p}{x_n} \right) \quad (2.22)$$

$$\frac{p-1}{x_{n+1}} = \frac{p-1}{x_n} + \Delta t \frac{p}{x_n} + \beta_1 \Delta t \left(\frac{p}{x_{n+1}} - \frac{p}{x_n} \right) \quad (2.23)$$

For a cubic solution of the dynamic equation (GN32) the above expressions reduce to:

$$x_{n+1} = x_n + \Delta t \dot{x}_n + \frac{\Delta t^2}{2} \ddot{x}_n + \frac{\Delta t^3}{6} \dddot{x}_n + \beta_3 \frac{\Delta t^3}{6} (\ddot{x}_{n+1} - \ddot{x}_n) \quad (2.24)$$

$$\dot{x}_{n+1} = \dot{x}_n + \Delta t \ddot{x}_n + \frac{\Delta t^2}{2} \dddot{x}_n + \beta_2 \frac{\Delta t^2}{2} (\ddot{x}_{n+1} - \ddot{x}_n) \quad (2.25)$$

$$\ddot{x}_{n+1} = \ddot{x}_n + \Delta t \dddot{x}_n + \beta_1 \Delta t (\ddot{x}_{n+1} - \ddot{x}_n) \quad (2.26)$$

The β_i parameters control the variation of the i th differential term over the time interval n to $n+1$; in the GN22 case this parameter dictates the acceleration behaviour. The dynamic equation becomes the following standard matrix equation with an effective stiffness and effective load term when equations (2.24)-(2.26) are substituted into the general equation (1.2) at time step $n+1$.

$$\hat{K} \ddot{x}_{n+1} = -\hat{R} \quad (2.27)$$

$$\hat{K} = M\beta_1\Delta t + C\beta_2\frac{\Delta t^2}{2} + K\beta_3\frac{\Delta t^3}{6} \quad (2.28)$$

$$\begin{aligned} \hat{R} = & M \left\{ \ddot{x}_n + \ddot{x}_n \Delta t (1 - \beta_1) \right\} + C \left\{ \dot{x}_n + \ddot{x}_n \Delta t + \ddot{x}_n \frac{\Delta t^2}{2} (1 - \beta_2) \right\} \\ & + K \left\{ x_n + \dot{x}_n \Delta t + \ddot{x}_n \frac{\Delta t^2}{2} + \ddot{x}_n \frac{\Delta t^3}{6} (1 - \beta_3) \right\} - f \end{aligned} \quad (2.29)$$

Once the matrix equation is solved the \ddot{x}_{n+1} term can be substituted back into equations

(2.24)-(2.26) to solve for the new displacement, velocity and acceleration. The load vector f is dealt with in the next section 2.2.2.

2.2.2 SINGLE-STEP TIME MARCHING SCHEMES

This set of solution schemes follows the same line of argument as the multi-step Weighted Residual Method by Zienkiewicz described in section 2.1.2. This scheme is known as the SSpj method, Zienkiewicz *et. al.* (1984) and Zienkiewicz and Taylor (1991), where the p represents the order of the approximation polynomial and the j the order of the dynamic equation which is two.

A function x is approximated as the following polynomial:

$$x = x_n + \dot{x}_n t + \frac{\ddot{x}_n t^2}{2} + \dots + \alpha_n^{(p)} t^p \frac{1}{p!} \quad (2.30)$$

This function and its derivatives are used to solve the Weighted Residual equation (2.10) with time interval from 0 to Δt , expression (2.31) is defined to reduce the final equation.

$$\int_0^{\Delta t} W [M\ddot{x} + C\dot{x} + Kx - f] dt = 0 \quad (2.10)$$

$$\frac{\int_0^{\Delta t} W t^q dt}{\int_0^{\Delta t} W dt} = \theta_q \Delta t^q \quad (2.31)$$

The process just discussed is simplified with the substitution of the following equations which define displacement, velocity and acceleration at the end of the time step in terms

of previous known values at the start:

$$x_{n+1} = \sum_{q=0}^{p-1} \frac{\dot{x}_n \Delta t^q \theta_q}{q!} + \frac{\alpha_n^{(p)} \Delta t^p \theta_p}{p!} \quad (2.32)$$

$$\dot{x}_{n+1} = \sum_{q=1}^{p-1} \frac{\dot{x}_n \Delta t^{q-1} \theta_{q-1}}{(q-1)!} + \frac{\alpha_n^{(p)} \Delta t^{p-1} \theta_{p-1}}{(p-1)!} \quad (2.33)$$

$$\ddot{x}_{n+1} = \sum_{q=2}^{p-1} \frac{\dot{x}_n \Delta t^{q-2} \theta_{q-2}}{(q-2)!} + \frac{\alpha_n^{(p)} \Delta t^{p-2} \theta_{p-2}}{(p-2)!} \quad (2.34)$$

Parameter $\alpha_n^{(p)}$ represents some average value of the pth derivative for the time interval. See Figure 2.2 for graphical representation of the SSpj method.

If p is set equal to three the kinematic equations (2.32)-(2.34) become:

$$x_{n+1} = x_n + \dot{x}_n \Delta t \theta_1 + \ddot{x}_n \frac{\Delta t^2 \theta_2}{2} + \alpha_n^{(3)} \frac{\Delta t^3 \theta_3}{6} \quad (2.35)$$

$$\dot{x}_{n+1} = \dot{x}_n + \ddot{x}_n \Delta t \theta_1 + \alpha_n^{(3)} \frac{\Delta t^2 \theta_2}{2} \quad (2.36)$$

$$\ddot{x}_{n+1} = \ddot{x}_n + \alpha_n^{(3)} \Delta t \theta_1 \quad (2.37)$$

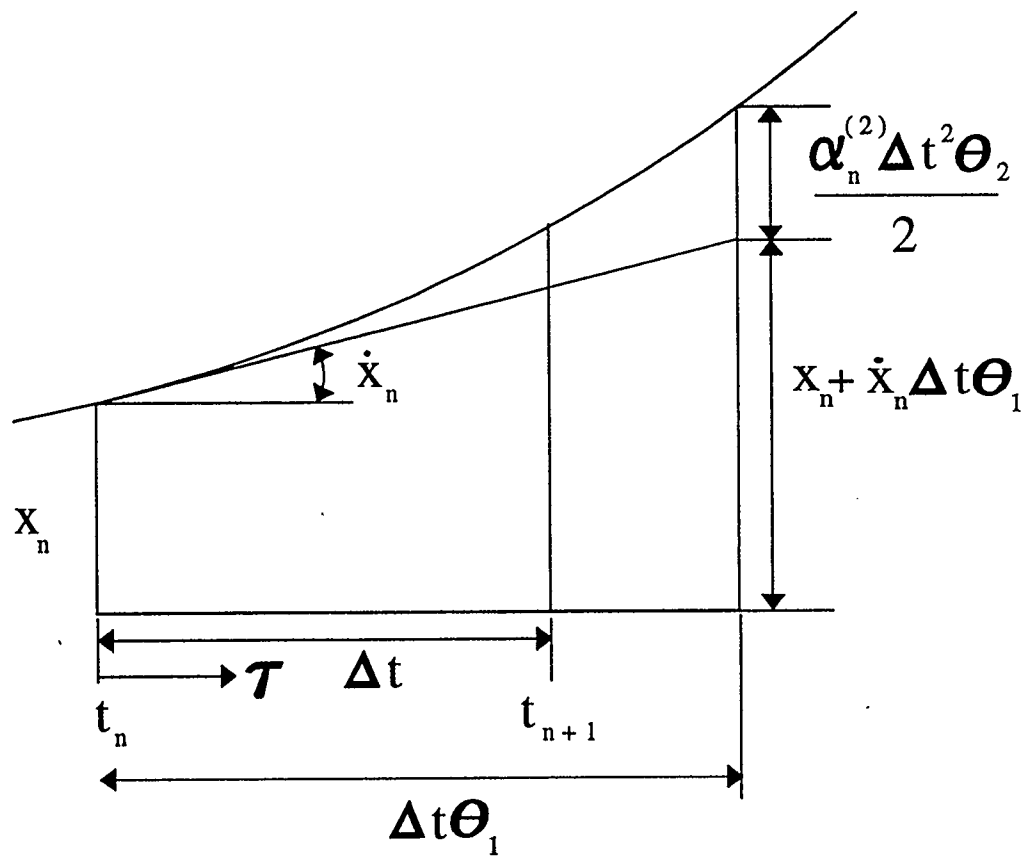


FIGURE 2.2 APPROXIMATION OF x FOR A QUADRATIC ALGORITHM

After substitution of the kinematic equations into (2.10) the following matrix equation is generated:

$$\alpha_n^{(3)} = -\hat{K}^{-1} \hat{R} \quad (2.38)$$

$$\hat{K} = M \Delta t \theta_1 + \frac{C \Delta t^2 \theta_2}{2} + \frac{K \Delta t^3 \theta_3}{6} \quad (2.39)$$

$$\hat{R} = M(\ddot{x}_n) + C(\dot{x}_n + \ddot{x}_n \Delta t \theta_1) + K \left(x_n + \dot{x}_n \Delta t \theta_1 + \ddot{x}_n \frac{\Delta t^2 \theta_2}{2} \right) - f \quad (2.40)$$

The final displacement, velocity and acceleration values at the end of the time step are then obtained by solving the three polynomial expansions:

$$x_{n+1} = x_n + \dot{x}_n \Delta t + \frac{\ddot{x}_n \Delta t^2}{2} + \alpha_n^{(3)} \frac{\Delta t^3}{6} \quad (2.41)$$

$$\dot{x}_{n+1} = \dot{x}_n + \ddot{x}_n \Delta t + \alpha_n^{(3)} \frac{\Delta t^2}{2} \quad (2.42)$$

$$\ddot{x}_{n+1} = \ddot{x}_n + \alpha_n^{(3)} \Delta t \quad (2.43)$$

The load vector f is assumed to vary linearly over the time interval:

$$f = \theta_1 f_{n+1} + (1 - \theta_1) f_n \quad (2.44)$$

Table 2.2 contains the parameter values required to emulate some more common solution schemes using either the GNpj or the SSpj general equations. The Bossak and the Hilber,

Hughes and Taylor (HHT) parameters represent a dynamic equation with no damping or force effects. The parameters required for a dynamic equation with damping and force can be found in references Zienkiewicz *et. al.* (1984), Katona and Zienkiewicz (1985) and Zienkiewicz and Taylor (1991).

General Equation And Solution Scheme		β_1 GNpj θ_1 SSpj	β_2 GNpj θ_2 SSpj	β_3 GNpj θ_3 SSpj
SS22/GN22	Newmark	δ	2α	
SS32/GN32	Houbolt	2	$11/3$	6
	Wilson Theta	θ_w	θ_w^2	θ_w^3
	Bossak [Wood <i>et. al.</i> (1980)] $\gamma_B = 1/2 - \alpha_B$	$1 - \alpha_B$	$2\beta_B + 2/3 - \alpha_B$	$6\beta_B$
	HHT [Hilber <i>et. al.</i> (1977)] $\gamma_H = 1/2 - \alpha_H$	1	$2\beta_H + 2/3 - 2\alpha_H^2$	$6\beta_H(1 + \alpha_H)$

TABLE 2.2 PARAMETER VALUES FOR COMMON SOLUTION SCHEMES
USING EITHER THE GNpj OR THE SSpj GENERAL EQUATION

CHAPTER THREE

STABILITY AND ACCURACY OF METHODS

The solution schemes discussed in chapter two each behave differently with varying time increment and parameter values, this behaviour is of interest to us as it represents the stability and accuracy of the schemes. To study the significance of the variables chosen, two C programs were written: see appendix B for program documentation and listings. The first program, `dynamic.c`, incorporated some of the common solution schemes including the Single-Step, SSPj, algorithm for a one dimensional problem with no damping effect. The SSPj scheme was selected because it simulates all the common solution methods and also eliminates the requirement to make assumptions about the force and displacement history prior to time step $t-\Delta t$. `Dynamic.c` calculates the oscillating displacement pattern for a prescribed time period, this data is used to determine the amplitude decay and period elongation of the solution. During the displacement calculations the amplification matrix A is formulated, for examples of the amplification matrix see equation (1.17) and section 3.2 of this chapter. This matrix is output to a file for the second program `eigen.c` to calculate the spectral radius of the A matrix for the prescribed conditions, also used for the assessment of accuracy. Figures 3.1 and 3.2 present flow charts of the programs `dynamic.c` and `eigen.c`.

This chapter deals with the various mathematical methods used to assess the solution schemes so that an optimum algorithm can be selected for a particular dynamic problem. The results from the programs are then compared with previous research and some conclusions about the effectiveness of the common schemes are made.

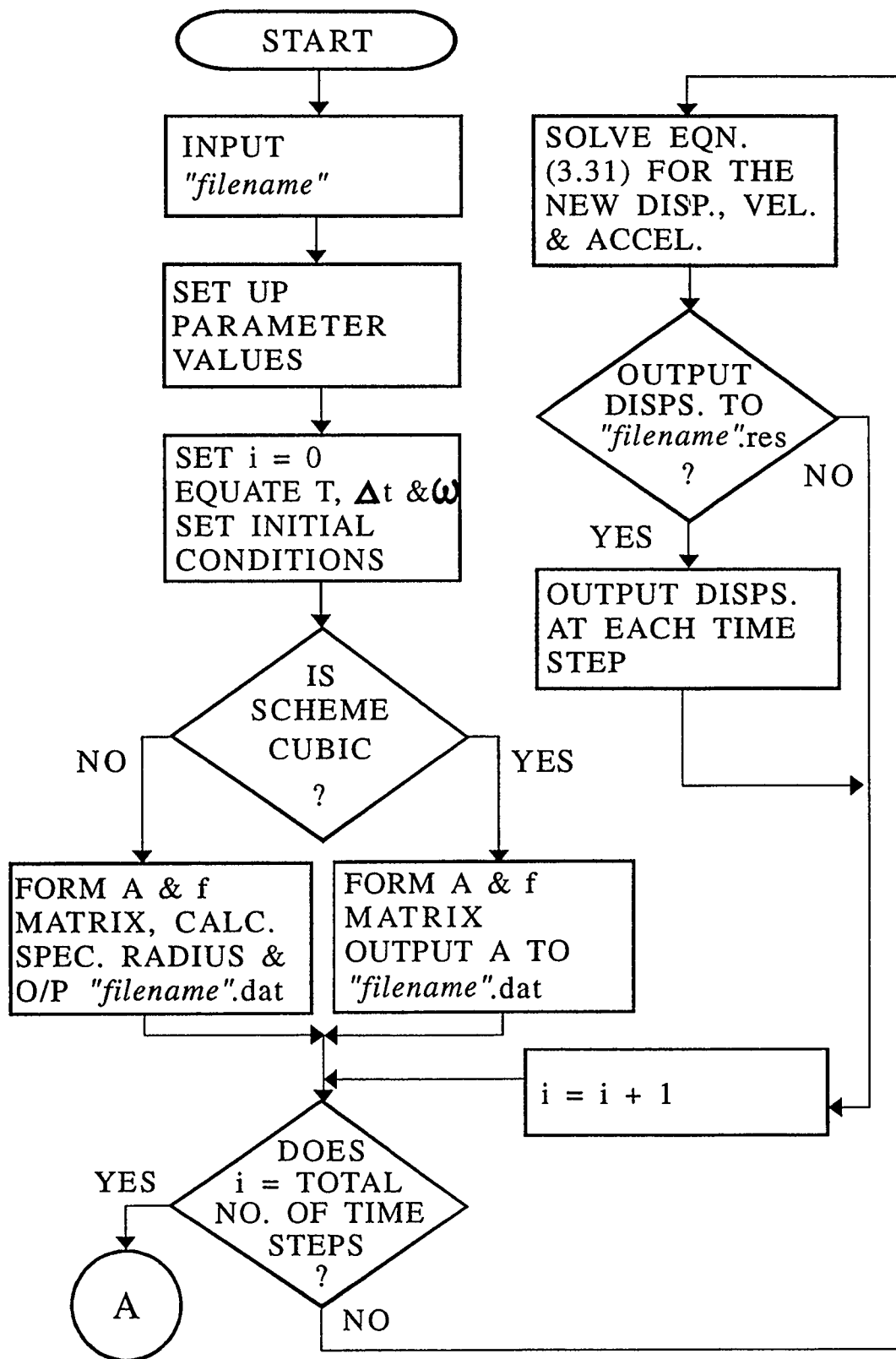


FIGURE 3.1 FLOW CHART OF DYNAMIC.C

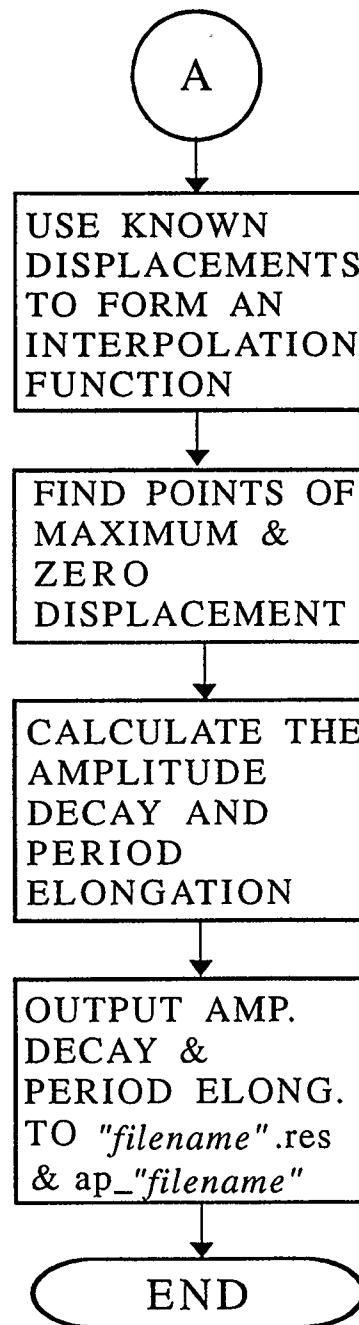


FIGURE 3.1 FLOW CHART OF DYNAMIC.C CONTINUED

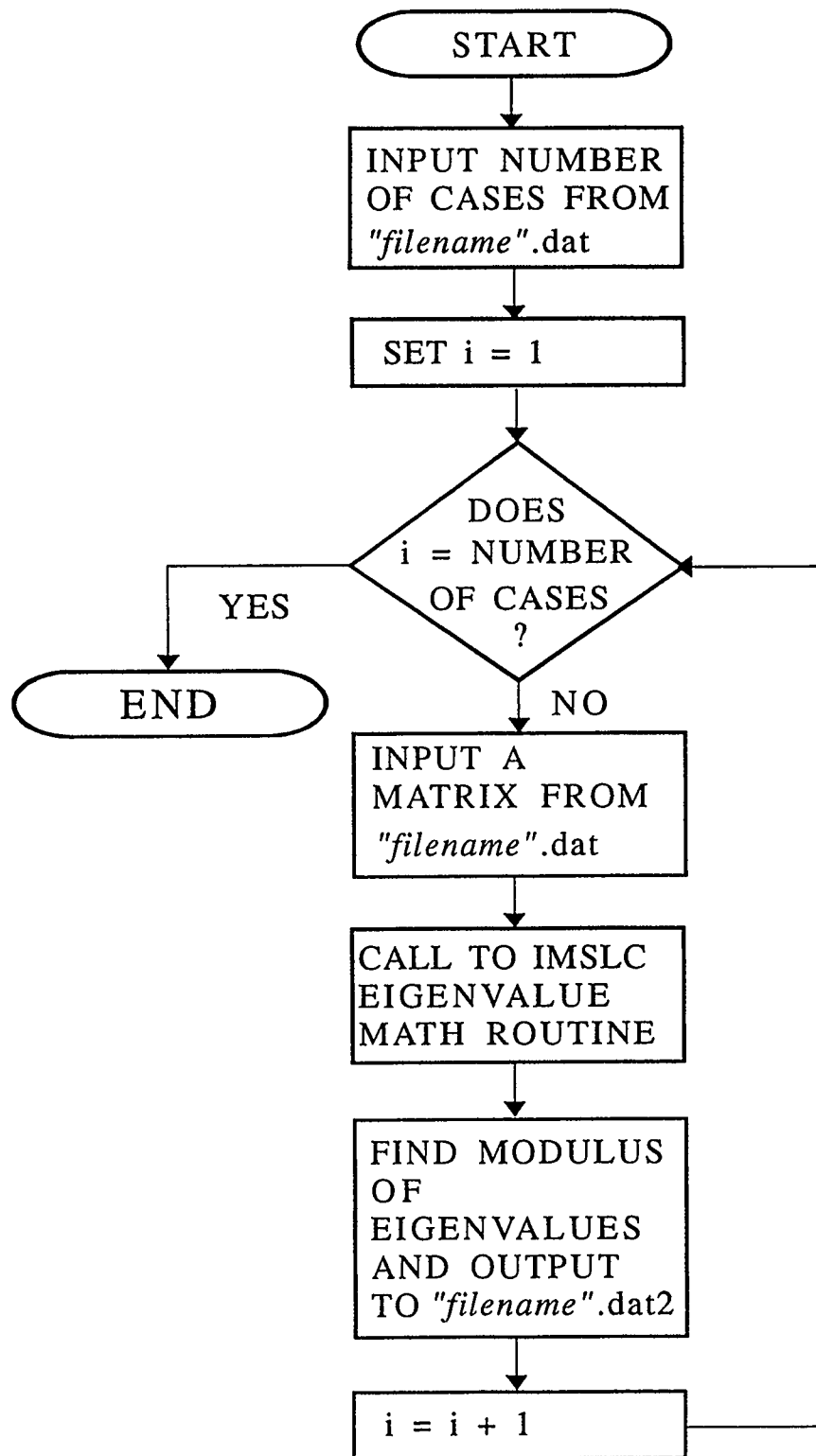


FIGURE 3.2 FLOW CHART OF EIGEN.C

3.1 STABILITY AND ACCURACY

A number of methods which estimate the integration errors for the approximation of the dynamic equation have been developed. These methods examine the stability and accuracy of the solution schemes. For a very accurate solution the schemes require a small time step size which is inversely proportional to the smallest period. However the primary response of the structure may be due to lower frequencies, and hence a much larger time step size which is inversely proportional to the primary period may give excellent results and reduce the computational effort significantly. We therefore need to know how the solution schemes behave for the higher frequencies when the $\Delta t/T$ is large. Bathe and Wilson (1976) stated that the stability of the schemes for these large $\Delta t/T$'s is governed by the behaviour of the initial conditions, they must not be amplified artificially making integration of the lower frequencies worthless. Bathe and Wilson also said that errors in the displacements, velocities and accelerations due to computer round-off must not grow during the integration. Stability of a scheme is determined by examining its behaviour for the prescribed initial conditions with zero load applied.

The selection of time step size for conditionally stable schemes is dictated by the imposed stability conditions. For an un-conditionally stable scheme a time step size must be selected which is just small enough to produce accurate results but no smaller so as not to increase computational time. Therefore accuracy behaviour of the approximation methods over a range of time step sizes is required. We begin by looking at mathematical approaches to determining the stability and accuracy of these solution schemes.

Lambert (1973) stated the following theorem, "A necessary and sufficient condition for a linear multi-step method to be convergent is that it be consistent and zero-stable. Consistency controls magnitude of local truncation error, zero-stability controls manner in which error is propagated."

Equation (2.19) written again below, is rearranged forming a linear difference operator \mathfrak{L} : see (3.2).

$$\sum_{j=0}^k v_j x_{n+j} = \Delta t^2 \sum_{j=0}^k \eta_j g_{n+j} \quad (3.1)$$

$$\mathfrak{L}[x; \Delta t] = \sum_{j=0}^k \left[v_j x_{n+j} - \Delta t^2 \eta_j g_{n+j} \right] \quad (3.2)$$

The functions x_{n+j} and g_{n+j} are expandable by Taylor's series:

$$x_{n+j} = x_n + j \Delta t \dot{x}_n + \frac{(j \Delta t)^2}{2!} \ddot{x}_n + \dots \quad (3.3)$$

$$g_{n+j} = \ddot{x}_{n+j} = \ddot{x}_n + j \Delta t \ddot{\ddot{x}}_n + \dots \quad (3.4)$$

Substitution of the above truncated Taylor expansions into (3.2) yields the following equation:

$$\begin{aligned} \mathfrak{L}[x; \Delta t] = & v_0 x_n + v_1 \left[x_n + \Delta t \dot{x}_n + \frac{\Delta t^2}{2!} \ddot{x}_n \right] + v_2 \left[x_n + 2 \Delta t \dot{x}_n + \frac{(2 \Delta t)^2}{2!} \ddot{x}_n \right] \\ & + \dots + v_k \left[x_n + k \Delta t \dot{x}_n + \frac{(k \Delta t)^2}{2!} \ddot{x}_n \right] \\ & - \Delta t^2 \left\{ \eta_0 \ddot{x}_n + \eta_1 [\ddot{x}_n + \Delta t \ddot{\ddot{x}}_n] + \eta_2 [\ddot{x}_n + 2 \Delta t \ddot{\ddot{x}}_n] + \dots + \eta_k [\ddot{x}_n + k \Delta t \ddot{\ddot{x}}_n] \right\} \end{aligned} \quad (3.5)$$

When like differential terms are collected together the linear difference operator is reduced to a simpler form.

$$\mathcal{L}[x; \Delta t] = x_n C_0 + \Delta t \dot{x}_n C_1 + \Delta t^2 \ddot{x}_n C_2 + \dots + \Delta t^q \frac{d^q x_n}{dt^q} C_q \quad (3.6)$$

Written in full the C_i terms are:

$$C_0 = v_0 + v_1 + v_2 + \dots + v_k$$

$$C_1 = v_1 + 2v_2 + \dots + kv_k$$

$$C_2 = \frac{1}{2!} [v_1 + 2^2 v_2 + \dots + k^2 v_k] - [\eta_0 + \eta_1 + \eta_2 + \dots + \eta_k]$$

$$C_q = \frac{1}{q!} [v_1 + 2^q v_2 + \dots + k^q v_k] - \frac{1}{(q-2)!} [\eta_1 + 2^{q-2} \eta_2 + \dots + k^{q-2} \eta_k] \quad (3.7)$$

Lambert stated that the order of the multi-step method is deduced using the following relationship:

$$C_0 = C_1 = C_2 = \dots = C_p = C_{p+1} = 0 \quad C_{p+2} \neq 0 \quad (3.8)$$

With this rule applying, the truncation error associated with the linear operator equation is:

$$C_{p+2} \Delta t^{p+2} \frac{d^{(p+2)} x_n}{dt^{(p+2)}} \quad (3.9)$$

A linear multi-step method is said to be consistent if it has order greater than or equal to

one, hence C_0 , C_1 and C_2 must always be equal to zero for consistency to apply. The weighted residual multi-step equation, with k set to three, is consistent since it has order p equal to two. This is derived by substituting the v_j and η_j values in equation (2.20) into the C_i terms and solving. The truncation error can be reduced if the parameter values are specifically selected to produce further zero C_i terms, in this example C_4 and C_5 are:

$$C_4 = \frac{1}{12(\gamma-1)} [18\gamma - 6\beta - 11] \quad (3.10)$$

$$C_5 = \frac{1}{12(\gamma-1)} [15\gamma - 2\alpha - 12] \quad (3.11)$$

To increase the order of p the following values are assigned to the α and β parameters producing zero C_4 and C_5 terms.

$$\alpha = \frac{15\gamma}{2} - 6 \quad \beta = 3\gamma - \frac{11}{6} \quad (3.12)$$

The truncation error is derived by substituting the α and β values into the C_6 term and solving.

$$C_6 = \frac{-1}{240(\gamma-1)} \quad (3.13)$$

The fact that C_0 and C_1 are zero implies that there is a double root equal to one, this is known as the double principal root. Examining equation (3.2) the difference operator can be split into two characteristic polynomials, one in terms of v_j and the other in terms of the η_j factor.

$$\phi(\zeta) = \sum_{j=0}^k v_j \zeta^j \quad \sigma(\zeta) = \sum_{j=0}^k \eta_j \zeta^j \quad (3.14)$$

For zero-stability the first characteristic polynomial $\phi(\zeta)$ should not have a root of modulus greater than one. If the polynomial has a root of modulus one, it lies on the circumference of the unit circle, then its multiplicity should be no greater than two, this is the Dahlquist Stability rule, Lambert (1973). Further roots are spurious and they should lie within the unit circle to satisfy the zero-stability criterion. In the weighted residual example $\phi(1) = \phi'(1) = 0$, implying there is a double root at one. The third root is $(\gamma - 2)/(\gamma - 1)$ which is less than one if γ satisfies the conditions $3/2 \leq \gamma < \infty$, therefore the scheme satisfies the zero-stability and consistency conditions.

If the amplification matrix is stable, that is the algorithm does not keep growing without bounds, for all parameter values and time increment size, then the algorithm is unconditionally stable. When the scheme requires limits on these variables the algorithm is conditionally stable. This stability behaviour can be examined further using the Routh-Herwitz criterion, see Gantmacher (1960). This method of evaluation uses the stability polynomial (3.15) transformed from a real to an imaginary axis using the transformation function $r = (1+z)/(1-z)$. This transformation simply maps the unit circle on to the real negative z plane, see Figure 3.3.

$$\sum_{j=0}^k v_j r^j + (\omega \Delta t)^2 \sum_{j=0}^k \eta_j r^j = 0 \quad (3.15)$$

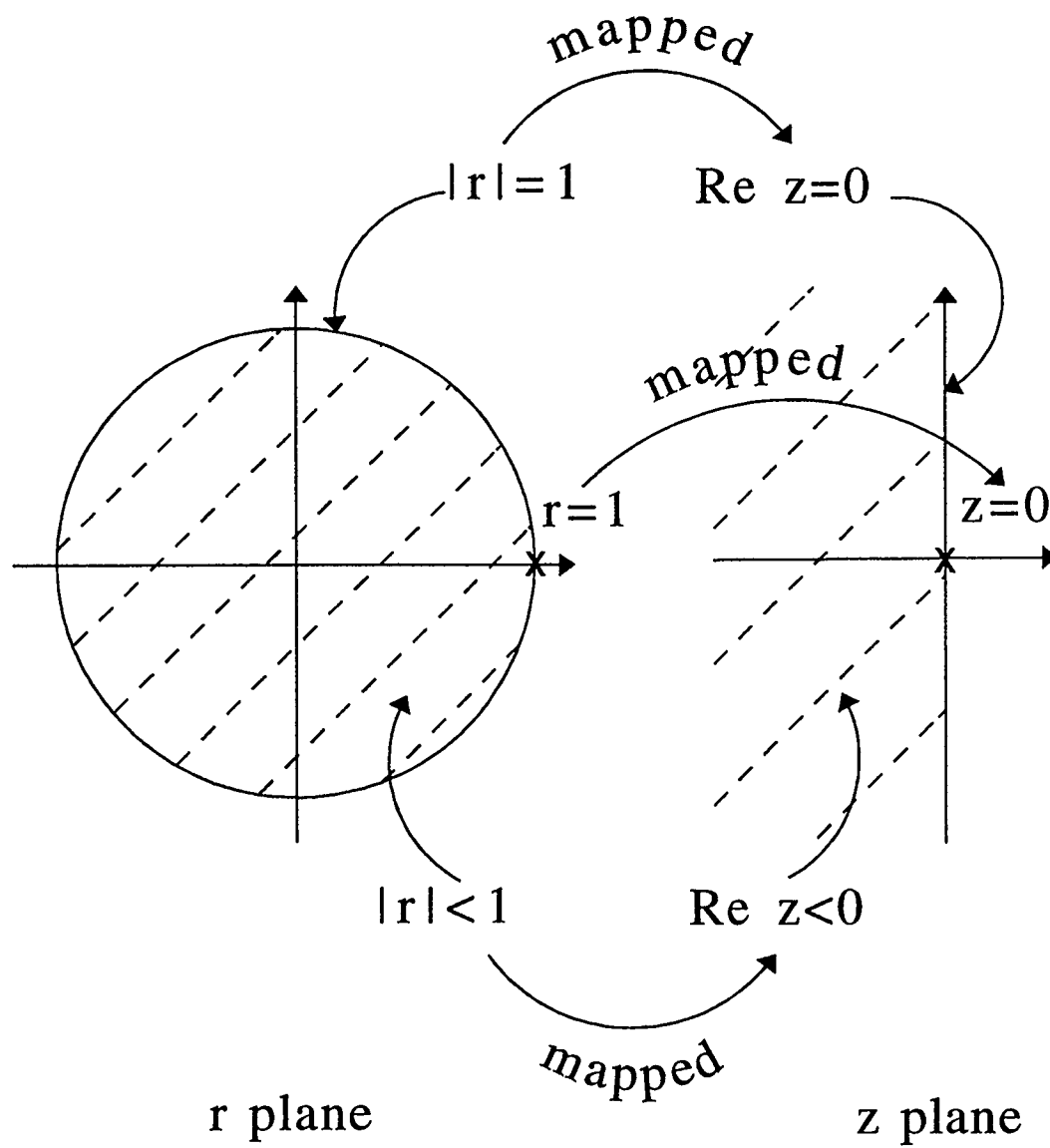


FIGURE 3.3 MAPPING UNIT CIRCLE ON TO THE IMAGINARY AXIS $\text{Re } z$ EQUAL TO ZERO

After the transformation the stability polynomial becomes the following if k is equal to three.

$$\begin{aligned} & \{v_3 + (\omega\Delta t)^2\eta_3\}(1+z)^3 + \{v_2 + (\omega\Delta t)^2\eta_2\}(1+z)^2(1-z) \\ & + \{v_1 + (\omega\Delta t)^2\eta_1\}(1+z)(1-z)^2 + \{v_0 + (\omega\Delta t)^2\eta_0\}(1-z)^3 = 0 \end{aligned} \quad (3.16)$$

This equation is simplified by collecting like z factors.

$$b_0 z^3 + b_1 z^2 + b_2 z + b_3 = 0 \quad (3.17)$$

The b_i coefficients in equation (3.17) are as follows:

$$\begin{aligned} b_0 &= (v_3 - v_2 + v_1 - v_0) + (\omega\Delta t)^2(\eta_3 - \eta_2 + \eta_1 - \eta_0) \\ b_1 &= (3v_3 - v_2 - v_1 + 3v_0) + (\omega\Delta t)^2(3\eta_3 - \eta_2 - \eta_1 + 3\eta_0) \\ b_2 &= (3v_3 + v_2 - v_1 - 3v_0) + (\omega\Delta t)^2(3\eta_3 + \eta_2 - \eta_1 - 3\eta_0) \\ b_3 &= (v_3 + v_2 + v_1 + v_0) + (\omega\Delta t)^2(\eta_3 + \eta_2 + \eta_1 + \eta_0) \end{aligned} \quad (3.18)$$

The Routh-Herwitz criterion states that the following equalities must be satisfied for a stable algorithm:

$$\begin{aligned} \text{SS22: } & b_0 > 0, \quad b_1 \geq 0, \quad b_2 \geq 0 \\ \text{SS32: } & b_0 > 0, \quad b_1 \geq 0, \quad b_2 \geq 0, \quad b_3 \geq 0, \quad b_1 b_2 - b_3 b_0 \geq 0 \end{aligned} \quad (3.19)$$

When the Routh-Herwitz criterion is used to determine the stability of the weighted residual general equation the b_i values obtained are:

$$\begin{aligned}
 b_0 &= \frac{1}{\gamma-1} \{8\gamma-12+q^2(24-16\gamma)\} \\
 b_1 &= \frac{4-8q^2}{1-\gamma} \\
 b_2 &= \frac{q^2(24\gamma-36)}{\gamma-1} \\
 b_3 &= \frac{12q^2}{\gamma-1}
 \end{aligned} \tag{3.20}$$

Note the notation $q^2=(\omega \Delta t)^2 / 12$. From the b_2 value it is deduced that $\gamma \geq 3/2$ and from b_1 $\omega\Delta t \leq \sqrt{6}$ in order to satisfy the equalities.

Returning to the SSpj and GNpj schemes, a method is required to analyze the stability and accuracy of a single-step solution scheme with no damping or force term. A cubic solution of the characteristic polynomial is calculated by solving $\det(A - \lambda I)$ where A is the amplification matrix, I the identity matrix and λ_j are the eigenvalues.

$$\det(A - \lambda I) = \lambda^3 - 2A_1\lambda^2 + A_2\lambda - A_3 = \sum_{j=0}^p q_j \lambda^{3-j} = 0 \tag{3.21}$$

The A_1, A_2 and A_3 variables are invariants of the amplification matrix and they equal the following:

$$\begin{aligned}
 A_1 &= \frac{1}{2} \text{trace } A = \frac{1}{2} \sum_{i=1}^3 A_{ii} \\
 A_2 &= \sum (\text{principal minors}) \\
 &= A_{22} A_{33} - A_{32} A_{23} + A_{11} A_{33} - A_{31} A_{13} + A_{11} A_{22} - A_{21} A_{12} \\
 A_3 &= \det A
 \end{aligned} \tag{3.22}$$

The single-step matrix equation is written as:

$$X_{n+1} = AX_n, \quad X_{n+2} = A^2 X_n, \quad \dots, \quad X_{n+p} = A^p X_n \tag{3.23}$$

In this matrix equation A is the amplification matrix and X_n is:

$$X_n = \begin{Bmatrix} x_n \\ \Delta t \dot{x}_n \\ \Delta t^2 \ddot{x}_n \end{Bmatrix} \tag{3.24}$$

Taking a linear combination of equations (3.23) the following is written:

$$q_0 X_n + q_1 X_{n+1} + \dots + q_p X_{n+p} = \left(q_0 I + q_1 A + \dots + q_p A^p \right) X_n \tag{3.25}$$

If coefficients q_0, q_1, \dots, q_p equal the coefficients in the characteristic polynomial then equation (3.25) equals zero by the Cayley-Hamilton theorem, Froberg (1969).

$$\sum_{j=0}^p q_j X_{n+j} = x_n - 2A_1 x_{n+1} + A_2 x_{n+2} - A_3 x_{n+3} = 0 \tag{3.26}$$

It can be seen that the characteristic polynomial (3.21) is similar to the multi-step formulation (3.26), therefore the invariants of A for either the GNpj or SSpj schemes can be used to determine the multi-step form of the single-step equations. Wood (1990) followed this procedure and hence defined the SS22 and SS32 coefficients which are listed in Table 3.1. Note that the SS32 v_3 variable has been normalized as it was in the weighted residual formulation in section 2.1.2. The stability and accuracy conditions of the SSpj and GNpj algorithms are identical so for the rest of this chapter the SSpj scheme only will be studied.

The truncation error C_{p+2} of the single-step SSpj schemes can now be calculated by substituting in the v_j and η_j values in Table 3.1 into equations (3.7). All the SSpj schemes are consistent, that is they have order greater than or equal to one, this is due to the algorithm construction, see Wood (1990). Table 3.2 lists the coefficients required to calculate the truncation errors.

Stability of the SSpj schemes can also be calculated by substituting the v_j and η_j parameters into equations (3.18) and then satisfying the Routh-Herwitz equalities. Note that $(\omega\Delta t)^2$ is always positive so that many of the solution schemes are stable for all time increment sizes. Table 3.2 lists the equalities that must be satisfied for stability.

If zero amplitude decay is required then the Routh-Herwitz equalities below must apply:

$$\begin{aligned} \text{SS22: } b_1 &= 0 \\ \text{SS32: } b_1 b_2 - b_3 b_0 &= 0 \end{aligned} \tag{3.27}$$

Solution Scheme	j	Coefficients For Multi-Step Equation	
		v_j	η_j
SS22	0	1	$(\theta_2 - 2\theta_1 + 1)/2$
	1	-2	$(2\theta_1 - 2\theta_2 + 1)/2$
	2	1	$\theta_2/2$
SS32	0	$(1 - \theta_1)/\theta_1$	$(3\theta_2 - 3\theta_1 - \theta_3 + 1)/6\theta_1$
	1	$(3\theta_1 - 2)/\theta_1$	$(3\theta_3 - 6\theta_2 + 4)/6\theta_1$
	2	$(1 - 3\theta_1)/\theta_1$	$(3\theta_1 + 3\theta_2 - 3\theta_3 + 1)/6\theta_1$
	3	1	$\theta_3/6\theta_1$

TABLE 3.1 COEFFICIENTS FOR THE MULTI-STEP EQUIVALENT OF THE SINGLE-STEP EQUATIONS

		SS22	SS32
Truncation Coefficients	C_3	$1/2 - \theta_1$	0
	C_4	$(2-3\theta_2-3\theta_1)/6$	$(6\theta_1-6\theta_2-1)/12\theta_1$
	C_5	$(1-3\theta_2-\theta_1)/6$	$(7\theta_1-6\theta_2-2\theta_3-1)/12\theta_1$
	C_6	$(47-210\theta_2-30\theta_1)/720$	$(150\theta_1-105\theta_2-90\theta_3-19)/360\theta_1$
Routh-Herwitz Stability Criterion		$(\omega\Delta t)^2 > 2/(\theta_1 - \theta_2)$ $\theta_1 \geq 1/2$	$(\omega\Delta t)^2 > 12(1-2\theta_1)/(4\theta_3-6\theta_2+1)$ $(\omega\Delta t)^2 \geq -12/(6\theta_2-6\theta_1-1)$ $\theta_1 \geq 1/2$

TABLE 3.2 COEFFICIENTS TO CALCULATE TRUNCATION ERROR AND STABILITY FOR THE SINGLE-STEP EQUATIONS

This stability behaviour can easily be examined by plotting the spectral radius of the amplification matrix against $\Delta t/T$. The smaller the spectral radius the more rapid the convergence of the algorithm, this phenomenon can be taken advantage of when damping out higher modes, which is often desirable for multi-degree of freedom systems. The spectral radius is equal to the modulus of the largest eigenvalue of the amplification matrix:

$$\rho(r) = \max |\lambda_i| \leq 1 \quad (3.28)$$

For stability the spectral radius must be less than or equal to one.

We now look at an analytical development which determines the accuracy of the approximation methods. Numerical dissipation of the systems kinematics is measured by the magnitude of amplitude decay occurring; and the numerical dispersion, due to the approximation, is measured by the amount of period elongation or phase lag.

Amplitude decay is determined by taking two consecutive peaks, or troughs, in a system with no physical damping applied, and calculating the percentage decrement in the amplitude over time, see equation (3.29). Period elongation on the other hand is established by finding the percentage difference between the true dynamic period of the system, T , and the algorithmic period output by the solution scheme, \bar{T} , see equation (3.30).

$$\% \text{ Amplitude Decay} = 100 - \left[\frac{100}{|1stPeak|} \times |2ndPeak| \right] \quad (3.29)$$

$$\% \text{ Period Elongation} = \frac{100 \bar{T}}{T} - 100 \quad (3.30)$$

3.2 PROGRAMMING THEORY

Before the SSpj algorithm can be used it must be arranged into a matrix equation format which is programmable, the equation has an amplification matrix A and a force vector L. Dynamic.c was coded for a single degree of freedom system hence the notation for the mass, stiffness and damping terms is lower case.

$$\begin{Bmatrix} x_{t+\Delta t} \\ \Delta t \dot{x}_{t+\Delta t} \\ \Delta t^2 \ddot{x}_{t+\Delta t} \end{Bmatrix} = A \begin{Bmatrix} x_t \\ \Delta t \dot{x}_t \\ \Delta t^2 \ddot{x}_t \end{Bmatrix} + L f_{t+\Delta t} \quad (3.31)$$

The amplification matrix and force vector used in the SS32 formulation are generated by substituting the $\alpha_n^{(3)}$ equation (2.38) into the three kinematic equations (2.41)-(2.43).

$$A = \begin{bmatrix} 1 - \frac{k\Delta t^3}{6\kappa} & 1 - \frac{k\Delta t^3\theta_1}{6\kappa} & \frac{1}{2} - \frac{m\Delta t}{6\kappa} - \frac{k\Delta t^3\theta_2}{12\kappa} \\ -\frac{k\Delta t^3}{2\kappa} & 1 - \frac{k\Delta t^3\theta_1}{2\kappa} & 1 - \frac{m\Delta t}{2\kappa} - \frac{k\Delta t^3\theta_2}{4\kappa} \\ -\frac{k\Delta t^3}{\kappa} & -\frac{k\Delta t^3\theta_1}{\kappa} & 1 - \frac{m\Delta t}{\kappa} - \frac{k\Delta t^3\theta_2}{2\kappa} \end{bmatrix} \quad (3.32)$$

$$L = \frac{\Delta t^3}{\kappa} \begin{Bmatrix} \frac{1}{6} \\ \frac{1}{2} \\ 1 \end{Bmatrix} \quad (3.33)$$

$$\kappa = m\Delta t\theta_1 + \frac{k\Delta t^3\theta_3}{6}$$

The SS22 matrices are calculated in a similar fashion:

$$A = \begin{bmatrix} 1 - \frac{k\Delta t^2}{2\kappa} & 1 - \frac{k\Delta t^2\theta_1}{2\kappa} \\ -\frac{k\Delta t^2}{\kappa} & 1 - \frac{k\Delta t^2\theta_1}{\kappa} \end{bmatrix} \quad (3.34)$$

$$L = \frac{\Delta t^2}{\kappa} \begin{Bmatrix} \frac{1}{2} \\ 1 \end{Bmatrix} \quad (3.35)$$

$$\kappa = m + \frac{k\Delta t^2\theta_2}{2}$$

The above matrix calculations are performed in subroutines SS32 and SS22. The matrices are then used to solve (3.31) in routine Equation, the force term is calculated using equation (2.44). Subroutine Equation also calculates the frequency, period and time step size of the dynamic system.

$$\omega = \sqrt{\frac{k}{m}} \quad (3.36)$$

$$T = \frac{2\pi}{\omega} \quad (3.37)$$

$$\Delta t = \frac{T}{\text{Number of divisions specified}} \quad (3.38)$$

Subroutine Elong_decay determines the two points which span the displacement plots so called zero line of oscillation, and also the two points spanning the maximum displacement from the zero line. The first cycle of the plot is ignored as the initial conditions can deform its displacement pattern producing erroneous period and amplitude results. The points are used to calculate the exact crossing point and maximum displacement of the plot by applying an interpolation function in subroutine Interp. The interpolation function employed is the Divided Difference, see Gerald and Wheatley (1984). A nth degree polynomial can be used to find any point x_i along itself if several values of x are defined.

$$P_n(x_i) = f_i = d_0 + (x - x_0)d_1 + \dots + (x - x_0)(x - x_1)\dots(x - x_{n-1})d_n \quad (3.39)$$

The d_i values are formed using divided difference theory, below are examples of the zero, first and second order divided difference equations:

$$\begin{aligned}
 f[x_0] &= f_0 = d_0 \\
 f[x_0, x_1] &= \frac{f_1 - f_0}{x_1 - x_0} = d_1 \\
 f[x_0, x_1, x_2] &= \frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} = d_2
 \end{aligned} \tag{3.40}$$

Equations (3.40) are substituted into the polynomial (3.39), for a cubic solution n is set equal to three. In Interp points zero and one are known, point two which is half way between zero and one in time is to be calculated, therefore x_i becomes x_2 and the following equation is solved:

$$P_3(x_2) = f_0 + \left[\frac{f_1 - f_0}{x_1 - x_0} \right] (x_2 - x_0) + \left[\frac{f[x_1, x_2] - f[x_0, x_1]}{x_2 - x_0} \right] (x_2 - x_0)(x_2 - x_1) = f_2 \tag{3.41}$$

Once the crossing points are known the oscillations periods and amplitudes can be calculated. The period elongation is determined by comparing three consecutive periods with the dynamic equation period T which is calculated in (3.37), see equation (3.30). Two consecutive maximum amplitudes are used to calculate the amplitude decay for the algorithm, see equation (3.29).

Program `eigen.c` calls an `imsl` C maths routine which calculates the eigenvalues for the cubic SS32 amplification matrix, and `dynamic.c` calculates the eigenvalues for the quadratic SS22 schemes. The modulus of each eigenvalue is determined and the largest of these corresponds to the spectral radius.

3.3 EXAMPLES

Using the stability and truncation error equations in Table 3.2, Table 3.3 was tabulated for the examples that are to be discussed. For all the cases where the spectral radius, amplitude decay and period elongation are to be determined, the mass and stiffness equal one, therefore frequency ω equals 1 and period T equals 2π . The initial conditions were taken as $x(0)=3$, $x'(0)=0$ and $f(0)=0$.

The displacement plots illustrated in this section were taken for a time increment $T/12$, the initial conditions equalled those published in the paper by Katona and Zienkiewicz (1985), they were $x(0)=x'(0)=0.7071$ and $f(0)=0$. These conditions give an exact solution:

$$x(t) = \sin \left(t + \frac{\pi}{4} \right) \quad (3.42)$$

The trapezoidal scheme produces a spectral radius equal to one for all $\Delta t/T$, also by the Routh-Herwitz criterion equation (3.27), b_1 equals zero which means there is no natural damping within the solution scheme. As a check for accuracy of amplitude decay calculated by the computer the special case of the Newmark method was run and amplitude decay plotted. If no errors occur amplitude decay should be zero, looking at plot in Figure 3.4 it is apparent that errors are present. Up to $\Delta t/T$ equal to 0.1 there are virtually no errors but as the number of divisions within the period decreases the errors begin to rapidly increase. At $\Delta t/T$ equal to 0.25, corresponding to a period divided into four time steps the error is approximately six percent and at three divisions per period the errors incurred are a high nine percent. This means that the results obtained for the following examples give a reasonably accurate picture up to $\Delta t/T$ equal to 0.25 but give erroneous results for larger $\Delta t/T$.

Solution Scheme	θ_1	θ_2	θ_3	Stability Limit $(\omega\Delta t)^2_{\max}$	Truncation Error C_{p+2}
Fox-Goodwin	0.5	0.1667		6.0	$C_6=-0.0042$
Trapezium	0.5	0.5		∞	$C_4=-0.1667$
Central Diff.	0.5	0.0		4.0	$C_4=+0.0833$
Wilson Theta	1.37	1.8769	2.5714	∞	$C_4=-0.2458$
Houbolt	2.0	3.6667	6.0	∞	$C_4=-0.4583$
SS32 Case 1	0.5	0.6667	0.75	∞	$C_4=-0.3333$
SS32 Case 2	0.5	0.3333	0.25	6.0	$C_6=-0.0083$
Wilson Theta	1.4	1.96	2.744	∞	$C_4=-0.2595$
Wilson Theta	2.0	4.0	8.0	∞	$C_4=-0.5417$
Newmark	0.55	0.6		∞	$C_3=-0.0500$
Bossak Case 1	1.1	1.3717	1.815	∞	$C_4=-0.1992$
Bossak Case 2	1.1	1.7667	3.0	∞	$C_4=-0.3788$
Hilber, Hughes & Taylor (HHT)	1.0	1.2517	1.6335	∞	$C_4=-0.2092$

TABLE 3.3 SAMPLE SINGLE-STEP SCHEMES AND THEIR CORRESPONDING TRUNCATION ERRORS AND STABILITY REQUIREMENTS

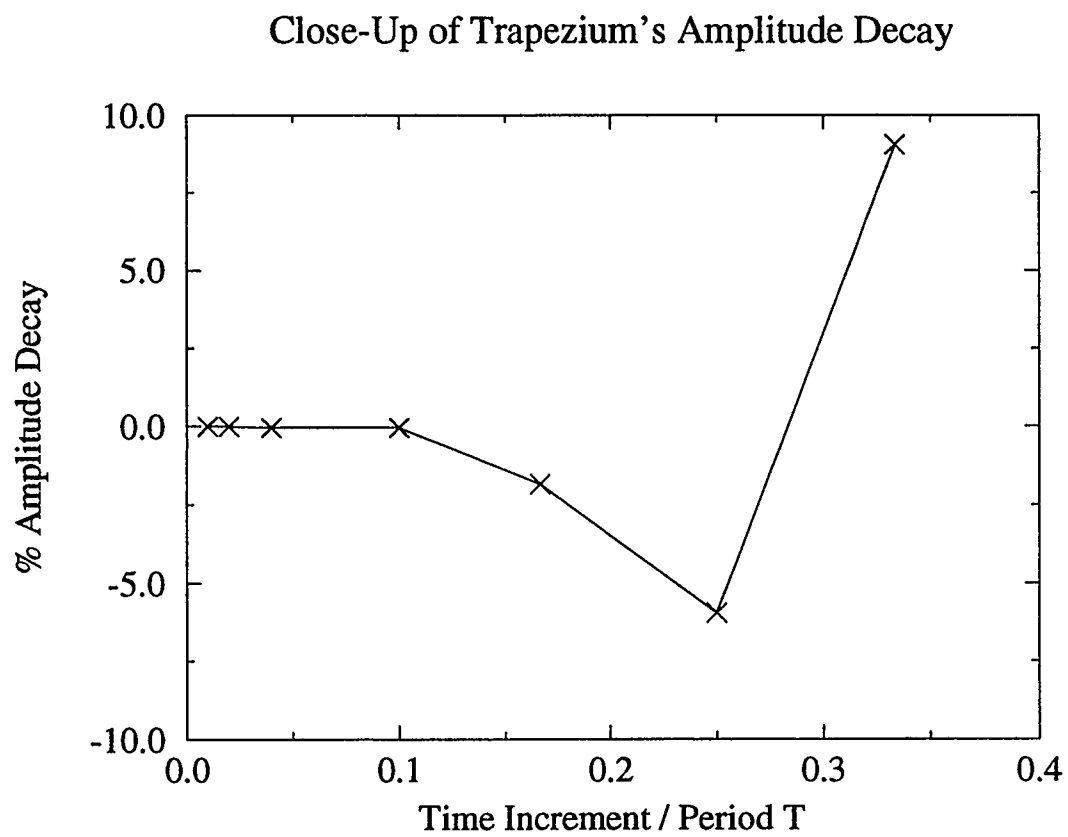
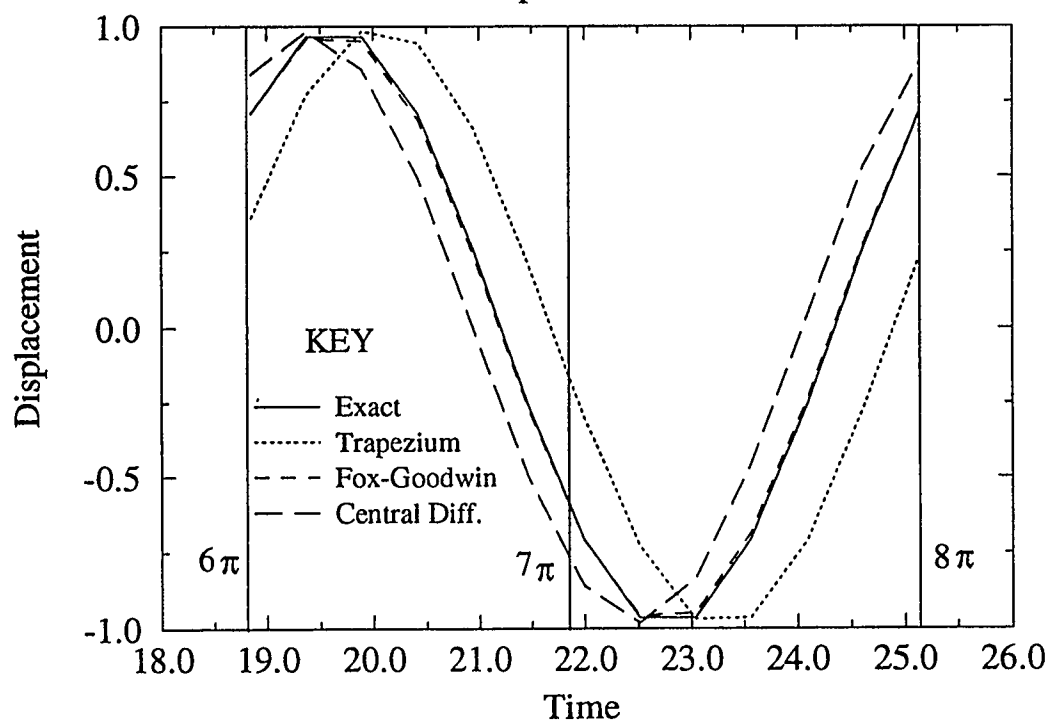


FIGURE 3.4 MEASURING PROGRAM ACCURACY USING THE TRAPEZIUM SOLUTION

The first tests run were used to check the program, results were cross checked with those produced in Katona and Zienkiewicz's paper (1985). The tests also act as a comparison for various SS22 and SS32 schemes. Figure 3.5 shows a comparison of three quadratic algorithms, these are the trapezium, Fox-Goodwin and central difference schemes. The Fox-Goodwin scheme has the smallest truncation error and therefore should be the most accurate; however stability limits exist and therefore affect the range of time step size over which this scheme gives reasonable results. The trapezium method is unconditionally stable but should not give as accurate results, it also has zero amplitude decay which may be undesirable if damping of unwanted modes of frequency is required. The central difference scheme is an explicit method. Explicit methods calculate new displacements by substituting central difference expressions for velocity and acceleration at time t into the dynamic equation, implicit schemes use kinematic equations at time $t+\Delta t$. The draw back with the central difference scheme is that it has limits on time increment size. Figure 3.5 a. shows the displacement plot for the fourth period, it is easy to see that the Fox-Goodwin scheme has no apparent period elongation and a slight amplitude decay, therefore proving that it is the most accurate scheme, if the time increment is below its critical stability value. The trapezium and central difference schemes both have no amplitude decay but possess period errors. Figure 3.5 b. shows the range of $\Delta t/T$ over which the Fox-Goodwin and central difference methods are stable, it also shows the trapezium schemes spectral radius equal to one for the entire range. Figure 3.6 a. and b. present a more accurate picture of the amplitude decay and period elongation seen in the displacement plot which shows $\Delta t/T$ at a set value of 0.0833. Both plots show that the trends of accuracy, as the time increment size increases, is variable with each scheme. It is interesting to note that the central difference scheme has period reduction while all the other schemes studied have period elongation.

a. Displacement Plot



b. Spectral Radius

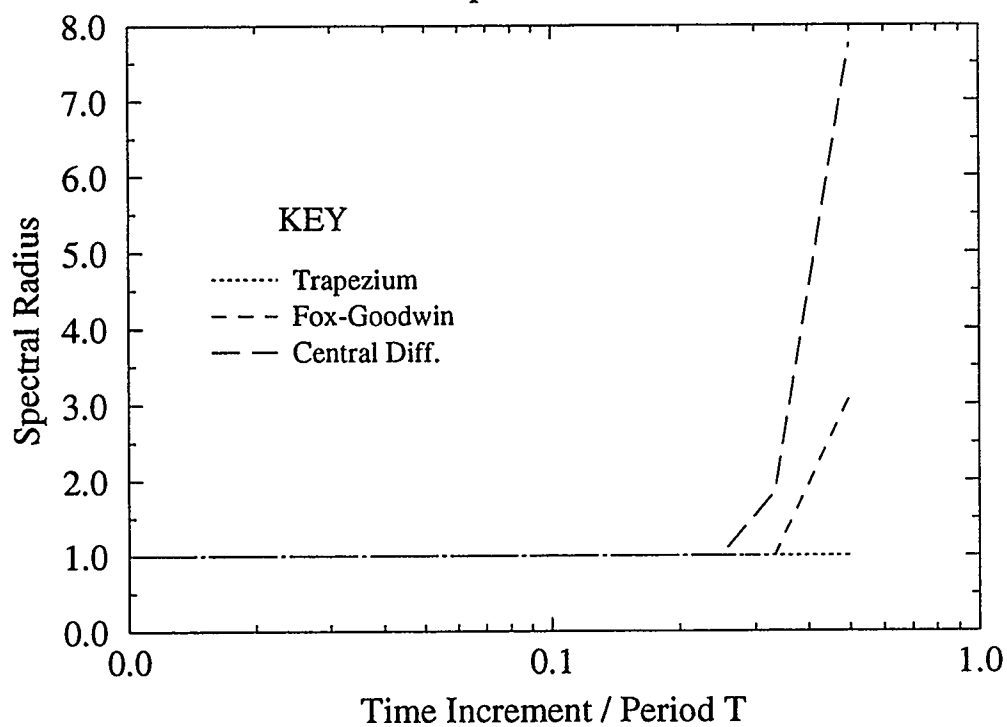


FIGURE 3.5 SS22 SOLUTION SCHEMES

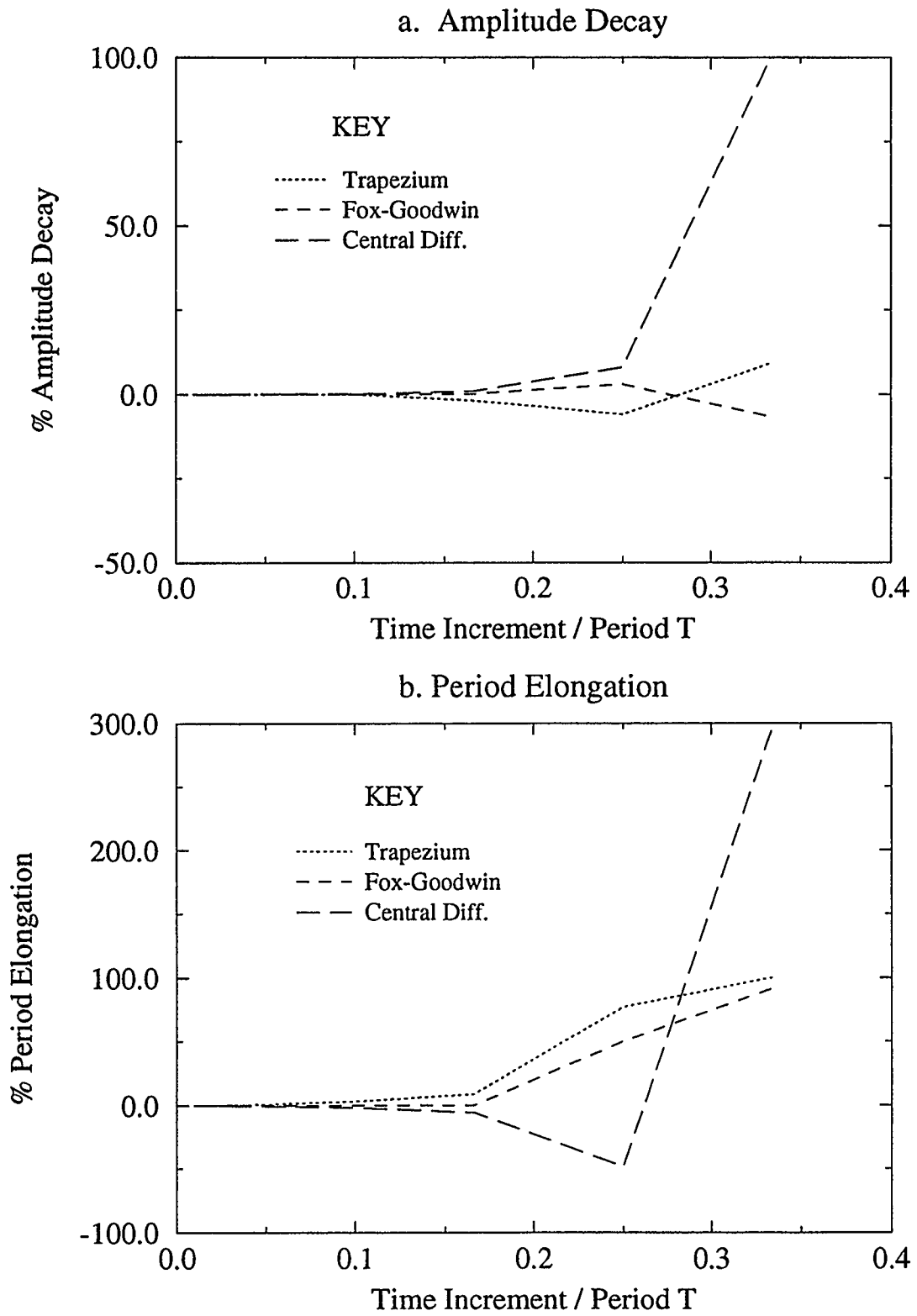


FIGURE 3.6 SS22 SOLUTION SCHEMES

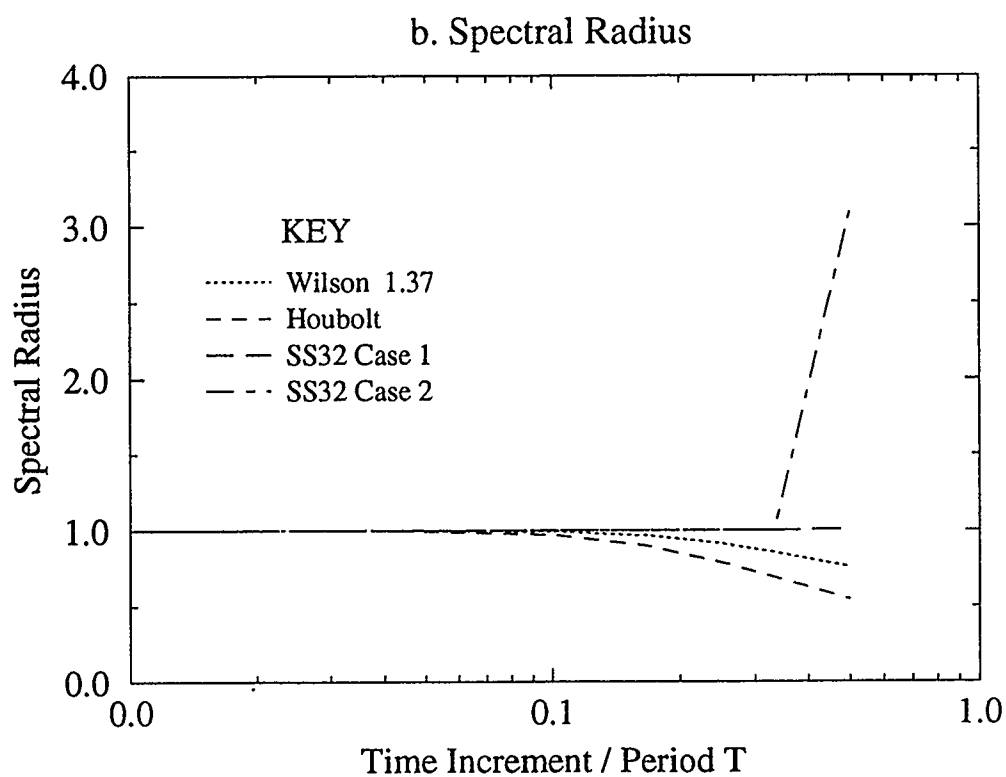
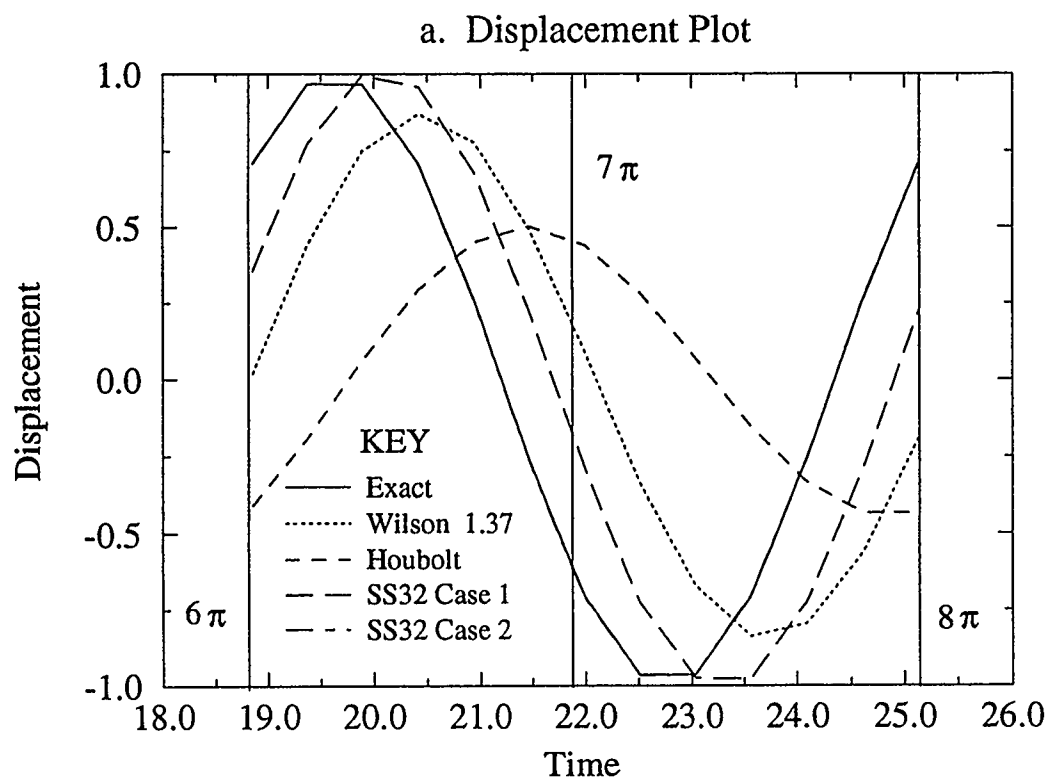


FIGURE 3.7 SS32 SOLUTION SCHEMES

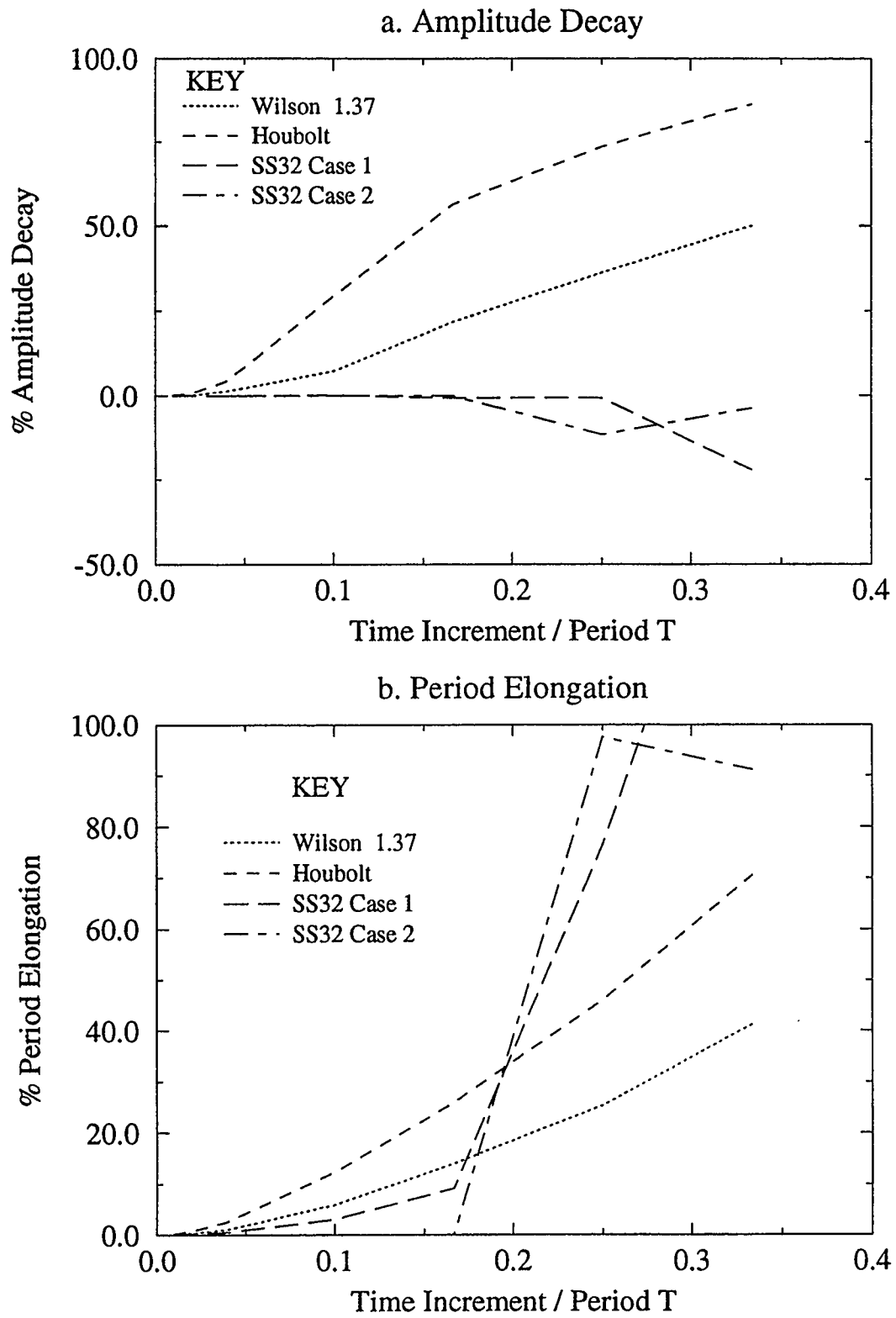


FIGURE 3.8 SS32 SOLUTION SCHEMES

The SS32 algorithms are compared in Figures 3.7 and 3.8. Four schemes were compared, the Wilson θ ($\theta=1.37$), Houbolt and two SS32 schemes, Case one with optimal stability and Case two with optimal accuracy. The Houbolt scheme shows clearly in Figure 3.7 a. that it has the most severe amplitude decay and period elongation, this is reflected in Figure 3.8 a. and b. for the corresponding value of $\Delta t/T$. An important point to make is that the Houbolt period elongation does not grow as rapidly as the SS32 cases which produce very inaccurate results at large time increments. The spectral radius, Figure 3.7 b., shows that the optimal accuracy scheme is unstable when $\Delta t/T$ exceeds a value around 0.3. This corresponds to the stability requirement in Table 3.3 where the maximum Δt permitted is approximately $T/(2.55)$ which results in a $\Delta t/T$ value of 0.39. Case one has a spectral radius of one and the condition in equation (3.27) is satisfied therefore this scheme has zero amplitude decay which may or may not be desirable.

Figures 3.9 and 3.10 examine the stability and accuracy of two Wilson θ and two Newmark schemes, the intention of this comparison is to see the effect that parameter values and the order of the algorithm can have on the solution. The Wilson θ schemes, which are cubic, were given values of θ of 2.0 and 1.4. The quadratic Newmark schemes selected were the special case with δ equal to 0.5 and α equal to 0.25 which will be referred to as the Trapezium method, and the case where δ equals 0.55 and α equals 0.3 which will be referred to as the Newmark case. For conversion to the SS22 and SS32 algorithms see Table 2.2. All these schemes have unconditional stability and all but the Newmark case have truncation error of order four. Examining the plots the Wilson θ methods vary dramatically in behaviour, the 2.0 case has the most severe damping of all the schemes and its spectral radius confirms this result. This scheme also has the worst period elongation especially for the smaller time increments. The Newmark scheme produces the best results if a small but non-zero amplitude decay is required.

The final set of plots in Figures 3.11 and 3.12 serve as a comparison of results produced by Wood *et. al.* (1980) and Hilber *et. al.* (1977). Two Bossak examples were selected and one Hilber, Hughes and Taylor example which is referred to as HHT in the plots. For

the conversion see Table 2.2.

$$\text{Bossak Case 1 : } \alpha_B = -0.1 \quad \beta_B = 0.3025 \quad \gamma_B = 0.6$$

$$\text{Bossak Case 2 : } \alpha_B = -0.1 \quad \beta_B = 0.5 \quad \gamma_B = 0.6$$

$$\text{HHT Case : } \alpha_H = -0.1 \quad \beta_H = 0.3025 \quad \gamma_H = 0.6$$

Before continuing with these examples a brief background of the algorithms is helpful. Both schemes introduced an extra parameter to control artificial damping and here they are determined for the case with no damping. The Bossak-Newmark scheme (Wood *et. al.* (1980)), uses the same method that Chan *et. al.* (1969) did in the truncated Taylor series collocation method to solve the dynamic equation:

$$M\ddot{x}_{n+1}(1 - \alpha_B) + M\ddot{x}_n\alpha_B + Kx_{n+1} = f_{n+1} \quad (3.43)$$

Hilber, Hughes and Taylor (Hilber *et. al.* (1977)), also adapted the general dynamic equation by using the following dynamic equation:

$$M\ddot{x}_{n+1} + Kx_{n+1}(1 + \alpha_H) - Kx_n\alpha_H = f_{n+1} \quad (3.44)$$

The three cases are unconditionally stable and all have second order accuracy, that is p equals two. Bossak Case one and the HHT example have very similar period elongations, their amplitude decays are similar but begin to diverge as the $\Delta t/T$ increases. Bossak Case two has poor amplitude decay and period elongation for small $\Delta t/T$ but between 0.15 and 0.25 the scheme becomes more accurate than the other two cases. It should be noted that all these schemes have much less amplitude decay than they have period elongation, see the displacement plot in Figure 3.11 a.

Of all the schemes studied the Newmark probably gives the best results for period elongation over an unbounded time increment size range, however if zero damping is required the Trapezium is the best solution. The SS32 Case two and the Fox-Goodwin give very good results but they become unstable and are therefore very limiting.

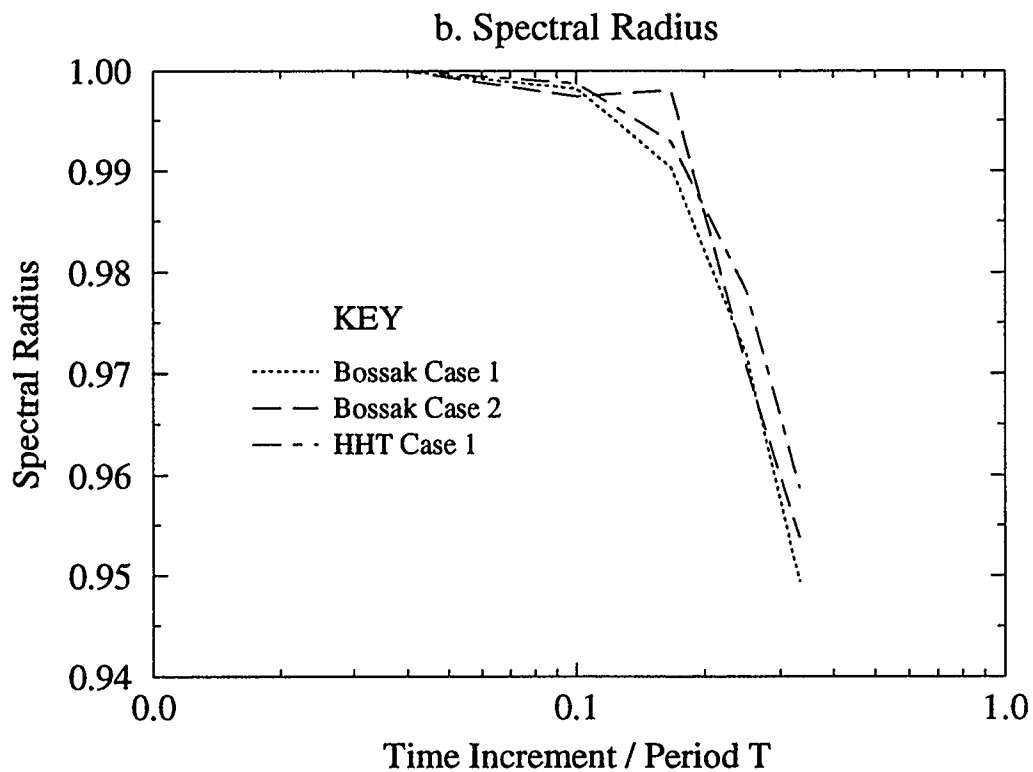
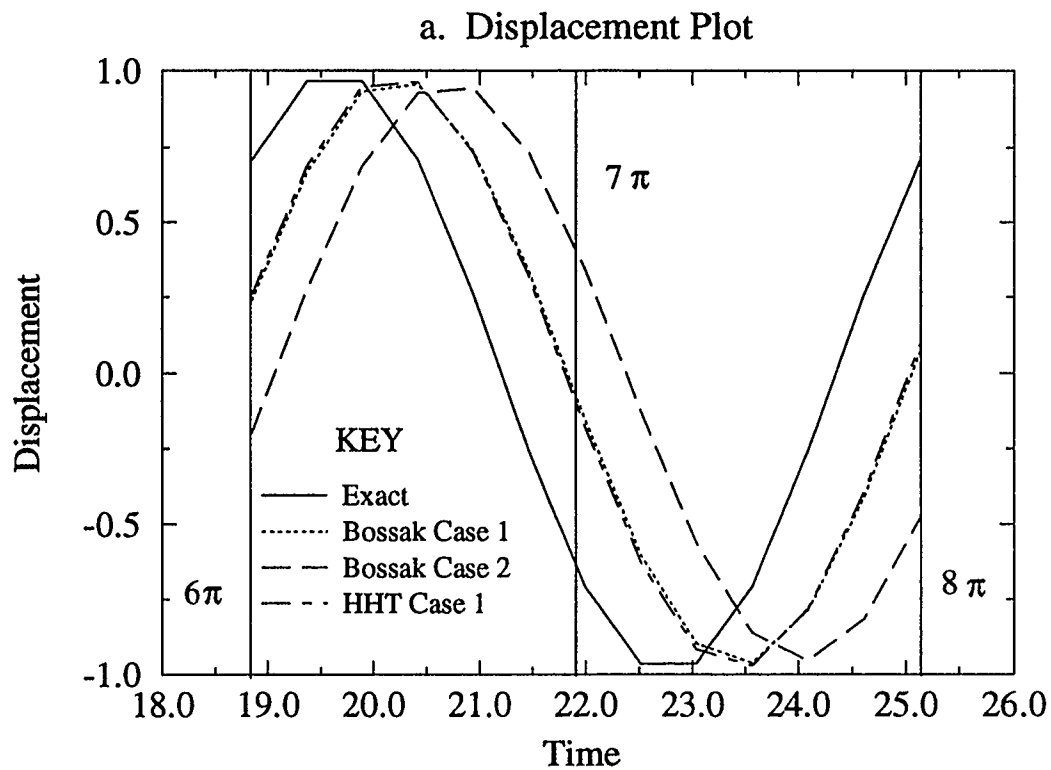


FIGURE 3.9 COMPARISON OF SCHEMES WITH DIFFERING PARAMETER VALUES

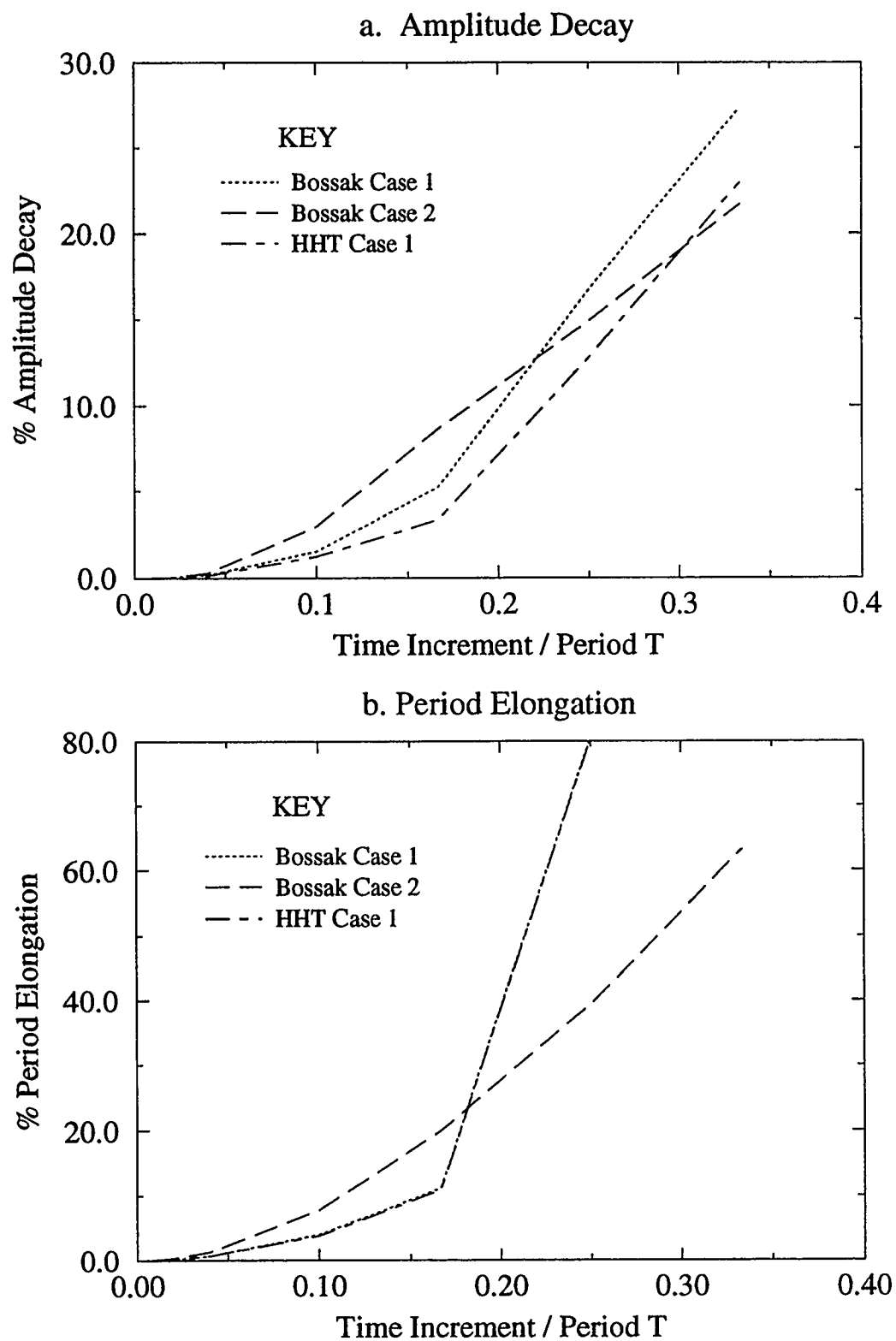


FIGURE 3.10

COMPARISON OF SCHEMES WITH DIFFERING PARAMETER
VALUES

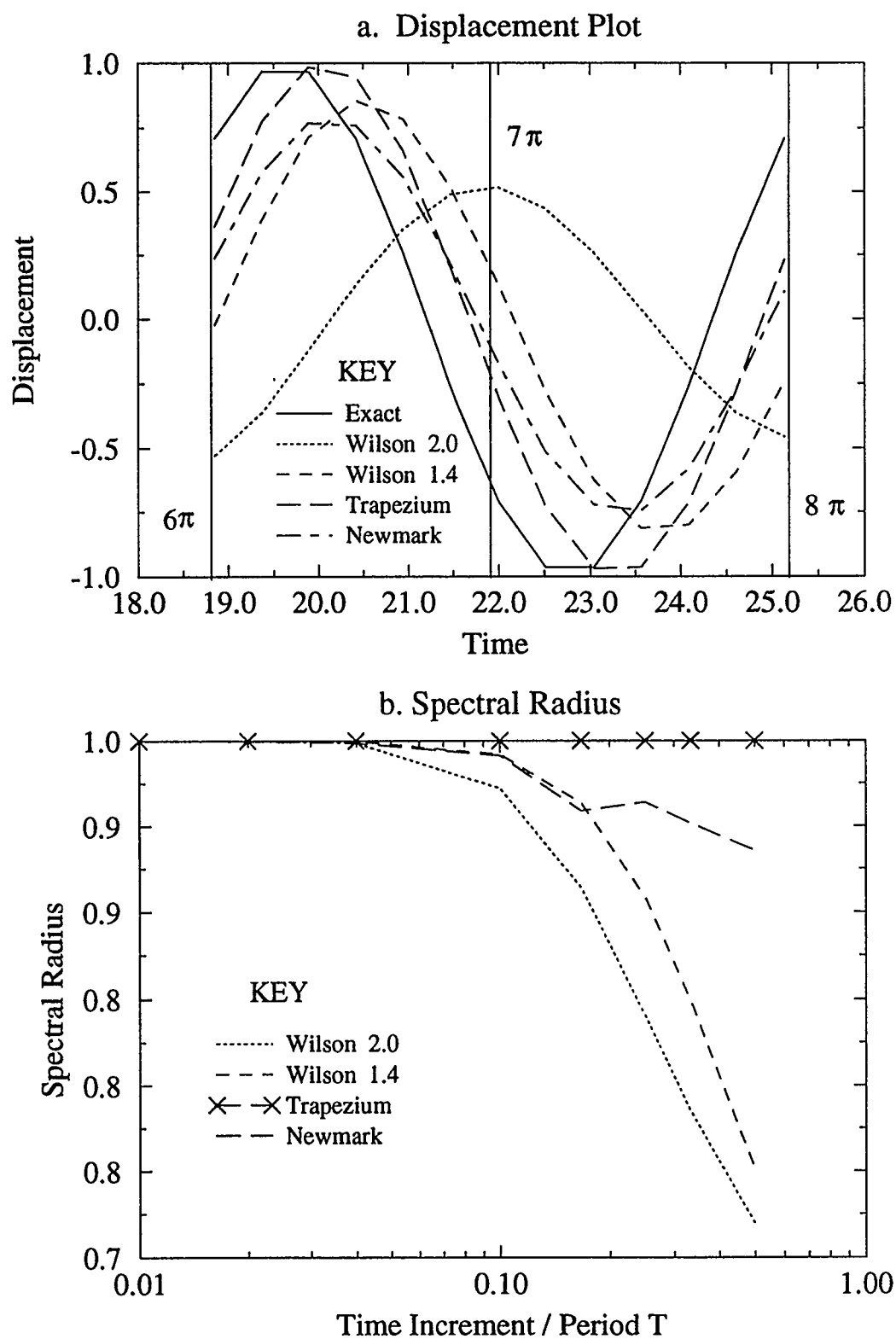


FIGURE 3.11 THE BOSSAK AND HILBER, HUGHES AND TAYLOR SCHEMES

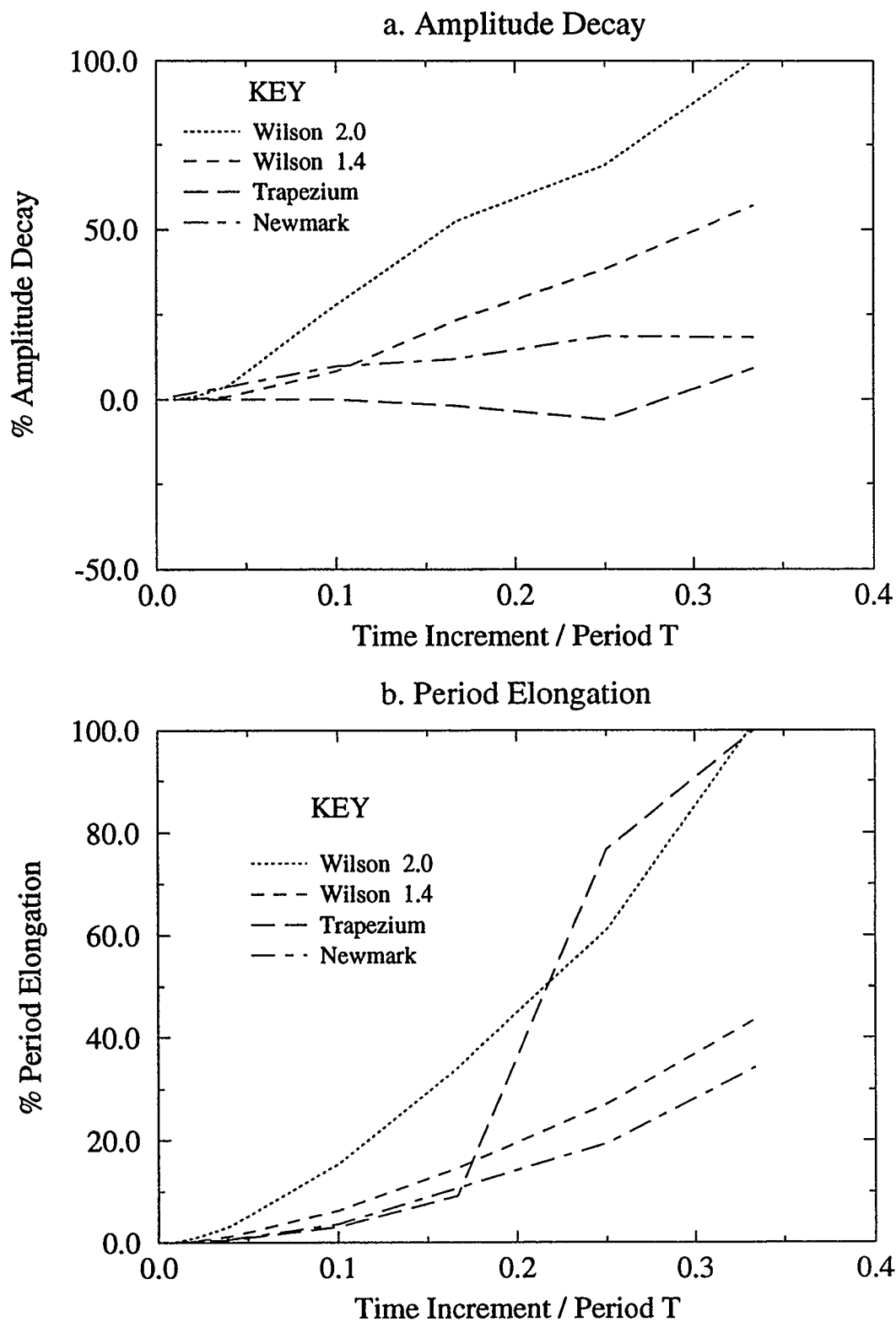


FIGURE 3.12 THE BOSSAK AND HILBER, HUGHES AND TAYLOR SCHEMES

CHAPTER FOUR

SOLUTION OF NON-LINEAR PROBLEMS USING ITERATIVE PROCEDURES

The second stage of the work introduced the concept of iterative procedures. Knowledge of these is a requirement when dealing with non-linear structural dynamic problems. Non-linearity in structures may be due to a number of reasons, for example a change from elastic to plastic behaviour, which is known as material non-linearity, or because of large structural deformations called geometric non-linearity. For more information on these types of structural behaviour see Zienkiewicz and Taylor (1991). This chapter begins by looking at some of the iterative methods in common use which model non-linear behaviour, and how effective they are. This leads to the next stage of research which was a study of the finite element program Nonsap. Nonsap, a structural analysis program for static and dynamic response of non-linear systems, was developed by Bathe, Wilson and Iding (1974), at the University of California, Berkeley in 1974. This version of Nonsap was selected because of its capability to perform non-linear dynamic problems and its adaptability to modification; in this case the inclusion of new dynamic schemes. It was decided that some amendments would be required to make Nonsap a tool that could be used in the study of time increment size effects for non-linear dynamic problems. The last section of this chapter looks at the amendments made and how the theory in Nonsap was altered.

4.1 ITERATIVE METHODS

The key to a rapidly converging iterative procedure is to first have a good starting point, this means selecting an accurate 'predictor' dynamic solution scheme, see Chapter Three. An efficient iterative procedure is then chosen as the 'corrector'. It should be remembered that choosing a method which requires repetitive updating and inverting of the effective stiffness matrix, which is called the Jacobian in matrix theory, will increase the solution time significantly. There are essentially two types of iterative method, the

'fixed tangent' which calculates the Jacobian and keeps it constant throughout the iterations, and the 'secant' schemes which update the Jacobian using values from each previous iteration.

Briefly, the iterative procedures use known displacements, velocities and accelerations from the previous time step to calculate an effective load vector and effective stiffness matrix. These are then used to calculate an incremental displacement using a ratio relationship, see Figure 4.1. A problem arises because the effective stiffness matrix behaves linearly while the solution is non-linear. The solution is to perform a convergence check to see if the incremental displacement is significantly different, if this is true then the above procedure is repeated until the incremental displacement converges to the required tolerance. These multiple iterations have the effect of dissipating the out-of-balance force vectors, graphically they appear to step along the non-linear force curve in a series of straight lines. In the following sections there are more detailed descriptions of examples of the two basic iterative procedures.

4.1.1 MODIFIED NEWTON ITERATION SCHEME

The Modified Newton method uses a constant stiffness matrix, it was named the 'initial stress' or 'load transfer' process by Zienkiewicz and his co-workers, Nayak and Zienkiewicz (1972). The Modified Newton Iteration method uses the effective load matrix for a time increment from n to $n+1$, and the displacement for the first iteration $x_{n+1}^{(i-1)}$, which is made equal to the known displacement at the start of the time increment n , to track along a curve of a non-linear force vector:

$$\hat{R}_{n+1} = P(x_{n+1}) - f \quad (4.1)$$

The $P(x_{n+1})$ is the internal force vector and f is the external force vector, see equation (2.44). An approximation is made about the total force vector for a selected incremental step from i to $i+1$.

$$\hat{R}_{n+1}^{(i)} \approx \hat{R}_{n+1}^{(i-1)} + \left(\frac{\partial \hat{R}}{\partial x} \right)_{n+1}^{(i-1)} \Delta x^{(i)} = 0 \quad (4.2)$$

$$\frac{\partial \hat{R}}{\partial x} = \hat{K}$$

The effective stiffness matrix, a constant approximation for this scheme, is formed at the beginning of the iterative procedure. Equation (4.2) can be rearranged into a form that is used to solve for the displacement increment $\Delta x^{(i)}$.

$$\Delta x^{(i)} = -\hat{K}^{-1} \hat{R}_{n+1}^{(i-1)} \quad (4.3)$$

The new displacement for the known force vector can be calculated.

$$x_{n+1}^{(i)} = x_{n+1}^{(i-1)} + \Delta x^{(i)} \quad (4.4)$$

This new displacement value is used to calculate the new effective load vector. A convergence check using a specified tolerance is applied to see if the $\Delta x^{(i)}$ incremental displacement is significantly smaller than the total displacement for the time step. If this is not the case then the whole process is repeated from equation (4.3) with i now equal to $i+1$. Figure 4.1 is a diagram showing the iterative process just described and below is a summary of the Modified Newton procedure.

- 1 Record $x_n = x_{n+1}^{(i-1)}$
- 2 Compute $\hat{R}_{n+1}^{(i-1)}$
- 3 Solve For $\Delta x^{(i)}$
- 4 Compute $x_{n+1}^{(i)}$
- 5 Convergence ? No, Begin At 2 Again

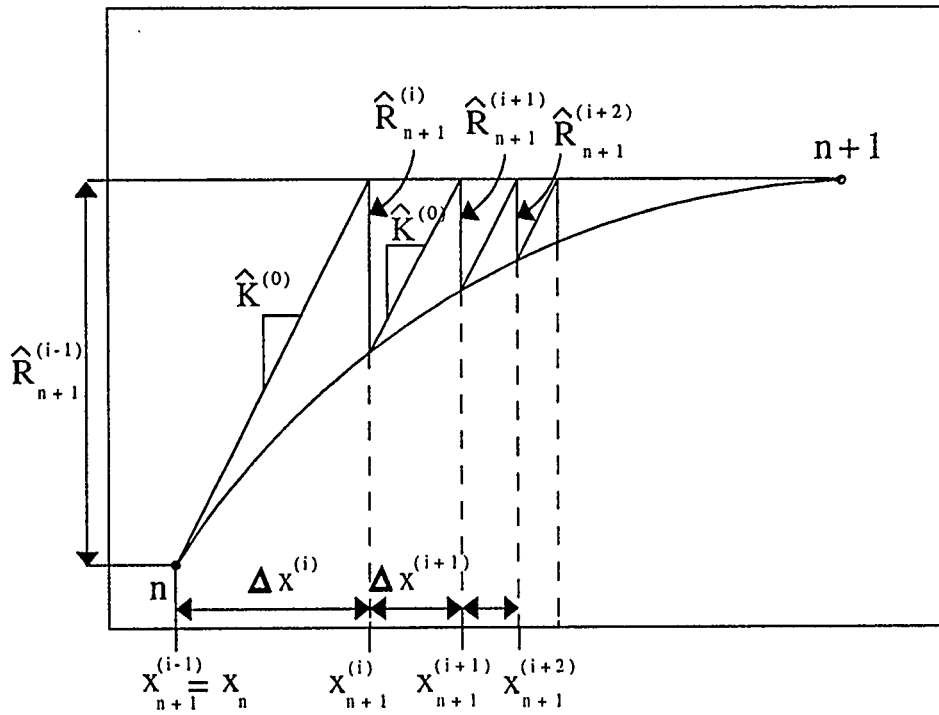


FIGURE 4.1 THE MODIFIED NEWTON ITERATION METHOD

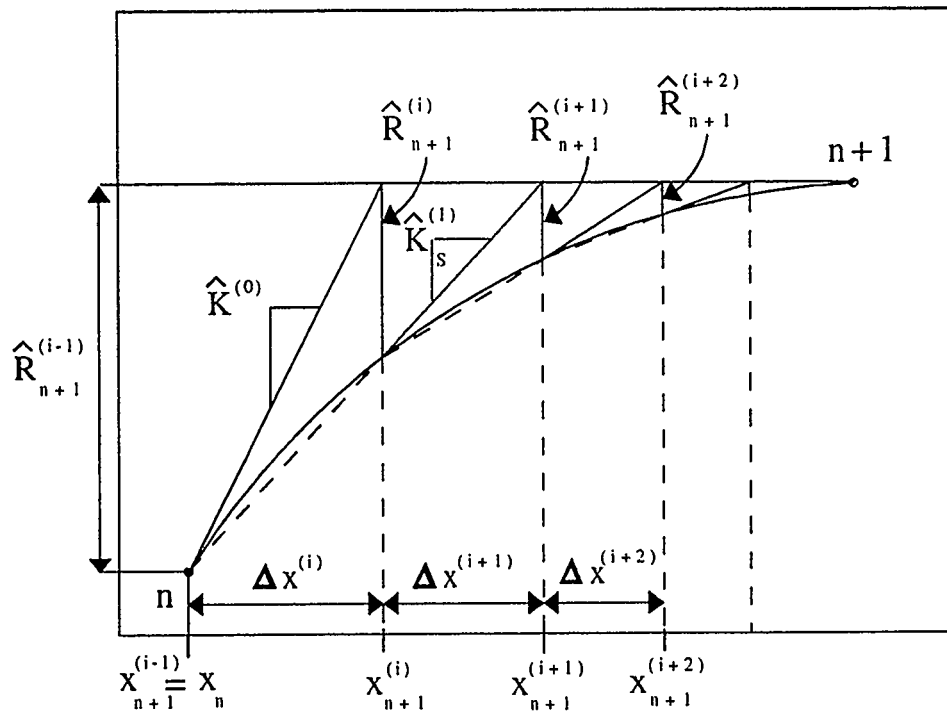


FIGURE 4.2 THE SECANT ITERATION METHOD

The Modified Newton method is less expensive on computer memory than the Newton-Raphson method which uses the same procedure but reforms the effective stiffness matrix before each iteration. It should be noted that the Newton-Raphson method converges more rapidly and therefore fewer computations are required but this does not compensate for the costly stiffness reformations.

4.1.2 QUASI-NEWTON METHOD

Improvement in convergence of schemes is often desirable in cases of very large plastic failure or displacement, hence the development of the quasi-Newton methods. The quasi-Newton, also referred to as the variable gradient method, improves on the Modified Newton, which only converges linearly, and the Newton-Raphson methods. Instead of totally reforming the effective stiffness matrix each iteration the quasi-Newton methods only slightly modify the stiffness matrix, see Figure 4.2. The secant method is one of the simplest of the quasi-Newton schemes, it begins by first calculating the value of $\Delta x^{(1)}$ using equation (4.3) of the Modified Newton method. This value is then used to calculate the new total displacement and then the effective load matrix at i equal to one. The secant 'slope' $K_s^{(1)}$ of the non-linear force curve is solved using the known values just calculated.

$$\Delta x^{(1)} = -(\hat{K}_s^{(1)})^{-1}(\hat{R}_{n+1}^{(0)} - \hat{R}_{n+1}^{(1)}) \quad (4.5)$$

The secant 'slope' term can now be used to calculate a new displacement increment value.

$$\Delta x^{(2)} = -(\hat{K}_s^{(1)})^{-1} \hat{R}_{n+1}^{(1)} \quad (4.6)$$

The $\Delta x^{(2)}$ is used to determine the total displacement in equation (4.4). As with the Modified Newton method a convergence check is carried out, if non-convergent then a new displacement increment is calculated by applying equations (4.5) and (4.6) again, these equations are written here in their general form:

$$\Delta \mathbf{x}^{(i)} = -(\hat{\mathbf{K}}_s^{(i)})^{-1}(\hat{\mathbf{R}}_{n+1}^{(i-1)} - \hat{\mathbf{R}}_{n+1}^{(i)}) = -(\hat{\mathbf{K}}_s^{(i)})^{-1} \gamma^{(i-1)} \quad (4.7)$$

$$\Delta \mathbf{x}^{(i+1)} = -(\hat{\mathbf{K}}_s^{(i)})^{-1} \hat{\mathbf{R}}_{n+1}^{(i)} \quad (4.8)$$

If the secant method is used to solve for a single degree of freedom system the matrix and vectors in equation (4.7) reduce to scalars, and hence the determination of the effective secant stiffness is a trivial solution. The secant scheme will converge almost as rapid as the Newton-Raphson method but with much greater computer efficiency.

For multi-degree of freedom systems the determination of the effective secant stiffness matrix, in equation (4.7), is no longer simple if matrix algebra is to be preserved. A number of secant update methods exist which calculate a new effective stiffness matrix using previously calculated matrices. One of these methods is the Broyden, Fletcher, Goldfarb and Shanno (BFGS) method (Zienkiewicz and Taylor (1991), and Owen and Hinton (1986)). The BFGS method converges quickly and also preserves symmetry and positive definiteness of the effective stiffness matrix. The secant method procedure is used but the determination of the new effective stiffness matrix is now calculated using the following expression:

$$(\hat{\mathbf{K}}_{\text{BFGS}}^{(i)})^{-1} = (\mathbf{I} + \mathbf{w}_i \mathbf{v}_i^T)(\hat{\mathbf{K}}_{\text{BFGS}}^{(i-1)})^{-1} (\mathbf{I} + \mathbf{v}_i \mathbf{w}_i^T) \quad (4.9)$$

Note that I is the identity matrix, and v_i and w_i equal the following expressions:

$$v_i = \hat{R}_{n+1}^{(i-1)} \left\{ 1 - \frac{(\Delta x^{(i)})^T \gamma^{(i-1)}}{(\Delta x^{(i+1)})^T \hat{R}_{n+1}^{(i-1)}} \right\} - \hat{R}_{n+1}^{(i)} \quad (4.10)$$

$$w_i = \frac{\Delta x^{(i)}}{\Delta x^{(i)T} \gamma^{(i-1)}}$$

Where γ equals:

$$\gamma^{(i-1)} = \hat{R}_{n+1}^{(i-1)} - \hat{R}_{n+1}^{(i)} \quad (4.11)$$

Matthies and Strang (1979) reported that the BFGS method converges under more extreme loading increment cases than the Modified Newton procedure, and there are substantial computational savings with the stiffness reformations. It was also found that the BFGS method consistently took fewer iterations to converge than the Modified Newton iteration scheme. For further information on the application of the BFGS method see Matthies and Strang (1979).

4.2 NONSAP, A FINITE ELEMENT PROGRAM

Nonsap was written using the Wilson θ and Newmark 'predictor' schemes in conjunction with the Modified Newton 'corrector' method to solve non-linear dynamic problems. The Wilson θ and Newmark schemes do not always give the best dynamic solution for a problem, therefore Nonsap was improved by introducing the SSpj scheme which can simulate a large selection of algorithms. A study on the effect of time increment size on a non-linear case was carried out after the amendments. Time step influences the stability and accuracy of the dynamic solution schemes and the rate of convergence of the iterative procedure, the results of this study are in Chapter Five.

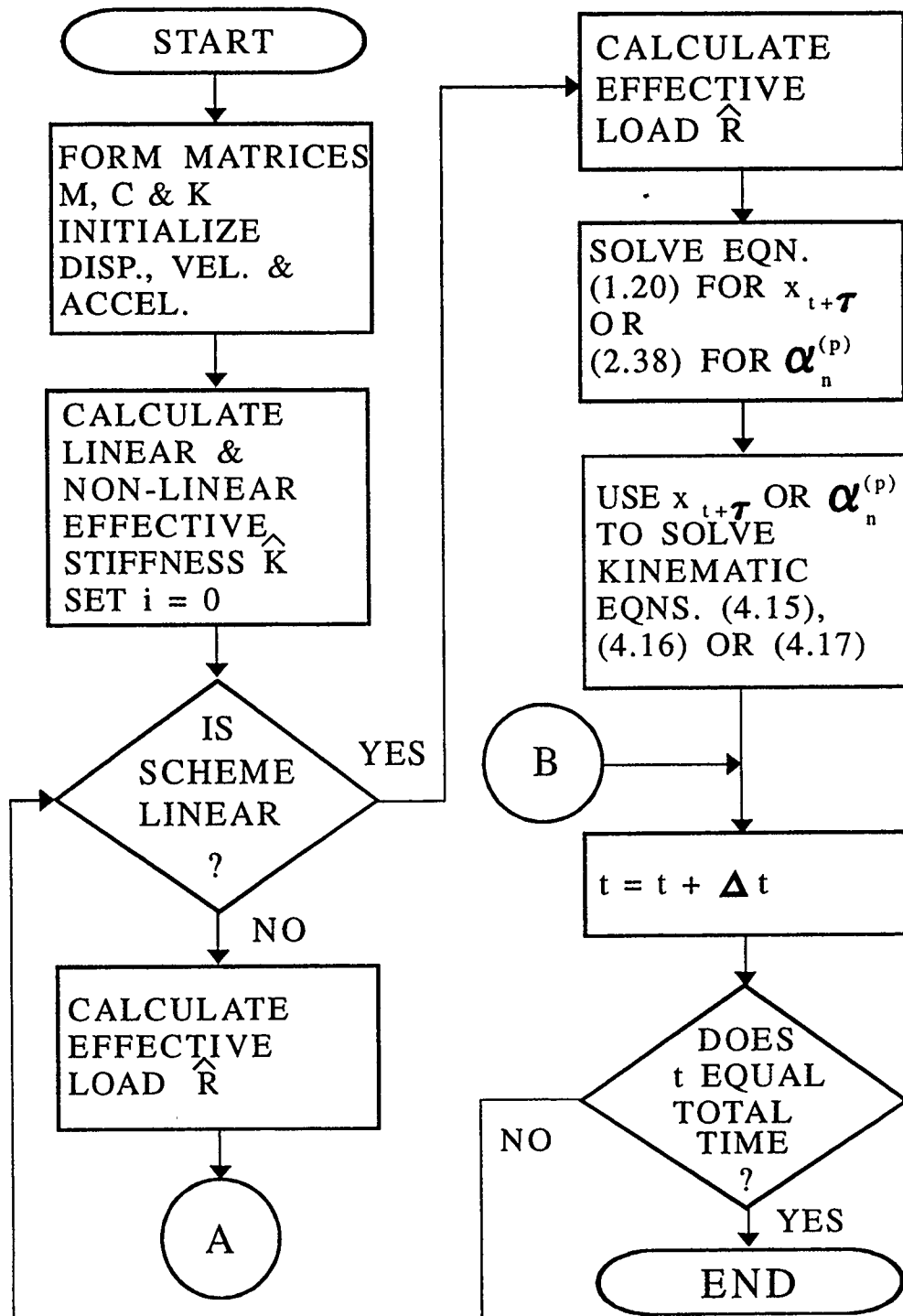


FIGURE 4.3 FLOW CHART OF NONSAP'S METHODOLOGY

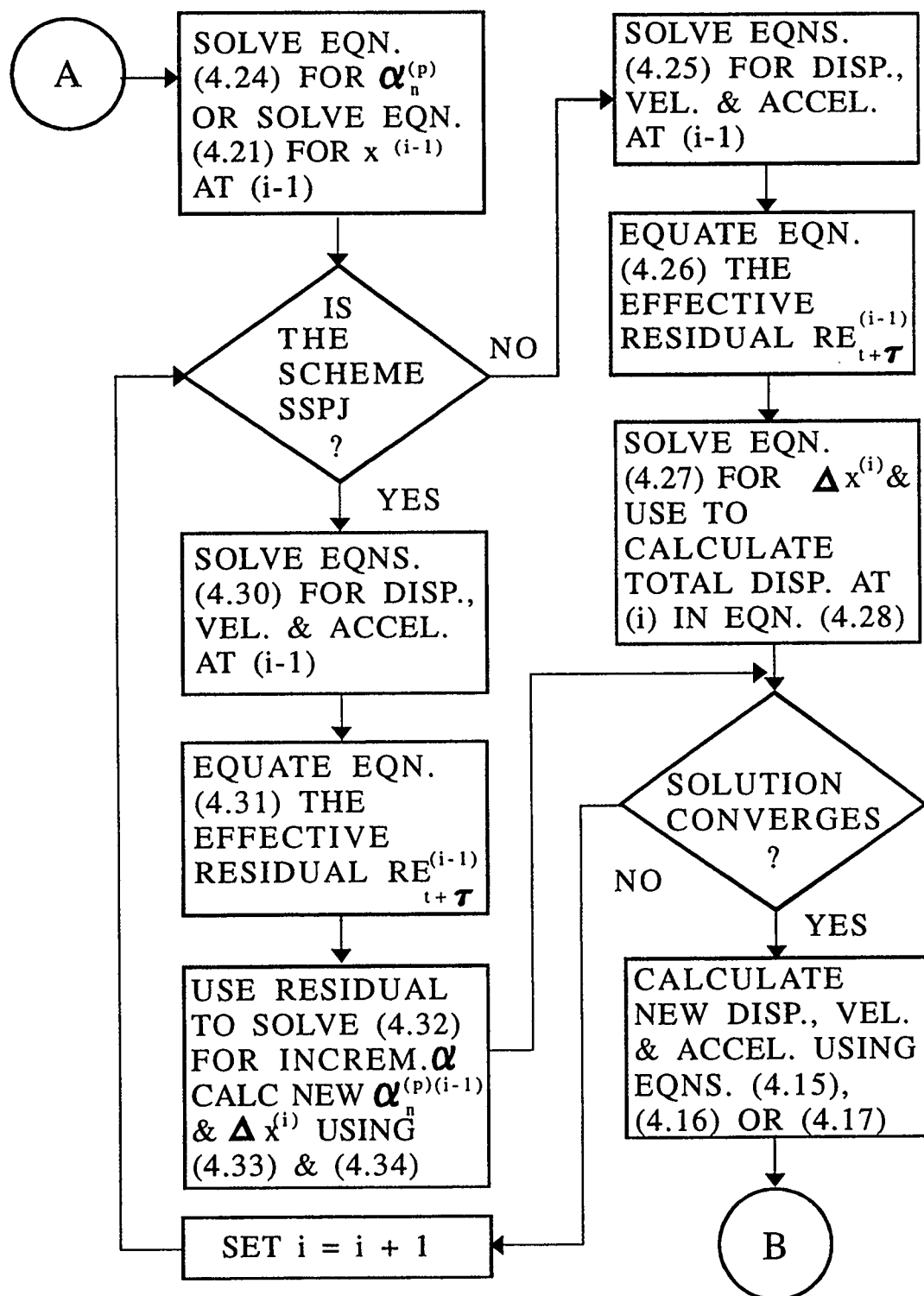


FIGURE 4.3 FLOW CHART OF NONSAP'S METHODOLOGY CONTINUED

4.2.1 A LINEAR FORMULATION

A review of Nonsap's step-by-step integration scheme with the SSpj amendments are discussed here. Reference to Figure 4.3, which illustrates Nonsap's methodology with a flow chart, will help during the following discussion. Firstly, following the procedure for a linear case, the effective stiffness matrix and effective load vector are formulated as in Chapter One.

$$\hat{K} = M(a_0 + a_1 \alpha) + C a_1 + K(a_2 + a_1 \beta) \quad (4.12)$$

The Wilson θ and Newmark effective load vector is:

$$\begin{aligned} \hat{R}_{t+\tau} = & R_t + \theta(R_{t+\Delta t} - R_t) + M [x_t(a_3 + a_6 \alpha) + \dot{x}_t(a_4 + a_7 \alpha) + \ddot{x}_t(a_5 + a_8 \alpha)] \\ & + C [x_t a_6 + \dot{x}_t a_7 + \ddot{x}_t a_8] + K [x_t(a_9 + a_6 \beta) + \dot{x}_t(a_{10} + a_7 \beta) + \ddot{x}_t(a_{11} + a_8 \beta)] \end{aligned} \quad (4.13)$$

While the SSpj effective load vector is:

$$\begin{aligned} \hat{R}_{t+\tau} = & R_t + \theta(R_{t+\Delta t} - R_t) - M [x_t(a_3 + a_6 \alpha) + \dot{x}_t(a_4 + a_7 \alpha) + \ddot{x}_t(a_5 + a_8 \alpha)] \\ & - C [x_t a_6 + \dot{x}_t a_7 + \ddot{x}_t a_8] - K [x_t(a_9 + a_6 \beta) + \dot{x}_t(a_{10} + a_7 \beta) + \ddot{x}_t(a_{11} + a_8 \beta)] \end{aligned} \quad (4.14)$$

The α and β terms are factors giving the Rayleigh damping contribution from the mass and stiffness matrices, see Clough and Penzien (1993), and θ equals one for the Newmark scheme. The a_1 to a_{11} terms are dependent on the solution scheme selected, for examples of their evaluation see the Wilson θ equations (1.21) and (1.22), and the SS32 equations (2.39) and (2.40). The values for a_i for the Wilson θ , Newmark, SS22 and SS32 are listed in Table 4.1.

Once the effective stiffness and load expressions are evaluated they are used to solve for the displacement at $t + \tau$ for the Wilson θ and Newmark schemes, or for the $\alpha_n^{(p)}$ term for the SSpj algorithms using equations (1.20) or (2.38) respectively. These terms are

	Wilson θ	Newmark	SS22	SS32
a_0	$6/(\theta\Delta t)^2$	$1/(\alpha\Delta t^2)$	1	$\theta_1\Delta t$
a_1	$3/(\theta\Delta t)$	$\delta/(\alpha\Delta t)$	$\theta_1\Delta t$	$(\theta_2\Delta t^2)/2$
a_2	1	1	$(\theta_2\Delta t^2)/2$	$(\theta_3\Delta t^3)/6$
a_3	a_0	a_0	0	0
a_4	$2a_1$	$1/(\alpha\Delta t)$	0	0
a_5	2	$1/(2\alpha) - 1$	0	1
a_6	a_1	a_1	0	0
a_7	a_5	$\delta/\alpha - 1$	a_0	a_5
a_8	$(\theta\Delta t)/2$	$\Delta t(\delta/\alpha - 2)/2$	0	a_0
a_9	0	0	a_0	a_5
a_{10}	0	0	a_1	a_0
a_{11}	0	0	0	a_1
a_{12}	a_0/θ	a_0	0	1
a_{13}	$(-2a_1)/\theta$	$-a_4$	1	Δt
a_{14}	$1 - 3/\theta$	$-a_5$	0	a_{13}
a_{15}	$\Delta t/2$	$\Delta t(1 - \delta)$	Δt	$(\Delta t^2)/2$
a_{16}	$\Delta t^2/6$	$\delta\Delta t$	a_{15}	a_{13}
a_{17}	-	-	0	a_{15}
a_{18}	-	-	$(\Delta t^2)/2$	$(\Delta t^3)/6$

TABLE 4.1 THE SOLUTION SCHEMES a_i PARAMETER VALUES.

then used to solve the following kinematic equations for displacement, velocity and acceleration at the end of the time step.

$$\begin{aligned}
 \ddot{x}_{t+\Delta t} &= a_{12}(x_{t+\theta\Delta t} - x_t) + a_{13} \dot{x}_t + a_{14} \ddot{x}_t \\
 \text{Wilson } \theta : \quad \dot{x}_{t+\Delta t} &= \dot{x}_t + a_{15}(\ddot{x}_{t+\Delta t} + \ddot{x}_t) \\
 x_{t+\Delta t} &= x_t + \Delta t \dot{x}_t + a_{16}(\ddot{x}_{t+\Delta t} + 2\ddot{x}_t)
 \end{aligned} \tag{4.15}$$

$$\begin{aligned}
 \ddot{x}_{t+\Delta t} &= a_{12}(x_{t+\Delta t} - x_t) + a_{13} \dot{x}_t + a_{14} \ddot{x}_t \\
 \text{Newmark} : \quad \dot{x}_{t+\Delta t} &= \dot{x}_t + a_{15}\ddot{x}_t + a_{16}\ddot{x}_{t+\Delta t} \\
 x_{t+\Delta t} &= x_t + \Delta t \dot{x}_t + \frac{\Delta t^2}{2} \ddot{x}_t
 \end{aligned} \tag{4.16}$$

$$\begin{aligned}
 \ddot{x}_{t+\Delta t} &= a_{12} \ddot{x}_t + a_{13} \alpha_n^{(p)} \\
 \text{SSpj} : \quad \dot{x}_{t+\Delta t} &= \dot{x}_t + a_{14} \ddot{x}_t + a_{15} \alpha_n^{(p)} \\
 x_{t+\Delta t} &= x_t + a_{16} \dot{x}_t + a_{17} \ddot{x}_t + a_{18} \alpha_n^{(p)}
 \end{aligned} \tag{4.17}$$

The following relationship applies:

$$x = x_{t+\tau} - x_t \tag{4.18}$$

4.2.2 A NON-LINEAR FORMULATION

The non-linear process requires small amendments to the linear versions of the effective stiffness matrix and load vector, this allows for the introduction of the Modified Newton iteration procedure illustrated in Figure 4.3. The non-linear formulation has a linear and a non-linear stiffness contribution, K_L represents the linear and K_{NL} the non-linear component. The Wilson θ and the Newmark methods begin by solving the following equations:

$$\hat{K} = M(a_0 + a_1 \alpha) + C a_1 + K_L(a_2 + a_1 \beta) + K_{NL} a_2 \quad (4.19)$$

$$\begin{aligned} \hat{R}_{i+\tau}^{(i-1)} = & R_i + \theta(R_{i+\Delta t} - R_i) + M [\dot{x}_i(a_4 + a_7 \alpha) + \ddot{x}_i(a_5 + a_8 \alpha)] \\ & + C [\dot{x}_i a_7 + \ddot{x}_i a_8] + K_L [-x_i a_2 + \dot{x}_i(a_{10} + a_7 \beta) + \ddot{x}_i(a_{11} + a_8 \beta)] - K_{NL} x_i a_2 \end{aligned} \quad (4.20)$$

Note that there is no capability for Rayleigh damping from the non-linear stiffness portion in Nonsap. The effective stiffness matrix and the effective load vector are used to solve for the incremental displacement.

$$x^{(i-1)} = \hat{K}^{-1} \hat{R}_{i+\tau}^{(i-1)} \quad (4.21)$$

The SSpj schemes use a slightly different approach, the following are the equations for the effective stiffness and effective load vectors:

$$\hat{K} = M(a_0 + a_1 \alpha) + C a_1 + (K_L + K_{NL})(a_2 + a_1 \beta) \quad (4.22)$$

$$\begin{aligned} \hat{R}_{i+\tau}^{(i-1)} = & R_i + \theta_1(R_{i+\Delta t} - R_i) - M [\dot{x}_i(a_3 + a_6 \alpha) + \dot{x}_i(a_4 + a_7 \alpha) + \ddot{x}_i(a_5 + a_8 \alpha)] \\ & - C [\dot{x}_i a_6 + \dot{x}_i a_7 + \ddot{x}_i a_8] \\ & - (K_L + K_{NL}) [\dot{x}_i(a_9 + a_6 \beta) + \dot{x}_i(a_{10} + a_7 \beta) + \ddot{x}_i(a_{11} + a_8 \beta)] \end{aligned} \quad (4.23)$$

The SSpj formulation allows for Rayleigh damping for both the linear and non-linear contributions of the stiffness matrix, this makes the scheme slightly more general and accurate. Unlike the Wilson θ and the Newmark formulations the SSpj method uses the effective stiffness matrix and the effective load matrix to solve an equation of the same form as (2.38) for $\alpha_n^{(p)}$ at $i - 1$.

$$\alpha_n^{(p)(i-1)} = \hat{K}^{-1} \hat{R}_{t+\tau}^{(i-1)} \quad (4.24)$$

The next stage of the Modified Newton scheme is to apply the first iteration and check for convergence. In Nonsap this is achieved for all dynamic solution schemes by first calculating the load residual which represents the error due to the simplification of the effective load term. Equations (4.25) are used to calculate the displacement, velocity and acceleration at the extended time step for the Wilson θ and the Newmark schemes.

$$\begin{aligned} \ddot{x}_{t+\tau}^{(i-1)} &= -a_0 x^{(i-1)} + a_4 \dot{x}_t + a_5 \ddot{x}_t \\ \dot{x}_{t+\tau}^{(i-1)} &= -a_1 x^{(i-1)} + a_7 \dot{x}_t + a_8 \ddot{x}_t \\ x_{t+\tau}^{(i-1)} &= -x^{(i-1)} - x_t \end{aligned} \quad (4.25)$$

These equations are used to solve the general residual equation for the $(i - 1)$ st effective out-of-balance loads. The effective residual equation for the Wilson θ and the Newmark schemes is:

$$\begin{aligned} RE_{t+\tau}^{(i-1)} &= R_t + \theta(R_{t+\Delta t} - R_t) + M \left[\ddot{x}_{t+\tau}^{(i-1)} + \alpha \dot{x}_{t+\tau}^{(i-1)} \right] + C \left[\dot{x}_{t+\tau}^{(i-1)} \right] \\ &\quad + K_L \left[x_{t+\tau}^{(i-1)} + \beta \dot{x}_{t+\tau}^{(i-1)} \right] + F_{NL} \left[x_{t+\tau}^{(i-1)} \right] \end{aligned} \quad (4.26)$$

The F_{NL} term represents the non-linear forces in the structure due to the material non-linearity. This effective residual term is then used in place of the effective load vector in equation (4.3) to solve for the incremental displacement error.

$$\Delta x^{(i)} = \hat{K}^{-1} RE_{t+\tau}^{(i-1)} \quad (4.27)$$

Nonsap uses the new total displacement for the time step and the incremental displacement from the iterative procedure to perform a convergence check.

$$x_{t+\tau}^{(i)} = x_t + x^{(i-1)} + \Delta x^{(i)} \quad (4.28)$$

$$\frac{\sum_{i=1}^k (\Delta x^{(i)})^2}{\sum_{i=1}^k (x_{t+\tau}^{(i)})^2} < \text{tolerance} \quad k = \text{number of equations} \quad (4.29)$$

For the SSpj cases use equations (4.30) at iteration $(i - 1)$ to solve the following effective residual equation (4.31).

$$\begin{aligned} \ddot{x}_{t+\tau}^{(i-1)} &= a_0 \alpha_n^{(p)(i-1)} + a_5 \ddot{x}_t \\ \dot{x}_{t+\tau}^{(i-1)} &= a_1 \alpha_n^{(p)(i-1)} + a_7 \dot{x}_t + a_8 \ddot{x}_t \\ x_{t+\tau}^{(i-1)} &= a_2 \alpha_n^{(p)(i-1)} + a_9 x_t + a_{10} \dot{x}_t + a_{11} \ddot{x}_t \end{aligned} \quad (4.30)$$

$$\begin{aligned} RE_{t+\tau}^{(i-1)} &= R_t + \theta_1 (R_{t+\Delta t} - R_t) - M \left[\ddot{x}_{t+\tau}^{(i-1)} + \alpha \dot{x}_{t+\tau}^{(i-1)} \right] - C \left[\dot{x}_{t+\tau}^{(i-1)} \right] \\ &\quad - (K_L + F_{NL}) \left[x_{t+\tau}^{(i-1)} + \beta \dot{x}_{t+\tau}^{(i-1)} \right] \end{aligned} \quad (4.31)$$

The total residual error is then applied in equation (4.32) to yield the incremental error in the alpha term.

$$\Delta \alpha_n^{(p)(i-1)} = \hat{K}^{-1} RE_{t+\tau}^{(i-1)} \quad (4.32)$$

This incremental error is added to the total alpha at (i - 1).

$$\alpha_n^{(p)(i)} = \alpha_n^{(p)(i-1)} + \Delta\alpha_n^{(p)(i-1)} \quad (4.33)$$

But as the SSpj scheme uses a slightly different procedure, the convergence check must be adapted. A new $x_{t+\tau}^{(i)}$ value is calculated with equation (4.30) using the new $\alpha_n^{(p)(i)}$ term. The following equation is then used to calculate the incremental change in displacement for the extended time period, the displacement value then allows the convergence check to be performed in equation (4.29).

$$\Delta x^{(i)} = x_{t+\tau}^{(i)} - x_{t+\tau}^{(i-1)} \quad (4.34)$$

If the schemes converge to the prescribed tolerance then in the Wilson θ and the Newmark methods $x_{t+\tau}^{(i)}$ is used to solve equations (4.15) and (4.16). The SSpj schemes use the $\alpha_n^{(p)(i)}$ to solve kinematic equations (4.17). If convergence does not occur then the iterative procedure is repeated as in the Modified Newton method in section 4.1.1 from the point of reforming the effective load residual, equation (4.26) and (4.31), with i now equal to i + 1.

Figures 4.4 and 4.5 demonstrate the iterative procedure for both the methods discussed.

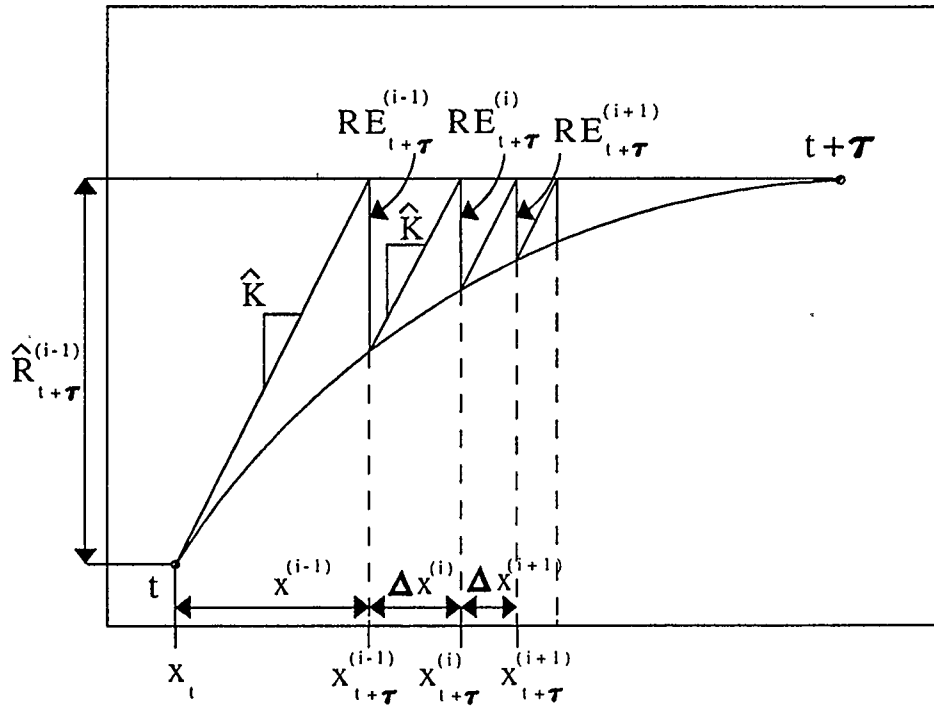
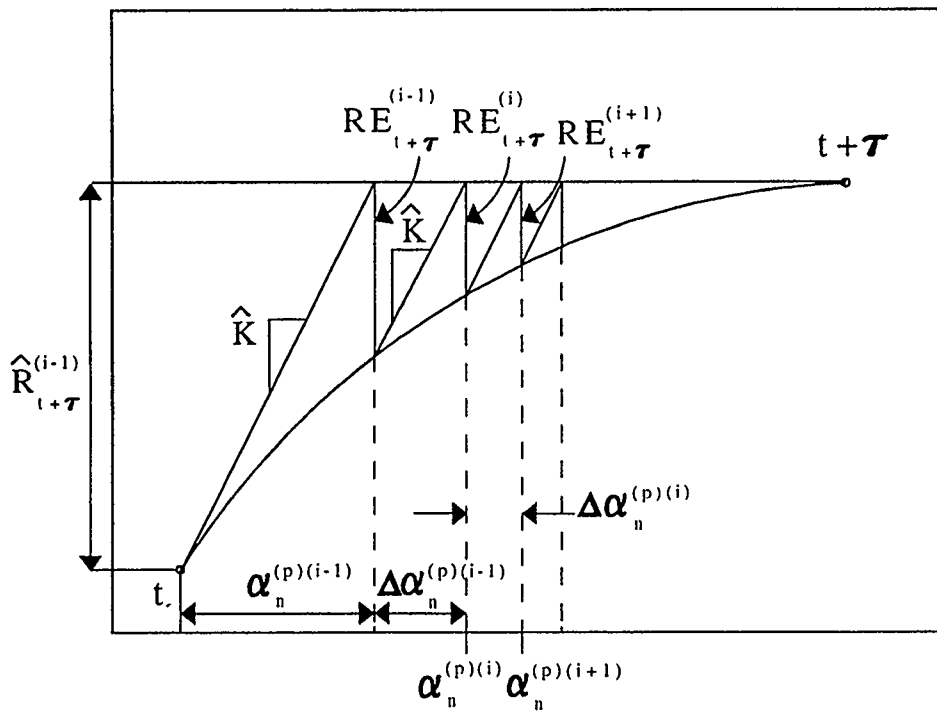
FIGURE 4.4 THE WILSON θ AND NEWMARK ITERATIVE METHOD

FIGURE 4.5 THE SSpj ITERATIVE METHOD

CHAPTER FIVE

EXAMPLES USING NONSAP

Before amendments were made to Nonsap a preliminary linear study was performed, this was designed to obtain a better understanding of Nonsap and its capabilities. Nonsap can perform linear, non-linear, static and dynamic analyses; therefore various scenarios using all of these options were investigated.

All the displacements in the following figures are those recorded at node twenty, the free end of the cantilever shown in Figure 5.1. Stresses are taken from integration point nine in element one, this is the stress output position closest to the point of maximum compression in the cantilever.

5.1 PRELIMINARY LINEAR STATIC AND DYNAMIC STUDY OF NONSAP

We begin by examining the problem to be studied in Figure 5.1. A cantilever $40" \times 1" \times 1"$ is divided into four eight-node isoparametric elements which were selected because of their ability to model linear stress patterns. The cantilever is given a Young's modulus $E = 30E+6 \text{ lb in}^{-2}$, Poissons ratio $\nu = 0.3$, and density $\rho = 0.00074 \text{ lb-sec}^2\text{in}^{-4}$. A downward forcing function is applied to the tip of the cantilever at its free end and the displacements and stresses produced by Nonsap are recorded.

To test the model an initial linear static analysis was conducted. The program output was checked against hand calculations for deflection and stresses in the z axis. The results proved the model was indeed behaving as the hand calculations indicated. The program results are illustrated in Figure 5.2. The displacements output at the free end were $-8.26"$ and $-8.43"$ for the four and eight element models. Hand calculations produced a displacement of $-8.53"$, therefore the four and eight element models had errors of 3.17 percent and 1.17 percent. The stresses output were of the correct order when compared to hand calculations, there was some difference which was to be expected because stresses

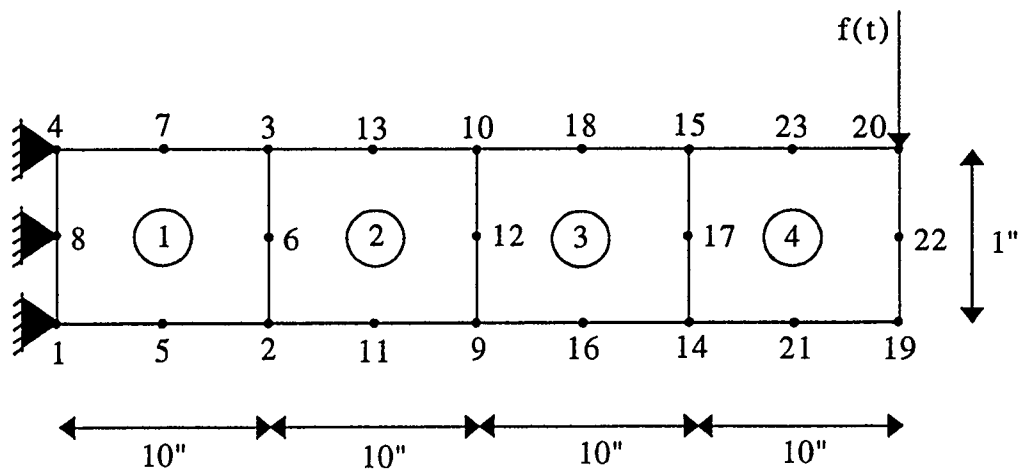


FIGURE 5.1 EXAMPLE CANTILEVER

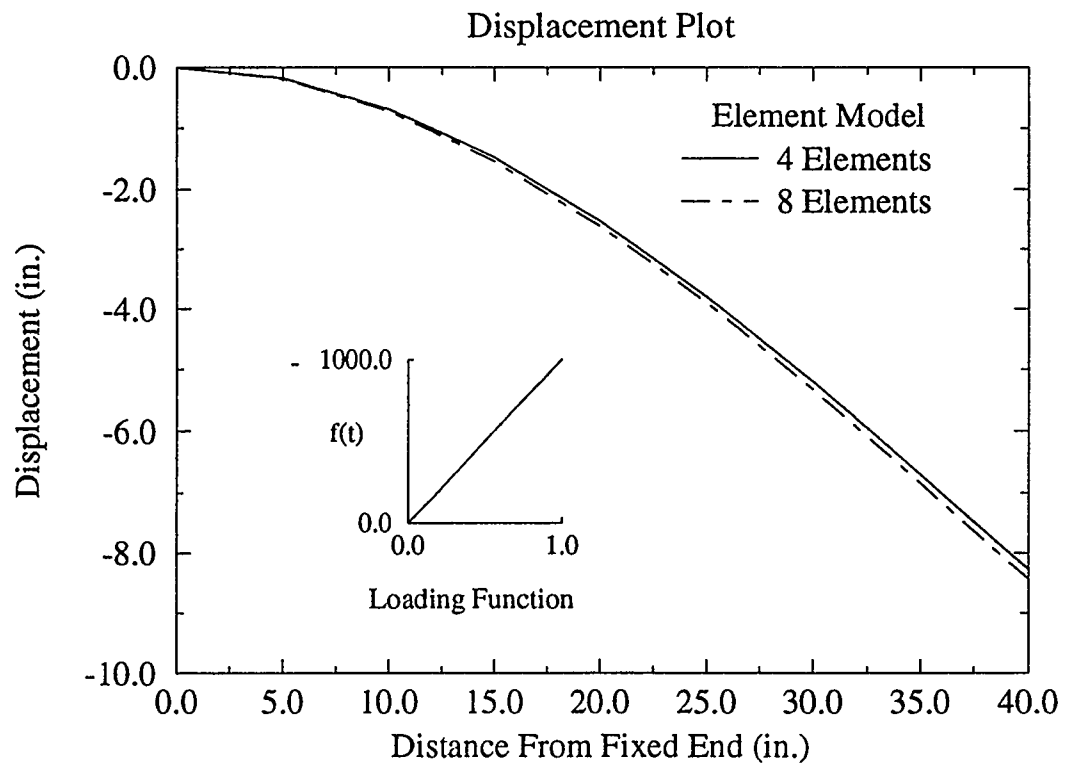


FIGURE 5.2 GRAPH FOR THE LINEAR STATIC CANTILEVER MODEL

in Nonsap are calculated at the integration points so they are slightly conservative when compared with hand calculations at the corner nodes of the elements.

The linear dynamic analysis was used to determine the best method of solution for the non-linear dynamic analysis. The first linear dynamic test modelled a four and an eight element cantilever using the Wilson θ method with θ equal to 1.4. Δt was set to 0.01 seconds and a triangular dynamic loading function applied as shown in Figure 5.3.

The test showed displacement variations at node twenty for both models were insignificant, see Figure 5.4. It was therefore deduced that the four isoparametric elements adequately modelled the cantilever deflection, this is the same conclusion arrived at for the static analysis.

The second test examined the differences in time dependent displacements for the two different mass matrices offered in Nonsap, these are the consistent and lumped mass matrix. The lumped mass matrix distributes the structure mass to the nodes, resulting in a diagonal matrix. The consistent mass matrix, a slightly more accurate representation of mass distribution, uses virtual displacements to model the mass coupling effects between coordinates, for further information see Clough and Penzien (1993).

Figure 5.5 shows minor variability in deflection for the two mass matrix formulations. These differences are due to the different natural frequencies for the two mass matrices, see Table 5.1 for the natural frequencies and periods for the two methods.

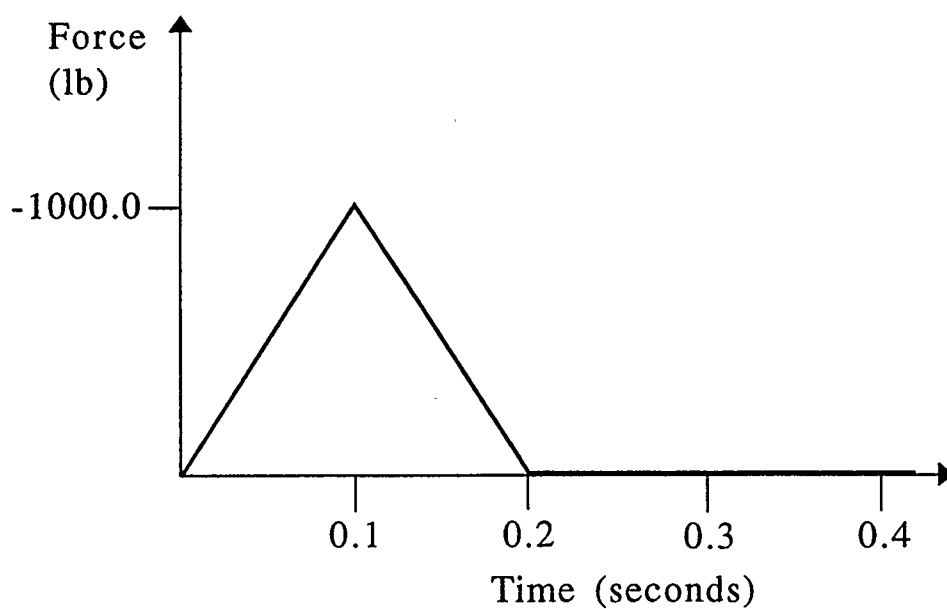


FIGURE 5.3 DYNAMIC LOADING FUNCTION

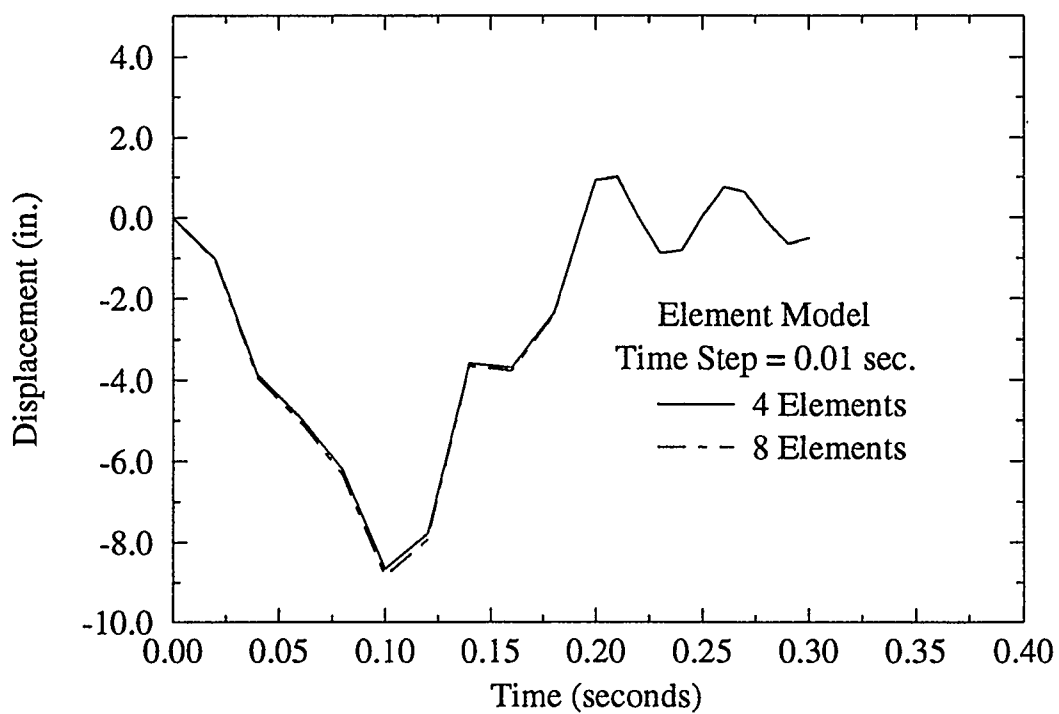


FIGURE 5.4 SIGNIFICANCE OF NUMBER OF ELEMENTS IN CANTILEVER MODEL

	Mass Matrix Formulation Method	
	Consistent Mass	Lumped Mass
Natural frequency, f	20.79 cps	20.41 cps
Period, T	0.048 sec	0.049 sec

TABLE 5.1 FREQUENCY DATA FOR DIFFERENT MASS MATRICES

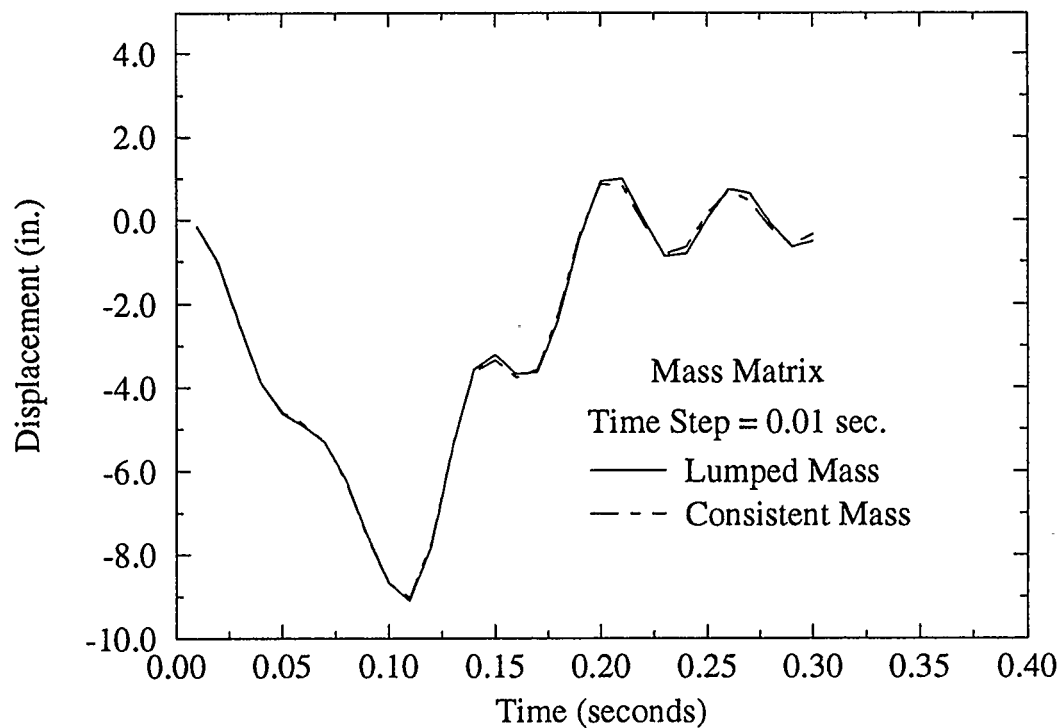


FIGURE 5.5 SIGNIFICANCE OF THE MASS MATRIX FORMULATION FOR THE CANTILEVER MODEL

5.1.1 STUDY OF TIME INCREMENT SIZE

The following tests were designed to examine the sensitivity of dynamic solution schemes with respect to time increment sizes. Two dynamic solution methods originally available in Nonsap, the Wilson θ and the Newmark schemes were used. All the runs used a lumped mass matrix so the period of free vibration was equal to 0.049 seconds, and the structure was modelled using the four element, eight-node isoparametric representation. Time increments investigated ranged from 0.1 to 0.0001 seconds, the corresponding $\Delta t/T$ values are given in Table 5.2. The forcing function was triangular causing a large displacement in the cantilever and then free vibration, no damping was used so all damping observed is a result of inaccurate approximation methods.

Time Increment / Period (seconds) (T=0.049 sec)					
Δt	0.1	0.01	0.005	0.001	0.0001
$\Delta t/T$	2.0408	0.2041	0.1020	0.0204	0.0020

TABLE 5.2 CORRESPONDING TIME INCREMENT SIZE AND PERIODS

Figure 5.6 a. shows the deflection output for the Wilson θ method ($\theta = 1.4$) for the prescribed time span. For a large time increment, Δt equal to 0.1 seconds, the results give a very crude picture of deflection. This is no surprise since a step size this large corresponds to a $\Delta t/T$ equal to 2.0, therefore deflections are only being calculated once every two periods of oscillation.

All the other time step sizes are small enough to model the vibration, the real question is how accurate are their solutions. The close-up plot in Figure 5.6 b. shows the free vibration in the cantilever from 0.2 to 0.3 seconds. It can be seen clearly that a time step

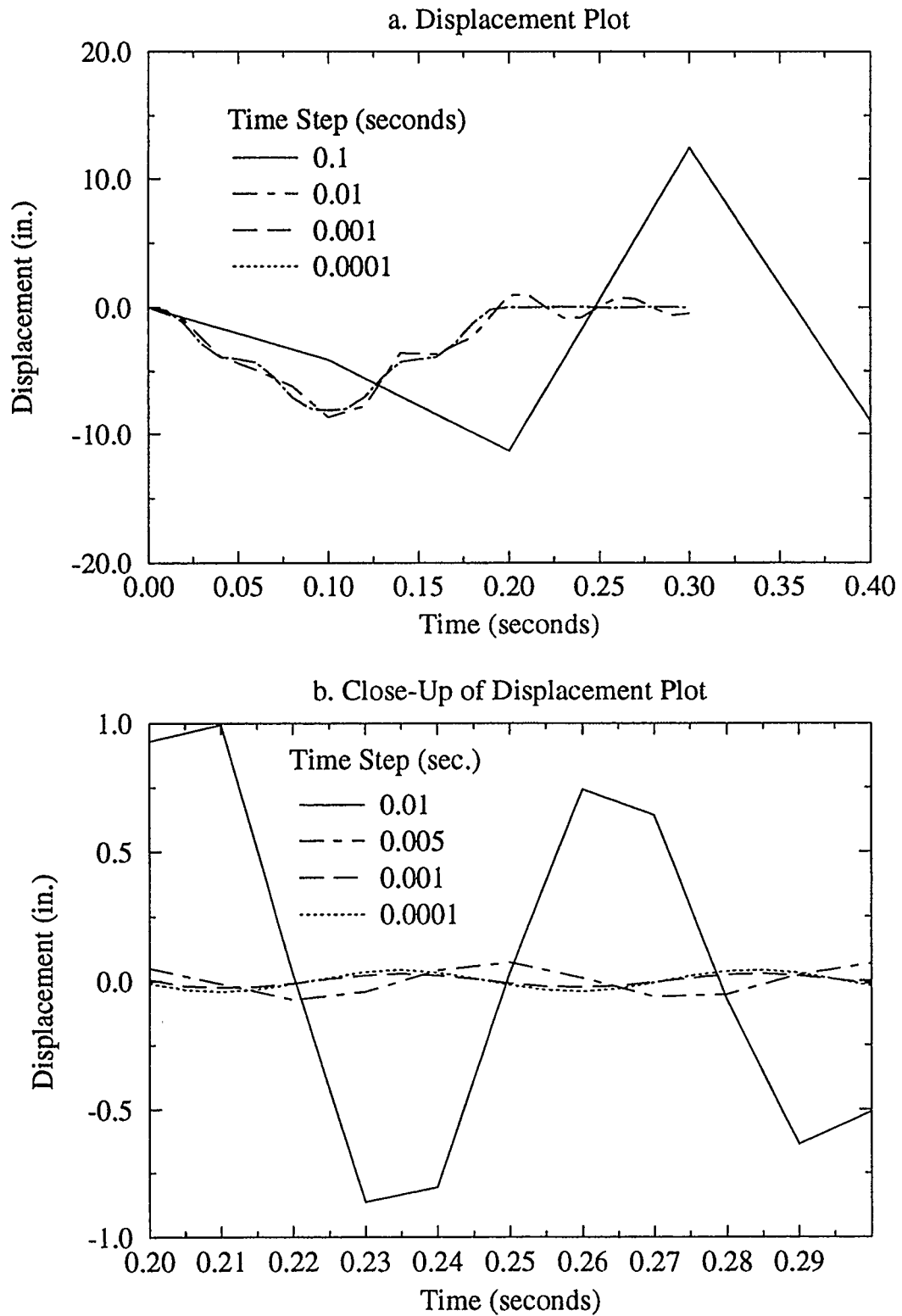


FIGURE 5.6 SENSITIVITY OF TIME INCREMENT SIZE FOR THE WILSON θ SCHEME

equal to 0.01 seconds has severe amplitude decay and that the solution is still quite crude compared to that resulting from smaller Δt values.

The line representing time step 0.005 seconds is slightly out of phase with the 0.001 and 0.0001 time step, this is probably due to a slight period elongation. The above observations correspond correctly to those published in a paper by Bathe and Wilson (1973), see Figure 5.7 and 5.8. The step size 0.005 seconds produced a period elongation of approximately five percent, and step size 0.01 seconds has period elongation and amplitude decay of approximately fifteen and nineteen percent respectively.

The Newmark scheme studied is a special case with δ equal to 0.5 and α equal to 0.25; it is commonly known as the trapezoidal method. The displacements output in Figure 5.9a., tell much the same story as the Wilson θ method, good results are only achieved when the step size is reduced to 0.005 seconds or less. One important observation is that the deflections do not show any amplitude decay, this is especially visible when looking at the close-up plot of Δt equal to 0.01 seconds. Bathe and Wilson (1973) also came to this conclusion when they examined the spectral radius and found that it has a constant value of one for all Δt , for the trapezoidal scheme.

5.1.2 STUDY OF PARAMETER VALUES

The final series of runs for the linear dynamic case compares both the Wilson θ and the Newmark methods with varying parameters. The following parameter values were used: for the Wilson θ method; θ equal to 1.4 and 2.0, and for the Newmark scheme; δ 's of 0.5 and 0.55 and α 's of 0.25 and 0.3. The results are displayed in Figure 5.10 a. and b.. The Wilson θ method with θ equal to 2.0 has the most period elongation of all the schemes. When θ is reduced to 1.4 the period elongation decreases, however the Newmark solution schemes yield even more accurate results. This is clearly seen in our results and the graphs by Bathe and Wilson. Wilson θ of 2.0 has severe amplitude decay which according to Bathe and Wilson should be significantly greater than twenty percent for a time step of 0.01 seconds. Finally our plot for the Trapezoidal case of the Newmark

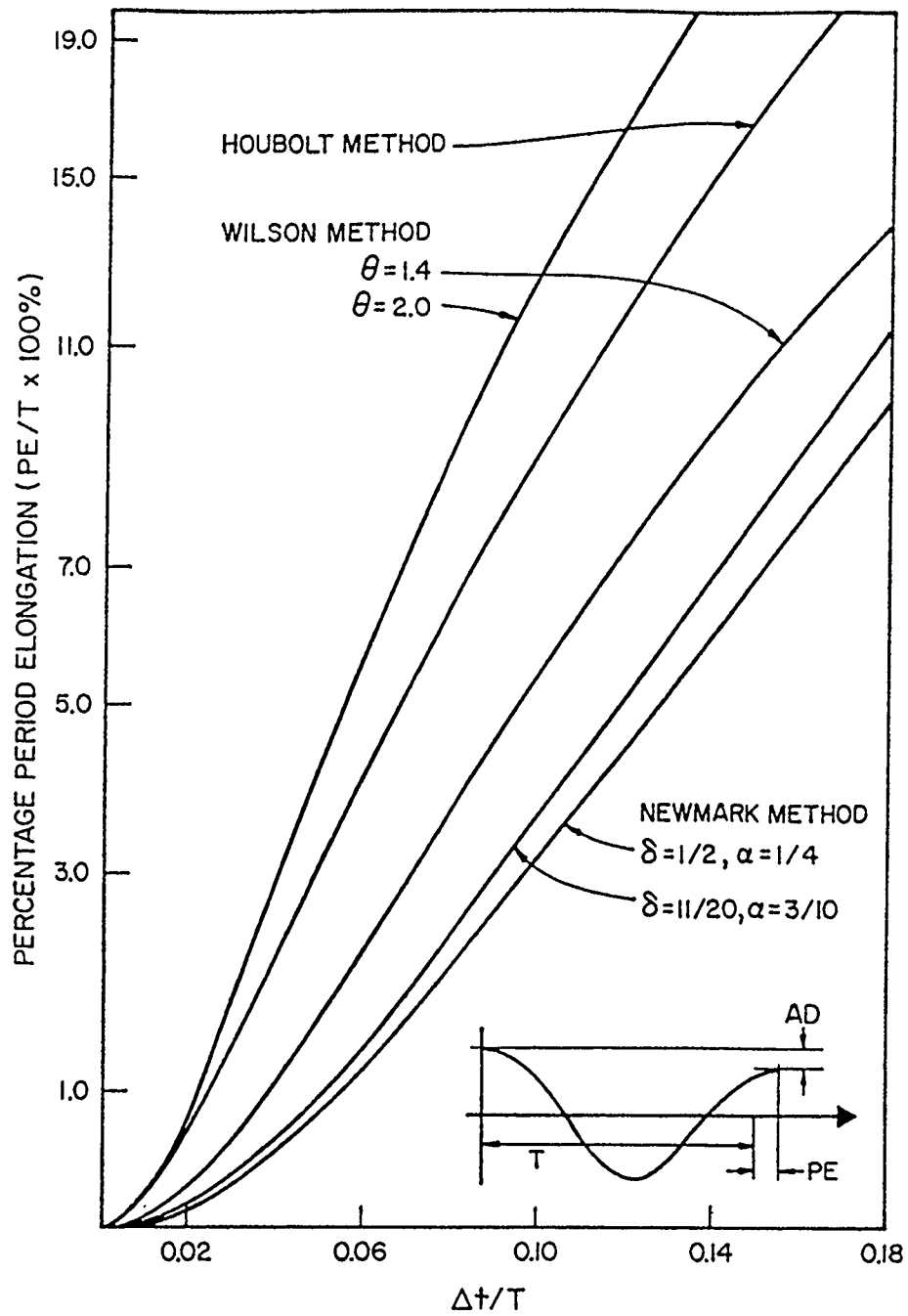


FIGURE 5.7 PERCENTAGE PERIOD ELONGATION RESULTS PUBLISHED BY BATHE AND WILSON (1973)

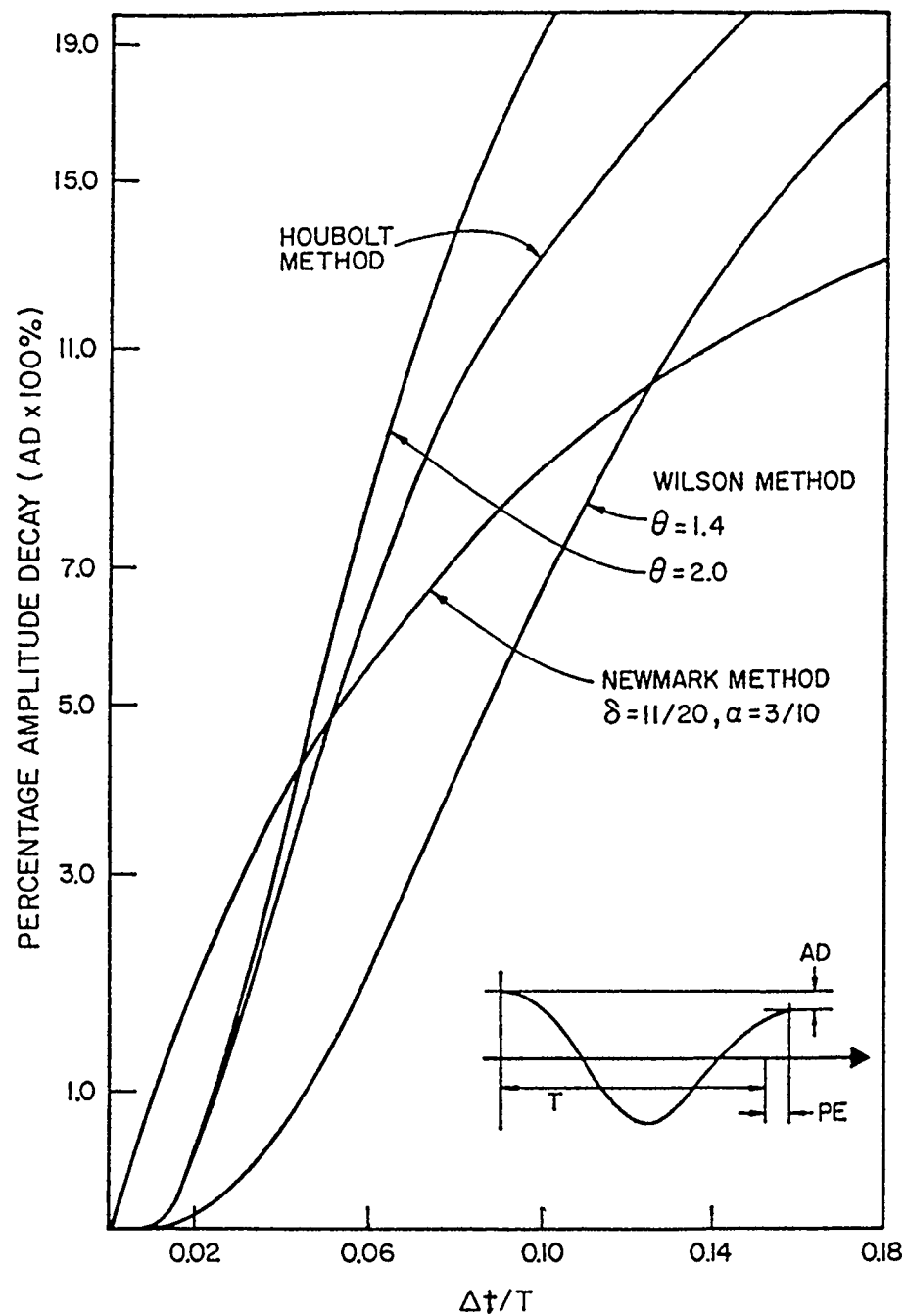


FIGURE 5.8 PERCENTAGE AMPLITUDE DECAY RESULTS PUBLISHED BY BATHE AND WILSON (1973)

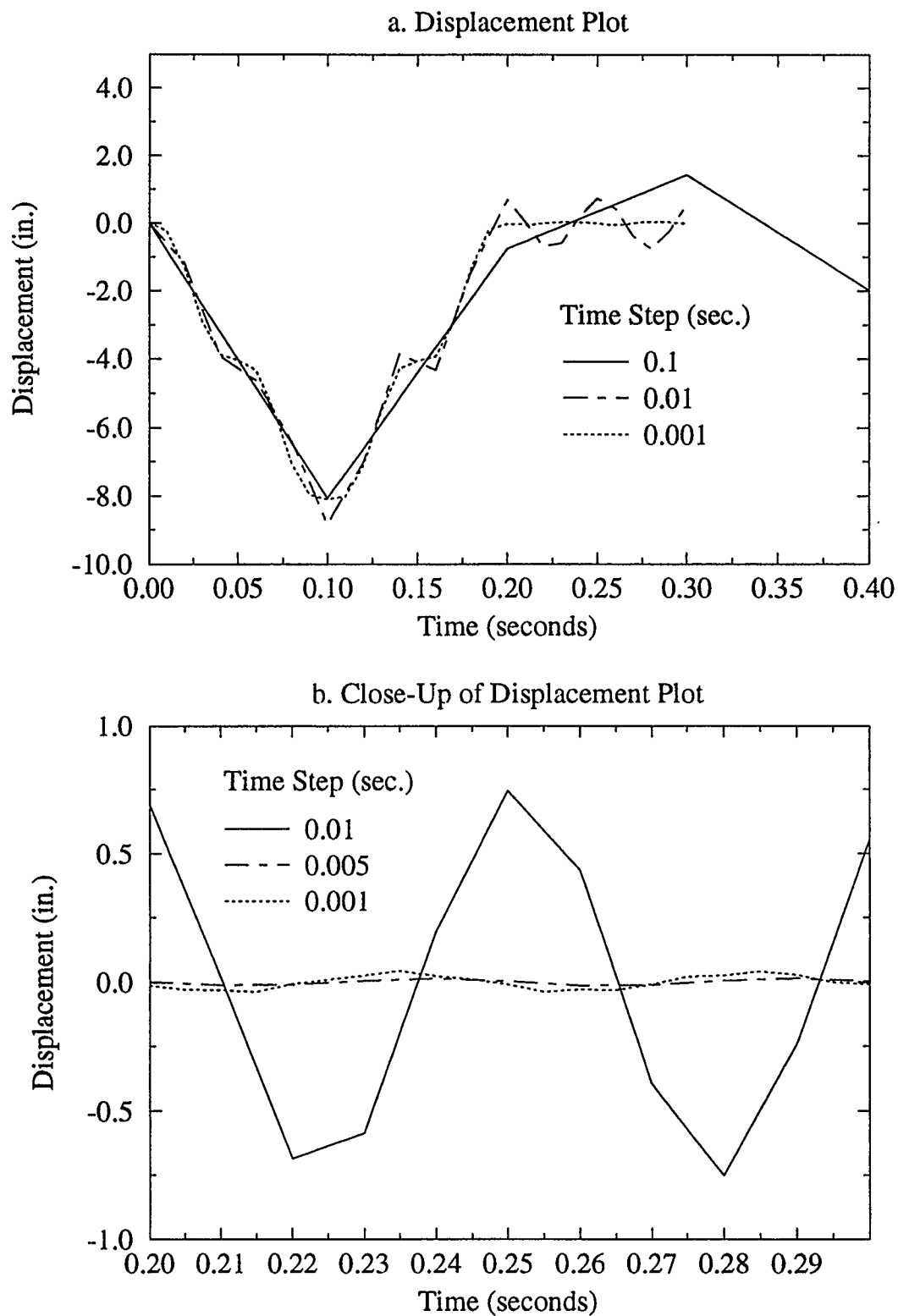


FIGURE 5.9 SENSITIVITY OF TIME INCREMENT ON THE NEWMARK SCHEME

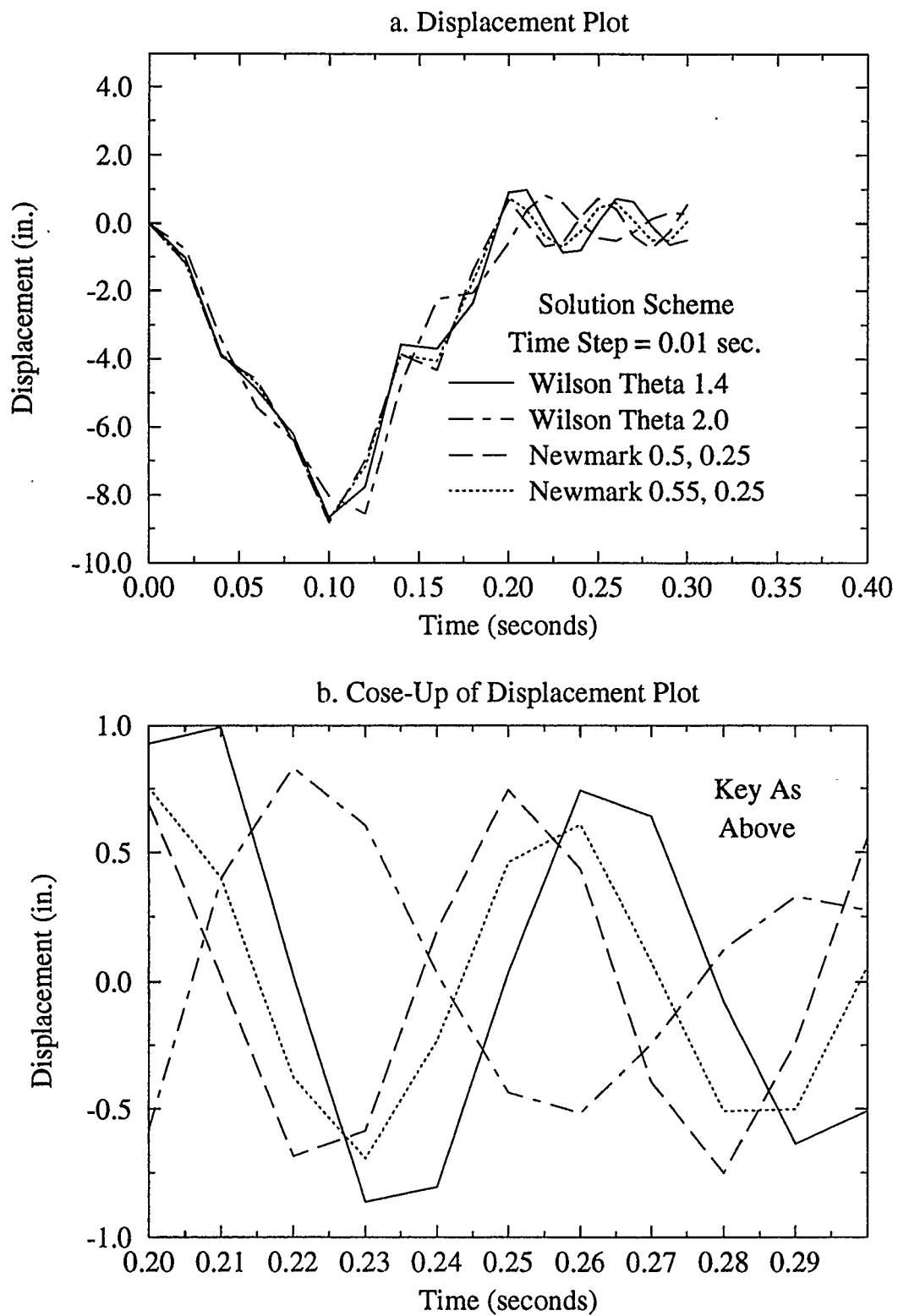


FIGURE 5.10

COMPARISON OF SOLUTION SCHEMES WITH DIFFERENT
PARAMETER VALUES

scheme re-confirms the conclusion that there is zero amplitude decay, this was pointed out in section 3.3.

5.2 NON-LINEAR DYNAMIC STUDY

A cantilever problem was also used to examine the non-linear behaviour of the approximation methods and the Modified Newton iterative procedure used by Nonsap. The model was subjected to material non-linearity using the Von Mises yield criterion model. The Von Mises model defines the plastic strain increments in relation to a yield surface, that is the point at which an elastic stress increment which exceeds the yield criterion becomes a plastic strain, see Figure 5.11 a.. The plasticity causes the internal forces of the material to behave non-linearly and therefore the dynamics of the structure are altered. Plastic deformation is non-recoverable so when the structural force is unloaded as in the case of the oscillating cantilever, the strain does not return to zero and when reloaded the strain increases in a linear elastic fashion. However if the load is increased beyond the previous maximum stress the structure experiences further plastic strain. Strain hardening allows the yield point stress to increase with plasticity. Figure 5.11 b. demonstrates the stress strain relationship discussed. For further reading refer to Zienkiewicz and Taylor (1991) and Spencer (1968).

The linear cantilever model with length reduced to 16"×1"×1" and a forcing function at the beam free end was selected; see Figure 5.12 for cantilever model and 5.13 for the forcing function. The degree of plasticity was controlled to prevent the beam failure and to preserve the numerical behaviour of the solution schemes. The following were prescribed values; Young's modulus $E = 30E+6 \text{ lb in}^{-2}$, Poissons ratio $\nu = 0.3$ and density $\rho = 0.00074 \text{ lb-sec}^2\text{in}^{-4}$. For the non-linear bilinear isotropic model a value for strain hardening modulus $E_t = 3E+6 \text{ lb in}^{-2}$, and a yield stress in simple tension $\sigma_y = 10E+3 \text{ lb in}^{-2}$ were applied. The fundamental frequencies and periods for the cantilever problem using a lumped mass matrix are listed in Table 5.3. Time steps used in the following cases are shown in Table 5.4; it should be remembered that a time step no greater than

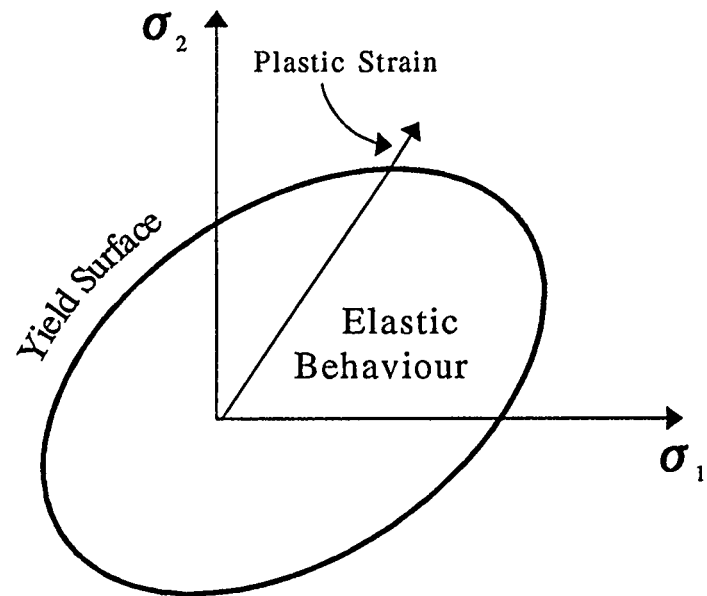


FIGURE 5.11 a. VON MISES YIELD MODEL

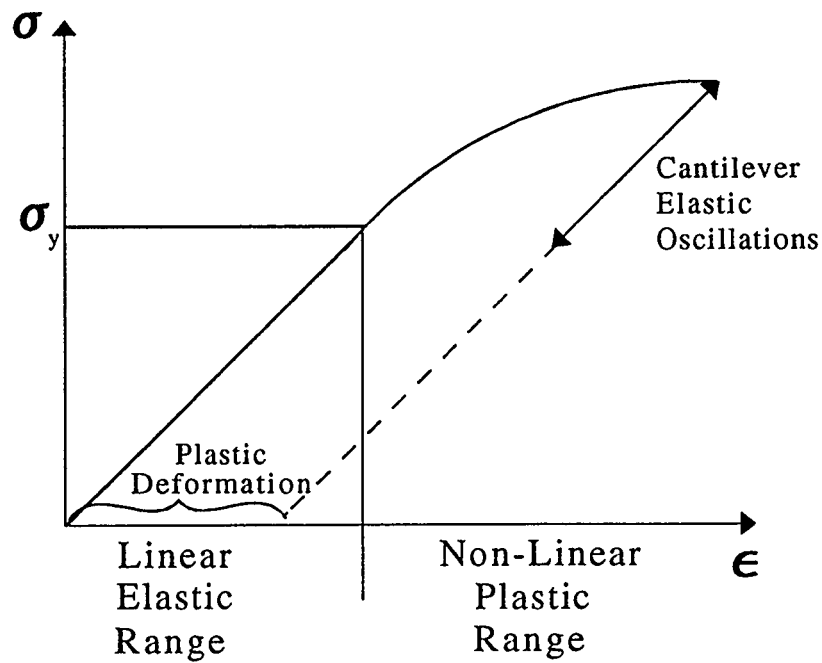


FIGURE 5.11 b. STRESS-STRAIN RELATIONSHIP

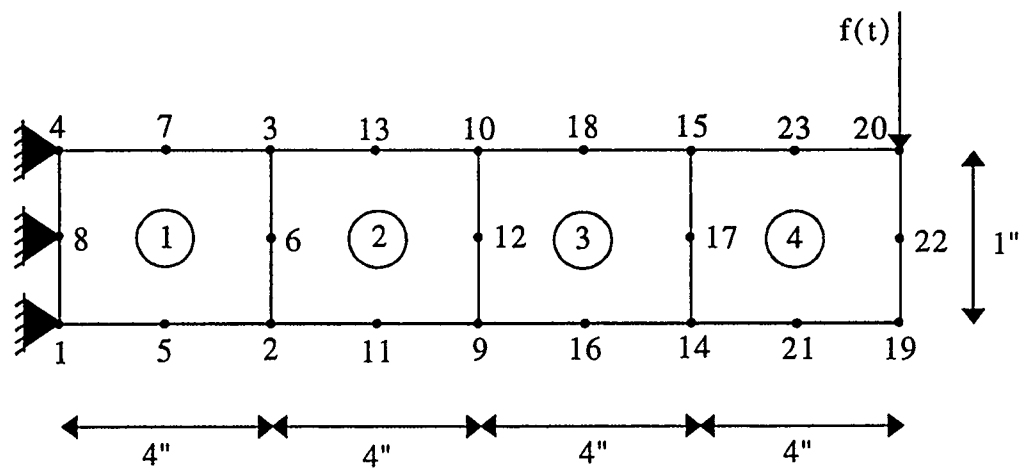


FIGURE 5.12 NON-LINEAR CANTILEVER MODEL

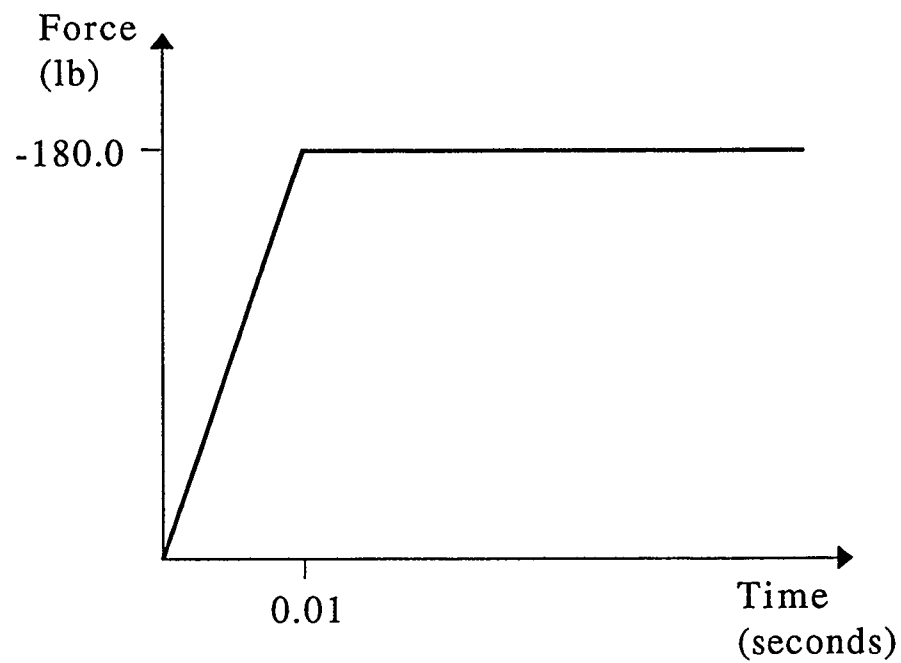


FIGURE 5.13 FORCING FUNCTION

Frequency Number	Frequency, f (rad/sec)	Frequency, ω (cycles/sec)	Period, T (seconds)
1	796.1	126.7	0.007892
2	4870.0	775.0	0.001290
3	13660.0	2175.0	0.0004599

TABLE 5.3 FREQUENCY AND PERIOD DATA FOR THE NON-LINEAR CANTILEVER PROBLEM USING A LUMPED MASS MATRIX

Δt (sec.)	$\Delta t/T_1$	$\Delta t/T_2$	$\Delta t/T_3$	$T_1/\Delta t$
0.002	0.25342	1.55039	4.34877	3.946
0.001	0.12671	0.77519	2.17439	7.892
0.0005	0.06336	0.38760	1.08719	15.784
0.00025	0.03168	0.19380	0.54360	31.568
0.000125	0.01584	0.09690	0.27180	63.136

TABLE 5.4 TIME INCREMENT SIZE AND PERIOD DATA

$T_1/10$ is recommended, T_1 is the period of the primary mode.

The force curve selected had an increasing ramp load which caused yielding in the model in approximately the same time span as the primary period of the structure. The load was then held constant, consequently structural oscillations at a frequency close to the structure's natural frequency occurred.

5.2.1 VERIFICATION OF NEW SSPj SCHEMES

To test the new version of Nonsap the original Wilson θ and Newmark schemes were run and the results compared with the corresponding SSPj schemes using the new SSPj source code. The Wilson θ case used had θ equal to 1.4, and the special Trapezium form of the Newmark scheme was run. For the equivalent SSPj parameters see Table 3.3. The displacement results for time step Δt equal to $5.0E-4$ seconds are illustrated in Figure 5.14 a. and b..

Initially all the schemes analyzed used a Modified Newton iteration convergence tolerance of 0.005 (100 iterations were allowed), the displacement plot is shown in Figure 5.14 a.. The plot indicated that the Wilson θ and the Trapezium schemes had comparable results, however the equivalent SSPj methods produced displacement plots which were severely damped and elongated, suggesting that the structure was experiencing different stress levels. It was concluded that the SSPj iterative method was less sensitive than the original Wilson θ and Newmark iteration process which iterates using the incremental displacement rather than the total $\alpha_n^{(p)}$. If sensitivity was the problem then the results would be improved by adjusting the convergence tolerance; hence an investigation of the tolerance was carried out. The final tolerance selected was reduced to 0.001, see the displacement plot in Figure 5.14 b.. An element of confidence in the accuracy of the results was achieved, the Wilson θ and SS32 schemes were almost identical and the Trapezium and SS22 schemes produced results with only a small margin of variation. The tolerance could have been reduced further to get even better results but this would have resulted in a significant increase in solution time due to the added number of

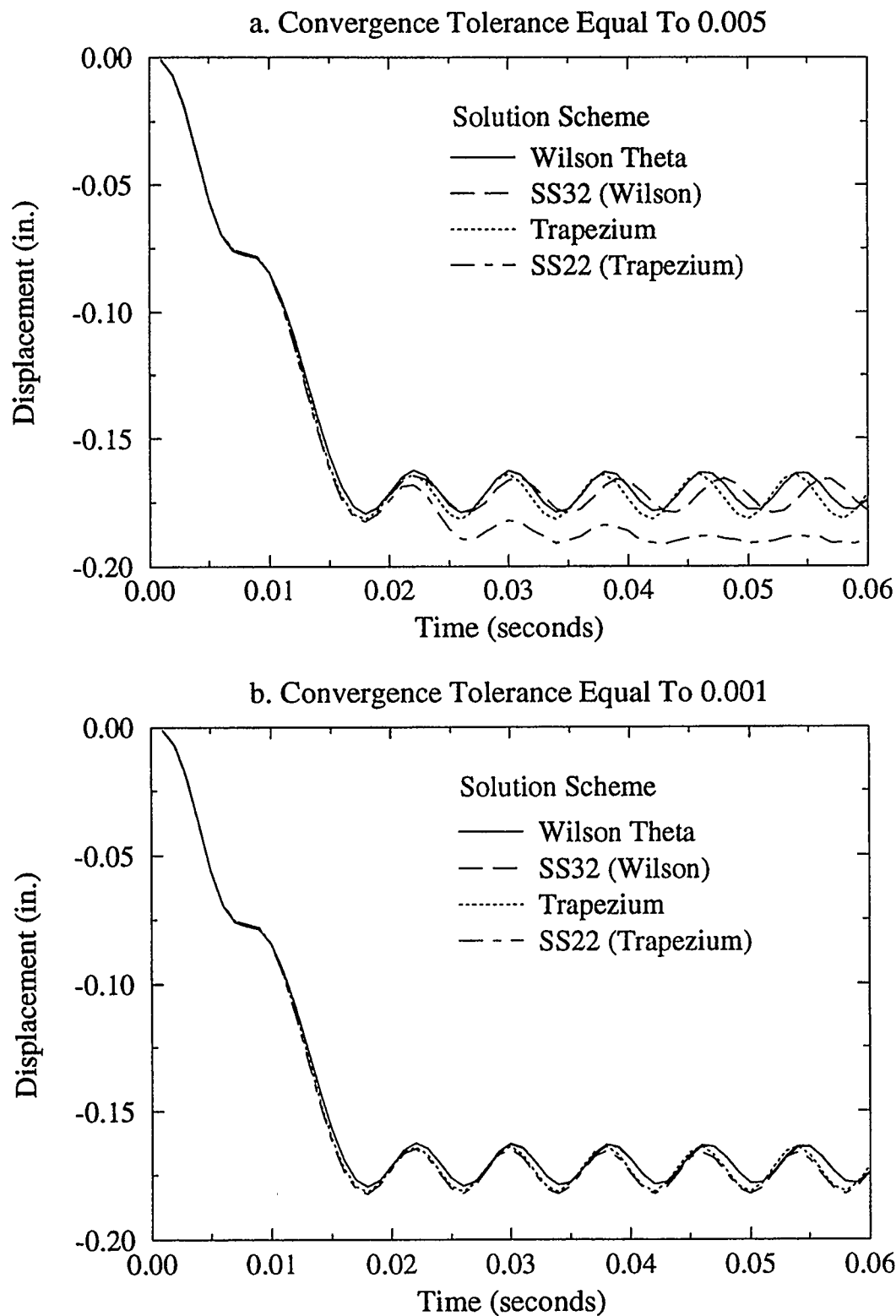


FIGURE 5.14 COMPARISON PLOTS OF THE WILSON θ AND NEWMARK SCHEMES WITH THE NEW SS_{pj} SCHEMES

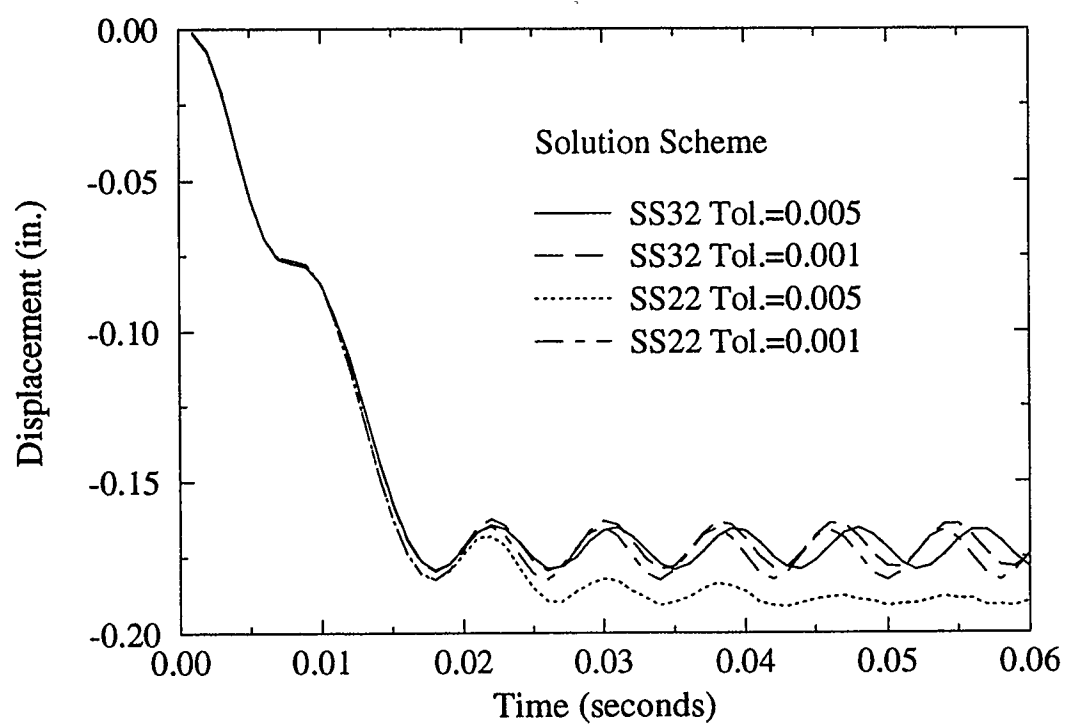


FIGURE 5.15 COMPARISON PLOTS OF THE SS_{pj} SCHEMES WITH DIFFERENT ITERATION TOLERANCES

iterations, and hence would be counter-productive.

Figure 5.15 demonstrates the improvement in the predicted displacements for the SSpj schemes by decreasing the tolerance for the iteration scheme from 0.005 to 0.001. The improvement is especially marked for the SS22 form of the Trapezium method which was over damping. This over damping was due to inaccurate calculation of the stresses which produced plastic behaviour where the other schemes were elastic.

Two studies were performed, they examined the effect of time increment size and solution scheme on the total solution time and displacement history accuracy. In some cases the stresses were also examined as they helped to give a better picture of the dynamics which the schemes predicted. For all the following tests the convergence tolerance was fixed at 0.001, and the SSpj forms of the Wilson θ and Newmark schemes were used.

5.2.2 STUDY OF TIME INCREMENT SIZE

The first set of results illustrated in Figure 5.16 a. show the displacements at various time steps using the Wilson θ method with parameter θ equal to 1.4. The time increments ranged from 2.0E-3 down to 1.25E-4 seconds, these sizes were selected because they represent approximately one quarter down to one sixty-fourth of the primary period. A tenth of the first two fundamental periods also falls within this range, see Table 5.4.

The displacement plot for time increment equal to 2.0E-3 seconds is cut short at time 0.018 seconds due to a diverging iterative procedure. It is not surprising that the numerical behaviour breaks down at this time step since it corresponds to approximately a quarter of the primary period, this is a fairly coarse time step. This conclusion is consistent with the discussion in section 3.3 where it was found that the program dynamic.c could not accurately predict displacements for time steps greater than a tenth of the primary period.

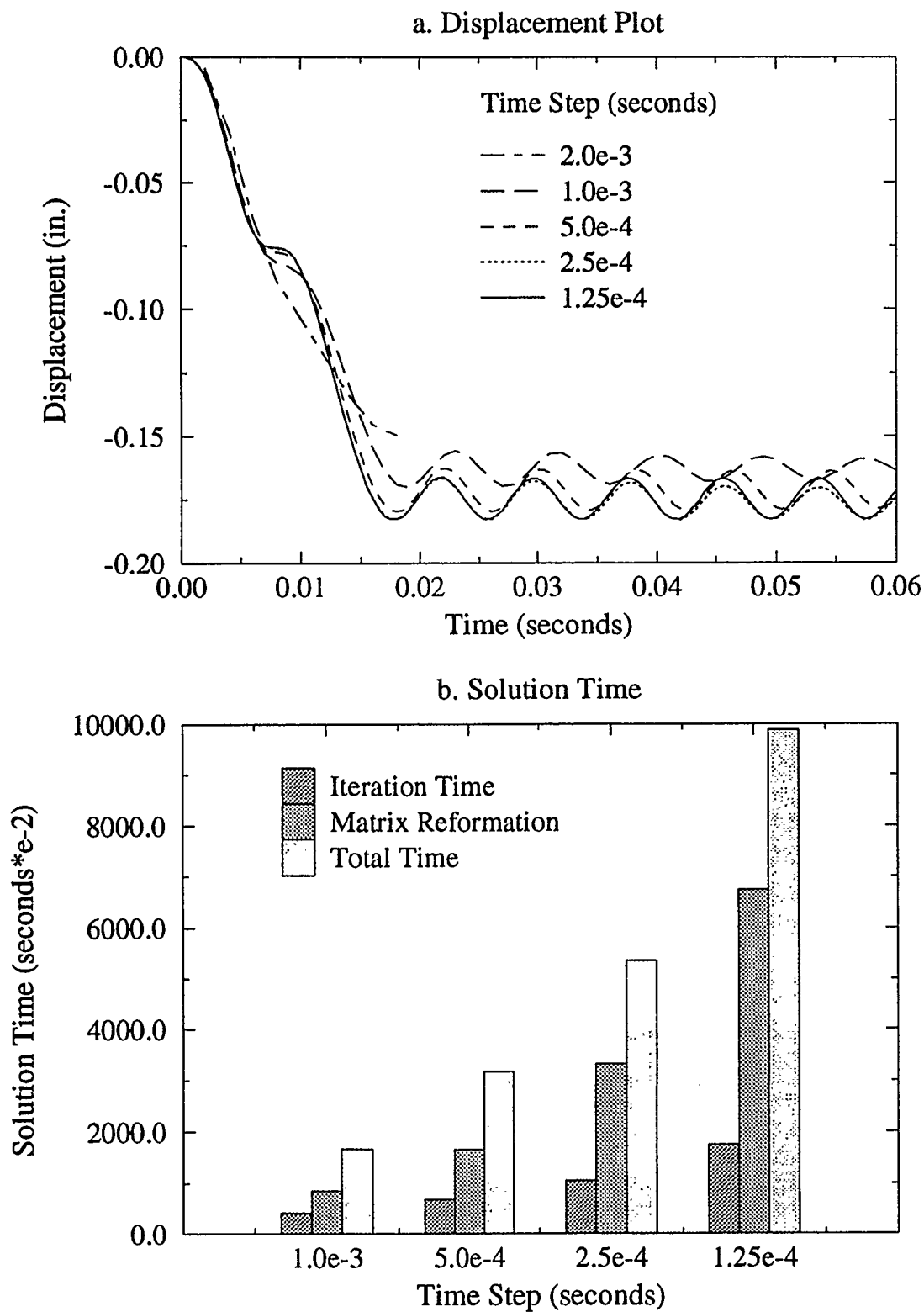


FIGURE 5.16

COMPARISON PLOTS OF THE WILSON θ SCHEME WITH
DIFFERENT TIME STEP SIZES

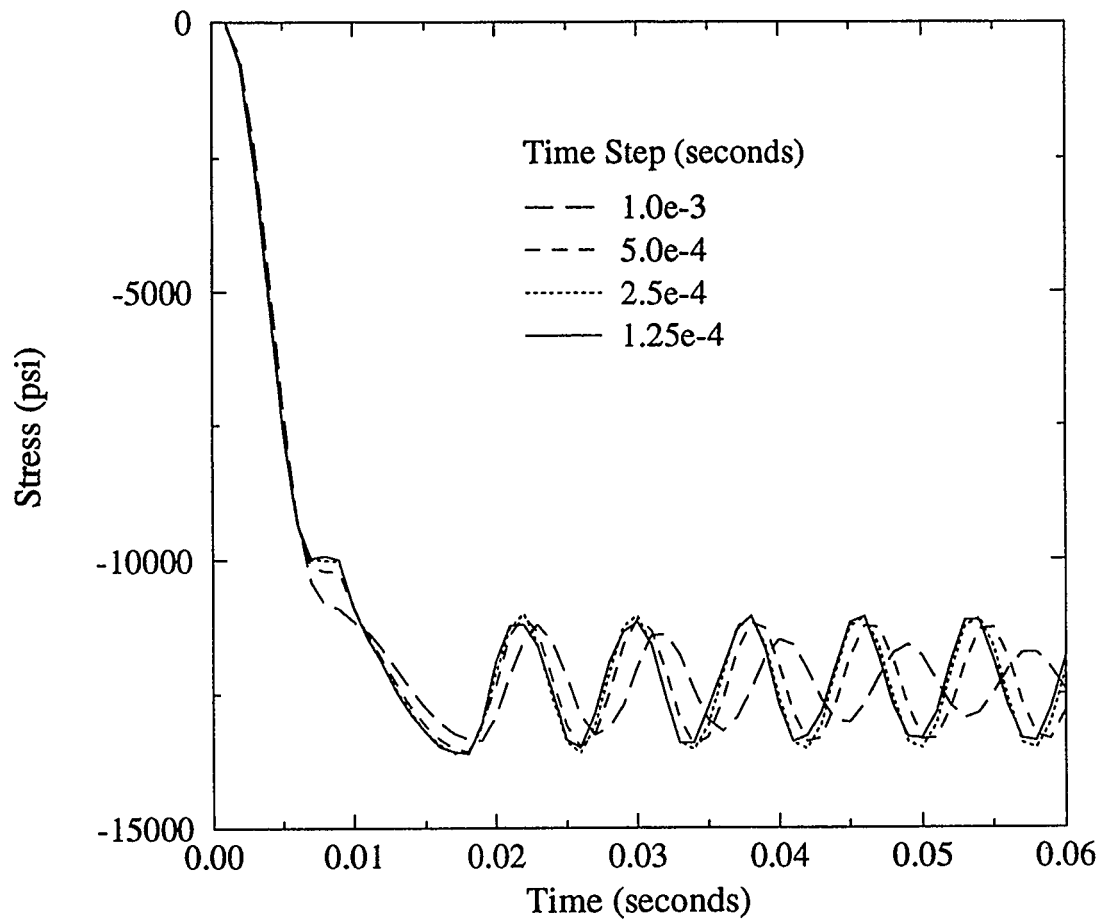


FIGURE 5.17

COMPARISON STRESS PLOT OF THE WILSON θ SCHEME
WITH DIFFERENT TIME STEP SIZES

It is clear that the results slowly converge as the time step is reduced. Note that time steps of $2.5\text{E-}4$ and $1.25\text{E-}4$ seconds show minor divergence in their displacement history, therefore reducing the step further would probably yield insignificant improvements in accuracy. The larger time steps display more amplitude decay resulting in a conservative estimate of the maximum displacement. The displacement plots begin to go out of phase during the plastic deformation of the structure approximately between 0.01 and 0.019 seconds depending on the time step used (in this case $\Delta t = 1.0\text{E-}3$ seconds). The larger time increments under-estimate the stress level during plasticity but remain in the plastic mode for a slightly longer period of time, see Figure 5.17. The Wilson θ method also shows more severe period elongation as the time step is increased.

The bar graph in Figure 5.16 b. presents the distribution and the total time taken for each case run. We are interested in the time incurred during the iterative procedure and reformation of the effective stiffness matrix and load vectors. The iterative procedure is far less draining on computer time than reforming the matrices, the procedure takes approximately twenty percent of the total time verses sixty to seventy percent for the reformations. Matrix reformation occurs every time step therefore the more time increments required, that is the smaller the time step, the greater the increase in total solution time. This trend is apparent in Figure 5.16 b., as the time step is decreased the total solution time becomes much larger. Since the significance of the iterative procedure is less it is preferable to increase the time step size and increase the number of iterations required per time step. Note that the user should be careful not to sacrifice the accuracy of the displacement history by increasing the step size too much.

Appendix C contains plots of the displacements and solution times for various other solution schemes. The conclusions made about the Wilson θ method apply to them also. One point to be made is that the Hilber, Hughes and Taylor scheme (HHT) in Figure C.3, like the Wilson θ method was capable of solving for a time step equal to $1.0\text{E-}3$ seconds, all the other methods failed during the plastic range and therefore are numerically more sensitive. The Trapezium method in Figure C.1 showed it had the least variation of

accuracy as the time increment was increased.

5.2.3 STUDY OF SOLUTION SCHEMES AND PARAMETER VALUES

The final set of tests examines the influence of the solution schemes on the displacement accuracy and the total solution times. Five of the solution schemes previously examined in section 3.3 were compared at two different time steps, $5.0\text{E-}4$ and $1.25\text{E-}4$ seconds. The five schemes tested were; Trapezium, Bossak case 2, HHT, Houbolt and Wilson θ ($\theta = 1.4$), parameter values are listed in Table 3.3.

For the non-linear problem no exact solution is known, however no damping is introduced into the system and we also know that the trapezium method has zero amplitude decay. The trapezium method had the least period elongation for the linear analysis in section 3.3 when the time increment was one sixteenth of period T_1 . Therefore the following results will be compared against the trapezium method assuming that it yields the best solution.

Figure 5.18 illustrates the displacement and time graphs for the five schemes for a time increment equal to $5.0\text{E-}4$ seconds. It is clear that the Houbolt scheme has the most amplitude decay and period elongation. All the other schemes do not suffer much amplitude decay although the Bossak and Wilson θ methods show a significant degree of period elongation. The phase errors begin during the plasticity stage, the corresponding stress plot in Figure 5.19 also shows variations of stresses during the plasticity range, therefore indicating errors. The Houbolt scheme had the most severely damped stress plot and hence the poor representation of displacement history in Figure 5.18 a.. Both the Trapezium and HHT methods produce roughly identical displacement and stress plots. The displacement plot for time step equal to $1.25\text{E-}4$ seconds, in Figure 5.20 a., shows that all the schemes have converged. The Houbolt method was the only scheme which shows period elongation, however this elongation error is very small.

The bar graphs in Figures 5.18 b. and 5.20 b. show that there is little difference in total time between all the solution schemes, therefore it is concluded that the times taken are not solution dependant. This conclusion also applies to the time taken for the iterations and the matrix reformations.

The final test run examined the effect of varying the parameters of the Wilson θ and Newmark schemes, see Figure 5.21. The displacement plot shows that the Wilson θ scheme, with θ equal to 2.0, has severe amplitude decay and period elongation in comparison to the special trapezium form of the Newmark scheme. The other Wilson θ method and the Newmark scheme with δ equal to 0.55 and α equal to 0.3 have similar amplitude decay, but the Wilson θ method shows signs of greater period elongation. These findings are consistent with those discussed in section 5.1.2 for the linear cases of these solution schemes. The solution time bar graph also confirms the conclusion that solution time is independent of solution scheme.

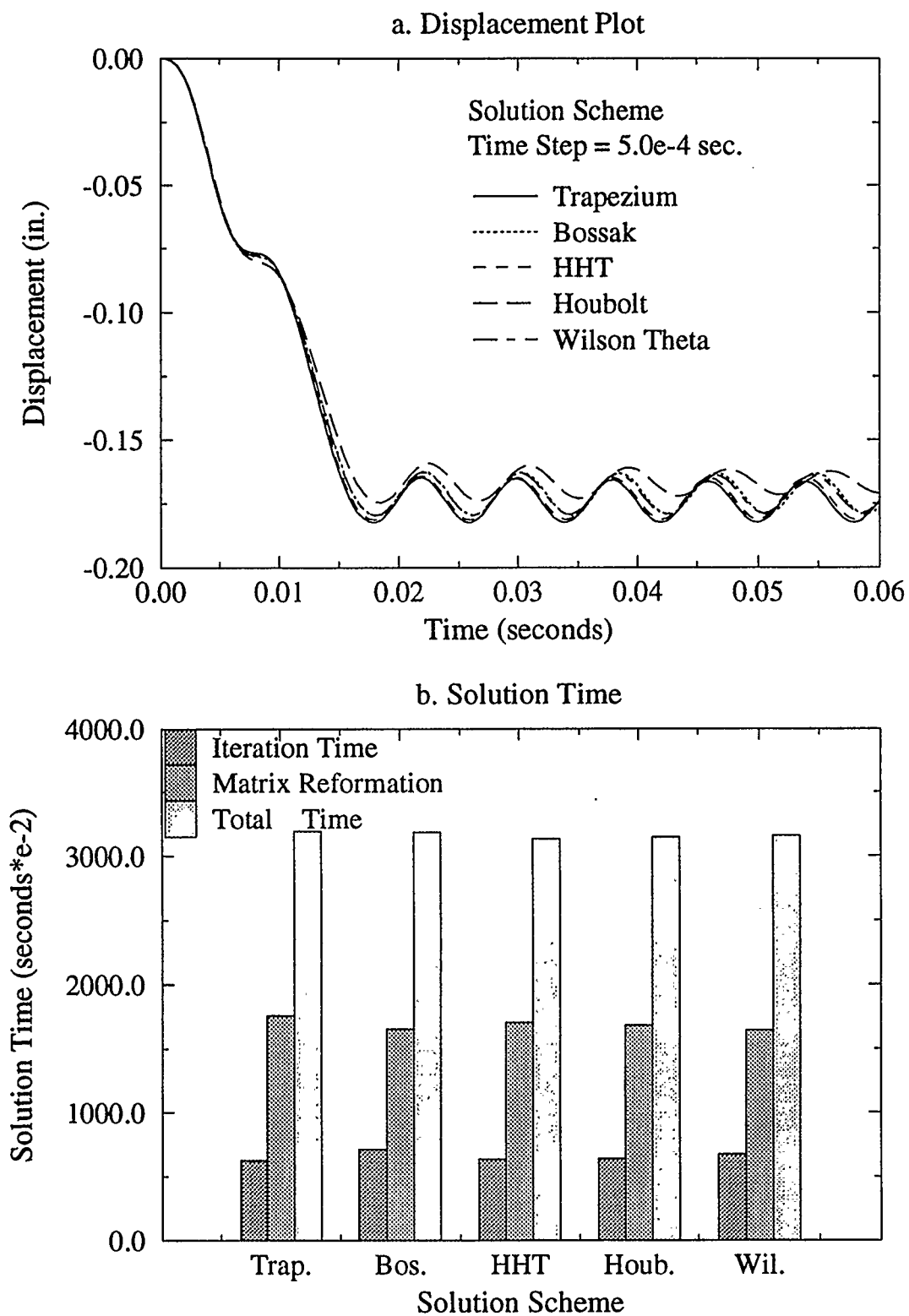


FIGURE 5.18

COMPARISON PLOTS OF DIFFERENT SOLUTION SCHEMES
WITH TIME STEP EQUAL TO 5.0E-4 SECONDS

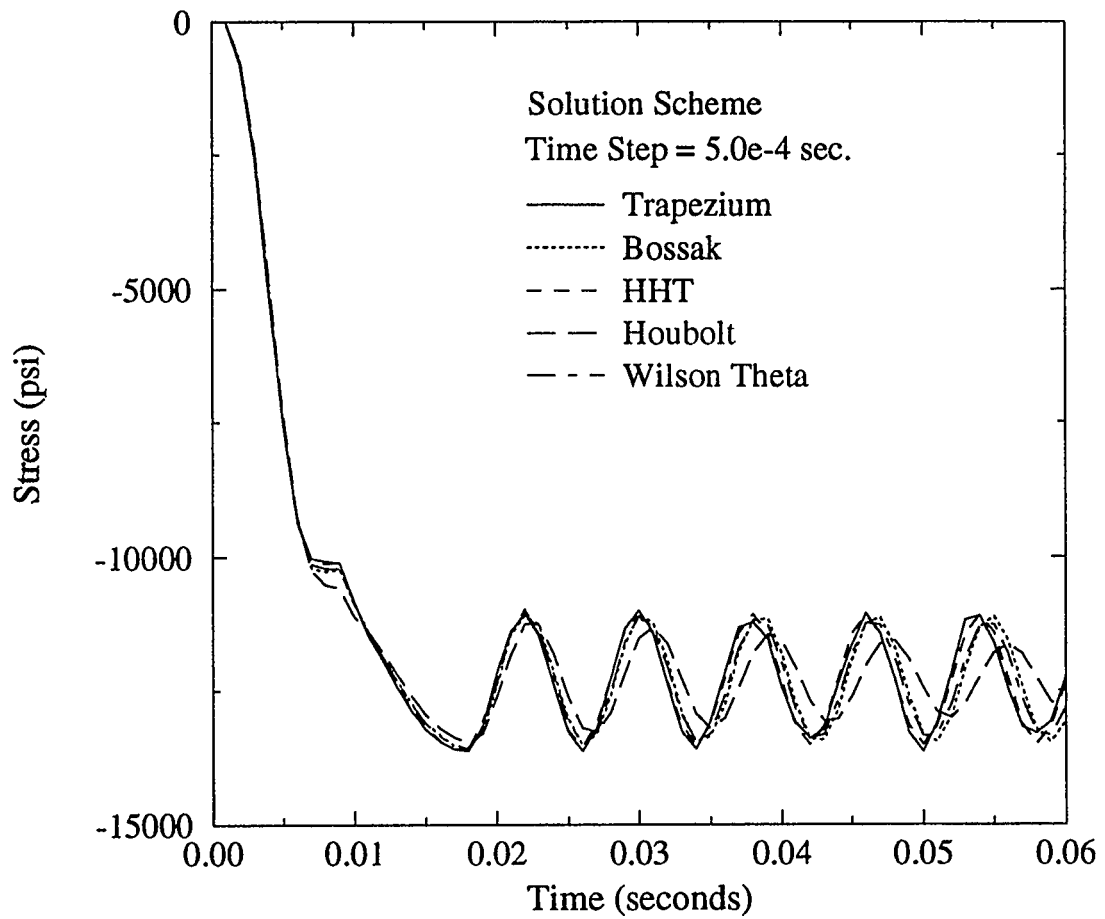


FIGURE 5.19

COMPARISON STRESS PLOT OF DIFFERENT SOLUTION
SCHEMES WITH TIME STEP EQUAL TO 5.0×10^{-4} SECONDS

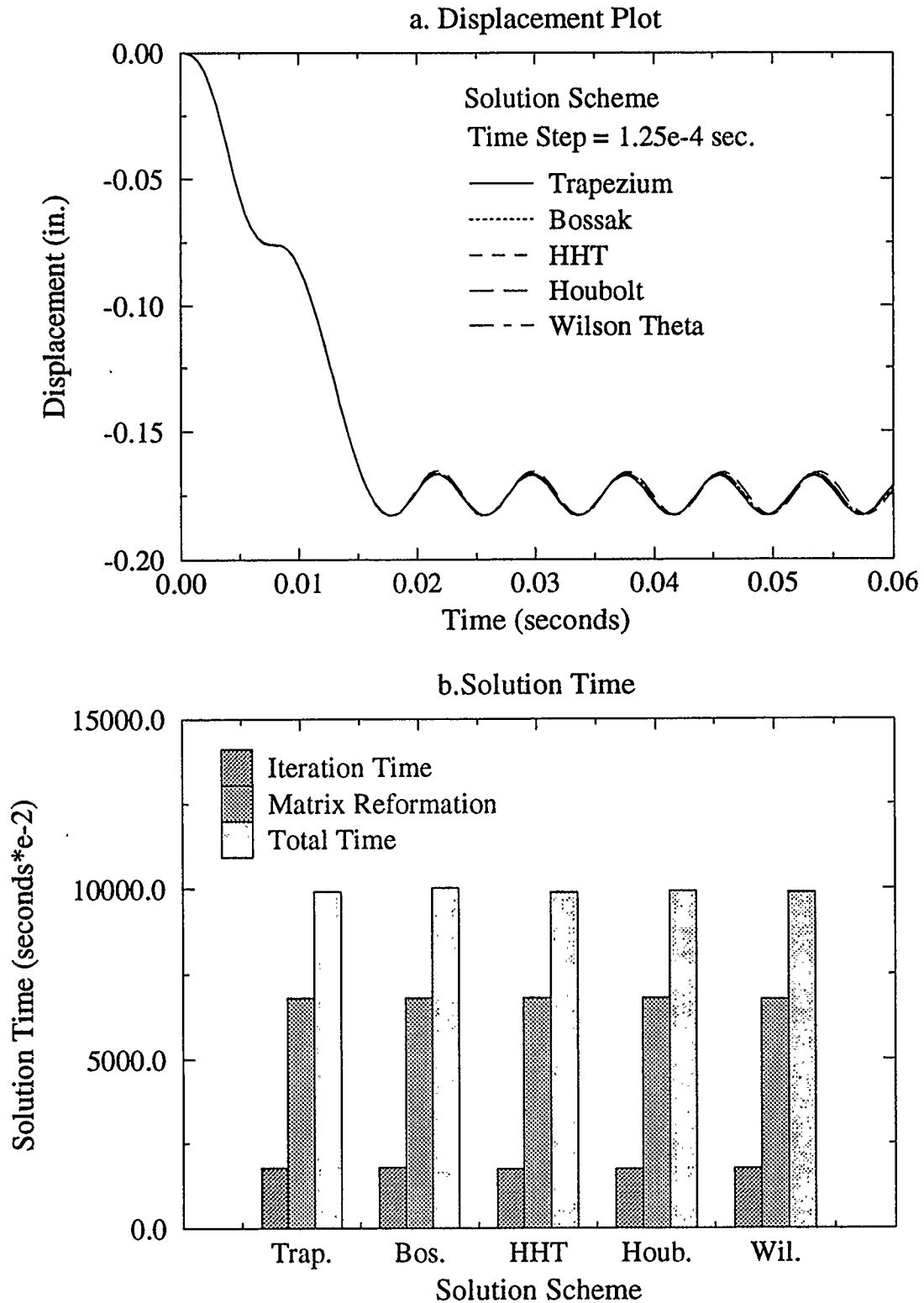


FIGURE 5.20

COMPARISON PLOTS OF DIFFERENT SOLUTION SCHEMES
WITH TIME STEP EQUAL TO 1.25E-4 SECONDS

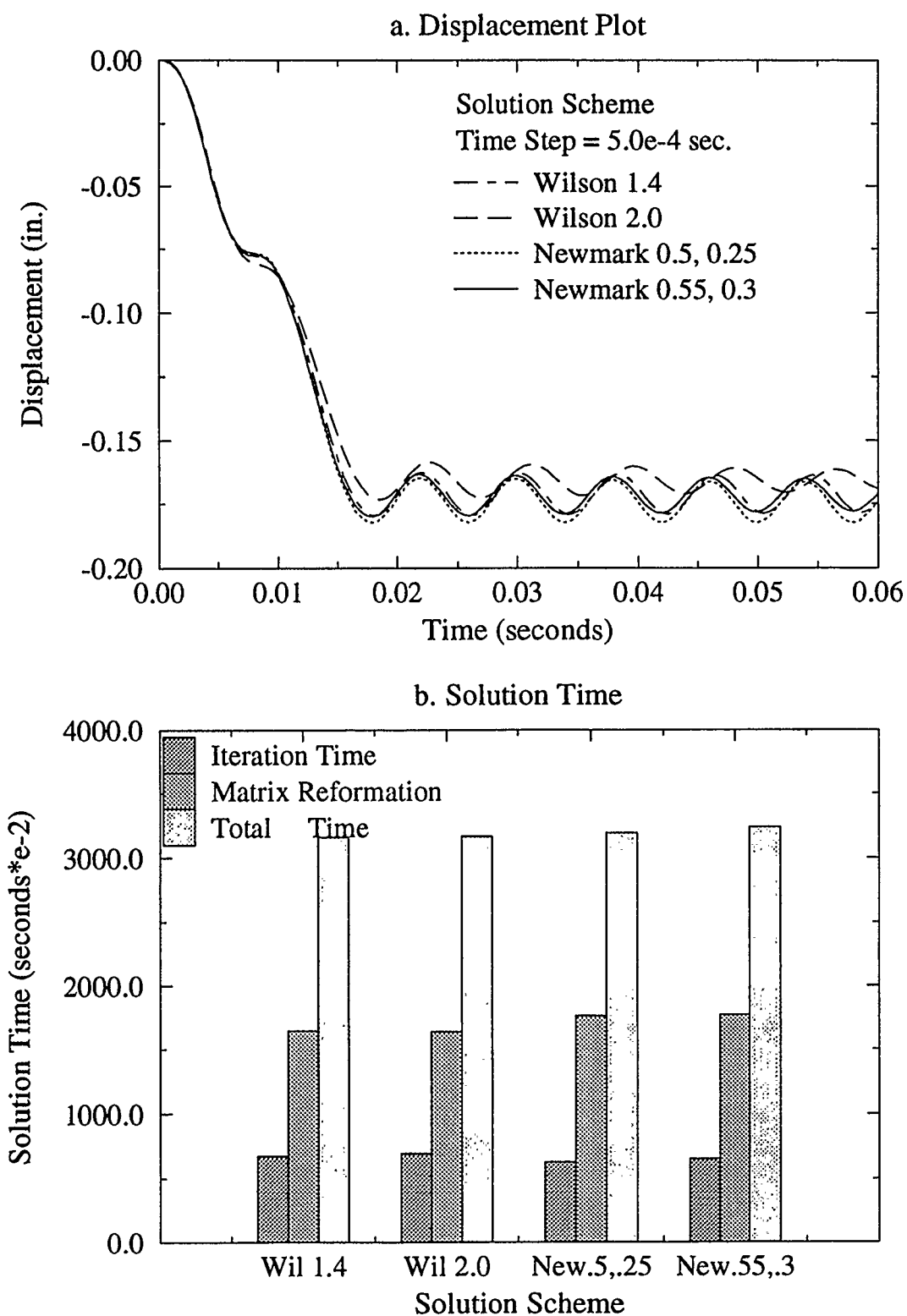


FIGURE 5.21

COMPARISON OF DIFFERENT PARAMETER VALUES FOR
WILSON θ AND NEWMARK SCHEMES, Δt 5.0E-4 SECONDS

CHAPTER SIX

CONCLUSIONS AND RECOMMENDATIONS

6.1 CONCLUSIONS

The conclusions made about the various approximation methods and how they behave for a range of time step sizes are based on the examples in this thesis. However these conclusions are consistent with results published by other authors for linear dynamic problems.

The multi-step finite difference approximations for the dynamic equation of equilibrium require knowledge of the displacement and force history over a number of time steps. When writing the first version of program `dynamic.c` difficulties arose while making assumptions about the displacement and force history prior to time step one. During the investigation of the single-step procedures it was found that information about the kinematics and forces were only required for the previous time step and therefore less assumptions about the system were made. The general single-step approximations also made testing of a large number of possible solution methods available and hence for our research purposes it was concluded that the analyses should be carried out using the SSpj scheme. Note that the SSpj approximation is essentially the same as the GNpj algorithm and therefore it was only necessary to study one of the methods.

In Chapter Two various procedures were used to investigate the accuracy and stability of approximation methods. Lambert's theorem (1973) highlighted the importance of consistency and zero-stability for the schemes as they determine the magnitude of the truncation error and its propagational behaviour. The Routh-Herwitz stability criterion was also introduced. Table 3.2 lists the conditions that the SSpj schemes must satisfy in order to meet the stability and accuracy requirements. Table 3.3 shows the corresponding conditions for a selection of solution cases. The SSpj scheme is always consistent but unconditional stability will only apply for some of the prescribed parameter values. For

example the Fox-Goodwin and Central Difference cases are only stable for a bounded range of time increment size where they satisfy the Routh-Herwitz criterion.

For additional information on the stability and accuracy of approximation schemes, methods of measuring spectral radius, and percentage amplitude decay and period elongation were examined. This analytical approach was more useful than the mathematical developments discussed in the last paragraph as a greater understanding of each solution case's behaviour for a range of time step sizes was achieved. This kind of information is useful when examining a model which has a wide range of frequency modes. For example it maybe desirable to filter out higher modes of oscillation with amplitude decay and therefore knowledge of the decay behaviour over a range of time step sizes will be required. The spectral radius was of particular interest as it identified those schemes which were conditionally stable and at what point the time step size caused instability.

It was concluded that the spectral radius and the Routh-Herwitz criteria were consistent in their identification of solution schemes with conditional stability, and for the time step size where the instability begins. Both procedures were in agreement when pinpointing the schemes that have zero amplitude decay. The truncation error indicated which schemes were the most accurate but did not specify if the accuracy applied to the period elongation or the amplitude decay and to what order the accuracy was. Therefore it is not of much benefit in comparison to the information that the percentage amplitude decay and period elongation can provide.

The forcing functions used in the analyses using Nonsap were selected to produce an oscillating displacement response. In the linear study a simple triangular function was adequate in producing free oscillations which easily highlighted accuracy problems. The non-linear analysis was more sensitive to the forcing function. Too much load in a time span close to the natural frequency of the structure or a severe change in the load, for example the apex of a triangular forcing function, caused the Modified Newton iterations

to diverge resulting in failure of the solution. If the load was not large enough the oscillations were too small making it impossible to see any amplitude decay or period elongation in the displacements. The conclusion is that when selecting a forcing function to study approximation schemes careful consideration of the structures dimensions and natural frequency must be made first.

The Modified Newton iteration convergence tolerance size is very important if good displacement results are to be achieved. The SSpj schemes were less sensitive than the Wilson θ and Newmark schemes so when selecting a tolerance value this must be taken into consideration. The smaller the tolerance value the more accurate the displacement results, however there is an increase in solution time due to an increase in the number of iterations.

In Chapter Three `dynamic.c` was used to determine the time increment size where the displacements were no longer satisfactory. According to the Routh-Herwitz criteria and spectral radius the Trapezium scheme has zero amplitude decay but when the time step was greater than a tenth of the period, amplitude decay was observed. In both the linear and non-linear analyses using Nonsap the same conclusion was deduced. For all the schemes tested the results were very poor when the time step was larger than a tenth of the fundamental period. Bathe and Wilson (1973) also observed this behaviour and they suggest using a smaller time step than the period divided by ten.

All the analyses showed that reduction of the time step decreased errors in accuracy. The linear study showed that the amplitude decay and period elongation for all the solution schemes tended towards zero as the time increment decreased, and schemes with conditional stability behaved well with small Δt 's. For the non-linear analysis displacement plots converged for all the schemes tested with diminishing time step. Some schemes failed when the time step was too large; this was due to the iterative procedure diverging.

The solution time of the non-linear problems increased dramatically when the time step was reduced. For all the cases matrix reformation and solution of the simultaneous equations were more draining on time than the iterative procedure, therefore to reduce solution time the time step should be made as large as possible but not to the extent that the accuracy of the solution is compromised.

The selection of solution scheme and parameter size is very important. The linear and non-linear studies indicated significant variations in displacement for different test cases. Schemes which showed greater accuracy in the linear study also excelled in the non-linear problems, for example, when the time step size was kept below a tenth of the primary period, the Trapezium scheme performed well. The trapezium method experienced the least period elongation and zero amplitude decay for the non-linear model, it was also the most accurate un-conditionally stable scheme for the linear tests. The Hilber, Hughes and Taylor method (HHT) was another un-conditionally stable scheme, but with some amplitude decay for both the linear and non-linear tests, this amplitude decay may be desirable for some problems. The Houbolt and Wilson θ scheme with θ equal to two, performed very poorly when the time step was large. In the non-linear case for accurate results the Houbolt scheme required a time step approximately one thirty-second of the primary period. All these results lead to the conclusion that the schemes behave in a similar manner whether the problem is linear or non-linear.

Hilber *et. al.* (1977) stated that the HHT approximation was developed to satisfy the following requirements:

1. Un-conditional stability for linear problems.
2. Control of numerical dissipation without changing time step sizes, zero dissipation should be achievable.
3. Dissipation should effect the lower frequencies as little as possible.

By making the α_H term zero the HHT method simulates the Newmark scheme. Zero

amplitude decay is achievable by reducing the HHT scheme into the special Trapezium form of the Newmark method, this is done by making β_H and γ_H equal to a quarter and a half respectively. The numerical damping is controlled by altering the values of the α_H and γ_H terms, it is for this reason that I recommend the use of the HHT approximation. The linear and non-linear examples showed that the Trapezium method was the best scheme if zero amplitude decay is required, and as mentioned we can simulate the Trapezium method with the HHT approximation. When damping of the high frequencies is necessary the HHT example in chapters three and five behaved the best. This HHT case had the least amplitude decay and period elongation for small time step sizes resulting in accurate calculation of the displacements for the lower frequencies; for large time step sizes the desirable amplitude decay was introduced.

Solution time was marginally effected by the scheme selected, therefore it was deduced that solution time is scheme independent.

6.2 RECOMMENDATIONS

The iterative procedure used in Nonsap was the Modified Newton method. This procedure is more computer efficient than the Newton-Raphson method but it does not converge as quickly. The Secant method converges almost as quickly as the Newton-Raphson procedure but is much more efficient. Various other Quasi-Newton iterative procedures have been developed which converge even more effectively than the Secant method. It is recommended that some of these Quasi-Newton iterative procedures be investigated further. With their introduction into Nonsap a study of these iterative procedures and their combined effect with the SSpj approximation methods on solution times would be very interesting.

Zienkiewicz and Xie (1991) and Zeng *et. al.* (1992) develop the idea of adaptive time stepping. This procedure adjusts time step sizes throughout the solution process to reduce computational errors. An error estimator predicts the optimal step size from time to time

to reduce the local error incurred at each time step within a prescribed tolerance. They reported that the error estimator introduces only a small increase in computational cost. The adaptive time stepping procedure could add considerable improvements to the accuracy of the displacements particularly for non-linear problems. It is suggested that the addition of such error estimators to the non-linear investigation be done to see their significance in terms of accuracy improvements, and computational cost.

6.3 COMMENTS

From the testing documented in this thesis it can be seen that the size of the time increment is very important when modelling vibrational deflection. Structures have differing natural frequencies therefore it is a good idea to determine the structures period of oscillation in order to calculate a good estimate of Δt by taking at minimum a tenth of the primary period.

Solution scheme and parameter size also play an important role in the accuracy of the displacements. With a little investigation an optimum solution scheme with prescribed parameter sizes can be selected for the required range of time step sizes.

BIBLIOGRAPHY

Bathe, K.J., Wilson, E.L. (1973) *Stability and Accuracy Analysis of Direct Integration Methods*, Earthquake Engineering and Structural Dynamics, Vol.1, pp.283-291

Bathe, K.J., Wilson, E.L., Iding, R.H. (1974) *Nonsap a Structural Analysis Program for Static and Dynamic Response of Non-Linear Systems*, University of California, Berkeley, U.S.A., Report Number UCSESM 74-3

Bathe, K.J., Wilson, E.L. (1976) *Numerical Methods in Finite Element Analysis*, Prentice-Hall, Inc., U.S.A.

Chan, S.P., Cox, H.L., Benfield, W.A. (1969) *Transient Analysis of Forced Vibrations of Complex Structural-Mechanical Systems*, The Royal Aeronautical Society Journal, Vol.66, pp.457-460

Clough, R.W., Penzien, J. (1993) *Dynamics of Structures, Second Edition*, McGraw-Hill Book Company, U.S.A.

Froberg, C.-E. (1969) *Introduction to Numerical Analysis, Second Edition*, Addison-Wesley Publishing Company, U.S.A.

Gantmacher, F.R. (1960) *The Theory of Matrices, Vol. 2*, Chelsea Publishing Company, New York, U.S.A.

Gerald, C.F., Wheatley, P.O. (1984) *Applied Numerical Analysis, Third Edition*, Addison-Wesley Publishing Company, U.S.A.

Hilber, H.M., Hughes, T.J.R., Taylor, R.L. (1977) *Improved Numerical Dissipation for Time Integration Algorithms in Structural Dynamics*, Earthquake Engineering and Structural Dynamics, Vol.5, pp.283-292

Hilber, H.M., Hughes, T.J.R. (1978) *Collocation, Dissipation and 'Overshoot' for Time Integration Schemes in Structural Dynamics*, Earthquake Engineering and Structural Dynamics, Vol.6, pp.99-117

Katona, M.G., Zienkiewicz, O.C. (1985) *A Unified Set of Single Step Algorithms, Part 3: The Beta-m Method, a Generalization of the Newmark Scheme*, International Journal for Numerical Methods in Engineering, Vol.21, pp.1345-1359

Lambert, J.D. (1973) *Computational Methods in Ordinary Differential Equations*, Wiley, London, U.K.

Matthies, H., Strang, G. (1979) *The Solution of Non-Linear Finite Element Equations*, International Journal for Numerical Methods in Engineering, Vol.14, pp.1613-1626

Nayak, G.C., Zienkiewicz, O.C. (1972) *Note on the 'Alpha'-Constant Stiffness Method for the Analysis of Non-Linear Problems*, International Journal for Numerical Methods in Engineering, Vol.4, pp.579-582

Owen, D.R.J., Hinton, E. (1986) *Finite Element in Plasticity, Theory and Practice*, Pineridge Press Limited, Swansea, U.K.

Spencer, G.C. (1968) *Introduction to Plasticity*, Chapman and Hall Ltd., London, U.K.

Wood, W.L. (1977) *On the Zienkiewicz Four-Time-Level Scheme for the Numerical Integration of Vibration Problems*, International Journal for Numerical Methods in Engineering, Vol.11, pp.1519-1528

Wood, W. L. (1990) *Practical Time Stepping Schemes*, Clarendon Press, Oxford, U.K.

Wood, W.L., Bossak, M., Zienkiewicz O.C. (1980) *An Alpha Modification of Newmark's Method*, International Journal for Numerical Methods in Engineering, Vol.15, pp.1562-1566

Zeng, L.F., Wiberg, N.-E., Li, X.D., Xie, Y.M. (1992) *A Posteriori Local Error Estimation and Adaptive Time-Stepping for Newmark Integration in Dynamic Analysis*, Earthquake Engineering and Structural Dynamics, Vol.21, pp.555-571

Zienkiewicz, O.C. (1977) *A New Look at the Newmark, Houbolt and Other Time Stepping Formulas. A Weighted Residual Approach*, Earthquake Engineering and Structural Dynamics, Vol.5, pp.413-418

Zienkiewicz, O.C., Wood, W.L., Hine, N.W., Taylor, R.L. (1984) *A Unified Set of Single Step Algorithms, Part 1: General Formulation and Applications*, International Journal for Numerical Methods in Engineering, Vol.20, pp.1529-1552

Zienkiewicz, O.C., Taylor, R.L. (1991) *The Finite Element Method Fourth Edition, Volume 2 Solid and Fluid Mechanics, Dynamics and Non-Linearity*, McGraw-Hill Book Company, London, U.K.

Zienkiewicz, O.C., Xie, Y.M. (1991) *A Simple Error Estimator and Adaptive Time Stepping Procedure for Dynamic Analysis*, Earthquake Engineering and Structural Dynamics, Vol.20, pp.871-887

APPENDIX A

A.1 TRUNCATED TAYLOR SERIES COLLOCATION

The general dynamic equation,

$$M\ddot{x} + C\dot{x} + Kx = f \quad (A1.1)$$

is written for three separate times $n-1$, n and $n+1$.

$$M\ddot{x}_{n-1} + C\dot{x}_{n-1} + Kx_{n-1} = f_{n-1} \quad (A1.2)$$

$$M\ddot{x}_n + C\dot{x}_n + Kx_n = f_n \quad (A1.3)$$

$$M\ddot{x}_{n+1} + C\dot{x}_{n+1} + Kx_{n+1} = f_{n+1} \quad (A1.4)$$

Equations (A1.2), (A1.3) and (A1.4) are multiplied by $\Delta t^2(\frac{1}{2}-\gamma+\beta)$, $\Delta t^2(\frac{1}{2}+\gamma-2\beta)$ and $\Delta t^2\beta$ respectively and are then added together.

$$\begin{aligned} & M \left[\ddot{x}_{n+1}\beta + \ddot{x}_n\left(\frac{1}{2}+\gamma-2\beta\right) + \ddot{x}_{n-1}\left(\frac{1}{2}-\gamma+\beta\right) \right] \Delta t^2 + \\ & C \left[\dot{x}_{n+1}\beta + \dot{x}_n\left(\frac{1}{2}+\gamma-2\beta\right) + \dot{x}_{n-1}\left(\frac{1}{2}-\gamma+\beta\right) \right] \Delta t^2 + \\ & K \left[x_{n+1}\beta + x_n\left(\frac{1}{2}+\gamma-2\beta\right) + x_{n-1}\left(\frac{1}{2}-\gamma+\beta\right) \right] \Delta t^2 - \\ & \left[f_{n+1}\beta + f_n\left(\frac{1}{2}+\gamma-2\beta\right) + f_{n-1}\left(\frac{1}{2}-\gamma+\beta\right) \right] \Delta t^2 = 0 \end{aligned} \quad (A1.5)$$

The resultant equation (A1.5) can be simplified further. The mass term is reduced by substitution of the acceleration term at $n+1$ with equation (2.6).

$$\begin{aligned} & \left[\ddot{x}_{n+1}\beta + \ddot{x}_n\left(\frac{1}{2}+\gamma-2\beta\right) + \ddot{x}_{n-1}\left(\frac{1}{2}-\gamma+\beta\right) \right] \Delta t^2 = \\ & x_{n+1} - x_n - \dot{x}_n \Delta t + \gamma \Delta t^2 (\ddot{x}_n - \ddot{x}_{n-1}) + \ddot{x}_{n-1} \Delta t^2 \left(\frac{1}{2} + \beta\right) - \ddot{x}_n \Delta t^2 \beta \end{aligned} \quad (A1.6)$$

The velocity and acceleration terms are eliminated by substituting equations (2.5) and (2.6) for acceleration at n therefore that the final mass term is:

$$= x_{n+1} - 2x_n + x_{n-1} \quad (A1.7)$$

The damping term is simplified by inputting equation (2.3) for the $\beta\Delta t$ and $(\gamma-\beta)$ velocity terms at time steps n+1 and n respectively.

$$\begin{aligned} & \left[\dot{x}_{n+1}\beta + \dot{x}_n\left(\frac{1}{2}+\gamma-2\beta\right) + \dot{x}_{n-1}\left(\frac{1}{2}-\gamma+\beta\right) \right] \Delta t = \\ & \frac{\Delta t}{2}(\dot{x}_n + \dot{x}_{n-1}) + \beta\Delta t(\dot{x}_{n+1} - \dot{x}_n) + (\gamma-\beta)(\dot{x}_n - \dot{x}_{n-1})\Delta t \end{aligned} \quad (A1.8)$$

$$\begin{aligned} & = \frac{\Delta t}{2}(\dot{x}_n + \dot{x}_{n-1}) + \beta\Delta t((1-\gamma)\ddot{x}_n\Delta t + \gamma\ddot{x}_{n+1}\Delta t) \\ & + (\gamma-\beta)((1-\gamma)\ddot{x}_{n-1}\Delta t + \gamma\ddot{x}_n\Delta t)\Delta t \end{aligned} \quad (A1.9)$$

The $\beta\Delta t$ acceleration terms are substituted by equation (2.6).

$$\begin{aligned} & = \frac{\Delta t}{2}(\dot{x}_n + \dot{x}_{n-1}) + \beta\Delta t(\ddot{x}_n\Delta t + \frac{\gamma}{\beta\Delta t}[x_{n+1} - x_n - \dot{x}_n\Delta t - \frac{\ddot{x}_n}{2}\Delta t^2]) \\ & + (\gamma-\beta)((1-\gamma)\ddot{x}_{n-1}\Delta t + \gamma\ddot{x}_n\Delta t)\Delta t \end{aligned} \quad (A1.10)$$

The previous substitution introduced two new acceleration terms at time n, these are replaced by equations (2.6) and (2.5).

$$\begin{aligned}
&= x_n - x_{n-1} + \gamma(x_{n+1} - x_n - \dot{x}_n \Delta t) \\
&+ \gamma \Delta t^2 \left[\left(\frac{1}{2} - \gamma \right) \ddot{x}_{n-1} + \gamma \ddot{x}_n \right] - \beta \Delta t^2 (\gamma \ddot{x}_n - \gamma \ddot{x}_{n-1})
\end{aligned} \tag{A1.11}$$

Further substitution of equations (2.5) and (2.6) for the acceleration terms at n will give the following final simplified form of the damping term.

$$= \gamma x_{n+1} + (1-2\gamma)x_n + (\gamma-1)x_{n-1} \tag{A1.12}$$

Hence (A1.5) the general equation, is reduced to the following:

$$\begin{aligned}
&M[x_{n+1} - 2x_n + x_{n-1}] + C\Delta t[\gamma x_{n+1} + (1-2\gamma)x_n + (\gamma-1)x_{n-1}] \\
&+ K\Delta t^2 \left[\beta x_{n+1} + \left(\frac{1}{2} + \gamma - 2\beta \right) x_n + \left(\frac{1}{2} - \gamma + \beta \right) x_{n-1} \right] \\
&- \Delta t^2 \left[\beta f_{n+1} + \left(\frac{1}{2} + \gamma - 2\beta \right) f_n + \left(\frac{1}{2} - \gamma + \beta \right) f_{n-1} \right] = 0
\end{aligned} \tag{A1.13}$$

A.2 WEIGHTED RESIDUAL

The multi-step Zienkiewicz Weighted Residual equation (2.10) is solved by first substituting in the following shape functions in equation (2.9) and their corresponding derivatives.

$$\begin{aligned}
N_{n-1} &= \frac{t}{2\Delta t} \left(\frac{t}{\Delta t} - 1 \right) & \dot{N}_{n-1} &= \frac{1}{\Delta t} \left(\frac{t}{\Delta t} - \frac{1}{2} \right) & \ddot{N}_{n-1} &= \frac{1}{\Delta t^2} \\
N_n &= 1 - \frac{t^2}{\Delta t^2} & \dot{N}_n &= -\frac{2t}{\Delta t^2} & \ddot{N}_n &= -\frac{2}{\Delta t^2} \\
N_{n+1} &= \frac{t}{2\Delta t} \left(\frac{t}{\Delta t} + 1 \right) & \dot{N}_{n+1} &= \frac{1}{\Delta t} \left(\frac{t}{\Delta t} + \frac{1}{2} \right) & \ddot{N}_{n+1} &= \frac{1}{\Delta t^2}
\end{aligned} \tag{A2.1}$$

If the weighting function W is set equal to one for the entire domain the weighted

residual equation is reduced to the following integral where p equals $t/\Delta t$ and q equals $(t/\Delta t)^2$:

$$\begin{aligned} \int_{-\Delta t}^{\Delta t} \frac{M}{\Delta t^2} \{x_{n+1} - 2x_n + x_{n-1}\} + \frac{C}{\Delta t} \left\{ \left(p + \frac{1}{2}\right)x_{n+1} - 2px_n + \left(p - \frac{1}{2}\right)x_{n-1} \right\} \\ + K \left\{ \frac{1}{2}(q+p)x_{n+1} + (1-q)x_n + \frac{1}{2}(q-p)x_{n-1} \right\} \\ - \left\{ \frac{1}{2}(q+p)f_{n+1} + (1-q)f_n + \frac{1}{2}(q-p)f_{n-1} \right\} .dt = 0 \end{aligned} \quad (A2.2)$$

After the integration and multiplying through by $\Delta t/2$ to simplify, general equation (A2.3) is yielded.

$$\begin{aligned} M \{x_{n+1} - 2x_n + x_{n-1}\} + \frac{C}{2} \{\Delta t x_{n+1} - \Delta t x_{n-1}\} \\ + \frac{K \Delta t}{2} \left\{ \frac{\Delta t}{3} x_{n+1} + \frac{4 \Delta t}{3} x_n + \frac{\Delta t}{3} x_{n-1} \right\} \\ - \frac{\Delta t}{2} \left\{ \frac{\Delta t}{3} f_{n+1} + \frac{4 \Delta t}{3} f_n + \frac{\Delta t}{3} f_{n-1} \right\} = 0 \end{aligned} \quad (A2.3)$$

APPENDIX B

B.1 DYNAMIC.C

B.1.1 INTRODUCTION

This program was designed to simulate one-step solution methods which solve linear dynamic problems. Dynamic.c will run the selection of common solution schemes discussed in Chapter Two. A copy of the program code is listed in section B.3.

Once a solution scheme is selected by the user, appropriate parameters and initial conditions must be chosen. The program uses the solution scheme to calculate the displacement history of the structure for a prescribed time, the displacement history is translated into a displacement plot against time using an interpolation function. Amplitude decay and period elongation for the displacement plot are determined by locating the maximum and zero amplitude points.

The spectral radius is calculated within dynamic.c if the scheme is quadratic. However if the scheme is cubic the A matrix is output to "*filename*".dat, it is then used by eigen.c to calculate the modulus of A's eigenvalues.

Using the amplitude decay, period elongation and spectral radius the user can determine which of the solution schemes gives the most accurate results for the prescribed dynamic problem.

B.1.2 ASSUMPTIONS AND LIMITATIONS

The following rules apply:

1. All filenames have a maximum length of 80 characters which includes any file extensions which are added by dynamic.c.
2. A maximum of 1000 time steps are permitted so an appropriate Δt must be selected for the prescribed total time period for the solution.

3. A limit of 14 different cases may be run for each input file, note that each case only differs by the number of prescribed divisions per period T.
4. No damping is possible in dynamic.c, the following is the governing single degree of freedom structural dynamic equation for dynamic.c:

$$m\ddot{x} + kx = f(t) \quad (B1.1)$$

5. Initial acceleration is calculated using equation (B1.1) and the prescribed mass, stiffness, initial displacement and force.
6. Dynamic.c can only solve problems using the solution schemes in Table B.1.
7. The interpolation function gives very poor results for time steps larger than the period T / 3.

B.1.3 INPUT DESCRIPTION

One input file is required to run the program, it contains the following data, an example is seen in Figure B.1:

1. Mass - the mass of the agitated system (real).
2. Stiffness - the stiffness of the agitated system (real).
3. Force - the applied force (real).
4. Force duration - the period of time the force is applied (real).
5. Initial displacement - the systems displacement at time increment zero (real).
6. Initial velocity - the systems velocity at time increment zero (real).
7. Displacement switch (int) - 0 = no displacements output
1 = yes displacements output
8. Time, number of cases + (increments x number of cases) - total length of time for solution (real), number of cases (int) and the number of divisions per period T, for each case (int).
9. Scheme - solution scheme identification number (int).
10. Parameters - the parameter values to be applied in solution, see Table B.1 (real).

Scheme	Scheme I.D. Number	Parameters			Order Of Solution Scheme
		Param. 1	Param. 2	Param. 2	
Wilson θ	1	θ_w	-	-	Cubic
Bossak $\gamma_B = 1/2 - \alpha_B$	2	α_B	β_B	-	Cubic
HHT $\gamma_H = 1/2 - \alpha_H$	3	α_H	β_H	-	Cubic
Newmark	4	δ	2α	-	Quadratic
Houbolt	5	-	-	-	Cubic
SS22	6	θ_1	θ_2	-	Quadratic
SS32	7	θ_1	θ_2	θ_3	Cubic

TABLE B.1 POSSIBLE SOLUTION SCHEMES IN DYNAMIC.C

```

1.0
1.0
0.0
0.0
3.0
0.0
0
26.0 2 50 3
2
-0.1 0.3025

```

FIGURE B.1 EXAMPLE INPUT FILE

B.1.4 OUTPUT DESCRIPTION

There are three output files, examples of which are in Figures (B.2)-(B.4):

1. *"filename".res* - first there is optional output of the structures displacement history. *Dynamic.c* outputs the period T , time increment size Δt and the percentage amplitude decay and period elongations. There is a new set of data for each case.
2. *"filename".dat* - there are two possible output files. The first in Figure (B.3a) is output for a cubic solution, the first integer represents the number of cases and the following multiples of three data lines represent the A matrices. The second output file Figure (B.3b) is output for a quadratic solution scheme, it lists the A matrix and its corresponding eigenvalues. Table B.1 lists the order of each solution scheme.
3. *ap_"filename"* - the first integer equals the number of cases and the following numbers are the amplitude decay and period elongation for each case.

B.1.5 RUNNING INSTRUCTIONS

To run *dynamic.c* type the following command at the prompt:

dynamic "filename"

Bossak1.res1

Time	Displacement
0.000000	3.000000
1.047198	1.643099
2.094395	-1.114105
3.141593	-2.904586
4.188790	-2.265658
5.235988	0.229382
6.283185	2.487263
7.330383	2.650537
8.377580	0.624844
9.424778	-1.877176
10.471976	-2.783015
11.519173	-1.376354
12.566371	1.138292
13.613568	2.665852
14.660766	1.965574
15.707963	-0.342190
16.755161	-2.323369
17.802358	-2.350422
18.849556	-0.438626
19.896753	1.798121
20.943951	2.508983
21.991149	1.137082
23.038346	-1.146151
24.085544	-2.440268
25.132741	-1.697070

Period	= 6.283185
Time increment	= 1.047198
Time increment/Period	= 0.166667
Amplitude decay	= 5.206276
Period Elongation (T & T[1])	= 11.140951
Period Elongation (T & T[2])	= 10.131836
Period Elongation (T & T[3])	= 13.199870

FIGURE B.2 EXAMPLE OUTPUT FILE "filename".res

2

OUTPUT 1

0.433914 0.716957
 -1.132173 0.433914

Eigenvalue 1 : $0.433914 + i(0.900954)$
 Eigenvalue 2 : $0.433914 - i(0.900954)$

OUTPUT 2

0.569667 0.784833
 -0.860666 0.569667

Eigenvalue 1 : $0.569667 + i(0.821876)$
 Eigenvalue 2 : $0.569667 - i(0.821876)$

FIGURE B.3 a EXAMPLE OUTPUT FILE "*filename*".dat FOR QUADRATIC
 SOLUTION SCHEME

2

0.833180 0.816498 0.279950
 -0.500459 0.449495 0.339849
 -1.000918 -1.101009 -0.320301
 0.872343 0.859577 0.296039
 -0.382972 0.578731 0.388117
 -0.765943 -0.842538 -0.223766

FIGURE B.3 b EXAMPLE OUTPUT FILE "*filename*".dat FOR CUBIC
 SOLUTION SCHEME

B.2 EIGEN.C

B.2.1 INTRODUCTION

Eigen.c was written to accompany dynamic.c, it simply calculates the eigenvalues and the modulus of the eigenvalues for any cubic A matrix output by dynamic.c. The eigenvalues are calculated using an imslc math library routine, the program listing is in section B.4.

B.2.2 ASSUMPTIONS AND LIMITATIONS

There are essentially only two limitations which apply to eigen.c. The first being that the program will only work for cubic solutions, that is it can only calculate the eigenvalues for a 3×3 matrix with an input file using the same format as "*filename*".dat which was output by dynamic.c. The second is that all file names have a limit in length of 80 characters.

B.2.3 INPUT AND OUTPUT DESCRIPTION

The input file is output by dynamic.c, it is described in section B.1.4 and is named "*filename*".dat.

There is one output file, "*filename*".dat2, see Figure B.5. The output file lists the A matrix for each case and the eigenvalues with their corresponding modulus.

B.2.4 RUNNING INSTRUCTIONS

As eigen.c uses an imslc math routine the computer environment must first be initialized using the following instruction at the prompt:

```
. use imslc
```

To run eigen.c type the following command at the prompt:

```
a.out "filename".dat
```

2
 8.600429 13.798828
 5.206276 11.140951

FIGURE B.4 EXAMPLE OUTPUT FILE ap_"filename"

0.83318 0.816498 0.27995
 -0.500459 0.449495 0.339849
 -1.00092 -1.10101 -0.320301
 OUTPUT 1

Eigenvalue 0 : 0.448445 + i(0.875695)
 Eigenvalue 1 : 0.448445 + i(-0.875695)
 Eigenvalue 2 : 0.0654832 + i(0)
 Eigenvalues modulus : 0.983842 0.983842 0.065483

0.872343 0.859577 0.296039
 -0.382972 0.578731 0.388117
 -0.765943 -0.842538 -0.223766
 OUTPUT 2

Eigenvalue 0 : 0.578046 + i(0.804132)
 Eigenvalue 1 : 0.578046 + i(-0.804132)
 Eigenvalue 2 : 0.0712157 + i(0)
 Eigenvalues modulus : 0.990336 0.990336 0.071216

FIGURE B.5 EXAMPLE OUTPUT FILE "filename".dat2

B.3 DYNAMIC.C LISTING

```

/* Program To Model Dynamic System */
/* By Tracy Greener, Jan, 1994 */
/* Version 2 */
/* Additional SSpj Formulation */

#include <string.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#define MOD(a)      (sqrt(pow(a,2)))

/* define global variables */
int scheme, num[15], n, count, swtch;
double x[15][1000], param[3], par[3], coeff[6], time, step, period, omega, zero;
char oname[80], oname1[80], oname2[80];

main(argc, argv)
    int argc;
    char **argv;
{
    FILE *opf,*opf1,*opf2;

    Arg_check(argc);
    Input(argv);
    Params();

    /* open files */
    strcpy(oname,argv[1]);
    strcat(oname, ".res");
    strcpy(oname2,argv[1]);
    opf=fopen(oname,"w");
    strcpy(oname1,"ap_");
    strcat(oname1,argv[1]);
    opf1=fopen(oname1,"w");
    strcat(oname2, ".dat");
    opf2=fopen(oname2,"w");

    fprintf(opf2,"%d\n",num[0]);
    fprintf(opf1,"%d\n",num[0]);

```

```

for(n=1;n<=num[0];++n)
    Equation(argv,opf,opf1,opf2);
fclose(opf);
fclose(opf1);
fclose(opf2);
}

/* Routine to check run line is correct */
Arg_check(argc)
    int argc;
{
    if(argc !=2){
        fprintf(stderr,"\n *** Usage: <prog> <i/p file> <o/p file> ***\n");
        exit(-1);
    }
}

/* Routine to read data from input file */
Input(argv)
    char **argv;
{
    int i;
    char iname[80];
    FILE *ipf;

    strcpy(iname,argv[1]);
    ipf=fopen(iname,"r");

    /* check file is open */
    if(ipf==NULL){
        fprintf(stderr,"\n *** Cannot open input file %s ***\n",iname);
        fclose(ipf);
        exit(-1);
    }

    /* [0]=mass [1]=stiff [2]=force [3]=dur [4]=initial disp [5]=initial vel */
    for(i=0;i<6;++i){
        fflush(ipf);
        fscanf(ipf,"%le\n",&coeff[i]);
    }
}

```



```

/* switch 0 = on, 1 = off, displacements */
fflush(ipf);
fscanf(ipf,"%d\n",&swtch);
fflush(ipf);
fscanf(ipf,"%le%d",&time,&num[0]);

for(i=1;i<=num[0];++i)
    fscanf(ipf,"%d",&num[i]);

fflush(ipf);
fscanf(ipf,"\n%d\n",&scheme);

/* Bossak, HHT, Newmark & other (p=2) input */
if(scheme==2 || scheme==3 || scheme==4 || scheme==7){
    fflush(ipf);
/* [0]=gamma [1]=beta or [0]=alpha [1]=beta */
    fscanf(ipf,"%le%le",&par[0],&par[1]);
}

/* Wilson Theta input [0]=theta */
if(scheme==1){
    fflush(ipf);
    fscanf(ipf,"%le",&par[0]);
}

/* Houbolt or other (p=3) schemes */
if(scheme==5 || scheme==6){
    fflush(ipf);
/* [0]=alpha [1]=beta [2]=gamma */
    fscanf(ipf,"%le%le%le",&param[0],&param[1],&param[2]);
}

fclose(ipf);
}

/* Routine to set up the parameters */
Params()
{
    int i;

    switch(scheme){

```

```

/*Wilson Theta*/
case 1:
param[0]=par[0];
param[1]=pow(par[0],2);
param[2]=pow(par[0],3);
break;

/*Bossak*/
case 2:
param[0]=1.0-par[0];
param[1]=(2.0/3.0)-par[0]+(2.0*par[1]);
param[2]=6.0*par[1];
break;

/*Hilber, Hughes & Taylor (HHT)*/
case 3:
param[0]=1.0;
param[1]=(2.0/3.0)+(2.0*par[1])-(2.0*(pow(par[0],2)));
param[2]=6.0*par[1]*(1.0+par[0]);
break;

/*Newmark*/
case 4:
param[0]=par[0];
param[1]=2.0*par[1];
break;

/*Houbolt*/
case 5:
break;

/*SSpj (p=3)*/
case 6:
break;

/*SSpj (p=2)*/
case 7:
param[0]=par[0];
param[1]=par[1];
break;

default:
fprintf(stderr, "\n *** Incorrect case input value ***\n");
exit(-1);

```

```

}
}

/* Routine to calculate displacements */
Equation(argv, opf, opf1, opf2)
    FILE *opf, *opf1, *opf2;
    char **argv;
    {
    int i, j, tmp, lapse, lap, tag;
    double Amat[1000], fmat[1000], xdot[1000], xddot[1000], f[1000], force, dummy;

    fprintf(opf, "\n %s%d\n *****\n\n", oname, n);
    if(scheme==7 || scheme==4) fprintf(opf2, " OUTPUT %d\n\n", n);
    if(swch==1) fprintf(opf, " Time      Displacement\n");

    /* Calculate frequency, period and step size */
    omega=sqrt(coeff[1]/coeff[0]);
    period=2.0*M_PI/omega;
    step=period/num[n];

    lapse=coeff[3]/step;
    lap=time/step;

    /* Prescribe initial conditions */
    x[n][0]=coeff[4];
    xdot[0]=coeff[5];
    xddot[0]=(coeff[2]-(coeff[1]*coeff[4]))/coeff[0];
    tag=0;

    for(i=0;i<lapse;++i){
        f[i]=coeff[2];
        tag=i;
    }

    dummy=pow(step,2);

    /* Call routine to calculate amplification and load matrices */
    if(scheme==4 || scheme==7)
        SS22(&Amat,&fmat,opf2);          /* Single-Step quadratic algorithm */
    else
        SS32(&Amat,&fmat,opf2);          /* Single-Step cubic algorithm */

```

```

for(i=0;i<lap;++i){

    if(swtch==1 && i==0)Output(&i,opf);
    force=(param[0]*f[i+1])+((1.0-param[0])*f[i]);

    /* Solve matrix equation */
    if(scheme==4 || scheme==7){
        x[n][i+1]=(Amat[0]*x[n][i])+(Amat[1]*xdot[i]*step)+(fmat[0]*force);
        xdot[i+1]=((Amat[2]*x[n][i])+(Amat[3]*xdot[i]*step)+(fmat[1]*force))/step;
    }
    else{

x[n][i+1]=(Amat[0]*x[n][i])+(Amat[1]*xdot[i]*step)+(Amat[2]*xddot[i]*dummy)+(fmat[0]*force);

xdot[i+1]=((Amat[3]*x[n][i])+(Amat[4]*xdot[i]*step)+(Amat[5]*xddot[i]*dummy)+(fmat[1]*force))/step;

xddot[i+1]=((Amat[6]*x[n][i])+(Amat[7]*xdot[i]*step)+(Amat[8]*xddot[i]*dummy)+(fmat[2]*force))/dummy;
    }

    tmp=i+1;
    if(swtch==1) Output(&tmp,opf);
}
Elong_decay(opf,opf1);
}

/* Formulation of amplification & force matrix */
SS22(Amat,fmat,opf2)
FILE *opf2;
double *Amat,*fmat;
{
double P, Q, R, sq_step, W, Y, Z, eigen1, eigen2 ,U;

sq_step=pow(step,2);
P=coeff[0]+((sq_step*param[1]*coeff[1])/2.0);
Q=sq_step/(2.0*P);
R=sq_step/P;

Amat[0]=1.0-(Q*coeff[1]);
Amat[1]=1.0-(Q*coeff[1]*param[0]);
Amat[2]=-R*coeff[1];

```

```

Amat[3]=1.0-(R*coeff[1]*param[0]);
fmat[0]=Q;
fmat[1]=R;

fprintf(opf2,"%f %f\n",Amat[0],Amat[1]);
fprintf(opf2,"%f %f\n\n",Amat[2],Amat[3]);

W=Amat[0]+Amat[3];
Y=W/2.0;
Z=(pow(W,2))-(4.0*((Amat[0]*Amat[3])-(Amat[1]*Amat[2])));
if(Z<0.0){
    U=(sqrt(-Z))/2.0;
    fprintf(opf2," Eigenvalue 1 : %f + i(%f)\n",Y,U);
    fprintf(opf2," Eigenvalue 2 : %f - i(%f)\n\n",Y,U);
}
else{
    U=(sqrt(Z))/2.0;
    eigen1=Y+U;
    eigen2=Y-U;
    fprintf(opf2," Eigenvalue 1 : %f\n",eigen1);
    fprintf(opf2," Eigenvalue 2 : %f\n\n",eigen2);
}

}

/* Formulation of amplification & force matrix */
SS32(Amat,fmat,opf2)
FILE *opf2;
double *Amat,*fmat;
{
int i;
double P, Q, R, S, cube_step;

cube_step=pow(step,3);
P=(step*param[0]*coeff[0])+((cube_step*param[2]*coeff[1])/6.0);
Q=cube_step/(P*6.0);
R=cube_step/(P*2.0);
S=cube_step/P;

Amat[0]=1.0-(Q*coeff[1]);
Amat[1]=1.0-(Q*coeff[1]*param[0]);
Amat[2]=(1.0/2.0)-((Q*coeff[0])/(pow(step,2)))-((Q*coeff[1]*param[1])/2.0);
Amat[3]=-1.0*R*coeff[1];

```

```

Amat[4]=1.0-(R*coeff[1]*param[0]);
Amat[5]=1.0-((R*coeff[0])/(pow(step,2)))-((R*coeff[1]*param[1])/2.0);
Amat[6]=-1.0*S*coeff[1];
Amat[7]=-1.0*S*coeff[1]*param[0];
Amat[8]=1.0-((S*coeff[0])/(pow(step,2)))-((S*coeff[1]*param[1])/2.0);
fmat[0]=Q;
fmat[1]=R;
fmat[2]=S;

```

```

for(i=0;i<3;++i)
    fprintf(opf2,"%f  %f  %f\n",Amat[i*3],Amat[(i*3)+1],Amat[(i*3)+2]);
}

```

/* Routine to output the results */

Output(no, ofile)

int *no;

FILE *ofile;

{

int temp;

double tmp;

temp=*no;

tmp=temp*step;

fprintf(ofile," %f %f\n",tmp,x[n][temp]);

}

/*Routine to calculate amplitude decay and period elongation */

Elong_decay(ofile,opf1)

FILE *ofile, *opf1;

{

double l[2], lint[2], T[3], Tint[3], Tag[4], Tagint[4], error, AD, LAD, PE[3], PEI[3],
value, temp[4];

int i, j, flag[4], dummy, ind, ptx[2], tmp, k;

dummy=1;

ind=0;

if(coeff[3]==time){

zero=coeff[2]/coeff[1];

}else

```

zero=0.0;

/* Locating points which bound the point of max. & zero amplitude */
for(i=0;i<4;++i){
    for(j=dummy;++j){
        if((x[n][j]>zero && x[n][j+1]>=zero) || (x[n][j]<zero && x[n][j+1]<=zero))
            continue;
        else{
            ind+=1;
            dummy=j+1;
            if(ind==((i+1)*2)){
                temp[i]=x[n][j];
                flag[i]=j;
                break;
            }
        }
    }
}

k=1;

if(MOD(x[n][flag[i]+1]-zero)<MOD(x[n][flag[i]-1]-zero)){
    tmp=flag[i]+1;
    Interp(&flag[i],&flag[i],&tmp,&Tagint[i],&k);
}
else{
    tmp=flag[i]-1;
    Interp(&flag[i],&tmp,&flag[i],&Tagint[i],&k);
}

if(i<2){
    for(j=dummy;++j){
        if(MOD(x[n][j]-zero)<MOD(x[n][j+1]-zero))
            continue;
        else{
            if(MOD(x[n][j+1]-zero)<MOD(x[n][j-1]-zero)){
                ptx[0]=j-1;
                ptx[1]=j;
            }
            else{
                ptx[0]=j;
                ptx[1]=j+1;
            }
            dummy=j+1;
            break;
        }
    }
}

```

```

    }
}
k=0;

/* Call routine performing interpolation function */
if((MOD(x[n][ptx[0]]-zero))>(MOD(x[n][ptx[1]]-zero))){
    Interp(&ptx[0],&ptx[0],&ptx[1],&lint[i],&k);
}
else{
    Interp(&ptx[1],&ptx[0],&ptx[1],&lint[i],&k);
}
}
}

/* Calculate period elongation */
for(i=0;i<3;++i){
    Tint[i]=Tagint[i+1]-Tagint[i];
    PEI[i]=(100.0/period*Tint[i])-100.0;
}

/* Calculate amplitude decay */
LAD=100.0-(100.0/lint[0]*lint[1]);
value=step/period;

/* Output data */
fprintf(ofile,"\n Period                = %f\n",period);
fprintf(ofile," Time increment          = %f\n",step);
fprintf(ofile," Time increment/Period        = %f\n",value);
fprintf(ofile," Amplitude decay                = %f\n",LAD);
fprintf(ofile," Period Elongation (T & T[1]) = %f\n",PEI[0]);
fprintf(ofile," Period Elongation (T & T[2]) = %f\n",PEI[1]);
fprintf(ofile," Period Elongation (T & T[3]) = %f\n",PEI[2]);
fprintf(opf1," %f %f\n",LAD,PEI[0]);
}

/* Interpolation scheme to calculate amplitude decay etc...*/
Interp(pt0,pt1,pt2,max,tmp)
double *max;
int *pt0, *pt1, *pt2, *tmp;
{
double bound[3], poly[3], AD, l, var[3];
int i, j, maxpt;

```



```

maxpt=*pt0;
bound[0]=*pt1;
bound[1]=*pt2;

/* construct the variables in the polynomial */
for(i=0;i<2;++i)
    var[i]=(x[n][maxpt+i]-x[n][maxpt+i-1])/(((maxpt+i)*step)-((maxpt+i-1)*step));

var[2]=(var[1]-var[0])/(((maxpt+1)*step)-(maxpt-1)*step);

poly[0]=x[n][(int)bound[0]];
poly[1]=x[n][(int)bound[1]];

/* Locate points of max. & zero amplitude */
for(;;){
    if(MOD(poly[1]-poly[0]>0.001){
        bound[2]=(bound[1]+bound[0])/2.0;

poly[2]=x[n][maxpt-1]+(var[0]*step*(bound[2]-(maxpt-1)))+(var[2]*step*step*(bound[2]
)-(maxpt-1))*(bound[2]-maxpt));
        if(MOD(poly[0]-zero)<MOD(poly[1]-zero)){
            if(*tmp==0){
                poly[0]=poly[2];
                bound[0]=bound[2];
            }
            else{
                poly[1]=poly[2];
                bound[1]=bound[2];
            }
        }
        else{
            if(*tmp==0){
                poly[1]=poly[2];
                bound[1]=bound[2];
            }
            else{
                poly[0]=poly[2];
                bound[0]=bound[2];
            }
        }
    }
    else
        break;
}

```

```
if(MOD(poly[0]-zero)>MOD(poly[1]-zero)){
    if(*tmp==0)
        l=(MOD(poly[0]-zero));
    else
        l=bound[1]*step;
}
else{
    if(*tmp==0)
        l=(MOD(poly[1]-zero));
    else
        l=bound[0]*step;
}

*max=l;
}
```

B.4 EIGEN.C LISTING

```

/* Program to calculate eigenvalues */
/* By Tracy Greener Jan, 1994 */

#include <imsl.h>
#include <stdlib.h>
#include <stdio.h>
#include <math.h>
#include <string.h>

char oname[80], oname2[80];

main(argc, argv)
    int argc;
    char **argv;
{
    FILE *opf, *opf2;
    double Amat[9],value[3];
    int n=3,i,j,num;
    d_complex *eval;

    /* Read Amat values from input file */
    strcpy(oname,argv[1]);
    opf=fopen(oname,"r");
    strcpy(oname2,argv[1]);
    strcat(oname2,"2");
    opf2=fopen(oname2,"w");

    fflush(opf);
    fscanf(opf,"%d\n",&num);

    for(j=0;j<num;++j){
        for(i=0;i<3;++i){
            fflush(opf);
            fscanf(opf,"%le %le %le\n",&Amat[i*3],&Amat[(i*3)+1],&Amat[(i*3)+2]);
            fprintf(opf2,"%g %g %g\n",Amat[i*3],Amat[(i*3)+1],Amat[(i*3)+2]);
        }

        /* Call imslc eigenvalue routine */
        eval=imsl_d_eig_gen(n,Amat,0);
    }
}

```

```
/* Output eigenvalues */
fprintf(opf2," OUTPUT %d\n\n",j+1);
for(i=0;i<3;++i){
    fprintf(opf2," Eigenvalue %d : %g + i(%g)\n",i,eval[i].re,eval[i].im);
    value[i]=sqrt((pow(eval[i].re,2))+pow(eval[i].im,2));
}
fprintf(opf2," Eigenvalues modulus : %f %f %f\n\n",value[0],value[1],value[2]);
}

}
```

APPENDIX C
GRAPHS FROM NON-LINEAR STUDY USING NONSAP

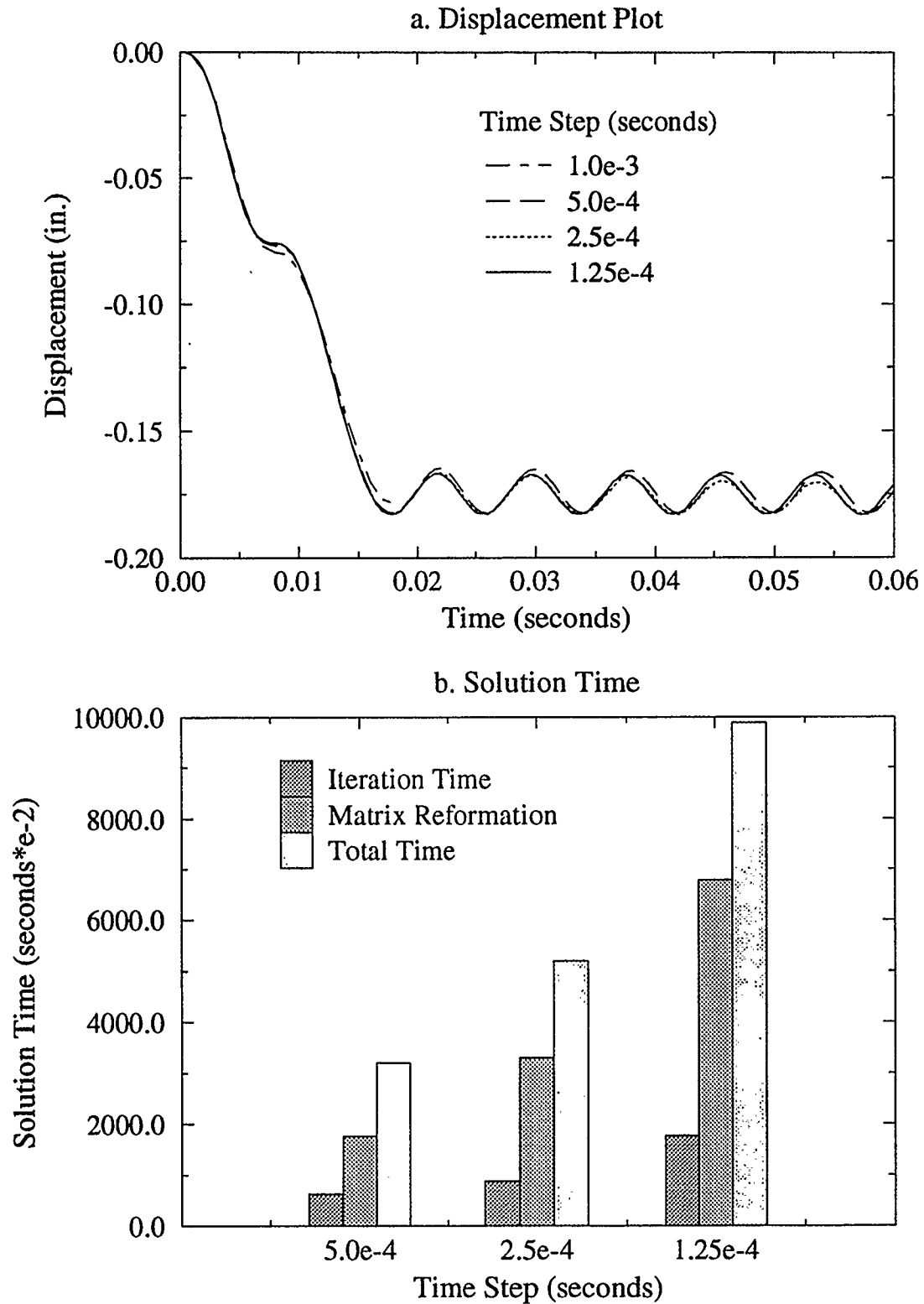


FIGURE C.1 COMPARISON PLOTS OF THE TRAPEZIUM SCHEME WITH DIFFERENT TIME STEP SIZES

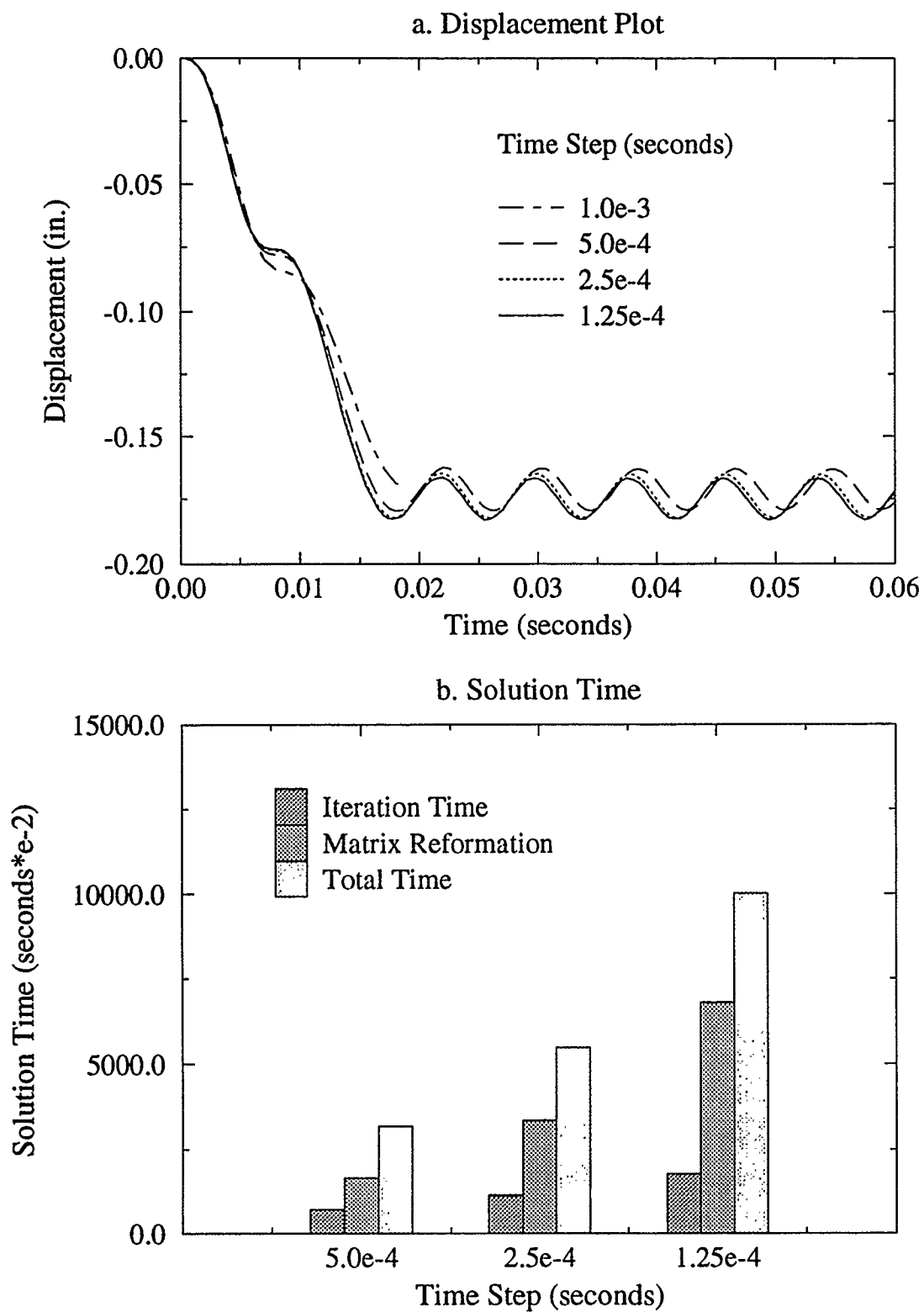


FIGURE C.2 COMPARISON PLOTS OF THE BOSSAK CASE 2 SCHEME WITH DIFFERENT TIME STEP SIZES

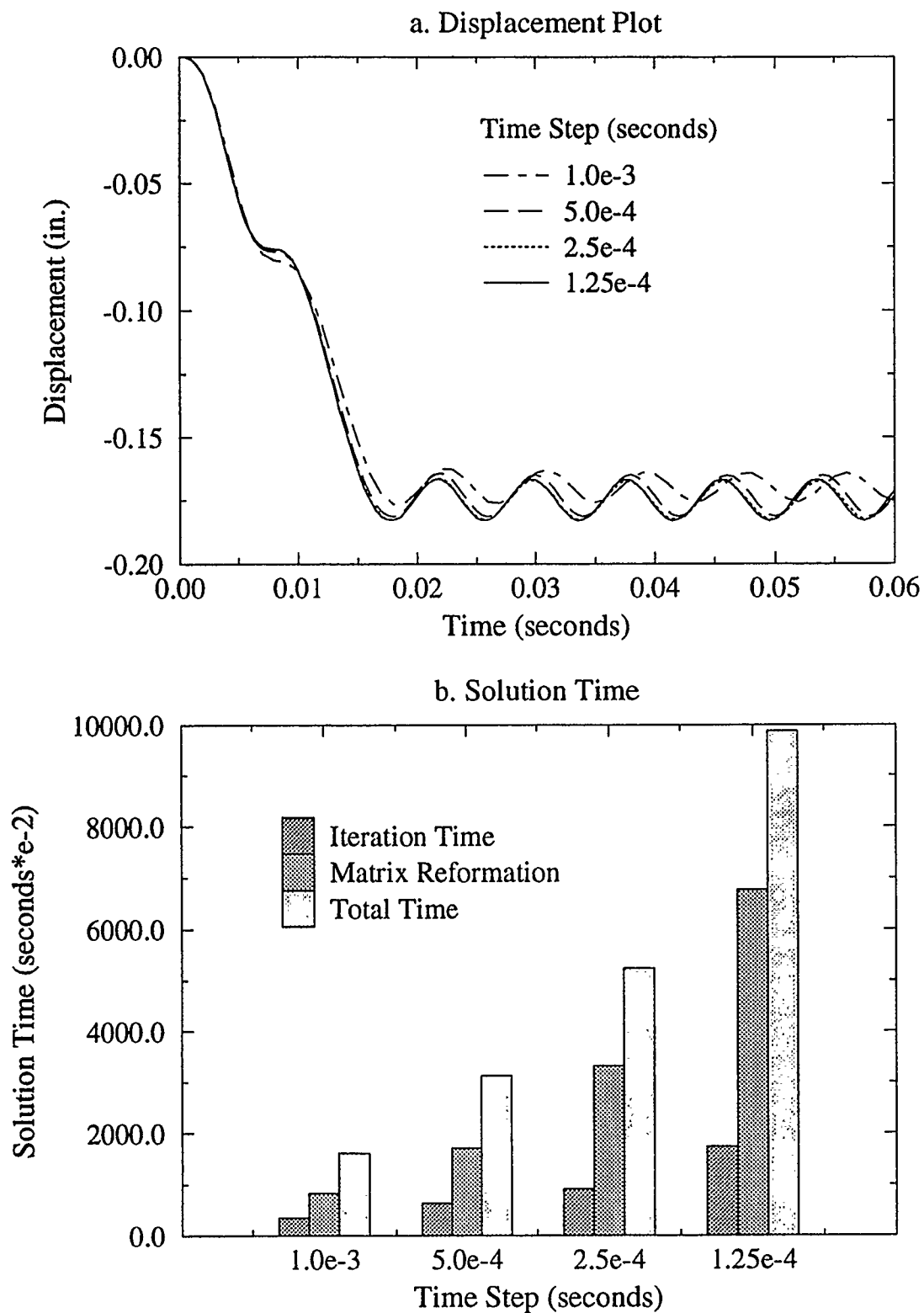


FIGURE C.3 COMPARISON PLOTS OF THE HILBER, HUGHES AND TAYLOR SCHEME WITH DIFFERENT TIME STEP SIZES

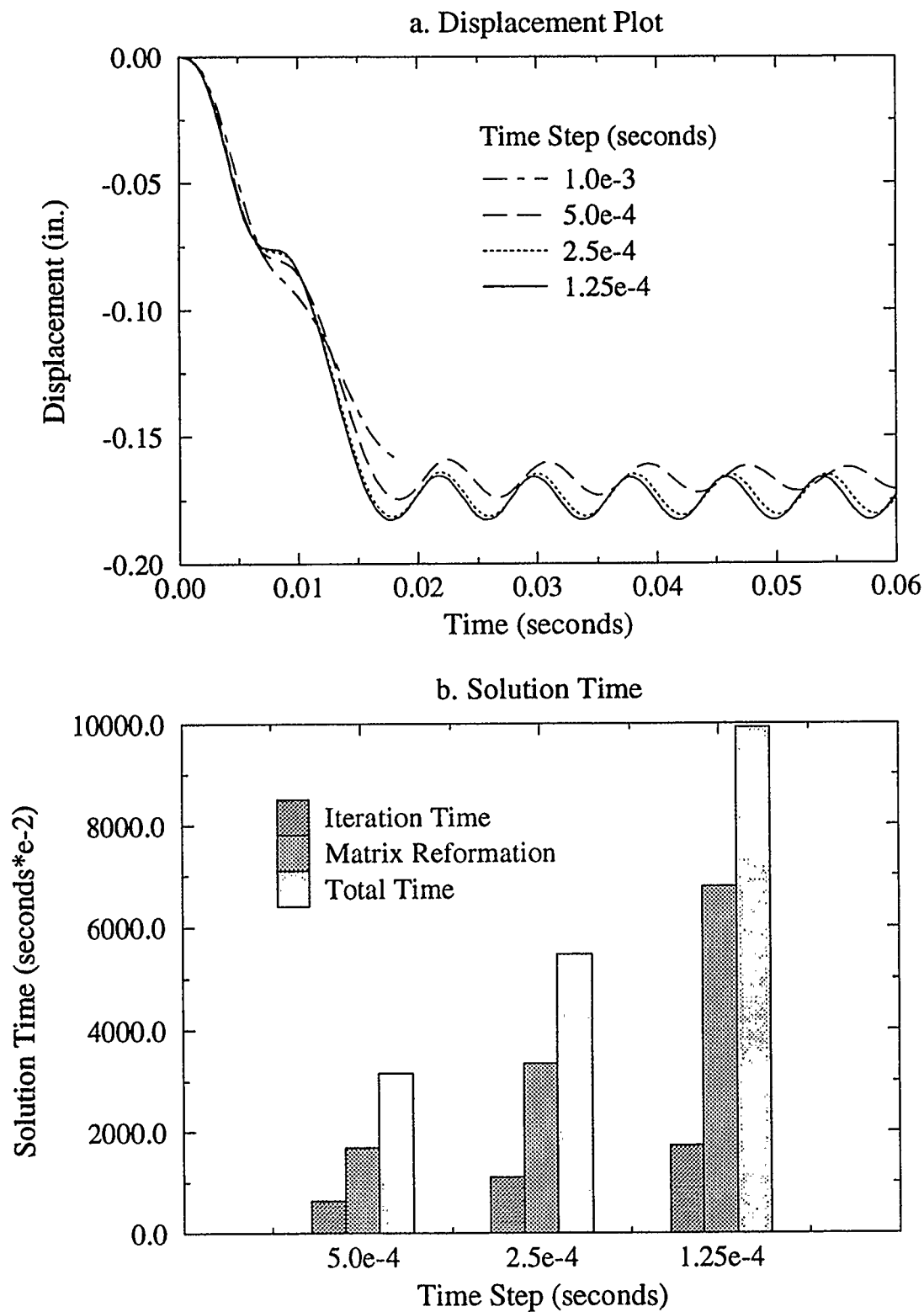


FIGURE C.4 COMPARISON PLOTS OF THE HOUBOLT SCHEME WITH DIFFERENT TIME STEP SIZES