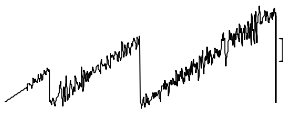
Draft

CU-NEES-09-11



NEES at CU Boulder
01000110 01001000 01010100
The George E Brown, Jr. Network for Earthquake Engineering Simulation

An Optimized Computational Environment for Real Time Hybrid Simulation

by

Dae Hung Kang

Gary Haussmann

Victor Saouma

August 31, 2009

April 2009

Department of Civil Environmental and Architectural Engineering
University of Colorado
UCB 428
Boulder, Colorado 80309-0428

Contents

1 INTRODUCTION **11**
1.1 Literature Survey . . . 11
1.1.1 Plasticity Models . . . 11
1.1.1.1 Lumped Plasticity Models . . . 11
1.1.1.2 Distributed Plasticity Models . . . 15
1.2 Parallel Computation for transient analysis . . . 17

2 ELEMENT FORMULATION **19**
2.1 Truss Element Formulation . . . 19
2.1.1 Formulation . . . 19
2.1.2 Coordinate system for 2D truss element . . . 21
2.1.3 State determination . . . 22
2.2 Beam-Column Element Formulations . . . 22
2.2.1 Stiffness-Based 2D Beam-Column Element . . . 23
2.2.1.1 Formulation . . . 23
2.2.1.2 Coordinate system for stiffness-based 2D beam-column element . 25
2.2.1.3 State determination . . . 27
2.2.1.4 Nonlinear analysis using stiffness-based formulation . . . 30
2.2.2 Flexibility Method 2D Beam-Column Element . . . 35
2.2.2.1 Formulation . . . 36
2.2.2.2 Coordinate system for flexibility-based 2D beam-column element 39
2.2.2.3 State determination . . . 41
2.2.2.4 Nonlinear analysis using flexibility-based method . . . 50
2.3 Layer/Fiber Section . . . 64
2.4 Zero-Length 2D Element Formulation . . . 66
2.4.1 Formulation . . . 66
2.4.2 Coordinate system in zero-length 2D element . . . 68
2.4.3 Element state determination . . . 69
2.5 Zero-Length Section Element Formulation . . . 71
2.5.1 Formulation . . . 72
2.5.2 Coordinate system in zero-length 2D section element . . . 73
2.5.3 Element state determination . . . 73
2.6 Element Force . . . 75
2.7 Comparion . . . 78
2.7.1 Example 1 . . . 78

2.7.2 Example 2 . . . 78
2.7.3 Example 3 . . . 78

3 CONSTITUTIVE MODELS 85
3.1 Steel Models . . . 85
3.1.1 Classical plasticity . . . 85
3.1.1.1 Isotropic hardening model . . . 87
3.1.1.2 Combined isotropic and kinematic hardening model . . . 91
3.1.1.3 Determination for isotropic and kinematic harding parameters . 94
3.1.2 Bilinear model with isotropic hardening . . . 94
3.1.2.1 Stress-strain relation . . . 94
3.1.2.2 Determination for bilinear model . . . 96
3.1.3 Giuffre-Menegotto-Pinto Model Modified by Filippou et al. . . . 96
3.1.3.1 Stress-strain relationship . . . 96
3.1.3.2 Determination for modified Giuffre-Menegotto-Pinto Model . . . 101
3.2 Concrete Models . . . 105
3.2.1 Modified Kent And Park Model . . . 105
3.2.1.1 Stress-strain relation . . . 105
3.2.1.2 Determination for modified Kent and Park model . . . 111
3.2.2 Anisotropic damage model with effective damage and stiffness recovery . 111
3.2.2.1 Constitutive model . . . 111
3.2.2.2 Uniaxial multi-fiber formulation . . . 113
3.2.2.3 Determination for anisotropic damage model . . . 113

4 NONLINEAR SOLUTION FOR ELLIPTIC PROBLEMS 123
4.1 Force And Displacement Control . . . 123
4.2 Nolinear Solution . . . 124
4.2.1 Newton-Raphson iterative method . . . 125
4.2.2 Initial stiffness iterative method . . . 127
4.2.3 Modified Newton-Raphson iterative mathod . . . 127
4.3 Convergence criteria . . . 128
4.4 The spherical arc-length method . . . 130

5 TIME INTEGRATION METHODS for TRANSIENT ANALYSIS 132
5.1 Introduction . . . 132
5.1.1 Mass Representation . . . 132
5.1.1.1 Lumped mass . . . 133
5.1.1.2 Consistent mass . . . 133
5.1.2 Damping Representation . . . 134
5.2 Time Integration Methods . . . 135
5.2.1 Newmark β method . . . 137
5.2.1.1 General Formulation . . . 137
5.2.1.2 Newmark β implicit method . . . 139
5.2.2 Hilber-Hughes-Taylor method(HHT method) . . . 143
5.2.2.1 General Formulation . . . 143
5.2.3 Modification of Newmark β and HHT method by P.B. Shing . . . 145

6 MERCURY IMPLEMENTATION **149**
6.1 General Description . . . 149
6.2 MatLab Implementation Strategy . . . 151
6.3 c++ Implementation Strategy . . . 153
6.3.1 General Desired Functionality . . . 153
6.3.2 Named Composite Hierarchy (NCH) . . . 154
6.3.3 Data Streaming and Serialization . . . 154
6.3.4 Configuration and Input . . . 156
6.3.5 Domain and Structural Assembly . . . 156
6.3.6 Analysis . . . 156
6.3.7 Timing and Synchronization . . . 157
6.3.8 General Source Code Guidelines . . . 157

7 MERCURY VALIDATION **158**
7.1 Simple 2D Truss Validation . . . 158
7.1.1 Example 01 . . . 158
7.1.2 Example 02 . . . 158
7.1.3 Example 03 . . . 160
7.1.4 Example 04 . . . 160
7.1.5 Example 05 . . . 160
7.1.6 Example 06 . . . 160
7.1.7 Example 07 . . . 160
7.1.8 Example 08 . . . 160
7.1.9 Example 09 . . . 160
7.1.10 Example 10 . . . 164
7.1.11 Example 11 . . . 164
7.1.12 Example 12 . . . 164
7.1.13 Example 13 to 15 . . . 166
7.1.14 Example 16 and 17 . . . 166
7.1.15 Examples 18 and 19 . . . 166
7.2 Beam Column Validation . . . 170
7.2.1 Example 21 . . . 170
7.2.2 Example 22 . . . 170
7.2.3 Example 23 to 26 . . . 170
7.2.4 Example 27 . . . 174
7.2.5 Example 28 . . . 174
7.2.6 Example 29 . . . 174
7.2.7 Example 30 . . . 174
7.2.8 Ex110 to 113 . . . 177
7.2.9 Ex116 to 119 . . . 177
7.2.10 Example 140 . . . 183
7.2.11 Example 141 . . . 183
7.3 Comparison of Modified Kent & Park with the Anisotropic Model . . . 183
7.3.1 Description of Test Problems . . . 187
7.3.2 Results . . . 187
7.3.3 Effect of Fixed Number of Iterations . . . 191

8 HYBRID TEST **194**
8.1 Introduction . . . 194
8.1.1 Methods To Evaluate Seismic Performance . . . 194
8.1.2 Numerical Methods . . . 194
8.1.3 Experimental Methods . . . 195
8.2 Literature Survey . . . 195
8.2.1 Deveolpment Of Hybrid Test . . . 195
8.2.2 Time Integration Methods For Hybrid Test . . . 195
8.2.2.1 Substructure technique . . . 196
8.2.2.2 Explicit methods . . . 198
8.2.2.3 Implicit Methods . . . 202

9 PARALLEL COMPUTATION **217**
9.1 Using Parallel Computation in Finite-Element Analysis . . . 217
9.2 Parallel Computation in Real-Time Hybrid Testing . . . 218
9.3 Parallelism Using Task-Based Multithreading . . . 218
9.3.1 Parallel Speedup Using a Test Model . . . 219
9.4 Parallel computation with MPI . . . 219
9.4.1 Message Passing Interface(MPI) . . . 220
9.4.2 Determination of inverse matrix with MPI . . . 220
9.4.3 Element state determination with MPI . . . 220

A MATLAB USER's MANUAL **229**
A.1 Preface Block . . . 229
A.1.1 Unit . . . 229
A.1.2 Structural mode . . . 229
A.2 Control Block . . . 229
A.2.1 Analysis . . . 230
A.2.2 Iteration . . . 230
A.2.3 Iteration for element . . . 230
A.2.4 Integration . . . 230
A.2.5 MassInput . . . 231
A.2.6 Convergence Criteria . . . 231
A.3 Geometry Block . . . 232
A.3.1 Nodal coordinates . . . 232
A.3.2 Boundary condition . . . 232
A.4 Element Block . . . 232
A.4.1 Truss element (Sec. 2.1) . . . 233
A.4.2 Stiffness-based 2D beam-column element(Sec. 2.2.1) . . . 233
A.4.3 Flexibility-based 2D beam-column element (Sec. 2.2.2) . . . 233
A.4.4 Zero-length 2D element (Sec. 2.4) . . . 234
A.4.5 Zero-length 2D section element (Sec. 2.5) . . . 235
A.5 Section Block . . . 236
A.5.1 General section . . . 236
A.5.2 Layer section (Sec. 2.3) . . . 236
A.5.3 Fiber section (Sec. 2.3) . . . 237
A.6 Material Block . . . 238

A.6.1 Elastic material . . . 238
A.6.2 Hardening material (Sec. 3.1.1) . . . 238
A.6.3 Bilinear material with isotropic hardening (Sec. 3.1.2) . . . 239
A.6.4 Modified Giuffre-Menegotto-Pinto material (Sec. 3.1.3) . . . 240
A.6.5 Anisotropic damage 1D material (Sec. 3.2.2) . . . 241
A.6.6 Modified Kent and Park model (Sec. 3.2.1) . . . 241
A.7 Force Block . . . 242
A.7.1 Nodal force . . . 242
A.7.2 Nodal displacement . . . 243
A.7.3 Element distributed force . . . 243
A.7.4 Variabel nodal force . . . 243
A.7.5 Variable nodal displacement . . . 244
A.7.6 Variable element distributed force . . . 244
A.7.7 Ground acceleration . . . 244
A.7.8 Variable ground acceleration . . . 245
A.8 Output Block . . . 245
A.8.1 Nodal displacement . . . 245
A.8.2 Nodal velocity . . . 246
A.8.3 Nodal acceleration . . . 246
A.8.4 Nodal force . . . 246
A.8.5 Section axial force . . . 246
A.8.6 Section axial deformation . . . 246
A.8.7 Section moment . . . 247
A.8.8 Section curvature . . . 247
A.8.9 Uniaxial stress and strain . . . 247

B C++ USER's MANUAL **248**

C Notation **249**

BIBLIOGRAPHY **252**

List of Figures

1.1 Simple lumped plasticity models . . . 12
1.2 Constitutive models for nonlinear springs . . . 13
1.3 Lai's model . . . 14
1.4 Otani's model . . . 16
1.5 Multiple spring model . . . 17

2.1 Internal forces and corresponding displacements, 2D truss element . . . 21
2.2 Internal forces and corresponding displacements, stiffness-based 2D element . . . 26
2.3 State determination for stiffness-based method . . . 28
2.4 State determination procedure for stiffness-based method . . . 29
2.5 Flow chart of nonlinear analysis using stiffness-based method (1) . . . 31
2.6 Flow chart of nonlinear analysis using stiffness-based method (2) . . . 32
2.7 Flow chart of nonlinear analysis using stiffness-based method (3) . . . 33
2.8 2D beam-column element without rigid body modes . . . 36
2.9 Sign convention on section force . . . 37
2.10 Internal forces and corresponding displacement,flexibility-based 2D element . . . 39
2.11 The relationship between rigid body modes and no rigid body modes . . . 40
2.12 State determination for flexibility-based method with element iteration . . . 42
2.13 Element and section state determinations . . . 43
2.14 State determination for flexibility-based method without element iteration . . . 48
2.15 State determination procedure for flexibility-based method . . . 51
2.16 Flow chart of nonlinear analysis, flexibility-based w/ element iteration(1) . . . 52
2.17 Flow chart of nonlinear analysis, flexibility-based w/ element iteration(2)) . . . 53
2.18 Flow chart of nonlinear analysis, flexibility-based w/ element iteration(3) . . . 54
2.19 Flow chart of nonlinear analysis, flexibility-based w/o element iteration(1) . . . 58
2.20 Flow chart of nonlinear analysis, flexibility-based w/o element iteration(2) . . . 59
2.21 Flow chart of nonlinear analysis, flexibility-based w/o element iteration(3) . . . 60
2.22 Stress-strain curve of nonliner material . . . 64
2.23 Layer/fiber section . . . 65
2.24 Layer/fiber section state determination . . . 65
2.25 Zero-length 2D element(1) . . . 67
2.26 Zero-length 2D element(2) . . . 68
2.27 Flow chart of zero-length 2D element for element state determination . . . 70
2.28 Zero-length 2D section element(1) . . . 71
2.29 Zero-length 2D section element(2) . . . 72
2.30 Flow chart of zero-length 2D section element for element state determination . . . 74
2.31 Forces of an element . . . 76

2.32 Fixed end action of an element . . . 77
2.33 Example for equivalent fixed end action . . . 77
2.34 Example 1: Frame with layer sections and elastic sections . . . 78
2.35 Example 1: Displacement . . . 79
2.36 Example 1: Reaction . . . 79
2.37 Example 2: Beam . . . 80
2.38 Example 2: Force-displacement . . . 81
2.39 Example 3: Beam . . . 82
2.40 Example 3: Material properties . . . 83
2.41 Example 3: Force-displacement . . . 83
2.42 Example 3: Moment . . . 84

3.1 Uniaxial stress-strain curve for steel . . . 86
3.2 Idealized stress-strain response hardening material . . . 86
3.3 The tangent modulus for isotropic hardening model . . . 88
3.4 The evolution of elastic domain in isotropic hardening model . . . 88
3.5 Isotropic and kinematic hardening plasticity . . . 91
3.6 The evolution of elastic domain in isotropic and kinematic hardening model . . . 92
3.7 Determination for isotropic and kinematic hardening model . . . 95
3.8 Bilinear model . . . 96
3.9 Determination for bilinear model . . . 97
3.10 Menegotto-Pinto steel model . . . 98
3.11 Definition of curvature parameter R in Menegotto-Pinto steel model . . . 99
3.12 Determination (1) for modified Giuffre-Menegotto-Pinto Model . . . 101
3.13 Determination (2) for modified Giuffre-Menegotto-Pinto Model . . . 102
3.14 Determination (3) for modified Giuffre-Menegotto-Pinto Model . . . 103
3.15 Determination (4) for modified Giuffre-Menegotto-Pinto Model . . . 104
3.16 Concrete material model in compression . . . 106
3.17 Concrete material model under cyclic loading in compression . . . 107
3.18 Concrete material model under cyclic loading in tension . . . 109
3.19 Determination (1) for modified Kent and Park model . . . 118
3.20 Determination (2) for modified Kent and Park model . . . 119
3.21 Cracks induced by compression, tension, tension and compression . . . 120
3.22 Determination (1) for the anisotropic damage model . . . 120
3.23 Determination (2) for the anisotropic damage model . . . 121
3.24 Determination (3) for the anisotropic damage model . . . 122

4.1 Illustration of Newton-Raphson iterative method . . . 126
4.2 Illustration of initial stiffness iterative method . . . 128
4.3 Illustration of modified Newton-Raphson iterative method . . . 129

5.1 Rayleigh damping . . . 136
5.2 Flow chart (1) using the Newmark β implicit method . . . 141
5.3 Flow chart (2) using the Newmark β implicit method . . . 142
5.4 Flow chart using the HHT implicit method . . . 146
5.5 Shing iteration scheme using quadratic interpolation . . . 147
5.6 Flow chart using the Initial stiffness method modified P.B. Shing . . . 148

6.1 Integration of Mercury versions and supporting codes . . . 150

6.2 The program architecture of `Mercury` on Matlab in Mercury . . . 151
6.3 Named Composite Hierarchy. . . . 154
6.4 State Construction. . . . 155
6.5 Stiffness Matrix Assembly. . . . 155

7.1 Simple 2D Truss EX 01 . . . 158
7.2 Simple 2D Truss EX 02 . . . 160
7.3 Simple 2D Truss EX 03 . . . 161
7.4 Simple 2D Truss EX 04 . . . 161
7.5 Simple 2D Truss EX 05 . . . 162
7.6 Simple 2D Truss EX 06 . . . 162
7.7 Simple 2D Truss EX 07 . . . 162
7.8 Simple 2D Truss EX 08 . . . 163
7.9 Simple 2D Truss EX 09 . . . 163
7.10 Simple 2D Truss EX 10 . . . 164
7.11 Simple 2D Truss EX 11 . . . 164
7.12 Simple 2D Truss EX 12 . . . 165
7.13 Simple 2D Truss EX 13 to 15 . . . 166
7.14 Ex13 Isotropic Model, Ex 14 Kinematic Model, Ex 15 Hardening Model . . . 167
7.15 Example 13 to 15 Mercury Results, Complete Results for Example 13 to 15 . . . 167
7.16 Ex 16 and Ex 17 for Opensees and Mercury . . . 168
7.17 Ex16 and Ex17 Billinear/Bilinear Isotropic comprasion. . . . 168
7.18 Complete Results for Examples 16 and 17 . . . 168
7.19 Example 18, Example 19 Isotropic Material . . . 169
7.20 Example 18 and 19 Section Axial force/Deformation . . . 169
7.21 Example 21 . . . 170
7.22 Example 22 . . . 171
7.23 Example 23 . . . 171
7.24 Example 24 . . . 172
7.25 Example 25 . . . 172
7.26 Example 26 . . . 173
7.27 Example 23 to 26 . . . 173
7.28 Example 27, x, y, z direction . . . 174
7.29 Example 28, x, y, z direction . . . 175
7.30 Example 29, x direction . . . 176
7.31 Example 30 . . . 176
7.32 Example 110 to 113 setup . . . 177
7.33 Example 110 & 111 acceleration, velocity and displacement . . . 178
7.34 Example 112 & 113 acceleration, velocity and displacement . . . 179
7.35 Example 116 to 119 setup . . . 180
7.36 Example 116 & 117 acceleration, velocity and displacement . . . 181
7.37 Example 118 & 119 acceleration, velocity and displacement . . . 182
7.38 Example 140 setup . . . 183
7.39 Example 140 acceleration, velocity and displacement . . . 184
7.40 Example 141 setup . . . 185
7.41 Example 141 acceleration, velocity and displacement . . . 186
7.42 Test Beam Columns . . . 187
7.43 Load Diagrams . . . 189

7.44 Force Displacement for Column 1, Load Pattern 1 and 2 . . . 189
7.45 Force Displacement for Column 2, Load Pattern 2 . . . 190
7.46 Modified Kent & Park (OpenSees) vs Anisotropic Damage (Mercury), Tol. 10^{-8} 190
7.47 Force-Displacement for column 2 and load 2 . . . 192
7.48 Effect of fixed iteration number in the anisotropic damage model . . . 192
7.49 Number of iterations per increment in anisotropic damage model with 10^{-3} . . . 192
7.50 Effect of fixed iteration number in the anisotropic damage model . . . 193

8.1 Partitioning of a structure (Shing, 2008) . . . 197
8.2 Flow chart of central difference method for hybrid test . . . 199
8.3 Flow chart of Newmark β explicit method for hybrid test . . . 201
8.4 Flow chart of the modified Newmark β explicit method for hybrid test . . . 203
8.5 Modified Newton-Raphson iteration in hybrid test . . . 206
8.6 Iteration scheme using quadratic interpolation . . . 207
8.7 Flow chart (1) of the Newmark β implicit method for hybrid test . . . 208
8.8 Flow chart (2) of the Newmark β implicit method for hybrid test . . . 209
8.9 Flow chart (1) of the HHT implicit method for hybrid test . . . 213
8.10 Flow chart (2) of the HHT implicit method for hybrid test . . . 214
8.11 Flow chart of the Operator-Splitting method for hybrid test . . . 216

9.1 Speedup of multithreaded Mercury compared to non-parallel Mercury . . . 219
9.2 Flow chart of nonlinear analysis with truss element . . . 221
9.3 Flow chart of nonlinear analysis with truss element using the Shing method . . . 222
9.4 Modified Shing method using MPI . . . 223
9.5 Master processor for implementation . . . 224
9.6 Slave processor for determination of the inverse matrix . . . 225
9.7 External excitation loop in master processor . . . 226
9.8 Average elapsed time for element state determination . . . 227
9.9 Speed-up depending on number of processors . . . 228

A.1 Zero-length 2D element . . . 234
A.2 Zero-length 2D section element . . . 235
A.3 Fiber sections . . . 237
A.4 Zero-length 2D element . . . 239
A.5 Bilinear material with isotropic hardening . . . 239
A.6 Modified Giuffre-Menegotto-Pinto material with isotropic hardening . . . 240
A.7 Concrete 1D anisotropic damage model . . . 241
A.8 Concrete tension linear softening model . . . 242

List of Tables

2.1 Comparison on maximum excitation force of Y direction at point B 78

3.1 Steel models 87

5.1 Properties of the Newmark β method 139

7.1 Material Properties 188
7.2 Comparison of CPU time for various loads for a convergence criteria of 10^{-8} . . . 191
7.3 Comparison of CPU time between Modified Kent & Park and Anisotropic Damage 191
7.4 CPU Comparison for anisotropic damage model with various convergence criteria 192

Chapter 1

INTRODUCTION

1.1 Literature Survey

A Review of existing analytical models applicable to nonlinear response of reinforced concrete frame is introduced. Description of material nonlinearity in frame elements can be classified into two main groups: lumped plasticity and distributed plasticity. In the lumped plasticity model, a frame element is made of two zero length nonlinear rotational spring elements and an elastic beam-column element connecting them. The nonlinear material behavior of a frame element is characterized by a moment-rotation relation. Owing to the simplicity of the formulation, the lumped plasticity model is widely used when the computational cost of the analysis is high such as nonlinear time-history analysis of large structures. On the other hand, the main advantage of the distributed plasticity model is that material nonlinearity of the element can develop anywhere along the element and it is thus widely used for more accurate estimation of the nonlinear structural response.

The distributed plasticity model that subdivides the cross section of the member into fibers is also presented in details because of its promising performance and relevance to the frame element.

1.1.1 Plasticity Models

1.1.1.1 Lumped Plasticity Models

Under seismic excitation the inelastic behavior of frame members often concentrates at the end of girders and columns. Hence, an early approach to modeling this behavior was to use parallel or series model as shown in Fig. 1.1.

The earliest element with parallel component elements, shown in Fig. 1.1(a), allows for a bilinear moment-rotation relation (Clough and Johnston 1966). The member consists of two parallel component elements: one elastic component element for yielding and the other elasto-plastic component element to capture strain-hardening. The stiffness matrix of the member is then assumed to be the sum of the two stiffnesses. This model was generalized to multi-linear monotonic behavior allowing for the effect of cracking in reinforced concrete (Takizawa 1976).

The series model, shown in Fig. 1.1(b) (Giberson 1967), consists of a linear elastic com-

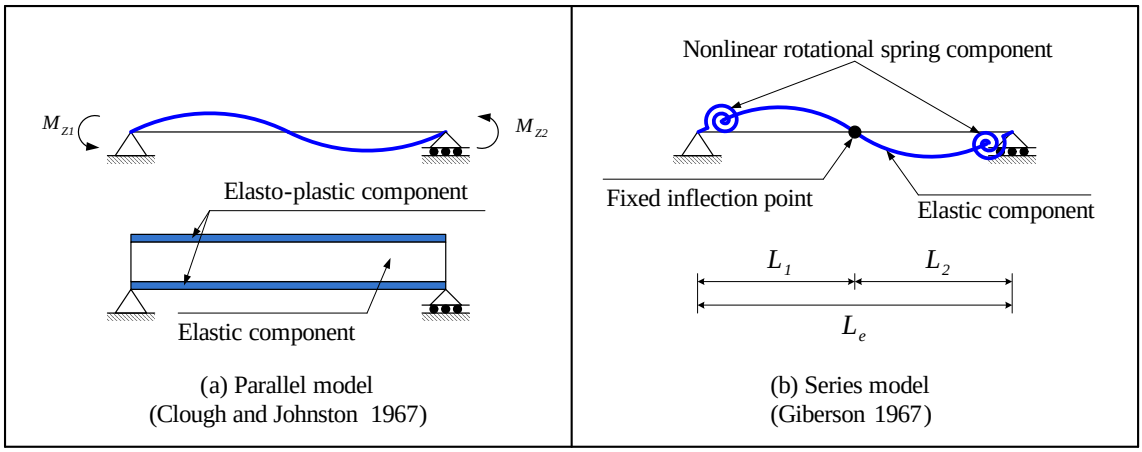


Fig. 1.1 Simple lumped plasticity models (Clough and Benuska 1967, Giberson 1967)

ponent and two equivalent nonlinear rotational spring components attached at the end of the element. The inelastic deformations are lumped into the end springs. This model is more adaptable than the original Clough and Johnston model, since it can describe more complex hysteretic behavior through the selection of appropriate moment-rotation relations for the end springs.

Several lumped plasticity constitutive models have been proposed, Fig. 1.2. Such models may include cyclic stiffness degradation in flexure and shear (Clough and Benuska 1967, Takeda, Sozen and Nielsen 1970, Brancaleoni, Ciampi and Antonio 1983), pinching under cyclic (Banon, Biggs and Irvine 1981, Brancaleoni et al. 1983), and fixed end rotations at the frame joint interface due to bar pull-out (Otani 1974, Filippou and Issa 1988). Usually, axial-flexural coupling is neglected. Nonlinear rate constitutive representations have also been generalized from the basic endochronic theory formulation (Ozdemir 1981) to provide continuous hysteretic relations for the nonlinear springs. An extensive discussion of the mathematical functions that are appropriate for such models is given by Iwan (1978). A critical issue for these models is the selection of parameters which can capture the experimental hysteretic behavior of reinforced concrete. Two basic problems are encountered: (a) the model parameters depend not only on the section characteristics but also on the load and deformation history, thus limiting the generality of the approach, and (b) a consistent and rational method for the selection of model parameters requires special algorithms for ensuring a least squares fit between analytical results and experimental data. Such an algorithm was first proposed by Ciampi and Nicoletti (1986) in a formal system identification method for the selection of parameters to describe the moment-curvature relation proposed by Brancaleoni et al. (1983).

The dependence of flexural strength on the axial force under uniaxial and biaxial flexure has been explicitly included in the modeling of frames and structural walls. In most lumped plasticity models the axial force-bending moment interaction is described by a yield surface for the stress resultants and an associated flow rule according to the tenets of classical plasticity theory (Prager and Hodge 1951). The response is assumed to be linear for stress states that fall within the yield surface in which case the flexural and axial stiffness of the member are uncoupled and independent of the element nodal forces. With the introduction of multiple yield and forcing surfaces and corresponding hardening rules multi-linear constitutive representations that include cracking and cyclic stiffness degradation are possible for the springs (Takayanagi and Schnobrich 1979).

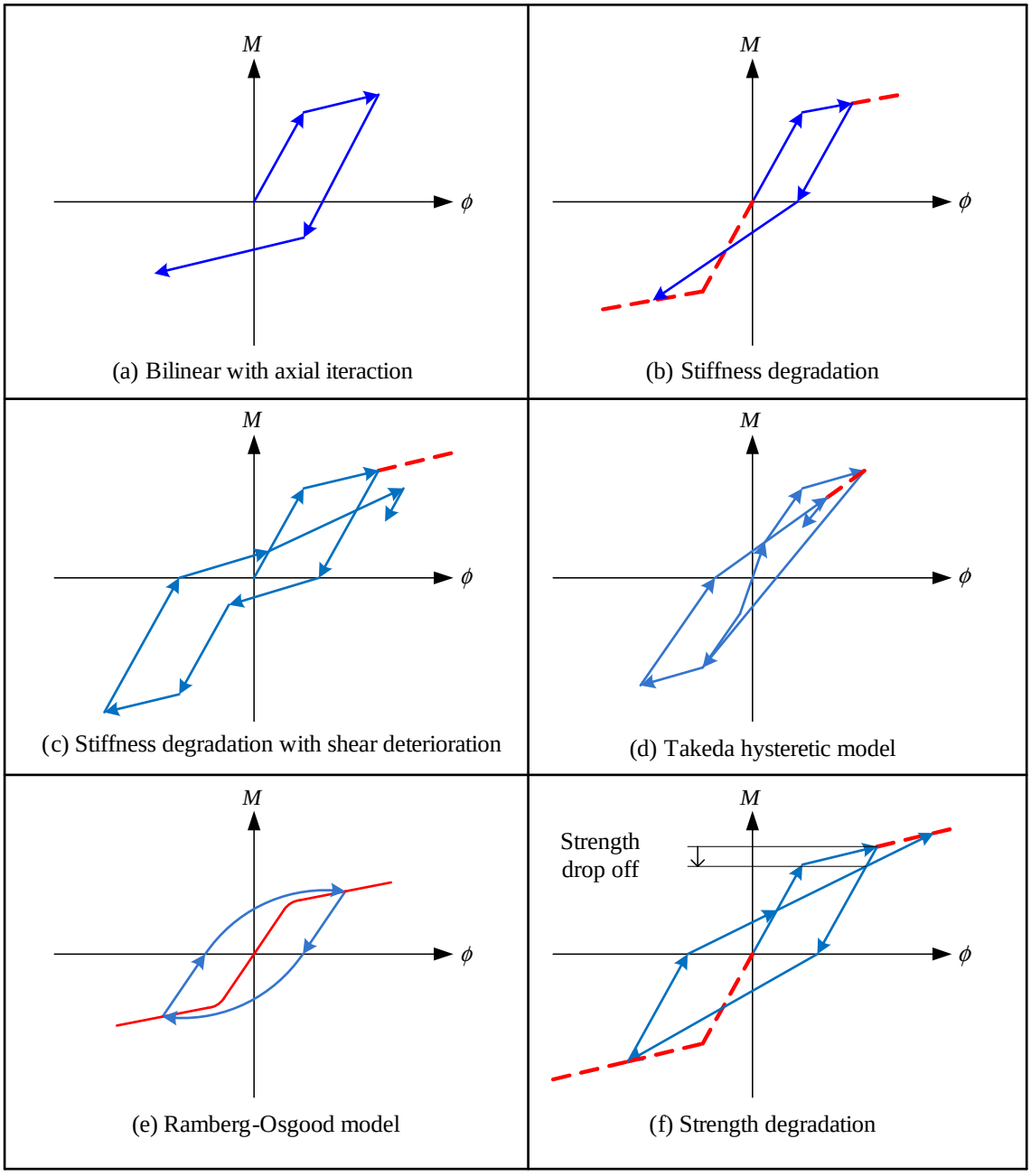


Fig. 1.2 Consititutive models for nonlinear springs in terms of moments and curvatures

A lumped plasticity model is a simplification of the actual behavior that usually involves the gradual spread of inelastic deformations into the member as a function of loading history. This modeling deficiency was recognized in several correlation studies, particularly, those related to large resisting elements of flexural wall-frame structures (Charney and Bertero 1982, Bertero, Aktan, Charney and Sause 1984). Yet the basic advantage of the lumped model remains its simplicity that reduces storage requirements and computational cost and improves the numerical stability of the computations which may no longer be an issue because of improvement in computing technology. Most lumped plasticity models, however, oversimplify certain important aspects of the hysteretic behavior of structures and are, therefore, limited in applicability. One such limitation derives from restrictive a priori assumptions for the determination of the spring parameters. Parametric and theoretical studies of girders under monotonic loading demonstrate a strong dependence between model parameters and the imposed loading pattern and level of inelastic deformation (Anagnostopoulos 1981). Neither factor is likely to remain constant during the dynamic response. The problem is further accentuated by the fluctuation of the axial forces in columns. Because of this history dependence, damage predictions at the global, but also particularly at the local level may be grossly inaccurate. Such information can only be obtained with more refined models capable of describing the hysteretic behavior of the section as a function of axial force. Another limitation of most lumped plasticity models proposed is their inability to describe adequately the deformation softening behavior of reinforced concrete. Such deformation softening can be observed as the reduction in lateral resistance of an axially loaded cantilever column under monotonically increasing lateral tip displacement. Again more advanced models are needed in this case.

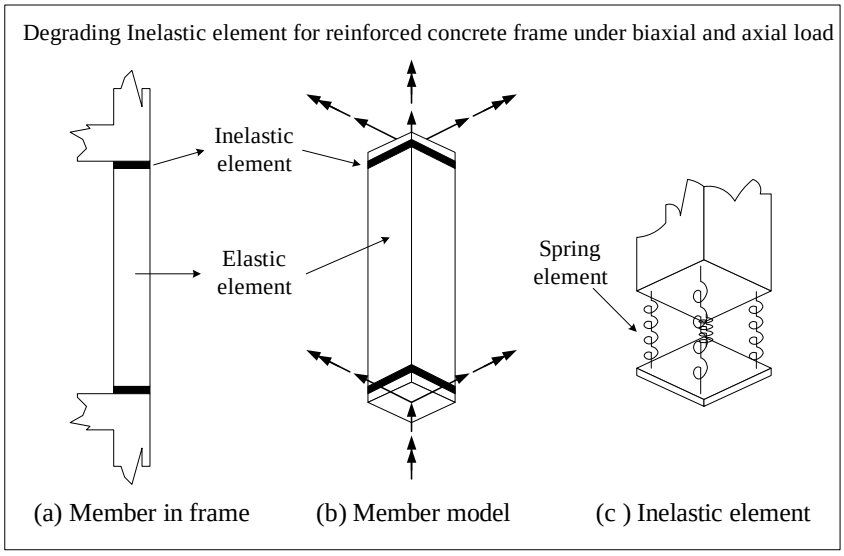


Fig. 1.3 Lai's model (Lai et al. 1984)

The generalization of the nonlinearity theory concepts to stress and strain resultant variables due to moment-rotation or axial force-deformation of reinforced concrete column, such as perfectly plasticity, limits the applicability of these models to well detailed members with large inelastic deformation capacity at the critical regions (Prager and Hodge 1951). For a reinforced concrete column section, the yield surface of the stress resultants is actually a function of a reference strain that couples the corresponding displacement components. This contradicts classical nonlinearity theory which does not account for deformation softening and assumes that

the section deformability is unlimited.

To overcome some of the limitations of classical nonlinearity theory in the description of the interaction between axial force and bending moments, a fiber hinge model that consists of a linear elastic element extending over the entire length of the reinforced concrete and has one inelastic element at each end, as shown in Fig. 1.3, is proposed by Lai et al. (1984). Each inelastic element is made of one inelastic spring at each section corner that represents the longitudinal reinforcing steel and central concrete spring that is effective in compression only. Discretization of these springs at the end sections can simulate the axial force-biaxial moment interaction in reinforced concrete in a more rational way than is possible by classical nonlinearity theory. In Fig. 1.3, the force deformation relation for the effective steel springs follows Takeda's model (Takeda et al. 1970), but the parameters that define the envelope are established from equilibrium considerations.

1.1.1.2 Distributed Plasticity Models

A more accurate model for the inelastic behavior of frame elements is possible with distributed plasticity models. In contrast to lumped plasticity models, material nonlinearity can now occur at any element section and the element behavior is derived by weighted integration of the section response. In practice, since the element integrals are evaluated numerically, only the behaviors of selected sections at the integration points by Gauss-Lobatto or Gauss-Legendre quadrature rule are monitored. Either the element deformations or the element internal forces are the primary unknowns of the model and these are obtained from suitable interpolation functions of the global element displacements or forces, respectively. Discrete cracks are represented as "smeared" over a finite length rather than treated explicitly. The constitutive behavior of the cross section is either formulated in accordance with classical plasticity theory in terms of stress and strain resultants or is explicitly derived by discretization of the cross section into fibers, as is the case in the spread plasticity fiber models. A common assumption of these models is that plane sections remain plane (Bernoulli beam theory), such that the strains vary linearly along the cross section.

Earlier frame models neglected the coupling between axial force and bending moment and, typically, consisted of two cantilever elements that are connected at the fixed point of contra-flexure of the member, as shown in Fig. 1.4. Independent hysteresis rules, the end moment-free end displacement and the end moment-free end rotation relations, are used for the derivation of the stiffness of the two cantilever elements. To overcome some of the numerical difficulties in the element formulations by independent hysteresis rules, Otani (1974) assumed that the inelastic deformations are lumped in two equivalent springs at the ends of the member, thus sacrificing the generality of the model. The global behavior of Otani (1974)'s model is derived by integration of the curvatures along the two cantilever components. The main limitation of this and similar models is the assumption of a fixed point of contra-flexure in the element.

A zone of inelastic deformations gradually spreads from the ends of frame element into the element as a function of loading history (Soleimani, Popov and Bertero 1979), while the rest of the frame element remains elastic. The fixed-end rotations at the ends of frame element are modeled through point hinges inserted at the ends of the element. These are related to the curvature at the corresponding end section through an "effective length" factor which remains constant during the entire response history.

In a model developed by Meyer, Roufaiel and Arzoumanidis (1983), the flexibility coefficients of the model are identical to those proposed by Soleimani et al. (1979), but the way of calculation

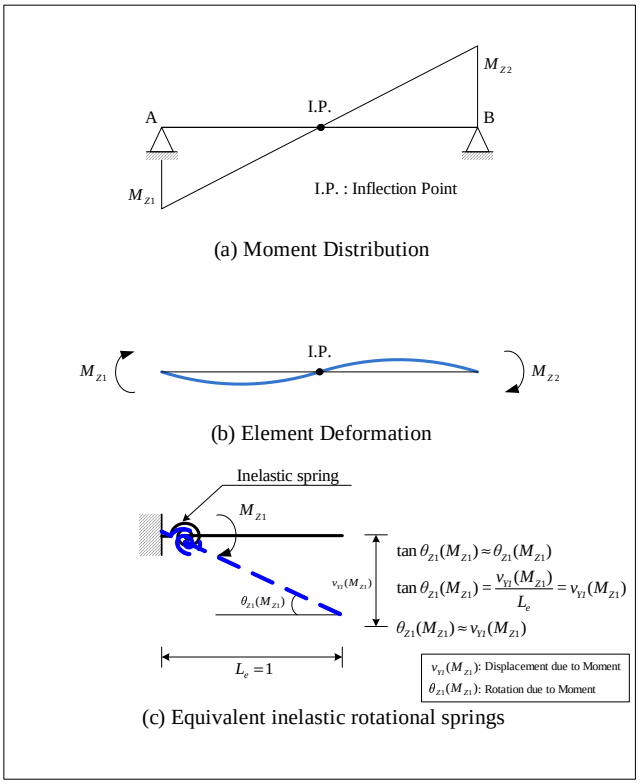


Fig. 1.4 Otani's model (Otani 1974)

the stiffness of the plastic zone during reloading is slightly different. Meyer et al. (1983) used Takeda's model (Takeda et al. 1970) to describe the hysteretic moment-curvature relation not to consider fixed-end rotations.

The original model was later extended to include the effect of shear and axial forces on the flexural hysteretic behavior based on a set of empirical rules (Roufaiel and Meyer 1987) without explaining the variation of axial forces due to overturning moments.

Darvall and Mendis (1985) proposed a similar but simpler model with end inelastic deformations defined through a trilinear moment-curvature relation. Once formed, the end hinges may remain perfectly plastic or exhibit plastic softening or hardening. Perfectly plastic hinges are concentrated at a point, while softening and hardening hinges have a user defined fixed length that is normally assumed to range from $0.75d$ to d where d is the effective depth of the cross section.

Takayanagi and Schnobrich (1979) proposed to divide the element into a finite number of short longitudinal elements, each represented by a nonlinear rotational spring as shown in Fig. 1.5. The properties of a segment depend on the bending moment at its midpoint and are assumed to be constant over the length of the segment. Static condensation is used to reduce this multi-spring model to a single frame element. Even though the nonlinear element behavior is eventually lumped at the end springs, this element belongs to the family of distributed plasticity models because it accounts for inelastic deformations that take place along the element. The multiple spring model was first used in the study of the seismic response of coupled shear walls, which exhibit significant variation of axial force. To account for the interaction between axial force and bending moment a three dimensional limit surface was introduced for the rotational

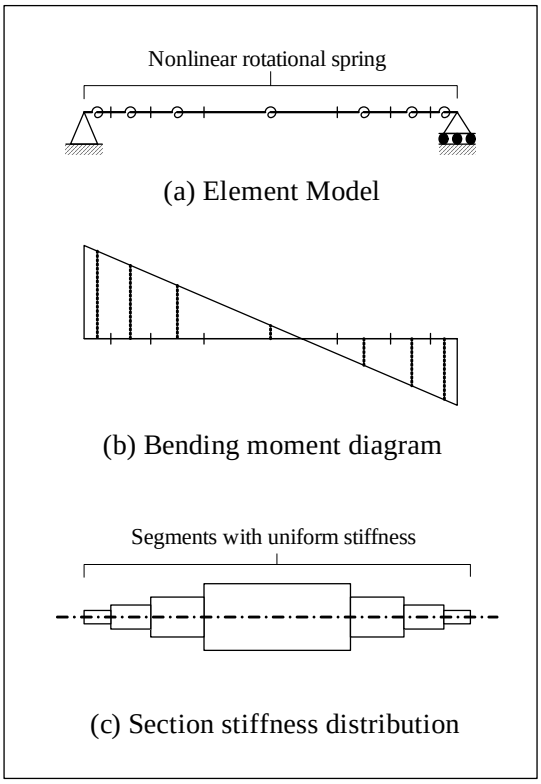


Fig. 1.5 Multiple spring model (Takayanagi and Schnobrich 1979)

springs.

Finally, Filippou and Issa (1988) also subdivide the element in different subelements, but follow a different approach. Each subelement describes a single effect, such as inelastic behavior due to bending, shear behavior at the interface or bond-slip behavior at the frame joint. The interaction between these effects is then achieved by combination of the subelements. This approach allows the hysteretic law of the individual subelement to be simpler, while the member still exhibits a complex hysteretic behavior through the interaction of the different subelements.

1.2 Parallel Computation for transient analysis

It is essential that the program of FE analysis can model as large structure as possible in real time. Generally, the time increment in a seismic simulation is 0.01 sec. Henceforth, it should be able to perform a full nonlinear analysis for a time step in exactly 0.01 sec. This requires a parallel program to decrease computational time.

In FE analysis, we note that there are two major CPU intensive operations:

1. Determination of inverse matrix
2. Element state determination for force recovery.

That is, computational time depends on determination of inverse matrix and element state determination.

Iterative methods except for initial stiffness method have determination of inverse matrix and element state determination every time step or iteration. Mercury which is the program of FE analysis in research uses several iterative methods for transient analysis. However, the program for parallel processing focuses on the Shing method which uses initial stiffness method. The Shing method has a disadvantage that as the analysis proceeds, the errors introduced by the initial stiffness matrix, and the fixed number of iterations (about 10) increase (due to nonlinearity) even if computational time of it is faster than it of other iterative methods. To overcome it, Mercury needs to update stiffness matrix which is made by nonlinearity during FE analysis, and to determine inverse matrix with it. It may increase computational time compared with the Shing method. For this reason, Mercury will introduce a modified Shing method which uses modified initial stiffness matrix instead of initial stiffness matrix. The modified Shing method includes two processors to determine inverse matrix with modified initial stiffness which is not computed regularly. One processor is for element state determination and other for determination of inverse matrix. It will be described in Sec. 9.4.

As other method to reduce computational time, Mercury will also introduce the parallel processing for element state determination. In case, it can use several processors. It will also be described in Sec. 9.4.

Chapter 2

ELEMENT FORMULATION

This chapter will describe the formulation of the elements in Mercury. It will start with the simplest one, truss and then proceed with beam-column of increasing levels of complexity. Finally zero length element and section element will be addressed.

2.1 Truss Element Formulation

The truss element has only one degree of freedom associated with each node in local reference whether in 2D or 3D.

2.1.1 Formulation

As with all finite elements, stiffness matrix derivation hinges on three requirements.

1. **Compatibility**

   Displacement Section displacements are determined from the element nodal displacements through the shape function. Because truss element has only one d.o.f per node, the generalized relationship between section displacement vector $\mathbf{d}_s(x)$ and element nodal displacement vector $\overline{\mathbf{d}}_e$ can be expressed as

$$\begin{aligned}\mathbf{d}_s(x) &= \left\{ \begin{array}{c} u(x) \end{array} \right\} \\ &= \underbrace{\left[\begin{array}{cc} -\frac{x}{L}+1 & \frac{x}{L} \end{array} \right]}_{\mathbf{N}_d(x)} \cdot \underbrace{\left\{ \begin{array}{c} \overline{u}_{x1} \\ \overline{u}_{x2} \end{array} \right\}}_{\overline{\mathbf{d}}_e}\end{aligned}$$

   where $\mathbf{N}_d(x)$ is the matrix of linear interpolation functions.

   Deformation Under the assumption that displacements are small, the section deforma-

tion vector $\boldsymbol{\varepsilon}_s(x)$ is related to the element nodal displacement vector by

$$\begin{aligned}\boldsymbol{\varepsilon}_s(x) &= \left\{ \begin{array}{c} \varepsilon_x(x) \end{array} \right\} \\ &= \underbrace{\left[\begin{array}{cc} -\frac{1}{L} & \frac{1}{L} \end{array} \right]}_{\mathbf{B}_d(x)} \cdot \overline{\mathbf{d}}_e \end{aligned} \tag{2.1}$$

where $\mathbf{B}_d(x)$ is the matrix obtained from the derivatives of $\mathbf{N}_d(x)$.

2. **Constitutive law**

   Section constitutive law is expressed as

$$\underbrace{\left\{ \begin{array}{c} N_x(x) \end{array} \right\}}_{\boldsymbol{\sigma}_s(x)} = \mathbf{k}_s(x) \cdot \boldsymbol{\varepsilon}_s(x) \tag{2.2}$$

   where $\boldsymbol{\sigma}_s(x)$ is the section[1] force vector, and $\mathbf{k}_s(x)$ is the section stiffness matrix.

   For linear elastic analysis $\mathbf{k}_s(x)$ is simply equal to

$$\mathbf{k}_s(x) = \left[\begin{array}{c} E(x) \cdot A(x) \end{array} \right]$$

   where, $E(x)$ and $A(x)$ are elastic modulus and cross sectional area.

3. **Equilibrium**

   For the force distribution to be in equilibrium, the relationship between the element nodal force vector $\overline{\mathbf{f}}_e$ in local reference and the section deformation vector is derived from the principle of virtual displacement

$$\underbrace{\delta \overline{\mathbf{d}}_e^T \cdot \overline{\mathbf{f}}_e}_{\text{External}} = \underbrace{\int_0^{L_e} \delta \boldsymbol{\varepsilon}_s(x)^T \boldsymbol{\sigma}_s(x) \, \mathrm{d}x}_{\text{Internal}} \tag{2.3}$$

   which is a weak form of equilibrium.

   Substitution of Eq. 2.1 and Eq. 2.2 into Eq. 2.3 and since the latter must hold for any

[1] The notion of section is not essential to understand the formulation of the truss element stiffness matrix. It is nevertheless introduced to be consistent with the subsequent formulation of beam-column (Sec. 2.2)

arbitrary $\delta\overline{\mathbf{d}}_e$ lead to

$$\begin{aligned}
\delta\overline{\mathbf{d}}_e^T \cdot \overline{\mathbf{f}}_e &= \int_0^{L_e} \delta\overline{\mathbf{d}}_e^T \cdot \mathbf{B}_d(x)^T \cdot \mathbf{k}_s(x) \cdot \boldsymbol{\varepsilon}_s(x)\ \mathrm{d}x \\
\overline{\mathbf{f}}_e &= \int_0^{L_e} \mathbf{B}_d(x)^T \cdot \mathbf{k}_s(x) \cdot \boldsymbol{\varepsilon}_s(x)\ \mathrm{d}x \\
&= \underbrace{\int_0^{L_e} \mathbf{B}_d(x)^T \cdot \mathbf{k}_s(x) \cdot \mathbf{B}_d(x)\ \mathrm{d}x}_{\overline{\mathbf{k}}_e} \cdot \overline{\mathbf{d}}_e \\
\overline{\mathbf{f}}_e &= \overline{\mathbf{k}}_e \cdot \overline{\mathbf{d}}_e
\end{aligned}$$

The element stiffness matrix in local reference is thus given by

$$\overline{\mathbf{k}}_e = \int_0^{L_e} \mathbf{B}_d(x)^T \cdot \mathbf{k}_s(x) \cdot \mathbf{B}_d(x)\ \mathrm{d}x$$

2.1.2 Coordinate system for 2D truss element

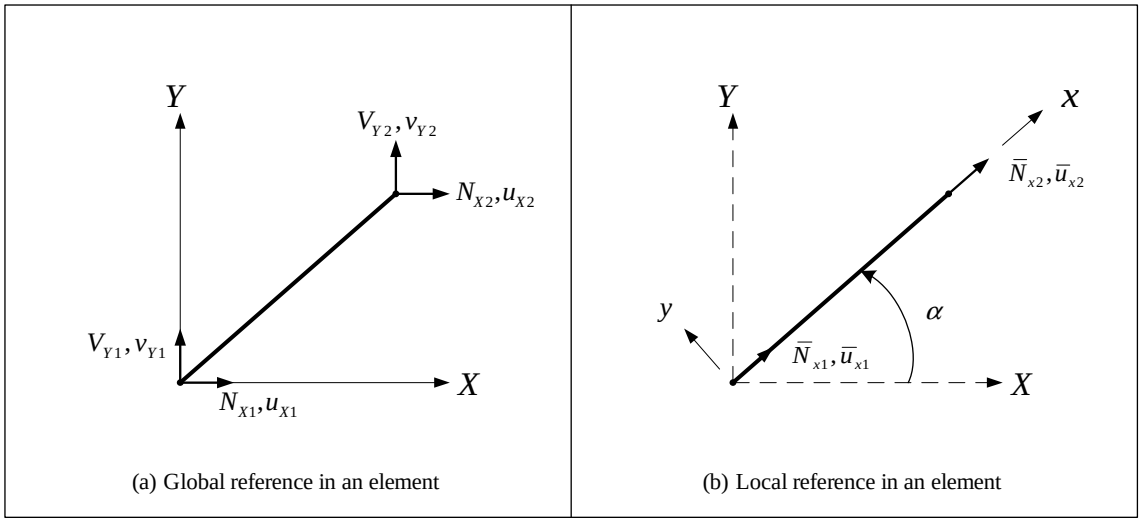


Fig. 2.1 Internal forces and corresponding displacements in global and local coordinate system for the 2D truss element

The element nodal forces and displacements are expressed with respect to the global reference, Fig. 2.1(a) as:

The element nodal force vector in global reference is given by

$$\mathbf{F}_e = \lfloor N_{X1},\ V_{Y1},\ N_{X2},\ V_{Y2} \rfloor^T$$

and the element nodal displacement vector by

$$\boldsymbol{\delta}_e = \lfloor u_{X1},\ v_{Y1},\ u_{X2},\ v_{Y2} \rfloor^T$$

Furthermore, the element nodal forces and displacements can be expressed with respect to the local reference as shown in Fig. 2.1(b). In which case we would have

$$\overline{\mathbf{f}}_e = \lfloor \overline{N}_{x1},\ \overline{N}_{x2} \rfloor^T$$

for the element nodal force vector, and

$$\overline{\mathbf{d}}_e = \lfloor \overline{u}_{x1},\ \overline{u}_{x2} \rfloor^T$$

for the element nodal displacement vector.

The rotation matrix which transform global reference, Fig. 2.1(a), to local reference ,Fig. 2.1(b), is given by $\boldsymbol{\Gamma}_e$ such that

$$\begin{aligned} \overline{\mathbf{f}}_e &= \boldsymbol{\Gamma}_e \cdot \mathbf{F}_e \\ \overline{\mathbf{d}}_e &= \boldsymbol{\Gamma}_e \cdot \boldsymbol{\delta}_e \\ \mathbf{K}_e &= \boldsymbol{\Gamma}_e^T \cdot \overline{\mathbf{k}}_e \cdot \boldsymbol{\Gamma}_e \end{aligned}$$

where, $\mathbf{K}_e$ is the element stiffness matrix in global reference and the rotation matrix is

$$\boldsymbol{\Gamma}_e = \left[\begin{array}{c|cccc} & N_{X1} & V_{Y1} & N_{X2} & V_{Y2} \\ \hline \overline{N}_{x1} & \cos\alpha & \sin\alpha & 0 & 0 \\ \overline{N}_{x2} & 0 & 0 & \cos\alpha & \sin\alpha \end{array}\right]$$

2.1.3 State determination

In a nonlinear structural analysis, state determination for 2D truss element is identical to the stiffness-based 2D beam-column element. It will be discussed in Sec. 2.2.1.3 and 2.2.1.4.

2.2 Beam-Column Element Formulations

The formulation of nonlinear frame structure using elements with layer/fiber sections can be achieved using the stiffness-based (displacement-based) method or the flexibility-based (force-based) method. In our implementation both methods will assume that deformations are small and plane sections remain plane during the loading history. The stiffness-based method assumes displacement interpolation functions along the element. This assumption will require a sufficient number of elements to achieve an accurate response. On the other hand, the flexibility-based method will have force interpolation functions along the element and a smaller number of elements than for the stiffness-based method will be needed to achieve the same level of accuracy.

However, the element formulation in the flexibility-based method is more complex than that of the stiffness-based method because of the need to iterate in order to obtain ε from material

constitutive models which are usually expressed as $\sigma = \sigma(\varepsilon)$ (Lee and Mosalam 2004).

In addition, the element formulation in the stiffness-based method is more straightforward and widely used in conventional finite element method. In this study both the stiffness and flexibility-based methods will be used to formulate the distributed nonlinear model with Bernoulli beam theory.

Elements will be characterized by several cross-sections located at the numerical integration points, defined by Gauss-Legendre quadrature rule for stiffness-based method or Gauss-Lobatto quadrature rule for flexibility-based method. Furthermore, each section will have a suitable number of layers/fibers where each one is under uniaxial state of stress.

2.2.1 Stiffness-Based 2D Beam-Column Element

We next formulate the stiffness matrix of the beam-column elements by considering the "classical" stiffness-based formulation first. The formulation assumes that the axial displacements are described by linear polynomial interpolation functions while the transverse displacements are modeled by cubic ones.

2.2.1.1 Formulation

As for the truss element we consider each of the three requirements which must be met.

1. **Compatibility**

   Displacement Section displacements are determined from the element nodal displacements through the shape functions. The generalized relationship between section displacement vector $\mathbf{d}_s(x)$ and the element nodal displacement vector $\overline{\mathbf{d}}_e$ can be expressed as

$$\begin{aligned}\mathbf{d}(x) &= \left\{ \begin{array}{c} u(x) \\ v(x) \end{array} \right\} \\ &= \mathbf{N}_d(x) \cdot \underbrace{\lfloor \; \overline{u}_{x1}, \quad \overline{v}_{y1}, \quad \overline{\theta}_{z1}, \quad \overline{u}_{x2}, \quad \overline{v}_{y2}, \quad \overline{\theta}_{z2} \; \rfloor^T}_{\overline{\mathbf{d}}_e}\end{aligned}$$

   where $\mathbf{N}_d(x)$ is the matrix of displacement interpolation functions.

   $\mathbf{N}_d(x)$ is a matrix of linear interpolation functions for axial displacements and cubic ones for the transverse displacements

$$\mathbf{N}_d(x) = \left[\begin{array}{cccccc} \psi_1(x) & 0 & 0 & \psi_2(x) & 0 & 0 \\ 0 & \phi_1(x) & \phi_2(x) & 0 & \phi_3(x) & \phi_4(x) \end{array} \right]$$

   where ψ_1, ψ_2, ϕ_1, ϕ_2, ϕ_3 and ϕ_4 are the interpolation functions for axial and trans-

verse displacements and notation expressed by

$$\begin{aligned} \psi_1(x) &= -\tfrac{x}{L_e} + 1 & \psi_2(x) &= \tfrac{x}{L_e} \\ \phi_1(x) &= 2\tfrac{x^3}{{L_e}^3} - 3\tfrac{x^2}{{L_e}^2} + 1 & \phi_2(x) &= \tfrac{x^3}{{L_e}^2} - 2\tfrac{x^2}{L_e} + x \\ \phi_3(x) &= -2\tfrac{x^3}{{L_e}^3} + 3\tfrac{x^2}{{L_e}^2} & \phi_4(x) &= \tfrac{x^3}{{L_e}^2} - \tfrac{x^2}{L_e} \end{aligned}$$

Deformation Under the assumptions that displacements are small and plane sections remain plane, the section deformation vector $\boldsymbol{\varepsilon}_s(x)$ (axial strain $\varepsilon_x(x)$ and curvature $\phi_z(x)$) is related to the element nodal displacement vector by

$$\begin{aligned} \boldsymbol{\varepsilon}_s(x) &= \left\{ \begin{array}{c} \varepsilon_x(x) \\ \phi_z(x) \end{array} \right\} \\ &= \mathbf{B}_d(x) \cdot \overline{\mathbf{d}}_e \end{aligned} \tag{2.4}$$

where $\mathbf{B}_d(x)$ is the matrix derived from the appropriate derivatives of the displacement interpolation functions.

$$\mathbf{B}_d(x) = \left[\begin{array}{cccccc} \psi_1'(x) & 0 & 0 & \psi_2'(x) & 0 & 0 \\ 0 & \phi_1''(x) & \phi_2''(x) & 0 & \phi_3''(x) & \phi_4''(x) \end{array} \right]$$

with

$$\begin{aligned} \psi_1'(x) &= -\tfrac{1}{L_e} & \psi_2'(x) &= \tfrac{1}{L_e} \\ \phi_1''(x) &= \tfrac{12x}{{L_e}^3} - \tfrac{6}{{L_e}^2} & \phi_2''(x) &= \tfrac{6x}{{L_e}^2} - \tfrac{4}{L_e} \\ \phi_3''(x) &= -\tfrac{12x}{{L_e}^3} + \tfrac{6}{{L_e}^2} & \phi_4''(x) &= \tfrac{6\,x}{{L_e}^2} - \tfrac{2}{L_e} \end{aligned}$$

2. **Constitutive law**

Section constitutive law relates axial strain and curvature to axial force and moment

$$\underbrace{\left\{ \begin{array}{c} N_x(x) \\ M_z(x) \end{array} \right\}}_{\boldsymbol{\sigma}_s(x)} = \mathbf{k}_s(x)\ \boldsymbol{\varepsilon}_s(x) \tag{2.5}$$

where $\boldsymbol{\sigma}_s(x)$ is the section force vector, and $\mathbf{k}_s(x)$ is the section stiffness matrix.

If $\mathbf{k}_s(x)$ is not derived from layer/fiber discretization of the cross section, and for linear elastic case $\mathbf{k}_s(x)$ is simply equal to

$$\mathbf{k}_s(x) = \left[\begin{array}{cc} E(x) \cdot A(x) & 0 \\ 0 & E(x) \cdot I_z(x) \end{array} \right] \tag{2.6}$$

where, $E(x)$, $A(x)$, and $I_z(x)$ are elastic modulus, cross sectional area, and section moment of inertia on cross sectional area.

3. **Equilibrium**

   Assuming a force distribution which is in equilibrium, the relationship between the element nodal force vector in local reference $\overline{\mathbf{f}}_e$ and the section deformation vector is derived from the principle of virtual work

$$\underbrace{\delta\overline{\mathbf{d}}_e^T \cdot \overline{\mathbf{f}}_e}_{\text{External}} = \underbrace{\int_0^{L_e} \delta\boldsymbol{\varepsilon}_s(x)^T \cdot \boldsymbol{\sigma}_s(x)\ \mathrm{d}x}_{\text{Internal}} \tag{2.7}$$

   which is a weak form of equilibrium.

   Substitution of Eq. 2.4 and Eq. 2.5 into Eq. 2.7 and since the latter must hold for any arbitrary $\delta\overline{\mathbf{d}}_e$, this leads to

$$\begin{aligned}
\delta\overline{\mathbf{d}}_e^T \cdot \overline{\mathbf{f}}_e &= \int_0^{L_e} \delta\overline{\mathbf{d}}_e^T \cdot \mathbf{B}_d(x)^T \cdot \mathbf{k}_s(x) \cdot \boldsymbol{\varepsilon}_s(x)\ \mathrm{d}x \\
\overline{\mathbf{f}}_e &= \int_0^{L_e} \mathbf{B}_d(x)^T \cdot \mathbf{k}_s(x) \cdot \boldsymbol{\varepsilon}_s(x)\ \mathrm{d}x \\
&= \underbrace{\int_0^{L_e} \mathbf{B}_d(x)^T \cdot \mathbf{k}_s(x) \cdot \mathbf{B}_d(x)\ \mathrm{d}x}_{\overline{\mathbf{k}}_e} \cdot \overline{\mathbf{d}}_e \\
\overline{\mathbf{f}}_e &= \overline{\mathbf{k}}_e \cdot \overline{\mathbf{d}}_e
\end{aligned}$$

   The element stiffness matrix in local reference is thus given by

$$\overline{\mathbf{k}}_e = \int_0^{L_e} \mathbf{B}_d(x)^T \cdot \mathbf{k}_s(x) \cdot \mathbf{B}_d(x)\ \mathrm{d}x \tag{2.8}$$

2.2.1.2 Coordinate system for stiffness-based 2D beam-column with Bernoulli beam theory

The element nodal forces and displacements are expressed with respect to the global reference, Fig. 2.2(a) as

$$\mathbf{F}_e = \lfloor N_{X1},\ V_{Y1},\ M_{Z1},\ N_{X2},\ V_{Y2},\ M_{Z2} \rfloor^T$$

for the element nodal force vector, and

$$\boldsymbol{\delta}_e = \lfloor u_{X1},\ v_{Y1},\ \theta_{Z1},\ u_{X2},\ v_{Y2},\ \theta_{Z2} \rfloor^T$$

for the element nodal displacement vector.

Furthermore, the element nodal forces and displacements of the element can be expressed

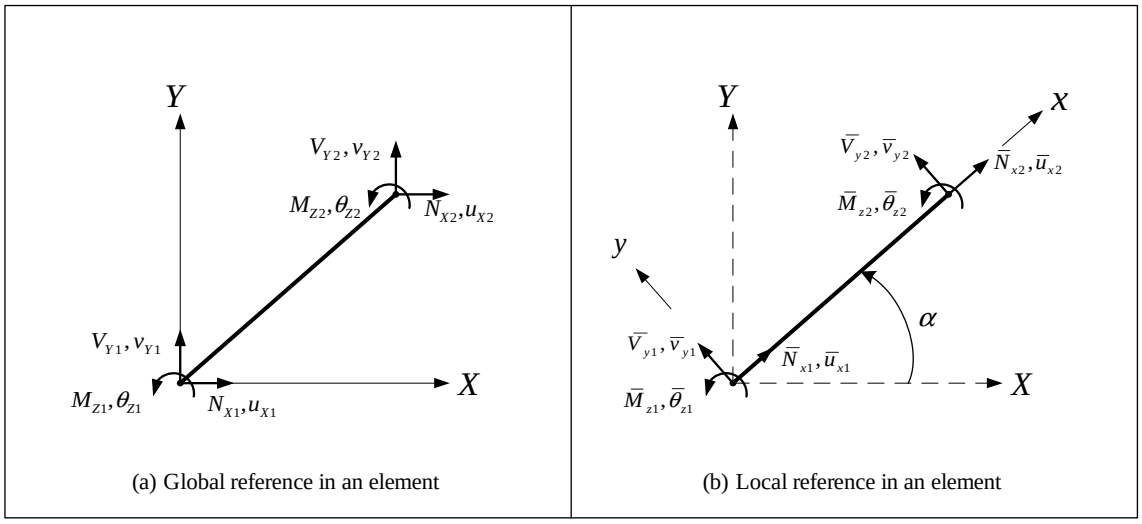


Fig. 2.2 Internal forces and corresponding displacements in global and local coordinate system for the stiffness-based 2D beam-column element

with respect to the local reference as shown in Fig. 2.2(b)

$$\overline{\mathbf{f}}_e = \lfloor \overline{N}_{x1},\ \overline{V}_{y1},\ \overline{M}_{z1},\ \overline{N}_{x2},\ \overline{V}_{y2},\ \overline{M}_{z2} \rfloor^T$$

for the element nodal force vector, and

$$\overline{\mathbf{d}}_e = \lfloor \overline{u}_{x1},\ \overline{v}_{y1},\ \overline{\theta}_{z1},\ \overline{u}_{x2},\ \overline{v}_{y2},\ \overline{\theta}_{z2} \rfloor^T$$

for the element nodal displacement vector.

The rotation matrix which transforms from global reference, Fig. 2.2(a), to local reference, Fig. 2.2(b), is given by $\mathbf{\Gamma}_e$ such that

$$\begin{aligned} \overline{\mathbf{f}}_e &= \mathbf{\Gamma}_e \cdot \mathbf{F}_e \\ \overline{\mathbf{d}}_e &= \mathbf{\Gamma}_e \cdot \boldsymbol{\delta}_e \\ \mathbf{K}_e &= \mathbf{\Gamma}_e^T \cdot \overline{\mathbf{k}}_e \cdot \mathbf{\Gamma}_e \end{aligned} \tag{2.9}$$

where, $\mathbf{K}_e$ is the element stiffness matrix in global reference.

The rotation matrix is then given by

$$\mathbf{\Gamma}_e = \left[\begin{array}{c|cccccc} & N_{X1} & V_{Y1} & M_{Z1} & N_{X2} & V_{Y2} & M_{Z2} \\ \hline \overline{N}_{x1} & \cos\alpha & \sin\alpha & 0 & 0 & 0 & 0 \\ \overline{V}_{y1} & -\sin\alpha & \cos\alpha & 0 & 0 & 0 & 0 \\ \overline{M}_{z1} & 0 & 0 & 1 & 0 & 0 & 0 \\ \overline{N}_{x2} & 0 & 0 & 0 & \cos\alpha & \sin\alpha & 0 \\ \overline{V}_{y2} & 0 & 0 & 0 & -\sin\alpha & \cos\alpha & 0 \\ \overline{M}_{z2} & 0 & 0 & 0 & 0 & 0 & 1 \end{array}\right]$$

2.2.1.3 State determination

In a nonlinear structural analysis, the nodal force vector at each step corresponds to an incremental external nodal force vector to the structure. The corresponding incremental nodal displacement vector of structure is determined from an incremental external nodal force vector and the element nodal displacement vectors are extracted for each element. The element nodal displacements are used to determine internal section forces. The process of determining the internal nodal forces that corresponds to the element nodal displacements is known as state determination. The state determination process is made up of three phases as shown in Fig. 2.3:

1. Section state determination, Fig. 2.3(c)

   Internal section forces are computed from section deformations which are in turn determined from element nodal displacements

2. Element state determination, Fig. 2.3(b)

   The element tangent stiffness matrices and internal element nodal forces of each element are determined from the internal section forces for each element which are in turn computed from section deformations

3. Structure state determination, Fig. 2.3(a)

   The element tangent stiffness matrices and internal element force vectors of each element are assembled to form the augmented tangent stiffness matrix $\mathbf{K}_S^{tan}$ and internal nodal force vector $\mathbf{P}_S^{int}$ ($\mathbf{P}_S^{int} = \mathbf{P}_t^{int} + \mathbf{P}_u^{int}$) of the structure. Where, subscript t and u are associated with free end constrained degrees of freedom respectively.

Once the structure state determination is complete, the internal nodal force vector is compared with the total applied external nodal force vector and the difference, if any, yields the residual nodal force vector which is then reapplied to the structure in an iterative solution process until the difference of total applied external force vector and internal nodal force vector satisfies equilibrium within a specified tolerance.

The state determination procedure is straightforward for a stiffness-based 2D beam-column element. The section deformation vectors $\boldsymbol{\varepsilon}_{s,e}(x)$ are determined from the element nodal displacement vector $\overline{\mathbf{d}}_e$ as shown in Eq. 2.4. The corresponding section tangent stiffness matrices $\mathbf{k}_{s,e}^{tan}(x)$ and the internal section force vectors $\boldsymbol{\sigma}_{s,e}^{int}(x)$ are determined from the section constitutive law, Eq. 2.5. The element tangent stiffness matrices $\overline{\mathbf{k}}_e^{tan}$ are obtained from Eq. 2.8, while the internal element nodal force vectors $\overline{\mathbf{f}}_e^{int}$ are determined from the principle of virtual work

$$\overline{\mathbf{f}}_e^{int} = \int_0^{L_e} \mathbf{B}_{d,e}(x)^T \cdot \boldsymbol{\sigma}_{s,e}^{int}(x) \mathrm{d}x \tag{2.10}$$

It is important to note that this method leads to an erroneous element response in the nonlinear case. This problem is again illustrated by Fig. 2.3 which shows the evolution of the structure, element and section states during one incremental external force vector at free degrees of freedom $\Delta\mathbf{P}_{t,n}$ that requires several iterations k (Spacone, Ciampi and Filippou 1992). Where, subscript n refers to external force step, superscript k the k^{th} iteration within the external force step n.

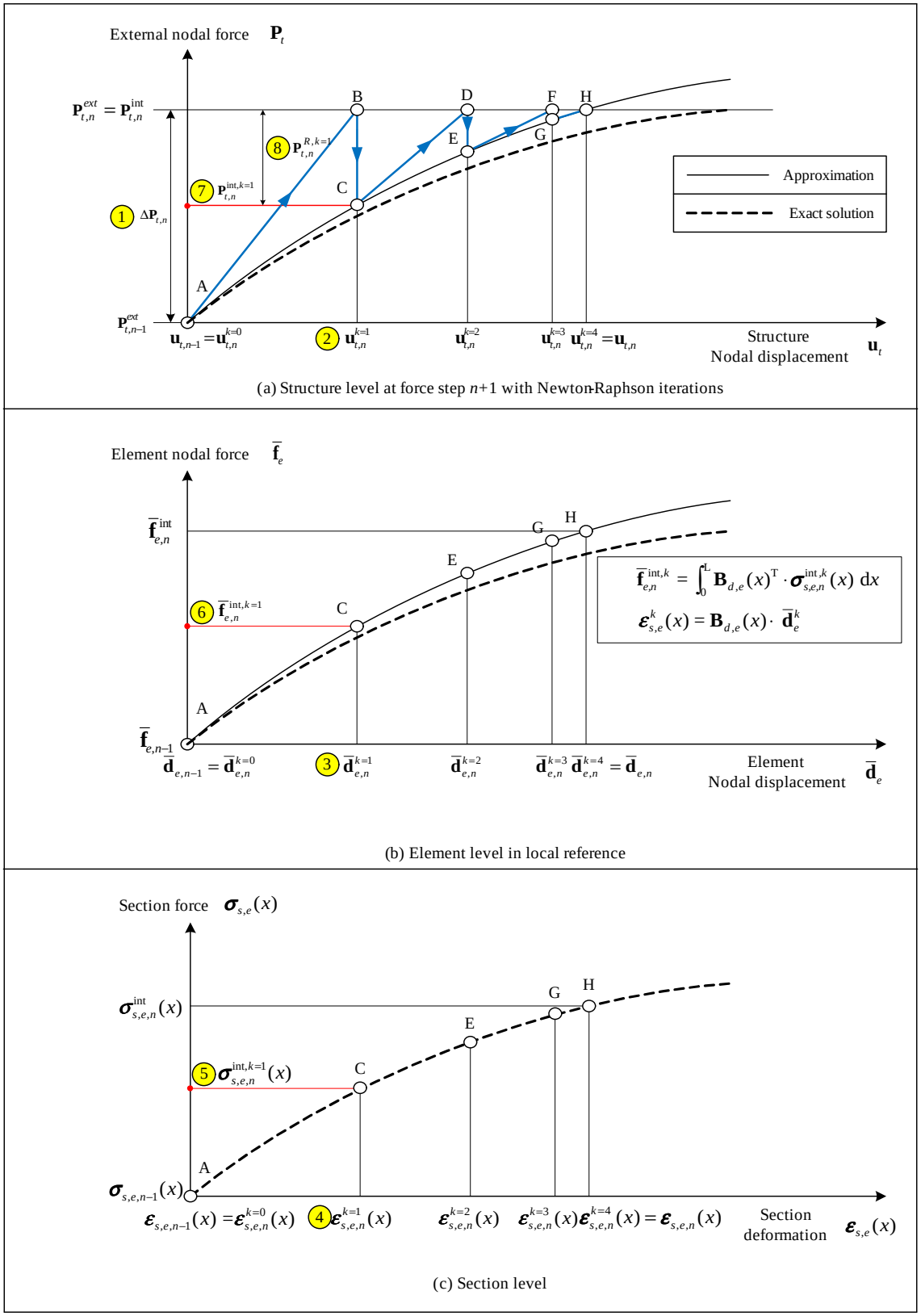


Fig. 2.3 State determination for stiffness-based method

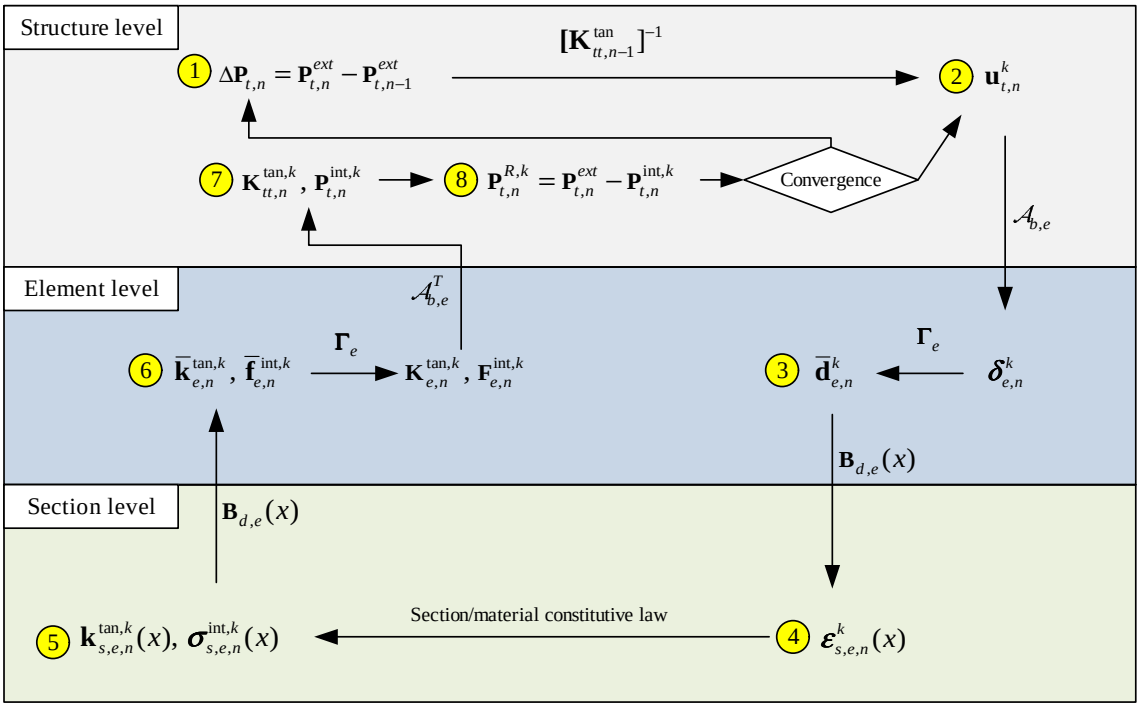
The Newton-Raphson iteration method operates in the global coordinate (structural level) system. At each Newton-Raphson iteration, the incremental nodal displacement vector at the structure level is determined from the nodal incremental force vector, and then the total element nodal displacement vectors are determined next.

At the k^{th} Newton-Raphson iteration, the element nodal displacement vectors in local reference $\overline{\mathbf{d}}_{e,n}$ of the nodal displacement vector $\mathbf{u}^k_{S,n}$ ($\mathbf{u}^k_{S,n} = \mathbf{u}^k_{t,n} + \mathbf{u}^k_{u,n}$) of degrees of freedom at the structure level are determined for each element.

Using Eq. 2.5 the section deformation vectors $\boldsymbol{\varepsilon}^k_{s,e,n}(x)$ for each element are computed. This is the first approximation of the element state determination, since $\mathbf{B}_{d,e}(x)$ is exact only in the linear elastic case of a prismatic member.

Assuming that the section constitutive law is explicitly known, the section tangent stiffness matrices $\mathbf{k}^{tan,k}_{s,e,n}(x)$ and the internal section force vectors $\boldsymbol{\sigma}^{int,k}_{s,e,n}(x)$ are readily determined from $\boldsymbol{\varepsilon}^k_{s,e,n}(x)$. Then, using Eq. 2.8 and Eq. 2.10 the element stiffness matrices $\overline{\mathbf{k}}^k_{e,n}$ in local reference and the internal element nodal force vectors in local reference $\overline{\mathbf{f}}^{int,k}_{e,n}$ are determined next.

Fig. 2.4 is the summary of the procedure on all state determinations.



where, $\mathbf{A}_{b,e}$ and $\mathbf{A}^T_{b,e}$ are the displacement extracting operator and the force assembling operator.

Fig. 2.4 State determination procedure for stiffness-based method

Since $\mathbf{B}_{d,e}(x)$ is only approximate, the integrals for the element tangent stiffness matrix in local reference and the internal element nodal force vector in local reference will also yield approximate results. The approximation of $\mathbf{B}_{d,e}(x)$ leads to stiffer solution, Fig. 2.3 (a) and (b). Note that the curve labeled "Exact solution" is only exact within the assumptions of the section constitutive law and the kinematic approximations of the problem within in assumptions that deformations are small and plane sections remain plane.

To overcome the numerical errors that arise from the approximation of $\mathbf{B}_{d,e}(x)$, analysts resort to fine mesh discretization of the structure, especially, in frame regions that undergo

highly nonlinear behaviors, such as the member ends. Even so, numerical convergence problems persist. This is precisely why the flexibility based element will yield better results. This will be discussed later.

2.2.1.4 Nonlinear analysis using stiffness-based formulation

With reference to Fig. 2.4 to 2.7, we will examine one single step of the Newton-Raphson mehthod for the nonlinear analysis with section constitutive law. Force and displacement control, mentioned here, are discussed in Sec. 4.1. Layer or fiber section procedure in Fig. 2.7 will be described in Sec. 2.3.

Step numbers are shown in Fig. 2.3.

Step 1 Compute the incemental nodal force vector $\Delta\mathbf{P}^{ext}_{t,n}$.

$$\Delta\mathbf{P}^{ext}_{t,n} = \mathbf{P}^{ext}_{t,n} - \mathbf{P}^{ext}_{t,n-1}$$

Step 2 Compute the incremental nodal displacement vector $\delta\mathbf{u}_{t,n}$ and corresponding total nodal displacement vector $\mathbf{u}_{t,n}$ in the structure level. Initially, iteration starts from Eq. 2.11 with $k = 1$.

If $k = 1$,

$$\begin{aligned}
\delta\mathbf{u}^k_{u,n} &= \mathbf{u}_{u,n} - \mathbf{u}_{u,n-1},\ \mathbf{u}_{u,n} = \mathbf{u}_{u,n-1} + \delta\mathbf{u}^k_{u,n} \\
\delta\mathbf{u}^k_{t,n} &= [\mathbf{K}^{tan,k-1}_{tt,n}]^{-1} \cdot [\Delta\mathbf{P}^{ext}_{t,n} - \mathbf{K}^{tan,k-1}_{tu,n}\delta\mathbf{u}^k_{u,n}] \\
\mathbf{u}^k_{t,n} &= \mathbf{u}^{k-1}_{t,n} + \delta\mathbf{u}^k_{t,n}
\end{aligned} \tag{2.11}$$

If $k \neq 1$,

$$\begin{aligned}
\mathbf{P}^{R,k}_{t,n} &= \mathbf{P}^{ext}_{t,n} - \mathbf{P}^{int,k}_{t,n} \\
\delta\mathbf{u}^{k+1}_{t,n} &= [\mathbf{K}^{tan,k}_{tt,n}]^{-1} \cdot \mathbf{P}^{R,k}_{t,n} \\
\mathbf{u}^{k+1}_{t,n} &= \mathbf{u}_{t,n-1} + \Delta\mathbf{u}^{k+1}_{t,n} = \mathbf{u}^k_{t,n} + \delta\mathbf{u}^{k+1}_{t,n}
\end{aligned} \tag{2.12}$$

where, superscript k is the iteration counter, $\mathbf{P}^R_t$ the residual nodal force vector in structural level, $\Delta\mathbf{u}^{k+1}_{t,n}$ the total incremental displacement vector from the last converged step, and $\delta\mathbf{u}^{k+1}_n$ the last incremental displacement vector.

The augmented stiffness matrix $\mathbf{K}_S$ is expressed as

$$\left\{ \begin{array}{c} \mathbf{P}_t \\ \mathbf{P}_u \end{array} \right\} = \left[\begin{array}{cc} \mathbf{K}_{tt} & \mathbf{K}_{tu} \\ \mathbf{K}_{ut} & \mathbf{K}_{uu} \end{array} \right] \left\{ \begin{array}{c} \mathbf{u}_t \\ \mathbf{u}_u \end{array} \right\} \tag{2.13}$$

where subscript t and u are associated with free end constrained degrees of freedom respectively.

Step 3 Loop over all the elements and determine their state.

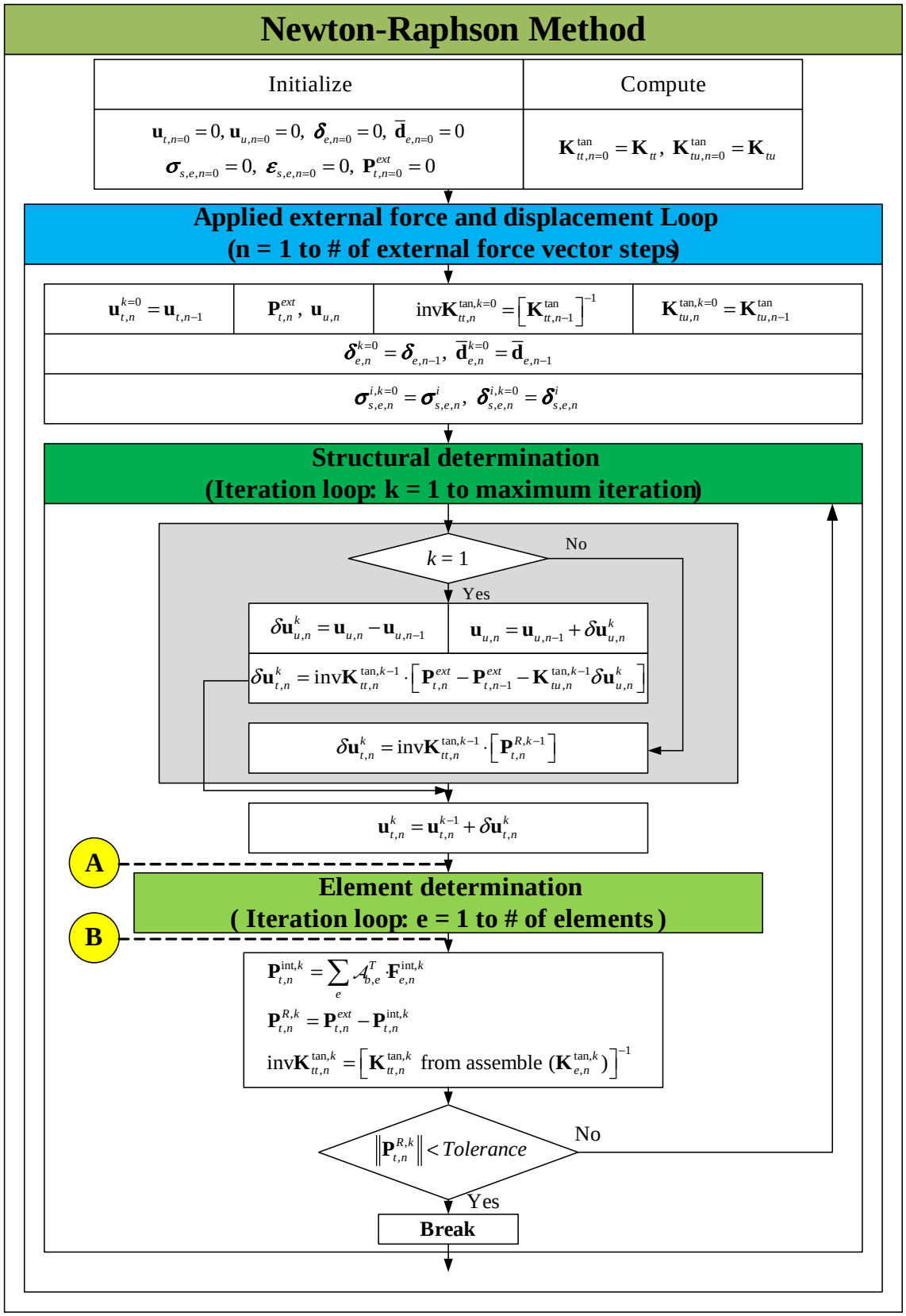


Fig. 2.5 Flow chart of nonlinear analysis using stiffness-based method (1)

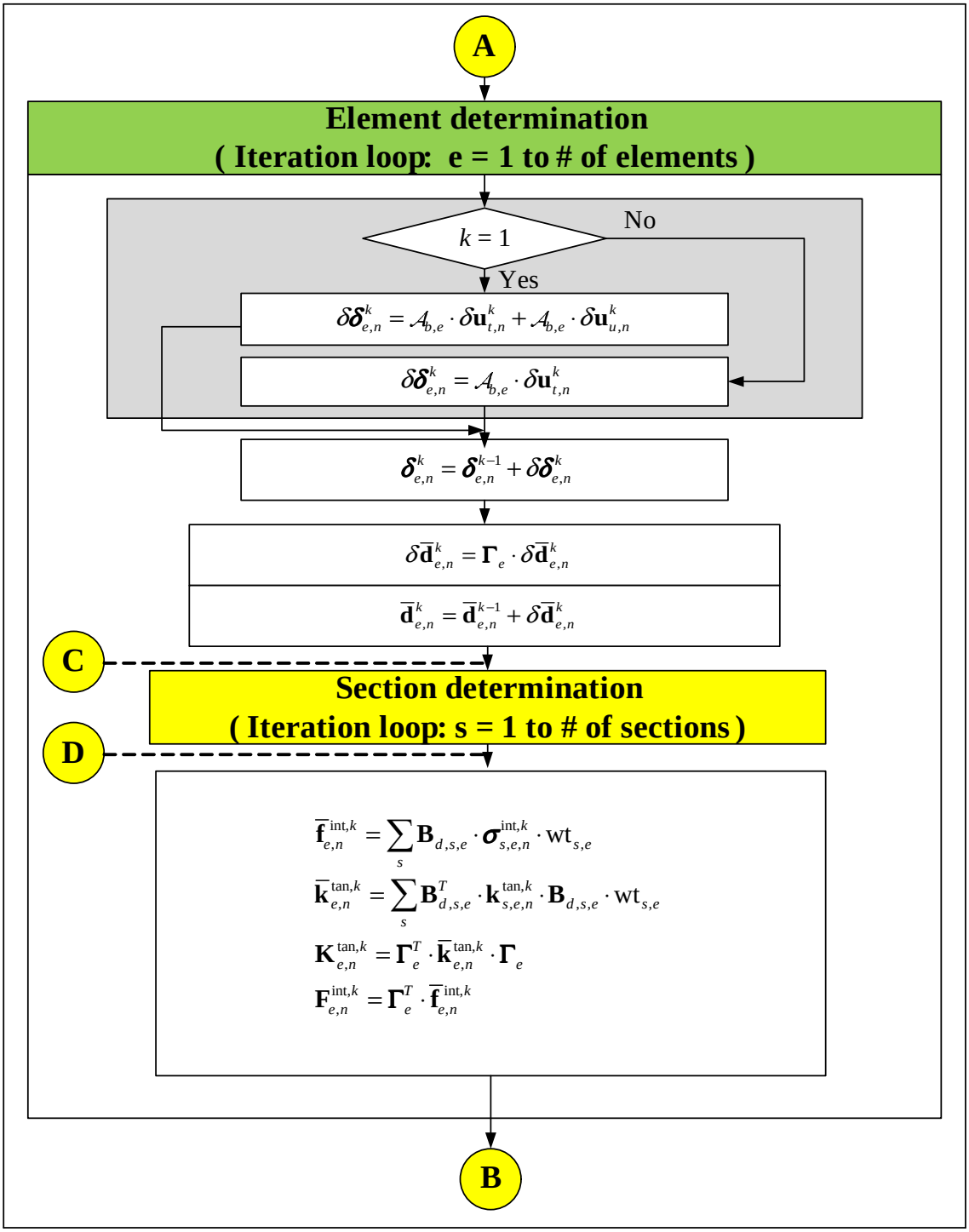


Fig. 2.6 Flow chart of nonlinear analysis using stiffness-based method (2)

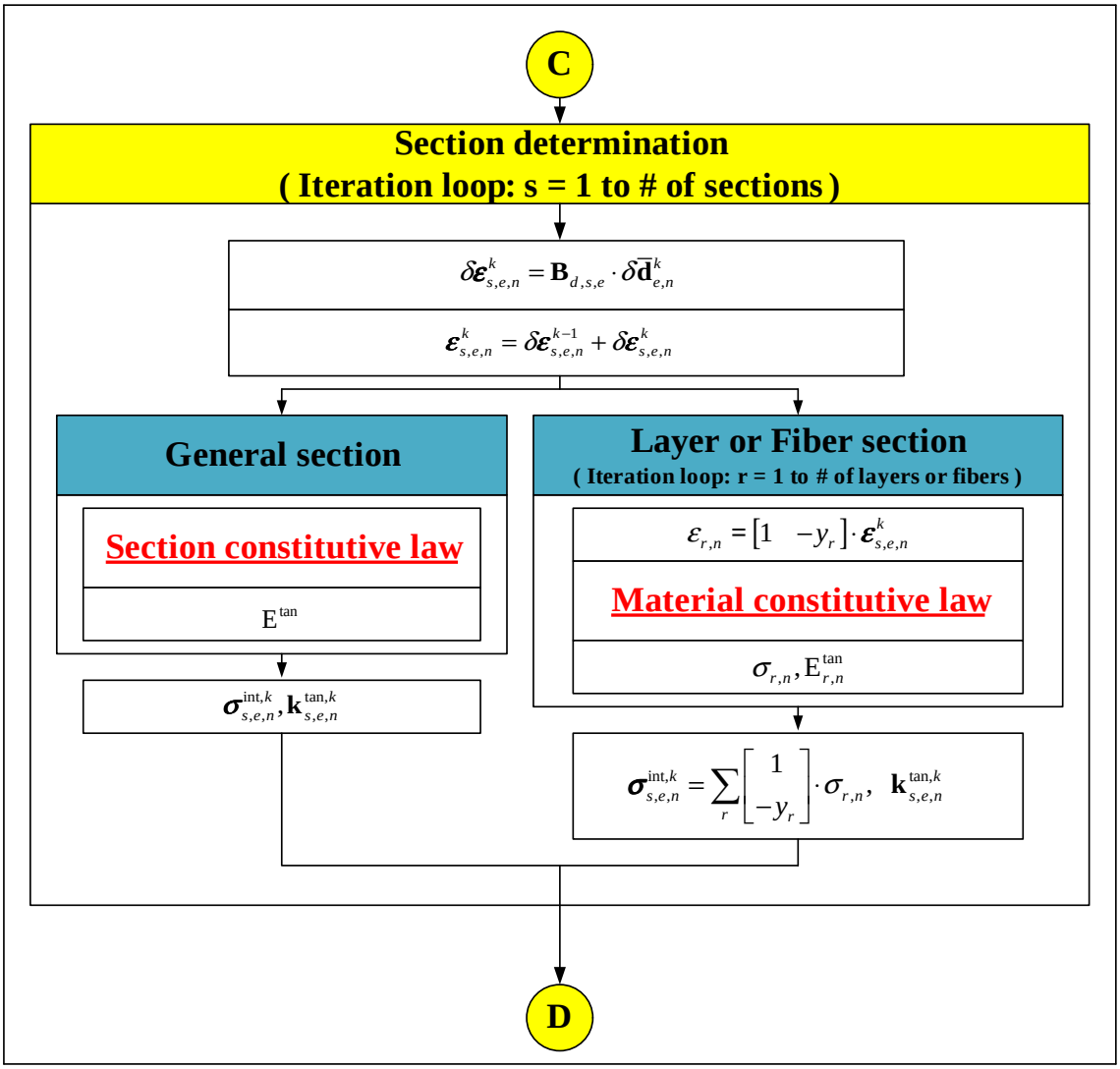


Fig. 2.7 Flow chart of nonlinear analysis using stiffness-based method (3)

- Determine the element nodal displacement vector in global reference.
  If $k = 1$,
  $$\delta\boldsymbol{\delta}^k_{e,n} = \mathcal{A}_{b,e} \cdot \delta\mathbf{u}^k_{t,n} + \mathcal{A}_{b,e} \cdot \delta\mathbf{u}^k_{u,n}$$
  If $k \neq 1$,
  $$\delta\boldsymbol{\delta}^k_{e,n} = \mathcal{A}_{b,e} \cdot \delta\mathbf{u}^k_{t,n}$$
  where, $\mathcal{A}_{b,e}$ is displacement extracting operator.
  $$\boldsymbol{\delta}^k_{e,n} = \boldsymbol{\delta}^{k-1}_{e,n} + \delta\boldsymbol{\delta}^k_{e,n}$$
  and, $\boldsymbol{\delta}^0_{e,n}$ corresponds to $\boldsymbol{\delta}_{e,n-1}$.
- Determine the element nodal displacement vector in local reference
  $$\begin{aligned} \delta\overline{\mathbf{d}}^k_{e,n} &= \boldsymbol{\Gamma}_e \cdot \delta\boldsymbol{\delta}^k_{e,n} \\ \overline{\mathbf{d}}^k_{e,n} &= \overline{\mathbf{d}}^{k-1}_{e,n} + \delta\overline{\mathbf{d}}^k_{e,n} \end{aligned}$$
  and, $\overline{\mathbf{d}}^0_{e,n}$ corresponds to $\overline{\mathbf{d}}_{e,n-1}$.

Step 4 Start the section state determination by looping over the element's sections. The total number of section may vary from element to element. Therefore, the total number of sections in an element is nIp_e and it depends on the number of integration points in the Gauss Legendre quadrature rule.

Determine the section deformation vector. For the section state determination, we first need to divide section into general section and fiber section as described at Sec. 2.3. Here we consider general section only. The state determination of each section is performed in loop s which refers to the s^{th} section.

$$\delta\boldsymbol{\varepsilon}^k_{s,e,n} = \mathbf{B}_{d,s,e} \cdot \delta\overline{\mathbf{d}}^k_{e,n}$$

where, $\mathbf{B}_{d,s,e}$ is the matrix derived (derivatives) from the displacement interpolation functions at s^{th} section of e^{th} element.

$$\boldsymbol{\varepsilon}^k_{s,e,n} = \boldsymbol{\varepsilon}^{k-1}_{s,e,n} + \delta\boldsymbol{\varepsilon}^k_{s,e,n}$$

and, $\boldsymbol{\varepsilon}^0_{s,e,n}$ corresponds to $\boldsymbol{\varepsilon}_{s,e,n-1}$.

Step 5 Determine the section tangent stiffness matrix and the internal section force vector. If the section constitutive law is explicitly known, then $\mathbf{k}^{tan,k}_{s,e,n}$ and $\boldsymbol{\sigma}^{int,k}_{s,e,n}$ are determined from $\boldsymbol{\varepsilon}^k_{s,e,n}$. For linear elastic section, we need not to recompute $\mathbf{k}^{tan,k}_{s,e,n}$ as it is identical to the initial section stiffness matrix $\mathbf{k}_{s,e}$.

If we have elastic section,

$$\begin{aligned} \mathbf{k}^{tan,k}_{s,e,n} &= \mathbf{k}_{s,e} \\ \boldsymbol{\sigma}^{int,k}_{s,e,n} &= \mathbf{k}^{tan,k}_{s,e,n} \cdot \boldsymbol{\varepsilon}^k_{s,e,n} \end{aligned}$$

where, $\mathbf{k}_{s,e,n}^{tan,k}$ is the section tangent stiffness matrix at k^{th} iteration.

Step 6 Determine first the internal element nodal force vector and the element tangent stiffness matrix, then from Eq. 2.10 and Eq. 2.8, determine $\overline{\mathbf{f}}_{e,n}^{int,k}$ and $\overline{\mathbf{k}}_{e,n}^{tan,k}$.

$$\overline{\mathbf{f}}_{e,n}^{int,k} = \sum_{s} \mathbf{B}_{d,s,e}^{T} \cdot \boldsymbol{\sigma}_{s,e,n}^{int,k} \ \mathrm{wt}_{s,e}$$

where, $\mathrm{wt}_{s,e}$ is the weight coefficient associated with the Jacobian at the s^{th} section of the e^{th} element.

$$\overline{\mathbf{k}}_{e,n}^{tan,k} = \sum_{s} \mathbf{B}_{d,s,e}^{T} \cdot \mathbf{k}_{s,e,n}^{tan,k} \cdot \mathbf{B}_{d,s,e} \cdot \mathrm{wt}_{s,e}$$

where, $\overline{\mathbf{k}}_{e,n}^{tan,k}$ is the element tangent stiffness matrix in local reference.

Finally, from Eq. 2.9, determine $\mathbf{F}_{e,n}^{int,k}$ and $\mathbf{K}_{e,n}^{tan,k}$.

$$\begin{aligned} \mathbf{F}_{e,n}^{int,k} &= \boldsymbol{\Gamma}_{e}^{T} \cdot \overline{\mathbf{f}}_{e,n}^{int,k} \\ \mathbf{K}_{e,n}^{tan,k} &= \boldsymbol{\Gamma}_{e}^{T} \cdot \overline{\mathbf{k}}_{e,n}^{tan,k} \cdot \boldsymbol{\Gamma}_{e} \end{aligned}$$

Step 7 Determine the internal nodal force vector and the augmented tangent stiffness matrix.

$$\begin{aligned} \mathbf{P}_{t,n}^{int,k} &= \sum_{e} \mathcal{A}_{b,e}^{T} \cdot \mathbf{F}_{e,n}^{int,k} \\ \mathbf{K}_{S,n}^{tan,k} &= \sum_{e} \mathcal{A}_{b,e}^{T} \cdot \mathbf{K}_{e,n}^{tan,k} \cdot \mathcal{A}_{b,e} \end{aligned}$$

where, $\mathcal{A}_{b,e}^{T}$ is force assembling operator, and $\mathbf{K}_{S,n}^{tan,k}$ has the four submatrices, $\mathbf{K}_{tt,n}^{tan,k}$, $\mathbf{K}_{tu,n}^{tan,k}$, $\mathbf{K}_{ut,n}^{tan,k}$, and $\mathbf{K}_{uu,n}^{tan,k}$ as shown Eq. 2.13.

Step 8 Compute the residual nodal force vector at the structural level from Eq. 2.12 and check for convergence:

- If $\mathbf{P}_{t,n}^{R,k}$ is within the specified tolerance, go to next force increment.
- If $\mathbf{P}_{t,n}^{R,k}$ is not within the specified tolerance, k is updated to $k+1$ and the next Newton-Raphson iteration initiates. Eq. 2.12 in **Step 2** through **Step 8** are repeated until convergence occurs at the structure level.

2.2.2 Flexibility Method 2D Beam-Column Element

Flexibility-based 2D beam-column elements are "nonconformist" finite elements since they yield the element flexibility matrix rather than the classical stiffness matrix, and are based on the equations of equilibrium rather than on assumed displacement field. Nevertheless, they do offer some important advantages which will be highlighted later.

2.2.2.1 Formulation

In this section we will derive the element flexibility matrix $\tilde{\mathbf{c}}_e$ without rigid body modes and then invert it to obtain the corresponding element stiffness matrix $\tilde{\mathbf{k}}_e$ (again without rigid body modes).

This is particularly interesting in those instances where stiffness-based method formulations are approximate and flexibility-based method formulations are exact such as a section varying along the element and elements with material nonlinearity. As previously, we will use the principle of complementary virtual work through the usual three major steps.

1. **Equilibrium**

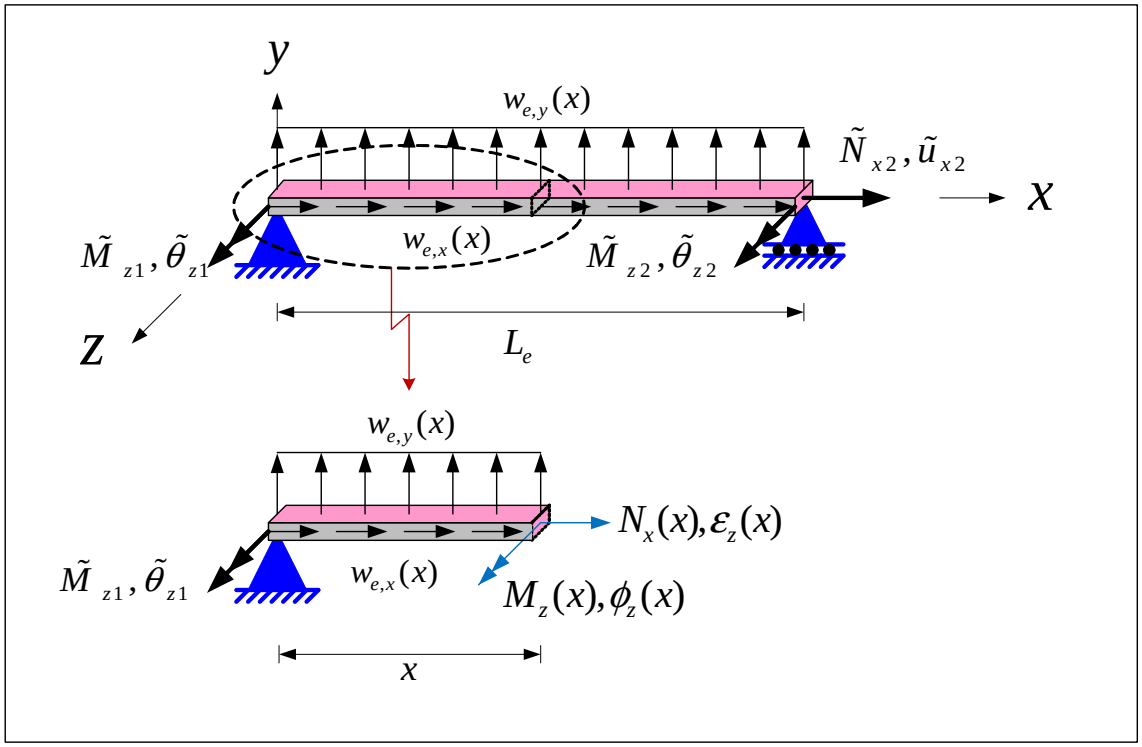


Fig. 2.8 2D beam-column element without rigid body modes

We start with the strong form of the equilibrium equation, Fig. 2.8

$$\begin{gathered} \underbrace{\mathbf{w}_e(x)}_{\text{External}} + \underbrace{\mathcal{L}_f \cdot \boldsymbol{\sigma}_s(x)}_{\text{Internal}} = \mathbf{0} \\ \left\{ \begin{array}{c} w_x^{(e)}(x) \\ w_y^{(e)}(x) \end{array} \right\} + \left[\begin{array}{cc} \frac{\mathrm{d}}{\mathrm{d}x} & 0 \\ 0 & \frac{\mathrm{d}^2}{\mathrm{d}x^2} \end{array} \right] \left\{ \begin{array}{c} N_x(x) \\ M_z(x) \end{array} \right\} = \mathbf{0} \end{gathered} \tag{2.14}$$

where, $\mathbf{w}_e(x)$ is the external element force vector and $\mathcal{L}_f$ is the force differential operator. We first assume that there are no external element forces, that is $\mathbf{w}_e(x) = \mathbf{0}$. The selection of force interpolation functions, $\mathbf{N}_f(x)$ results from the requirement that equilibrium, as expressed in Eq. 2.14, must be satisfied in the strong form. Thus, for the Euler-Bernoulli

beam,

$$\begin{bmatrix} \frac{\mathrm{d}}{\mathrm{d}x} & 0 \\ 0 & \frac{\mathrm{d}^2}{\mathrm{d}x^2} \end{bmatrix} \begin{Bmatrix} N_x(x) \\ M_z(x) \end{Bmatrix} = \mathbf{0}$$

will yield

$$\begin{aligned} \frac{\mathrm{d}N_x(x)}{\mathrm{d}x} &= 0 \\ \frac{\mathrm{d}^2 M_z(x)}{\mathrm{d}x^2} &= 0 \end{aligned}$$

Integrating these equations, we obtain

$$\begin{aligned} N_x(x) &= c_3 \\ M_z(x) &= c_1 x + c_2 \end{aligned} \tag{2.15}$$

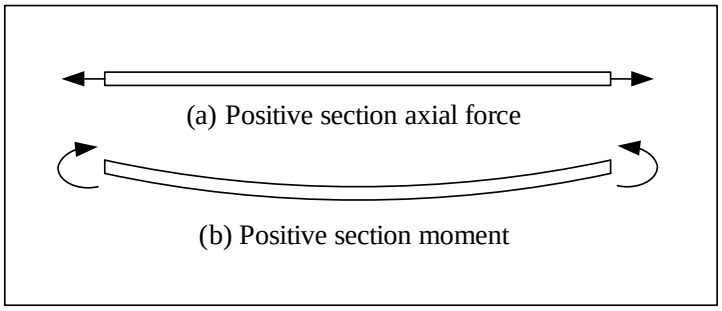


Fig. 2.9 Sign convention on section force

Applying the boundary condition from Fig. 2.8 with the sign convention (on section force) shown in Fig. 2.9,

$$\begin{aligned} N_x(L) &= N_{x2} \\ M_z(0) &= -M_{z1} \\ M_z(L) &= M_{z2} \end{aligned}$$

we obtain

$$\begin{aligned} c_1 &= \frac{M_{z1} + M_{z2}}{L_e} \\ c_2 &= -M_{z1} \\ c_3 &= N_{x2} \end{aligned} \tag{2.16}$$

Substituting Eq. 2.16 into Eq. 2.15,

$$N_x(x) = N_{x2}$$
$$M_z(x) = \left(\frac{x}{L_e} - 1\right) M_{z1} + \frac{x}{L_e} M_{z2}$$

or

$$\underbrace{\begin{Bmatrix} N_x(x) \\ M_z(x) \end{Bmatrix}}_{\boldsymbol{\sigma}_s(x)} = \underbrace{\begin{bmatrix} 0 & 0 & 1 \\ \frac{x}{L_e} - 1 & \frac{x}{L_e} & 0 \end{bmatrix}}_{\mathbf{N}_f(x)} \underbrace{\begin{Bmatrix} M_{z1} \\ M_{z2} \\ N_{x2} \end{Bmatrix}}_{\tilde{\mathbf{f}}_e} \tag{2.17}$$

where, $\tilde{\mathbf{f}}_e$ is the element nodal force vector without rigid body modes.

Finally, from Eq. 2.17, we extract the matrix of force interpolation functions.

$$\mathbf{N}_f(x) = \begin{bmatrix} 0 & 0 & 1 \\ \frac{x}{L_e} - 1 & \frac{x}{L_e} & 0 \end{bmatrix}$$

2. **Constitutive law**

   Whereas we have previously expressed section forces in terms of section deformations (Eq. 2.5), we now need to express section deformations in terms of section forces

$$\boldsymbol{\varepsilon}_s(x) = \mathbf{c}_s(x) \cdot \boldsymbol{\sigma}_s(x) \tag{2.18}$$

   where, $\mathbf{c}_s(x)$ is the section flexibility matrix. If $\mathbf{c}_s(x)$ is not derived from fiber section (which will be discussed in Sec. 2.3), then for linear elastic analysis $\mathbf{c}_s(x)$ is simply.

$$\mathbf{c}_s(x) = \begin{bmatrix} \frac{1}{E(x) \cdot A(x)} & 0 \\ 0 & \frac{1}{E(x) \cdot I_z(x)} \end{bmatrix}$$

   which is the inverse of Eq. 2.6.

3. **Compatibility**

   We will derive the compatibility equation from the weak form of the principle of complementary virtual work (as opposed to the principle of virtual work for the stiffness-based method in Eq. 2.7.

$$\underbrace{\delta\tilde{\mathbf{f}}_e{}^T \tilde{\mathbf{d}}_e}_{\text{External}} = \underbrace{\int_0^{L_e} \delta\boldsymbol{\sigma}_s(x)^T \cdot \boldsymbol{\varepsilon}_s(x) \mathrm{d}x}_{\text{Internal}} \tag{2.19}$$

   where, and $\tilde{\mathbf{d}}_e$ is the element nodal displacement vector without rigid body modes.

   Substituting Eq. 2.17 and Eq. 2.18 into Eq. 2.19 and since the latter must hold for any

arbitary $\delta\tilde{\mathbf{f}}_e$, we obtain

$$\begin{aligned}
\delta\tilde{\mathbf{f}}_e{}^T\tilde{\mathbf{d}}_e &= \int_0^{L_e} \delta\tilde{\mathbf{f}}_e{}^T \cdot \mathbf{N}_f(x)^T \cdot \mathbf{c}_s(x) \cdot \boldsymbol{\sigma}_s(x)\ \mathrm{d}x \\
\tilde{\mathbf{d}}_e &= \int_0^{L_e} \mathbf{N}_f(x)^T \cdot \mathbf{c}_s(x) \cdot \boldsymbol{\sigma}_s(x)\ \mathrm{d}x \\
&= \underbrace{\int_0^{L_e} \mathbf{N}_f(x)^T \cdot \mathbf{c}_s(x) \cdot \mathbf{N}_f(x)\ \mathrm{d}x}_{\tilde{\mathbf{c}}_e} \cdot \tilde{\mathbf{f}}_e \\
\tilde{\mathbf{d}}_e &= \tilde{\mathbf{c}}_e \cdot \tilde{\mathbf{f}}_e
\end{aligned}$$

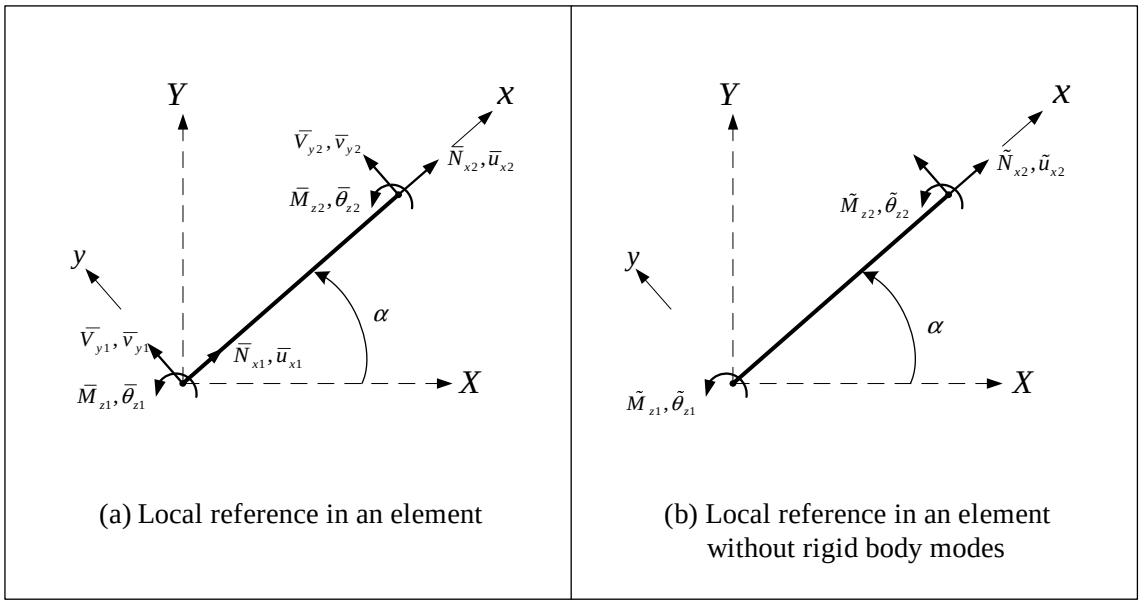
The element flexibility matrix without rigid body modes in local reference is thus given by

$$\tilde{\mathbf{c}}_e = \int_0^{L_e} \mathbf{N}_f(x)^T \cdot \mathbf{c}_s(x) \cdot \mathbf{N}_f(x)\ \mathrm{d}x \tag{2.20}$$

which is the counterpart of Eq. 2.8, and the corresponding element stiffness matrix without rigid body modes in local reference is simply

$$\tilde{\mathbf{k}}_e = [\tilde{\mathbf{c}}_e]^{-1}$$

2.2.2.2 Coordinate system for flexibility-based 2D beam-column element with Bernoulli beam theory



(a) Local reference in an element

(b) Local reference in an element without rigid body modes

Fig. 2.10 Internal forces and corresponding displacements in local coordinate system with or without rigid body for the flexibility-based 2D beam-column element

Contrarily to the reference system of the stiffness-based method in Fig. 2.2, we need to consider forces and displacements in local reference with and without rigid body modes as shown in Fig. 2.10.

Element nodal force vector without rigid body modes in local reference are (arbitrarily) selected as

$$\tilde{\mathbf{f}}_e = \lfloor \tilde{M}_{z1},\ \tilde{M}_{z2},\ \tilde{N}_{x2} \rfloor^T$$

and the corresponding element nodal displacement vector without rigid body modes in local reference are given by

$$\tilde{\mathbf{d}}_e = \lfloor \tilde{\theta}_{z1},\ \tilde{\theta}_{z2},\ \tilde{u}_{x2} \rfloor^T$$

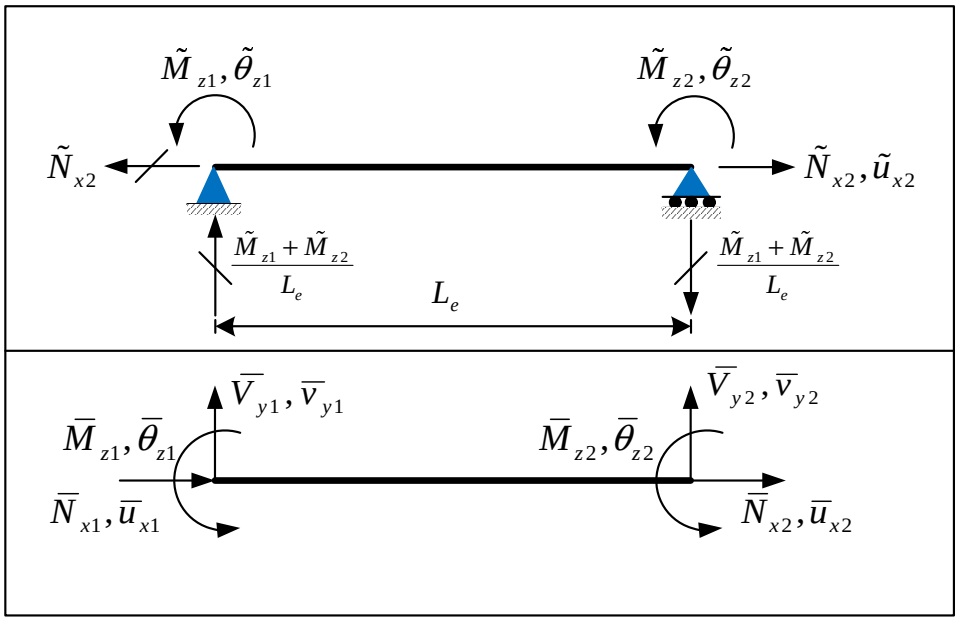


Fig. 2.11 The relationship between rigid body modes and no rigid body modes

The relationship between rigid body modes and no rigid body modes is obtained through equilibrium, Fig. 2.11.

$$\underbrace{\begin{Bmatrix} \overline{N}_{x1} \\ \overline{V}_{y1} \\ \overline{M}_{z1} \\ \overline{N}_{x2} \\ \overline{V}_{y2} \\ \overline{M}_{z2} \end{Bmatrix}}_{\overline{\mathbf{f}}_e} = \underbrace{\begin{bmatrix} 0 & 0 & -1 \\ \frac{1}{L_e} & \frac{1}{L_e} & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \\ -\frac{1}{L_e} & -\frac{1}{L_e} & 0 \\ 0 & 1 & 0 \end{bmatrix}}_{\tilde{\mathbf{\Gamma}}_e^T} \underbrace{\begin{Bmatrix} \tilde{M}_{z1} \\ \tilde{M}_{z2} \\ \tilde{N}_{x2} \end{Bmatrix}}_{\tilde{\mathbf{f}}_e} \tag{2.21}$$

$$\begin{aligned}\overline{\mathbf{f}}_e &= \tilde{\mathbf{\Gamma}}_e^T \cdot \tilde{\mathbf{f}}_e \\ \overline{\mathbf{d}}_e &= \tilde{\mathbf{\Gamma}}_e^T \cdot \tilde{\mathbf{d}}_e \\ \mathbf{K}_e &= \tilde{\mathbf{\Gamma}}_e^T \cdot \tilde{\mathbf{k}}_e \cdot \tilde{\mathbf{\Gamma}}_e\end{aligned} \tag{2.22}$$

which is the counterpart of Eq. 2.9.

2.2.2.3 State determination

In their early and pioneering publication Zeris and Mahin (1988) did not provide a clear and consistent method for calculating the internal element force vectors from element deformations. Possibly because the finite element formulation is based on the complementary principle of virtual work and corresponding flexibility-based 2D beam-column elements do not have shape functions that relate deformation field inside the element with element nodal displacement vector. To address this problem in the flexibility-based method, the numerical implementation would typically use the stiffness-based method in the structural level where the solution of the global equilibrium equations yields displacements. Also, the internal element force vectors of all elements in the structure need to be determined during the phase of state determination. In order to better describe the consistent state determination process, we will derive the governing equations by the mixed stiffness-based and flexibility-based methods.

The nonlinear algorithm for the mixed stiffness-based and flexibility-based methods will be divided into two methods (a) with Newton-Raphson iteration in element level to determine element state, (b) without iteration in element level to determine element state. The former is based on the formulation of (Spacone et al. 1992) and the later on the one of (Carol and Murcia 1989).

2.2.2.3.1 With element iterations As with the state determination of stiffness-based 2D beam-column in a nonlinear structural analysis the state determination process for the mixed stiffness-based and flexibility-based methods with Newton-Raphson iteration loop in element level is made up of three phases as shown in Fig. 2.12 (which is the counterpart of Fig. 2.3).

1. Section state determination, Fig. 2.12(c)
2. Element state determination, Fig. 2.12(b)
3. Structure state determination, Fig. 2.12(a)

Where, superscript j in Fig. 2.12(b) denotes the iteration scheme at the element level in the element state determination process. This iteration loop is necessary for the determination of the internal element nodal force vector $\tilde{\mathbf{f}}_e^{int,k}$ that correspond to the element nodal displacement vector $\tilde{\mathbf{d}}_e^k$ during the k^{th} Newton-Raphson iteration.

This method is explained with the assumption that the Newton-Raphson is used for the iteration process in structural level. The iteration process in structural level does not affect the strategy for iteration process in element level, which determines the internal element nodal force vector $\tilde{\mathbf{f}}_e^{int}$ for the given element nodal displacement vector $\tilde{\mathbf{d}}_e$.

In a flexibility-based 2D beam-column element, the first step is the determination of the element nodal force vector $\tilde{\mathbf{f}}_{e,n}^{k,j}$ from the current element nodal displacement vector using the element tangent stiffness matrix $\tilde{\mathbf{k}}_{e,n}^{tan,k,j-1}$ of the last iteration, then the force interpolation

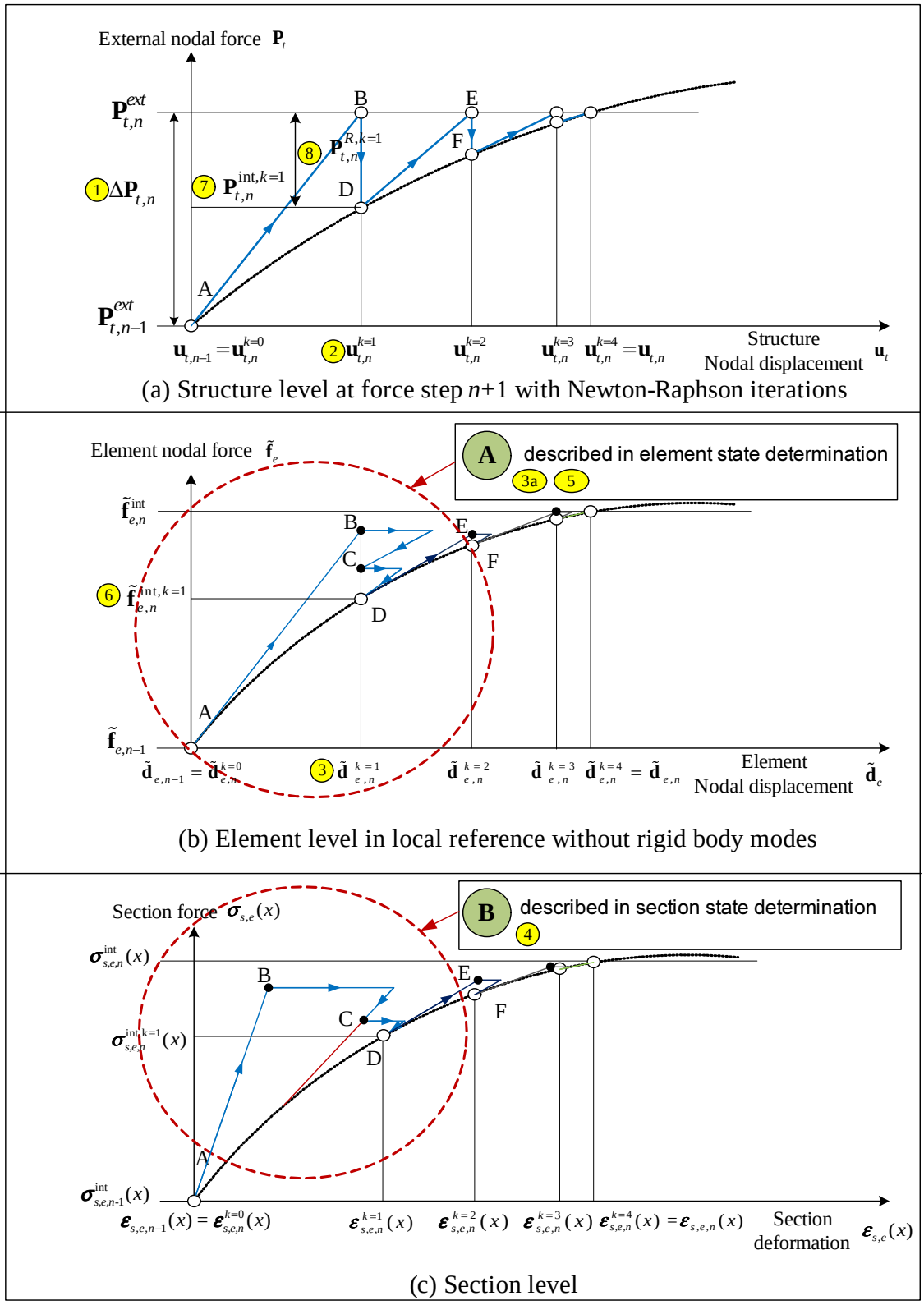
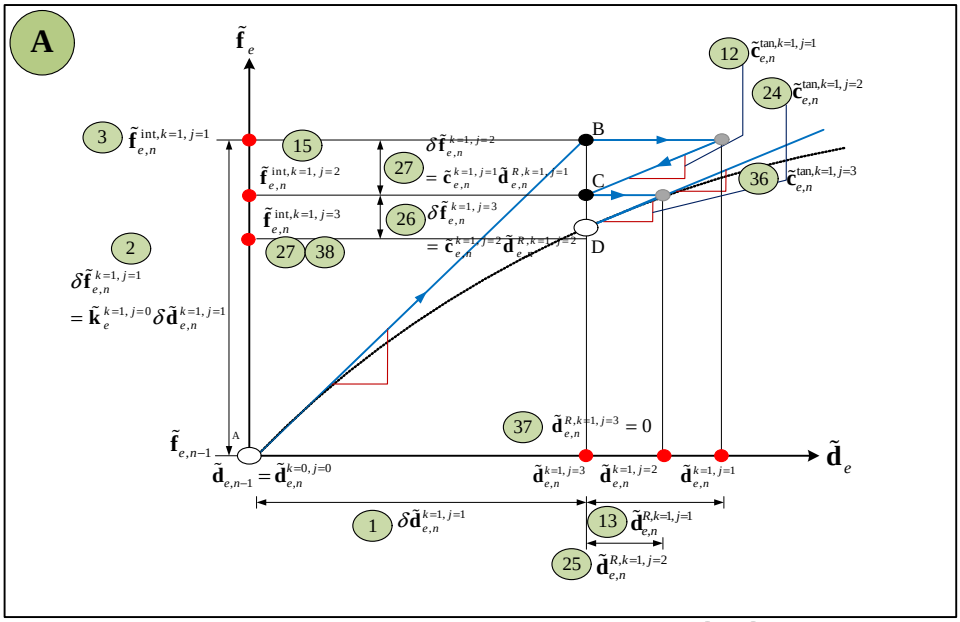
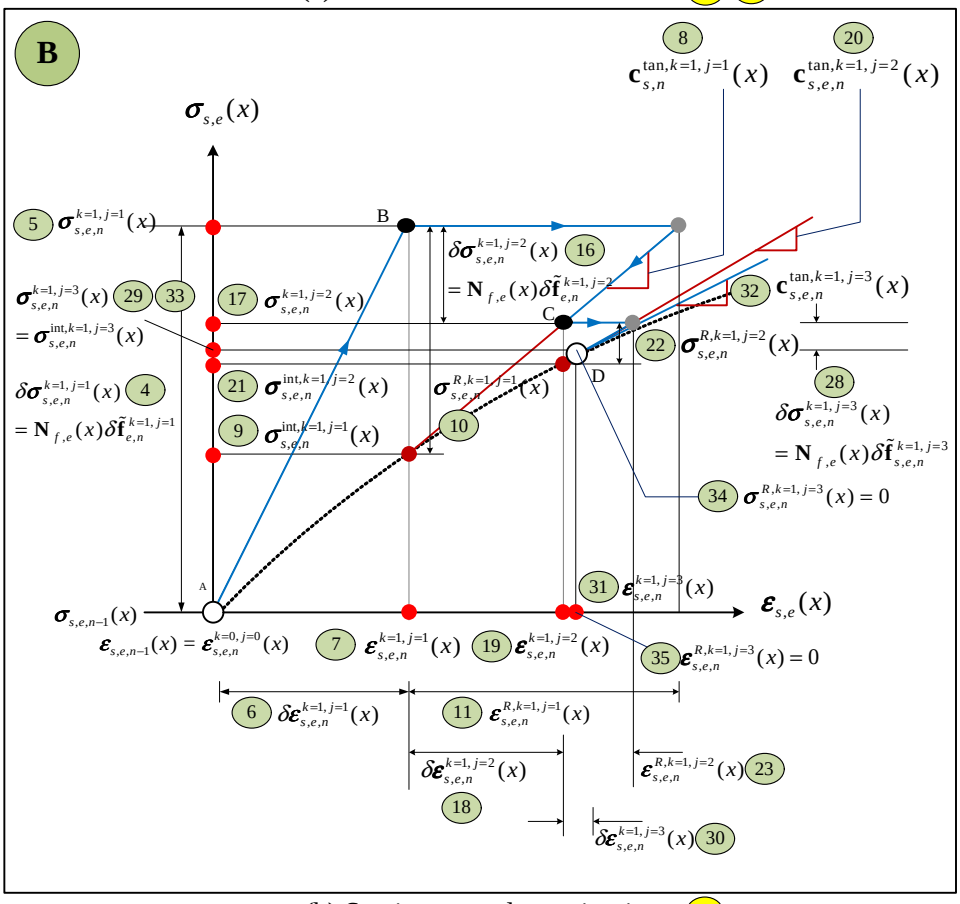


Fig. 2.12 State determination for flexibility-based method with element iteration (Taucer, Spacone and Filippou 1991)



(a) Element state determination 3a 5



(b) Section state determination 4

Fig. 2.13 Element and section state determinations for flexibility-based 2D beam-column element with Newton-Raphson iteration loop in element level

functions $\mathbf{N}_{f,e}(x)$ will then yield the section force vectors $\boldsymbol{\sigma}_{s,e,n}^{k,j}(x)$ along the element. These are too complications in this procedure. The first is the determination of the section deformation vectors $\boldsymbol{\varepsilon}_{s,e,n}^{k,j}(x)$ from section force vectors, since the nonlinear section force-deformation relation is commonly expressed as an explicit function of section deformation vector. The second arises from the fact that changes in the section tangent stiffness matrices $\mathbf{k}_{s,e,n}^{tan}(x)$ produce a new element tangent stiffness matrix which, in turn, changes the element nodal force vector for the given element nodal displacement vector.

These problems are solved through a nonlinear approach which first determines residual element nodal displacement vector $\tilde{\mathbf{d}}_{e,n}^{R,k,j}$ at each iteration. Then, compatibility of displacement at the structural level requires that this residual element nodal displacement vector be corrected. This is accomplished at the element level by applying corrective element nodal force vector based on the current element tangent stiffness matrix. The corresponding section force vectors are then determined from the force interpolation functions so that equilibrium will always be satisfied along the element. These section force vectors will not change during the section state determination in order to maintain equilibrium along the element. Finally, the linear approximation of the section force-deformation relation about the present state results in residual section deformation vectors $\boldsymbol{\sigma}_{s,e,n}^{R,k,j}(x)$. These are then integrated along the element to obtain new residual element nodal displacement vector and the whole process is repeated until convergence occurs. It is important to stress that compatibility of element nodal displacement vector and equilibrium along the element are always satisfied in this process.

The nonlinear solution procedure for the element and section state determinations in Fig. 2.12 are illustrated in details in Fig. 2.13 for one Newton-Raphson iteration k. For illustrative purposes, convergence in loop j is reached in three iterations in Fig. 2.13.

The goal of the Newton-Raphson iteration loop in element level is to determine the internal element nodal force vector for the current element nodal displacement vector at the k^{th} Newton-Raphson iteration, hence

$$\tilde{\mathbf{d}}_{e,n}^{k} = \tilde{\mathbf{d}}_{e,n}^{k-1} + \delta\tilde{\mathbf{d}}_{e,n}^{k}$$

An iterative process in element level denoted through the superscript j will be introduced inside the k^{th} Newton-Raphson iteration, and the first iteration corresponds to $j = 1$.

The initial state of the element, represented by the point **A**, and $j = 0$ and $k = 0$ in Fig. 2.13, corresponds to the state at the end of the last convergence in structural level. With the initial element tangent stiffness matrix given by

$$\tilde{\mathbf{c}}_{e,n}^{tan,k=1,j=0} = \tilde{\mathbf{c}}_{e,n-1}^{tan}$$

and the given incremental element nodal displacement vector

$$\delta\tilde{\mathbf{d}}_{e,n}^{k=1,j=1} = \delta\tilde{\mathbf{d}}_{e,n}^{k=1}$$

hence, the corresponding incremental element nodal force vector is

$$\delta\tilde{\mathbf{f}}_{e,n}^{k=1,j=1} = \left[\tilde{\mathbf{c}}_{e,n}^{tan,k=1,j=0}\right]^{-1} \cdot \delta\tilde{\mathbf{d}}_{e,n}^{k=1,j=1}$$

The incremental section force vectors can now be determined from the force interpolation functions

$$\delta\boldsymbol{\sigma}_{s,e,n}^{k=1,j=1}(x) = \mathbf{N}_{f,e}(x) \cdot \delta\tilde{\mathbf{f}}_{e,n}^{k=1,j=1}$$

With the section tangent flexibility matrices at end of the last convergence in structural level given by

$$\mathbf{c}_{s,e,n}^{tan,k=1,j=0}(x) = \mathbf{c}_{s,e,n-1}^{tan}(x)$$

the linearization of the section force-deformation relation yields the incremental section deformation vectors.

$$\delta\boldsymbol{\varepsilon}_{s,e,n}^{k=1,j=1}(x) = \mathbf{c}_{s,e,n}^{tan,k=1,j=0}(x) \cdot \delta\boldsymbol{\sigma}_{s,e,n}^{k=1,j=1}(x)$$

The section deformation vectors are updated to the state that corresponds to point **B** in Fig. 2.13(b), and the updated section deformation vector will be given by

$$\boldsymbol{\varepsilon}_{s,e,n}^{k=1,j=1}(x) = \boldsymbol{\varepsilon}_{s,e,n}^{k=1,j=0}(x) + \delta\boldsymbol{\varepsilon}_{s,e,n}^{k=1,j=1}(x)$$

For the sake of simplicity we will assume that the section force-deformation relation is explicitly known, then the section deformation vectors $\boldsymbol{\varepsilon}_{s,e,n}^{k=1,j=1}(x)$ will correspond to internal section force vectors $\boldsymbol{\sigma}_{s,e,n}^{int,k=1,j=1}(x)$ and updated section tangent flexibility matrices $\mathbf{c}_{s,e,n}^{tan,k=1,j=1}(x)$ in Fig. 2.13(b) can be defined.

The residual section force vectors are then determined

$$\boldsymbol{\sigma}_{s,e,n}^{R,k=1,j=1}(x) = \boldsymbol{\sigma}_{s,e,n}^{k=1,j=1}(x) - \boldsymbol{\sigma}_{s,e,n}^{int,k=1,j=1}(x)$$

and are transformed into residual section deformation vectors $\boldsymbol{\varepsilon}_{s,e,n}^{R,k=1,j=1}(x)$

$$\boldsymbol{\varepsilon}_{s,e,n}^{R,k=1,j=1}(x) = \mathbf{c}_{s,e,n}^{tan,k=1,j=1}(x) \cdot \boldsymbol{\sigma}_{s,e,n}^{R,k=1,j=1}(x)$$

The residual section deformation vectors are thus the linear approximation of the deformation error made in the linearization of the section force-deformation relation (Fig. 2.13(b)). While any suitable section flexibility matrix can be used to calculate the residual section deformation vector, the section tangent flexibility matrices offer the fastest convergence rate.

The residual section deformation vectors are integrated along the element using the complimentary principle of virtual work to obtain the residual element nodal displacement vector.

$$\tilde{\mathbf{d}}_{e,n}^{R,k=1,j=1} = \int_0^{L_e} \mathbf{N}_{f,e}(x)^T \cdot \boldsymbol{\varepsilon}_{s,e,n}^{R,k=1,j=1}(x)\mathrm{d}x$$

At this stage the first iteration ($j = 1$) is completed. The final element and section states for $j = 1$ correspond to point **B** in Fig. 2.13. The residual section deformation vectors $\boldsymbol{\varepsilon}_{s,e,n}^{R,k=1,j=1}(x)$ and the residual element nodal displacement vector $\tilde{\mathbf{d}}_{e,n}^{R,k=1,j=1}$ were determined in the first iteration, but the corresponding element nodal displacement vector have not yet been updated. Instead, they constitute the starting point of the remaining steps within iteration loop j.

The presence of residual element nodal displacement vector $\tilde{\mathbf{d}}_{e,n}^{R,k=1,j=1}$ will violate compatibility, since elements sharing a common node would now have different element nodal dis-

placement vector. In order to restore the inter-element compatibility, corrective force vector $\delta\tilde{\mathbf{f}}_{e,n}^{k=1,j=2}$ must be applied at the ends of the element as follows

$$\begin{aligned}\delta\tilde{\mathbf{f}}_{e,n}^{k=1,j=2} &= -\left[\tilde{\mathbf{c}}_{e,n}^{k=1,j=1}\right]^{-1}\cdot\tilde{\mathbf{d}}_{e,n}^{R,k=1,j=1}\\ \tilde{\mathbf{c}}_{e,n}^{k=1,j=1} &= \int_0^{L_e}\mathbf{N}_{f,e}(x)^T\cdot\mathbf{c}_{s,e,n}^{tan,k=1,j=1}(x)\cdot\mathbf{N}_{f,e}(x)\mathrm{d}x\end{aligned} \tag{2.23}$$

Thus, in the second iteration ($j=2$), the element nodal force vector is updated as

$$\tilde{\mathbf{f}}_{e,n}^{k=1,j=2} = \tilde{\mathbf{f}}_{e,n}^{k=1,j=1} + \delta\tilde{\mathbf{f}}_{e,n}^{k=1,j=2}$$

and the section force and deformation vectors are also updated to

$$\begin{aligned}\delta\boldsymbol{\sigma}_{s,e,n}^{k=1,j=2}(x) &= \mathbf{N}_{f,e}(x)\cdot\delta\tilde{\mathbf{f}}_{e,n}^{k=1,j=2}\\ \boldsymbol{\sigma}_{s,e,n}^{k=1,j=2}(x) &= \boldsymbol{\sigma}_{s,e,n}^{k=1,j=1}(x) + \delta\boldsymbol{\sigma}_{s,e,n}^{k=1,j=2}(x)\\ \delta\boldsymbol{\varepsilon}_{s,e,n}^{k=1,j=2}(x) &= \boldsymbol{\varepsilon}_{s,e,n}^{R,k=1,j=1}(x) + \mathbf{c}_{s,e,n}^{tan,k=1,j=1}(x)\cdot\delta\boldsymbol{\sigma}_{s,e,n}^{k=1,j=2}(x)\\ \boldsymbol{\varepsilon}_{s,e,n}^{k=1,j=2}(x) &= \boldsymbol{\varepsilon}_{s,e,n}^{k=1,j=1}(x) + \delta\boldsymbol{\varepsilon}_{s,e,n}^{k=1,j=2}(x)\end{aligned}$$

The state of the element and sections within the element at the end of the second iteration $j=2$ corresponds to point **C** in Fig. 2.13. It should be noted that the updated tangent flexibility matrices $\mathbf{c}_{s,e,n}^{tan,k=1,j=2}(x)$ and residual section deformation vectors $\boldsymbol{\varepsilon}_{s,e,n}^{R,k=1,j=2}(x)$ are computed for all sections.

The residual section deformation vectors are then integrated to obtain the residual element nodal deformation vector $\tilde{\mathbf{d}}_{e,n}^{R,k=1,j=2}$ and the new element tangent flexibility matrix $\tilde{\mathbf{c}}_{e,n}^{k=1,j=2}$ is determined by integration of the section flexibility matrices $\mathbf{c}_{s,e,n}^{tan,k=1,j=2}(x)$ according to Eq. 2.23. This completes the second iteration within loop j.

The third and subsequent iterations follow exactly the same scheme. Convergence is achieved when the selected convergence criterion is satisfied. With the conclusion of iteration loop j the internal element nodal force vector for the given element nodal displacement vector $\tilde{\mathbf{d}}_{e,n}^{k=1}$ are established, as represented by point **D** in Fig. 2.12 and Fig. 2.13. The Newton-Raphson iteration process can now proceed with step $k+1$.

When incremental element nodal displacement vector $\delta\tilde{\mathbf{d}}_{e,n}^{k,j=1} = \delta\tilde{\mathbf{d}}_{e,n}^{k}$ is added to the element nodal displacement vector $\tilde{\mathbf{d}}_{e,n}^{k-1}$ at the end of the previous Newton-Raphson iteration, it is important to make sure that the element nodal displacement vector $\tilde{\mathbf{d}}_{e,n}^{k}$ do not change except in the first iteration $j=1$ during iteration loop j

Equilibrium along the element is always strictly satisfied since section force vectors are derived from element nodal force vector by the force interpolation functions.

$$\boldsymbol{\sigma}_{s,e,n}^{k}(x) = \mathbf{N}_{f,e}(x)\cdot\tilde{\mathbf{f}}_{e,n}^{k} \quad\text{and}\quad \delta\boldsymbol{\sigma}_{s,e,n}^{k}(x) = \mathbf{N}_{f,e}(x)\cdot\delta\tilde{\mathbf{f}}_{e,n}^{k}$$

Compatibility is also satisfied, not only at the element ends, but also along the element.

$$\begin{aligned} \delta\tilde{\mathbf{f}}_{e,n}^{k,j} &= -\left[\tilde{\mathbf{c}}_{e,n}^{k,j-1}\right]^{-1} \cdot \tilde{\mathbf{d}}_{e,n}^{R,k,j-1} \\ \delta\boldsymbol{\sigma}_{s,e,n}^{k,j}(x) &= \mathbf{N}_{f,e}(x) \cdot \delta\tilde{\mathbf{f}}_{e,n}^{k,j} \\ \delta\boldsymbol{\varepsilon}_{s,e,n}^{k,j}(x) &= \boldsymbol{\varepsilon}_{s,e,n}^{R,k,j-1}(x) + \mathbf{c}_{s,e,n}^{tan,k,j-1}(x) \cdot \delta\boldsymbol{\sigma}_{s,e,n}^{k,j}(x) \end{aligned} \tag{2.24}$$

The second term of Eq. 2.24 expresses the relation between section deformation vectors and element nodal displacement vector. However, it should be noted that residual section deformation vectors $\boldsymbol{\varepsilon}_{s,e,n}^{R,k,j-1}(x)$ do not strictly satisfy this compatibility condition. This requirement can only be satisfied by integrating the residual section deformation vectors $\boldsymbol{\varepsilon}_{s,e,n}^{R,k,j-1}(x)$ to obtain $\tilde{\mathbf{d}}_{e,n}^{R,k,j-1}$. Since this is rather inefficient from a computational standpoint, the small compatibility error in the calculation of residual section deformation vectors $\boldsymbol{\varepsilon}_{s,e,n}^{R,k,j-1}(x)$ will be neglected.

While equilibrium and compatibility are satisfied along the element during each iteration of loop j, the section force-deformation relation and the element force-deformation relation is only satisfied within a specified tolerance when convergence is achieved at point in Fig. 2.13.

2.2.2.3.2 Without element iterations This alternative method, first introduced by (Carol and Murcia 1989) has the added advantage (over the previous approach) of avoiding an inner element loop, at the expenses however of additional iterations at the structural level.

As with the state determination of stiffness-based 2D beam-column in a nonlinear structural analysis the state determination process for the mixed stiffness-based and flexibility-based method without iteration in element level is made up of three phases as shown in Fig. 2.14.

1. Section state determination, Fig. 2.14(c)
2. Element state determination, Fig. 2.14(b)
3. Structure state determination, Fig. 2.14(a)

Starting with completion of the structure state determination, the internal nodal force vector at k^{th} iteration $\mathbf{P}_{t,n}^{int,k}$ is compared with the total applied external nodal force vector $\mathbf{P}_{t,n}^{ext}$ and the difference, if any, yields the residual nodal force vector $\mathbf{P}_{t,n}^{R,k}$ which is then reapplied to the structure in an iterative solution process until the difference of total applied external force vector and internal nodal force vector satisfies within a specified tolerance.

The state determination procedure is straightforward for a flexibility-based 2D beam-column element without element iteration like stiffness-based 2D beam-column element. The element nodal displacement vector $\tilde{\mathbf{d}}_{e,n}^{k}$ without rigid body modes at the k^{th} iteration is determined from Eq. 2.22

$$\begin{aligned} \delta\tilde{\mathbf{d}}_{e,n}^{k} &= \tilde{\mathbf{\Gamma}}_{e} \cdot \delta\tilde{\mathbf{f}}_{e,n}^{k} \\ \tilde{\mathbf{d}}_{e,n}^{k} &= \tilde{\mathbf{d}}_{e,n}^{k-1} + \delta\tilde{\mathbf{d}}_{e,n}^{k} \end{aligned}$$

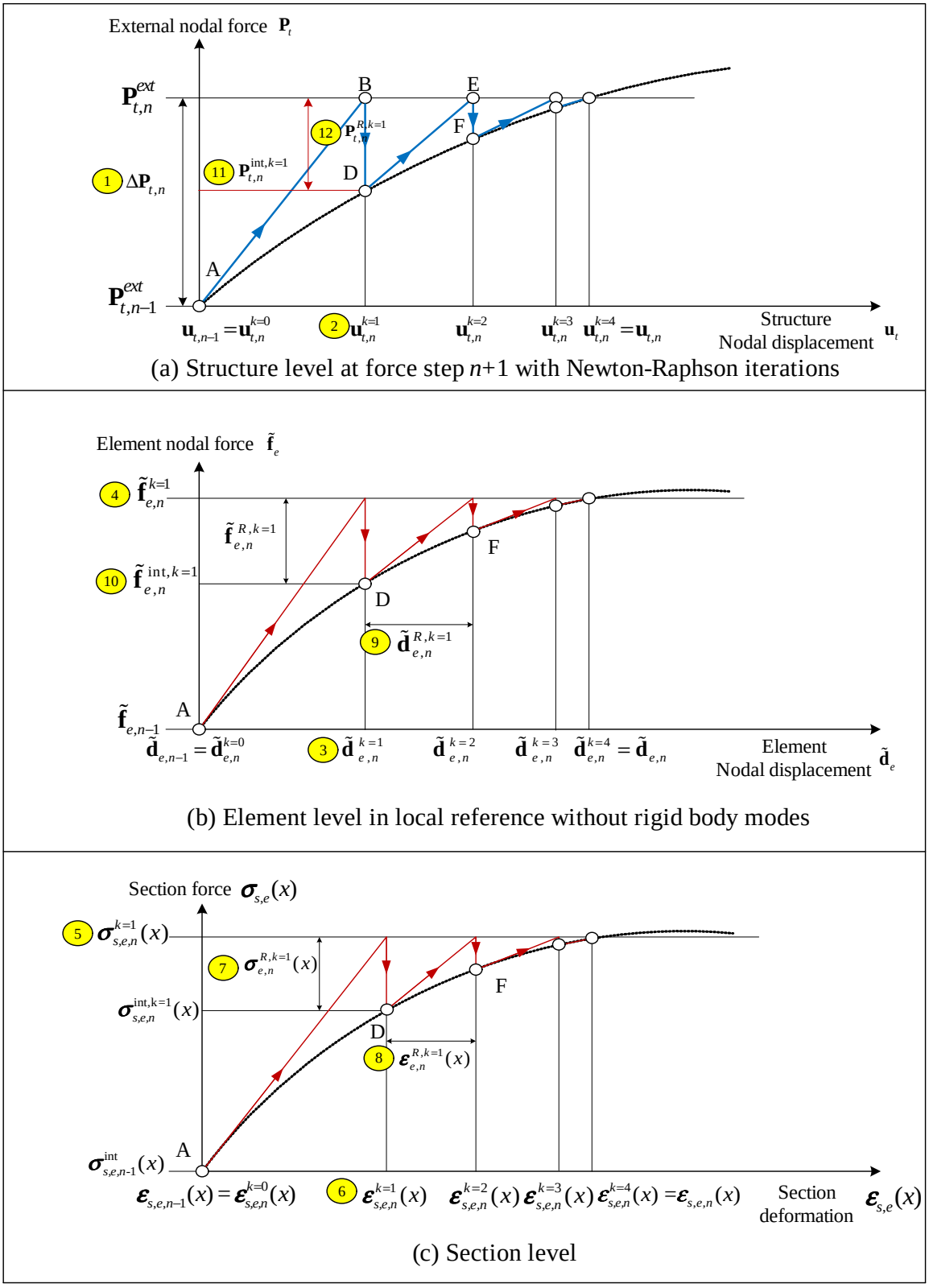


Fig. 2.14 State determination for flexibility-based method without element iteration (Carol and Murcia 1989)

and the element nodal force vector $\tilde{\mathbf{f}}_{e,n}^{k}$ without rigid body modes is determined from Eq. 2.20

$$\begin{aligned} \delta\tilde{\mathbf{f}}_{e,n}^{k} &= \tilde{\mathbf{k}}_{e,n}^{tan,k-1} \cdot \delta\tilde{\mathbf{d}}_{e,n}^{k} \\ \tilde{\mathbf{f}}_{e,n}^{k} &= \tilde{\mathbf{f}}_{e,n}^{k-1} + \delta\tilde{\mathbf{f}}_{e,n}^{k} \end{aligned} \tag{2.25}$$

The section force vectors at the k^{th} iteration $\boldsymbol{\sigma}_{s,e,n}^{k}(x)$ are determined from the element nodal force vector $\tilde{\mathbf{f}}_{e,n}^{k}$, and then the section deformation vectors at k^{th} iteration $\boldsymbol{\varepsilon}_{s,e,n}^{k}(x)$ are determined from the section force vectors $\boldsymbol{\sigma}_{s,e,n}^{k}(x)$.

$$\begin{aligned} \delta\boldsymbol{\sigma}_{s,e,n}^{k}(x) &= \mathbf{N}_{f,e} \cdot \delta\tilde{\mathbf{f}}_{e,n}^{k} \\ \boldsymbol{\sigma}_{s,e,n}^{k}(x) &= \boldsymbol{\sigma}_{s,e,n}^{k-1}(x) + \delta\boldsymbol{\sigma}_{s,e,n}^{k}(x) \\ \delta\boldsymbol{\varepsilon}_{s,e,n}^{k}(x) &= \mathbf{c}_{s,e,n}^{tan,k-1}(x) \cdot \delta\boldsymbol{\sigma}_{s,e,n}^{k}(x) \\ \boldsymbol{\varepsilon}_{s,e,n}^{k}(x) &= \boldsymbol{\varepsilon}_{s,e,n}^{k-1}(x) + \delta\boldsymbol{\varepsilon}_{s,e,n}^{k}(x) \end{aligned}$$

Assuming that the section constitutive law is explicitly known, the section tangent flexibility matrices $\mathbf{c}_{s,e,n}^{k}(x)$ and the internal section force vectors $\boldsymbol{\sigma}_{s,e,n}^{int,k}(x)$ are readily computed from $\boldsymbol{\varepsilon}_{s,e,n}^{k}(x)$.

In this approach the residual section force and deformation vectors are first determined from the section tangent flexibility matrices $\mathbf{c}_{s,e,n}^{k}(x)$.

$$\begin{aligned} \boldsymbol{\sigma}_{s,e,n}^{R,k}(x) &= \boldsymbol{\sigma}_{s,e,n}^{k}(x) - \boldsymbol{\sigma}_{s,e,n}^{int,k}(x) \\ \boldsymbol{\varepsilon}_{s,e,n}^{R,k}(x) &= \mathbf{c}_{s,e,n}^{k}(x) \cdot \boldsymbol{\sigma}_{s,e,n}^{R,k}(x) \end{aligned}$$

The residual section deformation vectors are integrated along the element using the complimentary principle of virtual work to obtain the residual element nodal displacement vector.

$$\tilde{\mathbf{d}}_{e,n}^{R,k} = \int_{0}^{L_e} \mathbf{N}_{f,e}(x)^{T} \cdot \boldsymbol{\varepsilon}_{s,e,n}^{R,k}(x)$$

The element flexibility matrix without rigid body modes $\tilde{\mathbf{c}}_{e,n}^{k}$ is determined from Eq. 2.20, and then the residual element nodal force vector will be determined form the residual element nodal displacement vector.

$$\tilde{\mathbf{f}}_{e,n}^{R,k} = [\tilde{\mathbf{c}}_{e,n}^{k}]^{-1} \cdot \tilde{\mathbf{d}}_{e,n}^{R,k} \tag{2.26}$$

The internal element nodal force vector without rigid body modes is determined from Eq. 2.25 and Eq. 2.26.

$$\tilde{\mathbf{f}}_{e,n}^{int,k} = \tilde{\mathbf{f}}_{e,n}^{k} - \tilde{\mathbf{f}}_{e,n}^{R,k}$$

Finally, the internal element nodal force vector is obtained from Eq. 2.22.

$$\overline{\mathbf{f}}_{e,n}^{int,k} = \tilde{\boldsymbol{\Gamma}}_{e}^{T} \cdot \tilde{\mathbf{f}}_{e,n}^{int,k}$$

The flexibility-based 2D beam-column element without iteration in element level to deter-

mine element state is straightforward method for the state determination procedure unlike the flexibility-based 2D beam-column element with Newton-Raphson iteration in element level to do. This element formulation uses the force interpolation functions without any assumption unlike stiffness based element with displacement interpolation functions. Due to the exact character of the force interpolation functions, no intrinsic errors exist in the flexibility-based 2D beam-column element formulation without iteration in element level to do. In addition, there are no limitations to the size of the elements, if the element has consistent section along longitudinal axis, for this reason (Carol and Murcia 1989)

Fig. 2.15(a) and (b) are the summary of the procedure on all state determinations with element iterations and without element iterations.

2.2.2.4 Nonlinear analysis using flexibility-based method

With reference to Fig. 2.12 and 2.13, and Fig. 2.16 to Fig. 2.18 for the method with Newton-Raphson iteration loop in element level and Fig. 2.19 to Fig. 2.21 for The method without iteration in element level, we will examine one single step of the Newton-Raphson method in structural level for nonlinear analysis using the flexibility-based method with section constitutive law due to force and displacement control. Force and displacement control are discussed in Sec. 4.1. Layer/fiber section procedure in Fig. 2.7 will be described in Sec. 2.3.

2.2.2.4.1 With element iterations We first examine the nonlinear analysis of flexibility-based 2D beam-column element with element iterations with reference to Fig. 2.16 to 2.18.

Step numbers are shown in Fig. 2.12, and numbers in Fig. 2.13 indicate the procedure for element and section determination during iterations.

Step 1 Compute the incremental nodal force vector $\Delta\mathbf{P}^{ext}_{t,n}$.

$$\Delta\mathbf{P}^{ext}_{t,n} = \mathbf{P}^{ext}_{t,n} - \mathbf{P}^{ext}_{t,n-1}$$

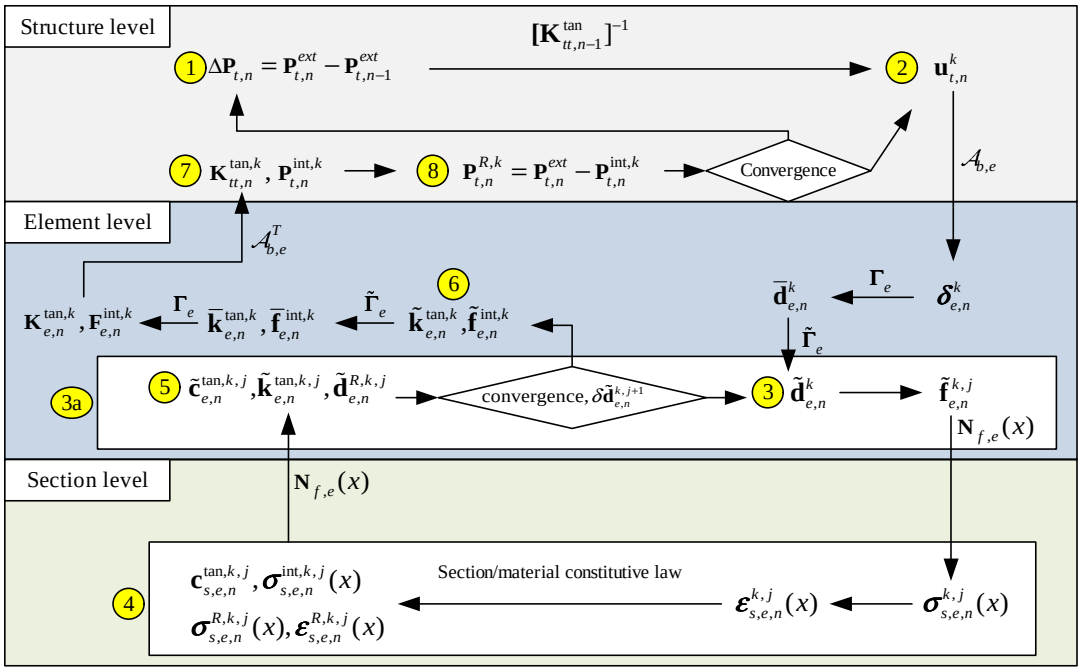
Step 2 Compute the incremental nodal displacement vector $\delta\mathbf{u}_{t,n}$ and total nodal displacement vector $\mathbf{u}_{t,n}$ in structure level. Initially, iteration started from Eq. 2.27 and $k = 1$.

If $k = 1$,

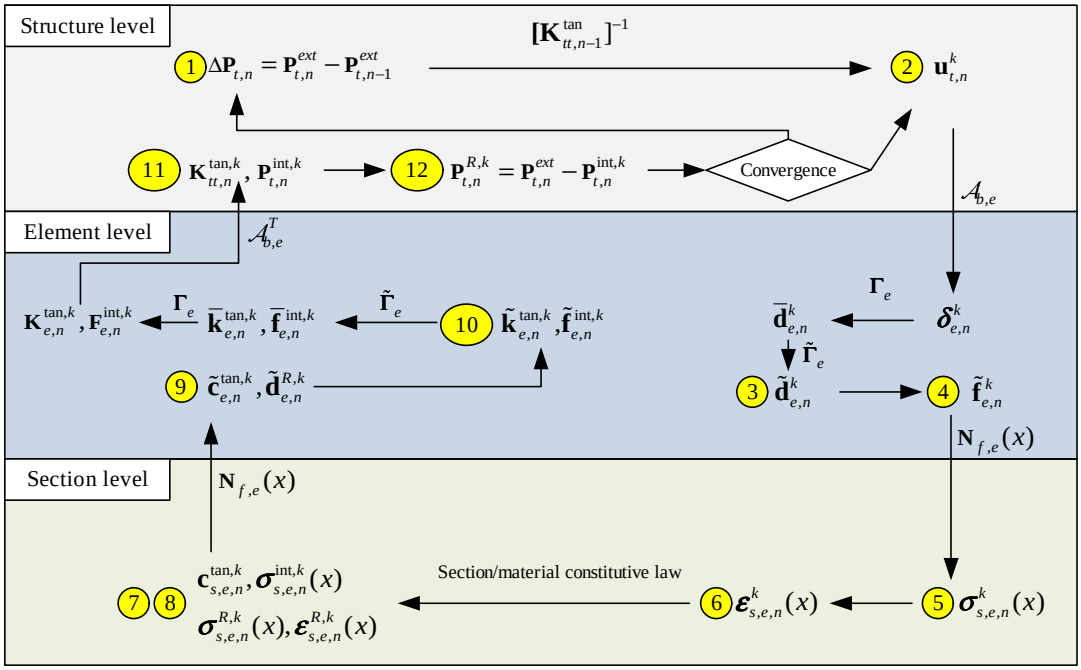
$$\begin{aligned}
\delta\mathbf{u}^{k}_{u,n} &= \mathbf{u}_{u,n} - \mathbf{u}_{u,n-1} \\
\mathbf{u}_{u,n} &= \mathbf{u}_{u,n-1} + \delta\mathbf{u}^{k}_{u,n} \\
\delta\mathbf{u}^{k}_{t,n} &= [\mathbf{K}^{tan,k-1}_{tt,n}]^{-1} \cdot [\mathbf{P}^{ext}_{t,n} - \mathbf{P}^{ext}_{t,n-1} - \mathbf{K}^{tan,k-1}_{tu,n}\delta\mathbf{u}^{k}_{u,n}] \\
\mathbf{u}^{k}_{t,n} &= \mathbf{u}^{k-1}_{t,n} + \delta\mathbf{u}^{k}_{t,n}
\end{aligned} \tag{2.27}$$

If $k \neq 1$,

$$\begin{aligned}
\mathbf{P}^{R,k}_{t,n} &= \mathbf{P}^{ext}_{t,n} - \mathbf{P}^{int,k}_{t,n} \\
\delta\mathbf{u}^{k+1}_{t,n} &= [\mathbf{K}^{tan,k}_{tt,n}]^{-1} \cdot \mathbf{P}^{R,k}_{t,n} \\
\mathbf{u}^{k+1}_{t,n} &= \mathbf{u}_{t,n-1} + \Delta\mathbf{u}^{k+1}_{t,n} = \mathbf{u}^{k}_{t,n} + \delta\mathbf{u}^{k+1}_{t,n}
\end{aligned} \tag{2.28}$$



(a) With Newton-Raphson iteration loop in element level



(b) Without iteration in element level

Fig. 2.15 State determination procedure for flexibility-based method

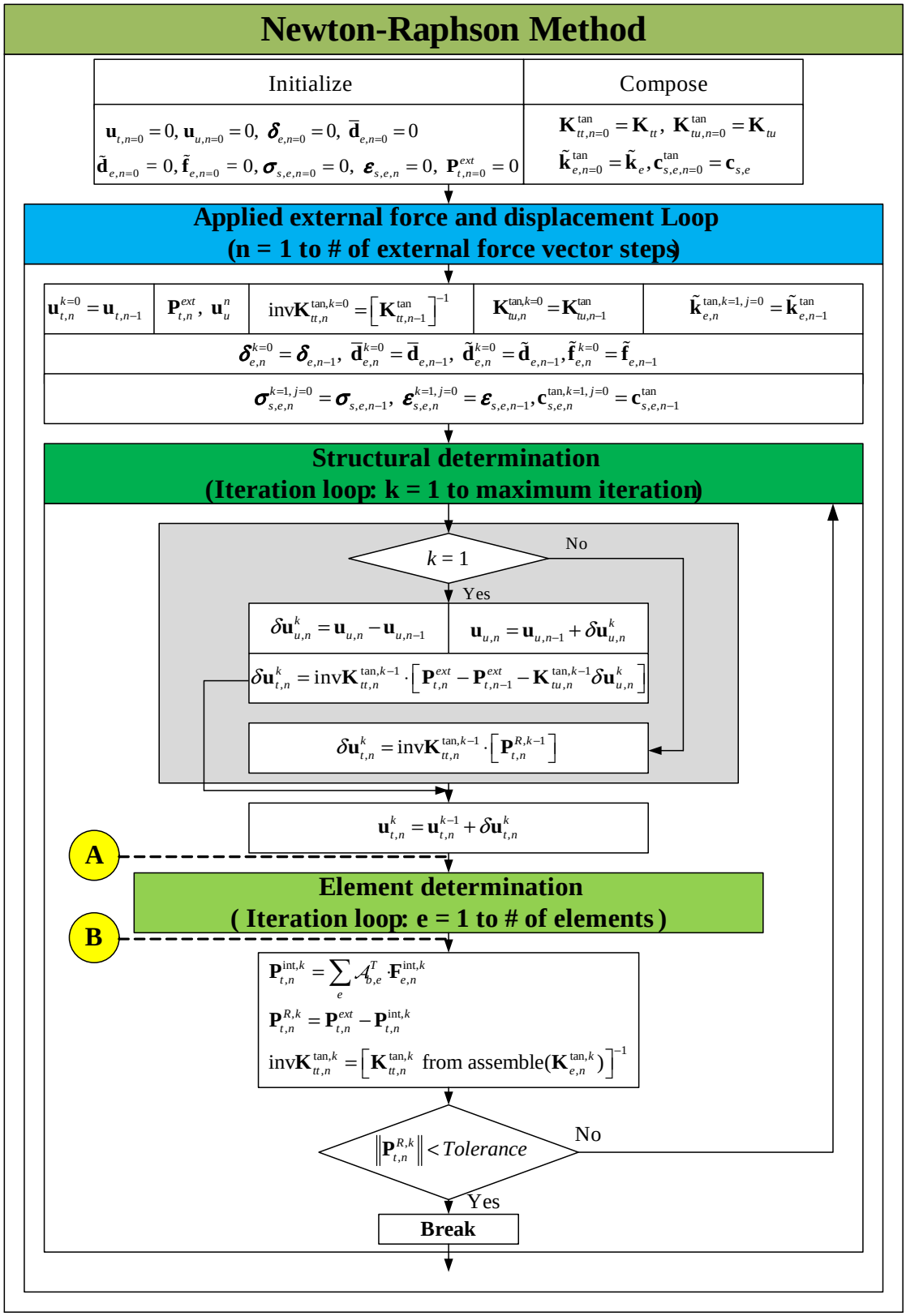


Fig. 2.16 Flow chart of nonlinear analysis using flexibility-based method with Newton-Raphson iteration in element level (1)

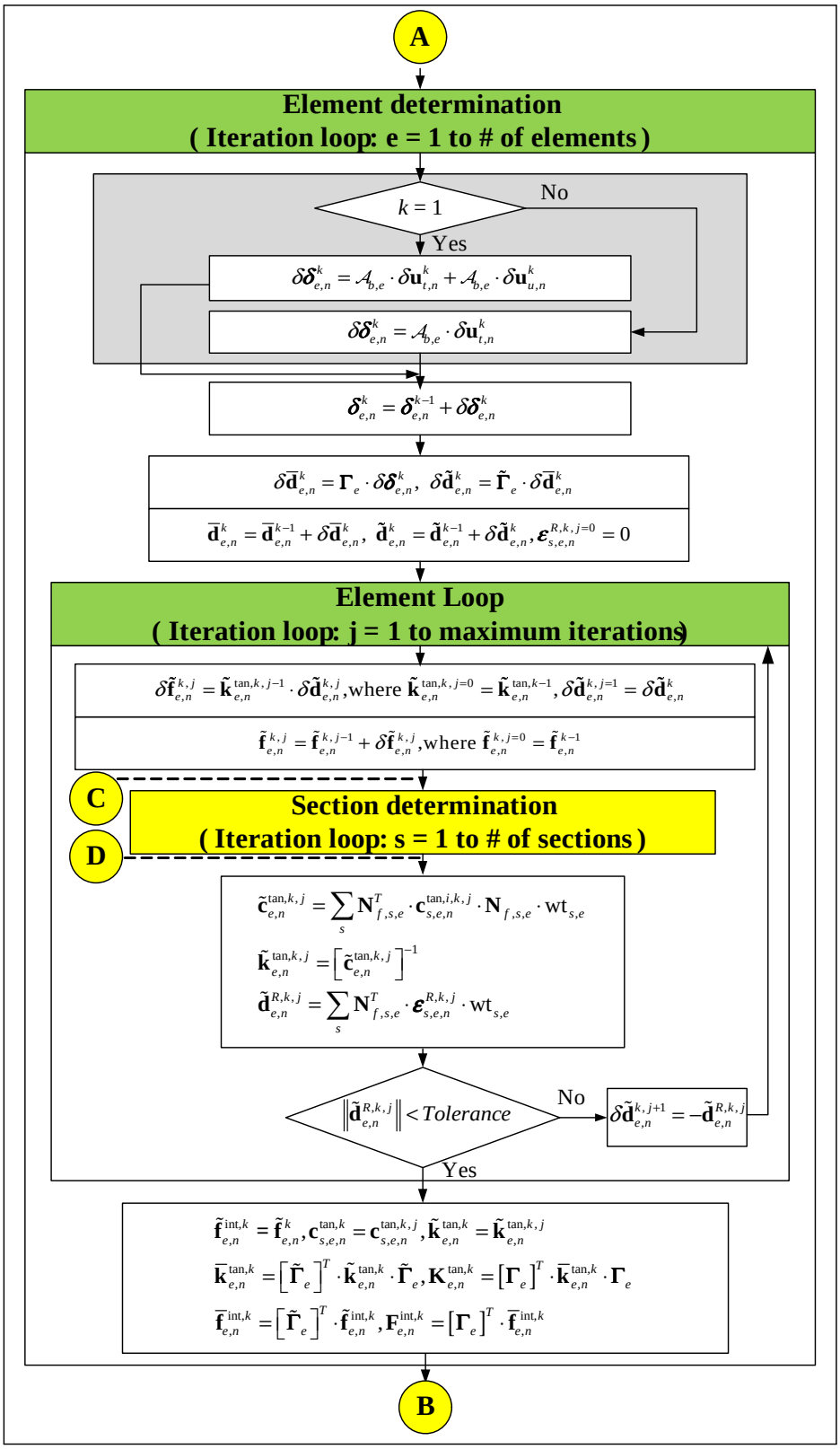


Fig. 2.17 Flow chart of nonlinear analysis using flexibility-based method with Newton-Raphson iteration in element level (2)

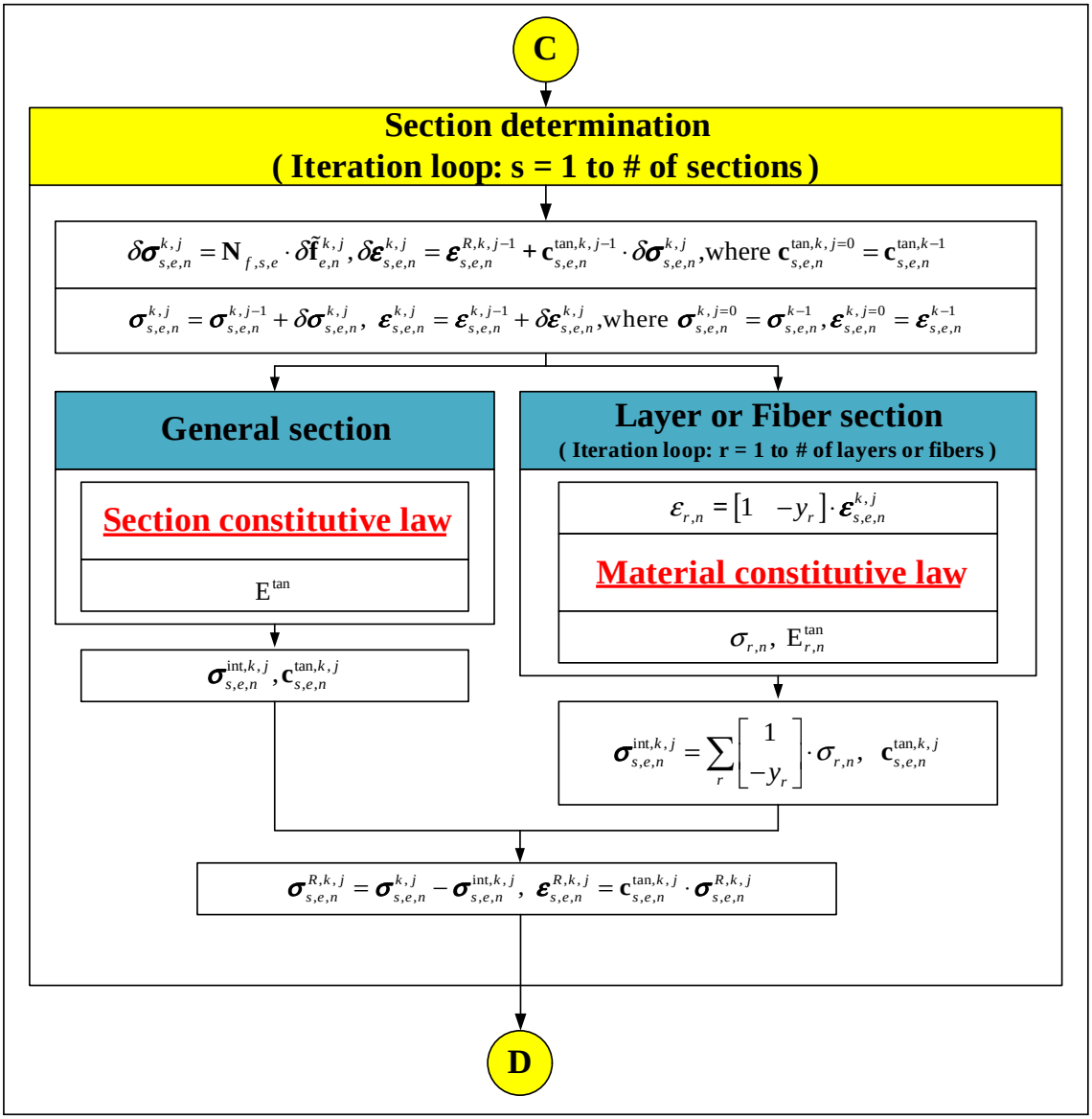


Fig. 2.18 Flow chart of nonlinear analysis using flexibility-based method with Newton-Raphson iteration in element level (3)

where, superscript k is the iteration counter, $\mathbf{P}_t^R$ the residual nodal force vector in structural level, $\Delta\mathbf{u}_{t,n}^{k+1}$ the total incremental displacement vector from the last converged step, and $\delta\mathbf{u}_n^{k+1}$ the last incremental displacement vector.

Step 3 Loop over all the elements and determine their state.

- Determine the element nodal displacement vector in global reference.
  If $k = 1$,
  $$\delta\boldsymbol{\delta}_{e,n}^k = \mathcal{A}_{b,e} \cdot \delta\mathbf{u}_{t,n}^k + \mathcal{A}_{b,e} \cdot \delta\mathbf{u}_{u,n}^k$$
  If $k \neq 1$,
  $$\delta\boldsymbol{\delta}_{e,n}^k = \mathcal{A}_{b,e} \cdot \delta\mathbf{u}_{t,n}^k$$
  where, $\mathcal{A}_{b,e}$ is displacement extracting operator.

  $$\boldsymbol{\delta}_{e,n}^k = \boldsymbol{\delta}_{e,n}^{k-1} + \delta\boldsymbol{\delta}_{e,n}^k$$

  and $\boldsymbol{\delta}_{e,n}^0$ corresponds to $\boldsymbol{\delta}_{e,n-1}$.
- Determine the element nodal displacement vector in local reference.

  $$\delta\overline{\mathbf{d}}_{e,n}^k = \boldsymbol{\Gamma}_e \cdot \delta\boldsymbol{\delta}_{e,n}^k$$
  $$\overline{\mathbf{d}}_{e,n}^k = \overline{\mathbf{d}}_{e,n}^{k-1} + \delta\overline{\mathbf{d}}_{e,n}^k$$

  and $\overline{\mathbf{d}}_{e,n}^0$ corresponds to $\overline{\mathbf{d}}_{e,n-1}$.
- Determine the element nodal displacement vector without rigid body modes in local reference.

  $$\delta\tilde{\mathbf{d}}_{e,n}^k = \tilde{\boldsymbol{\Gamma}}_e \cdot \delta\overline{\mathbf{d}}_{e,n}$$
  $$\tilde{\mathbf{d}}_{e,n}^k = \tilde{\mathbf{d}}_{e,n}^{k-1} + \delta\tilde{\mathbf{d}}_{e,n}^k$$

  and, $\overline{\mathbf{d}}_{e,n}^0$ and $\tilde{\mathbf{d}}_{e,n}^0$ correspond to $\overline{\mathbf{d}}_{e,n-1}$ and $\tilde{\mathbf{d}}_{e,n-1}$.

Step 3a Start the element state determination. Loop over all elements in the structure. The state determination of each element is performed within loop j.

- Determine the element nodal force vector without rigid body modes

  $$\delta\tilde{\mathbf{f}}_{e,n}^{k,j} = \tilde{\mathbf{k}}_{e,n}^{tan,k,j-1} \cdot \delta\tilde{\mathbf{d}}_{e,n}^{k,j}$$
  $$\tilde{\mathbf{f}}_{e,n}^{k,j} = \tilde{\mathbf{f}}_{e,n}^{k,j-1} + \delta\tilde{\mathbf{f}}_{e,n}^{k,j}$$

  $\delta\tilde{\mathbf{d}}_{e,n}^{k,j=1}$, $\tilde{\mathbf{f}}_{e,n}^{k,j=0}$, and and, $\tilde{\mathbf{k}}_{e,n}^{tan,k,j=0}$ correspond to $\delta\tilde{\mathbf{d}}_e^k$, $\tilde{\mathbf{f}}_{e,n}^{k-1}$, and $\tilde{\mathbf{k}}_{e,n}^{tan,k-1}$.

Step 4 Start the section state determination by looping over the element's sections. The total number of section may vary from element to element. Therefore, the total number of sections in an element is nIp_e and it depends on the number of integration points in the Gauss Lobatto quadrature rule.

- Determine the section force vector.
  For the section state determination, we first need to differentiate between general section and layer/fiber section as described in Sec. 2.3. For the sake of this description we consider general section only. The state determination of each section is performed within loop s where the subscript corresponds to the s^{th} section.

$$\delta\boldsymbol{\sigma}_{s,e,n}^{k,j} = \mathbf{N}_{f,s,e} \cdot \delta\tilde{\mathbf{f}}_{e,n}^{k,j}$$
$$\boldsymbol{\sigma}_{s,e,n}^{k,j} = \boldsymbol{\sigma}_{s,e,n}^{k,j-1} + \delta\boldsymbol{\sigma}_{s,e,n}^{k,j}$$

  where, $\mathbf{N}_{f,s,e}$ is the matrix derived from the force interpolation functions at s^{th} section of e^{th} element, and $\boldsymbol{\sigma}_{s,e,n}^{k,j=0}$ corresponds to $\boldsymbol{\sigma}_{s,e,n}^{k-1}$.
- Determine section deformation vector.

$$\delta\boldsymbol{\varepsilon}_{s,e,n}^{k,j} = \boldsymbol{\varepsilon}_{s,e,n}^{R,k,j-1} + \mathbf{c}_{s,e,n}^{tan,k,j-1} \cdot \delta\boldsymbol{\sigma}_{s,e,n}^{k,j}$$
$$\boldsymbol{\varepsilon}_{s,e,n}^{k,j} = \boldsymbol{\varepsilon}_{s,e,n}^{k,j-1} + \delta\boldsymbol{\varepsilon}_{s,e,n}^{k,j}$$

  and, $\boldsymbol{\varepsilon}_{s,e,n}^{k,j=0}$ and $\mathbf{c}_{s,e,n}^{tan,k,j=0}$ correspond to $\boldsymbol{\varepsilon}_{s,e,n}^{k-1}$ and $\mathbf{c}_{s,e,n}^{tan,k-1}$, and $\boldsymbol{\varepsilon}_{s,e,n}^{R,k,j=0} = 0$.
- Determine the section tangent flexibility matrix and the internal section force vector. If we assume that the section constitutive law is explicitly known, $\mathbf{c}_{s,e,n}^{tan,k,j}$ and $\boldsymbol{\sigma}_{s,e,n}^{int,k,j}$ are both determined from $\boldsymbol{\varepsilon}_{s,e,n}^{k,j}$. However, in elastic section, we need not compute $\mathbf{c}_{s,e,n}^{tan,k,j}$ again as it is identical to the initial section flexibility matrix $\tilde{\mathbf{c}}_{s,e}$.
- Determine the residual section force and deformation vector.

$$\boldsymbol{\sigma}_{s,e,n}^{R,k,j} = \boldsymbol{\sigma}_{s,e,n}^{k,j} - \boldsymbol{\sigma}_{s,e,n}^{int,k,j}$$
$$\boldsymbol{\varepsilon}_{s,e,n}^{R,k,j} = \mathbf{c}_{s,e,n}^{tan,k,j} \cdot \boldsymbol{\sigma}_{s,e,n}^{R,k,j}$$

Step 5 Determine the updated element flexibility matrix and the residual element nodal displacement vector.

The residual section deformation vectors and the section tangent flexibility matrices are integrated along the element using the complimentary principle of virtual work to obtain the residual element nodal displacement vector and the element tangent flexibility matrix.

$$\tilde{\mathbf{d}}_{e,n}^{R,k,j} = \sum_s \mathbf{N}_{f,s,e}^T \cdot \boldsymbol{\varepsilon}_{s,e,n}^{R,k,j} \cdot \mathrm{wt}_{s,e}$$
$$\tilde{\mathbf{c}}_{e,n}^{tan,k,j} = \sum_s \mathbf{N}_{f,s,e}^T \cdot \mathbf{c}_{s,e,n}^{tan,k,j} \cdot \mathbf{N}_{f,s,e} \cdot \mathrm{wt}_{s,e}$$
$$\tilde{\mathbf{k}}_{e,n}^{tan,k,j} = [\tilde{\mathbf{c}}_{e,n}^{tan,k,j}]^{-1}$$

where, $\mathrm{wt}_{s,e}$ is the weight coefficient associated with the Jacobian at the s^{th} section of the e^{th} element.

Convergence at the element level must be satisfied with the residual element nodal displacement vector.

- If $\tilde{\mathbf{d}}_{e,n}^{R,k,j}$ is within the specified tolerance, go to **Step 6**.

- If $\tilde{\mathbf{d}}_{e,n}^{R,k,j}$ is not within the specified tolerance, j and $\delta\tilde{\mathbf{d}}_{e,n}^{k,j+1}$ are updated to $j+1$ and $-\tilde{\mathbf{d}}_{e,n}^{R,k,j}$, and the next Newton-Raphson iteration initiates. We then repeat **Step 3a** through **Step 5** until convergence occurs at the element level.

Step 6 Determine the internal element nodal force vector and the element tangent stiffness matrix.

$$\begin{aligned}
\tilde{\mathbf{f}}_{e,n}^{int,k} &= \tilde{\mathbf{f}}_{e,n}^{int,k,j} \\
\overline{\mathbf{f}}_{e,n}^{int,k} &= \tilde{\mathbf{\Gamma}}_e^T \cdot \tilde{\mathbf{f}}_{e,n}^{int,k} \\
\mathbf{F}_{e,n}^{int,k} &= \mathbf{\Gamma}_e^T \cdot \overline{\mathbf{f}}_{e,n}^{int,k}
\end{aligned}$$

$$\begin{aligned}
\mathbf{c}_{s,e,n}^{tan,k} &= \mathbf{c}_{s,e,n}^{tan,k,j} \\
\tilde{\mathbf{k}}_{e,n}^{tan,k} &= \tilde{\mathbf{k}}_{e,n}^{tan,k,j} \\
\overline{\mathbf{k}}_{e,n}^{tan,k} &= \tilde{\mathbf{\Gamma}}_e^T \cdot \tilde{\mathbf{k}}_{e,n}^{tan,k} \cdot \tilde{\mathbf{\Gamma}}_e \\
\mathbf{K}_{e,n}^{tan,k} &= \mathbf{\Gamma}_e^T \cdot \overline{\mathbf{k}}_{e,n}^{tan,k} \cdot \mathbf{\Gamma}_e
\end{aligned}$$

Step 7 Determine the internal nodal force vector and the augmented tangent stiffness matrix.

$$\begin{aligned}
\mathbf{P}_{t,n}^{int,k} &= \sum_e \mathcal{A}_{b,e}^T \cdot \mathbf{F}_{e,n}^{int,k} \\
\mathbf{K}_{S,n}^{tan,k} &= \sum_e \mathcal{A}_{b,e}^T \cdot \mathbf{K}_{e,n}^{tan,k} \cdot \mathcal{A}_{b,e}
\end{aligned}$$

where, $\mathcal{A}_{b,e}^T$ is force assembling operator, and $\mathbf{K}_{S,n}^{tan,k}$ is consist of $\mathbf{K}_{tt,n}^{tan,k}$, $\mathbf{K}_{tu,n}^{tan,k}$, $\mathbf{K}_{ut,n}^{tan,k}$, and $\mathbf{K}_{uu,n}^{tan,k}$.

Step 8 Compute the residual nodal force vector at the structural level from Eq. 2.28. We then satisfy convergence with the residual nodal force vector.

- If $\mathbf{P}_{t,n}^{R,k}$ is within the specified tolerance, go to next force increment.
- If $\mathbf{P}_{t,n}^{R,k}$ is not within the specified tolerance, k is updated to $k+1$ and the next Newton-Raphson iteration initiates. Eq. 2.28 in **Step 2** through **Step 8** are repeated until convergence occurs at the structure level.

2.2.2.4.2 Without element iterations We then consider the methodology first proposed by (Carol and Murcia 1989) in which there are no element iteration with reference to Fig. 2.14 and Fig. 2.19 to 2.21.

Step 1 Compute the incemental nodal force vector $\Delta\mathbf{P}_{t,n}^{ext}$.

$$\Delta\mathbf{P}_{t,n}^{ext} = \mathbf{P}_{t,n}^{ext} - \mathbf{P}_{t,n-1}^{ext}$$

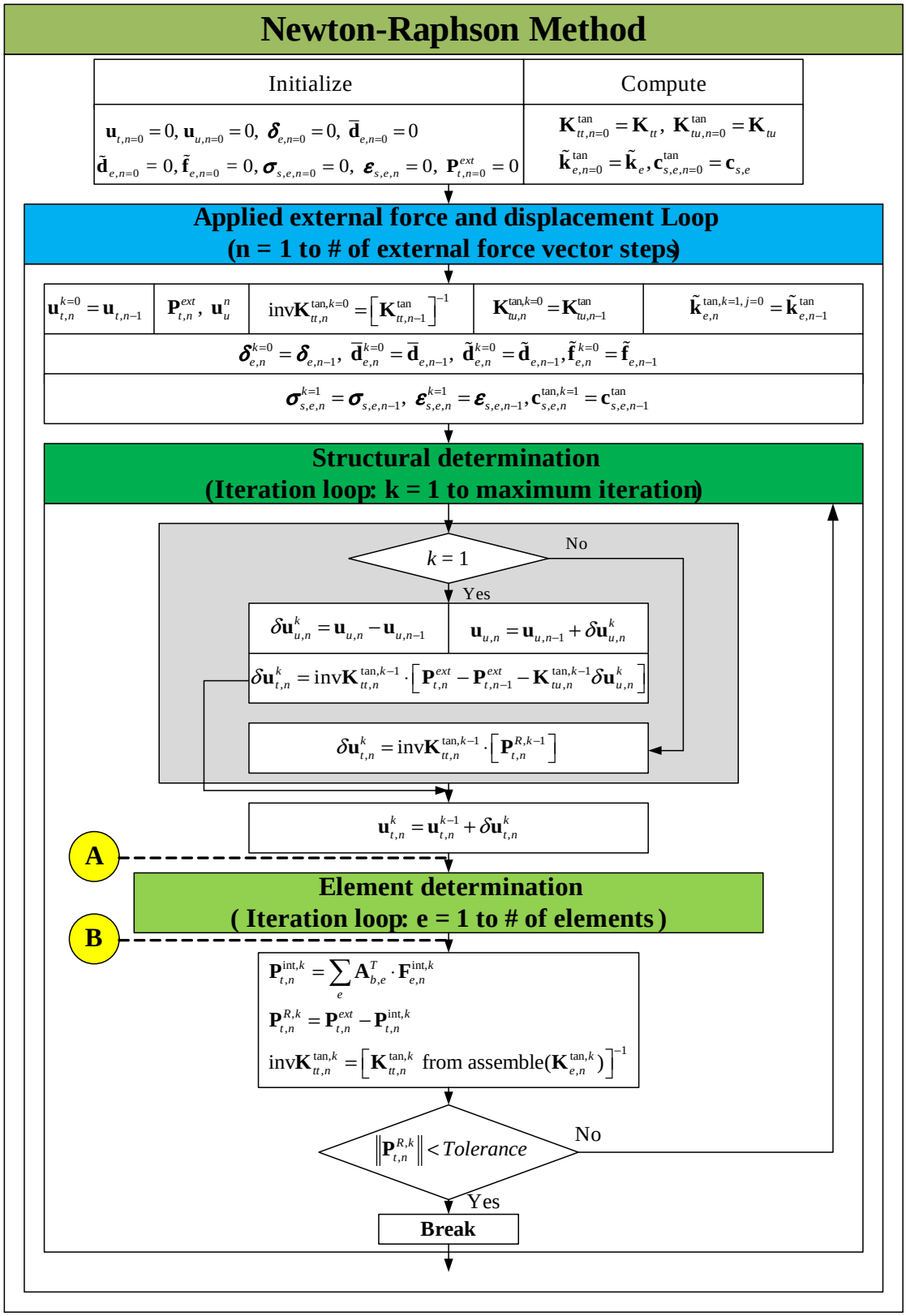


Fig. 2.19 Flow chart of nonlinear analysis using flexibility-based method without iteration in element level (1)

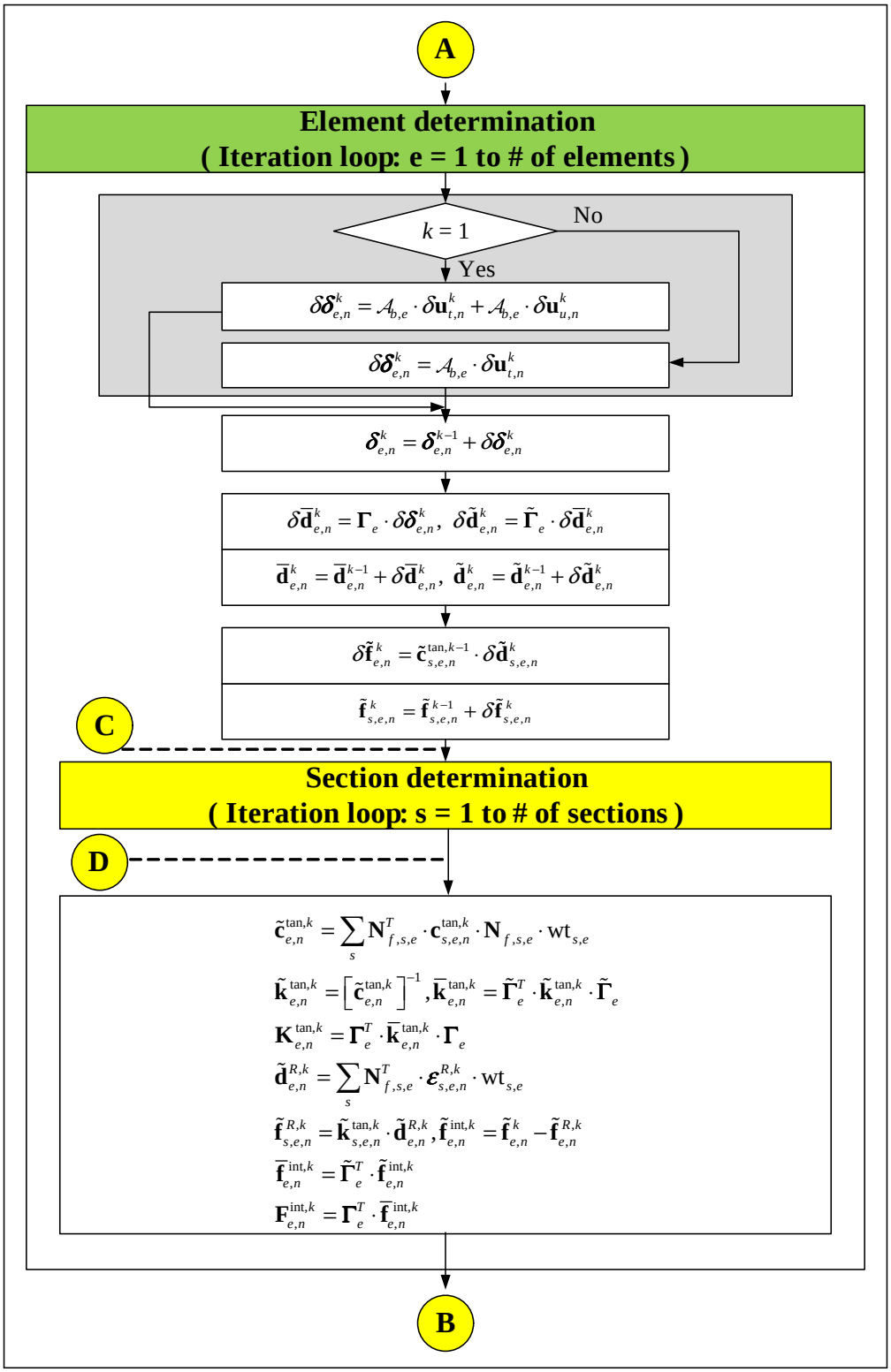


Fig. 2.20 Flow chart of nonlinear analysis using flexibility-based method without iteration in element level (2)

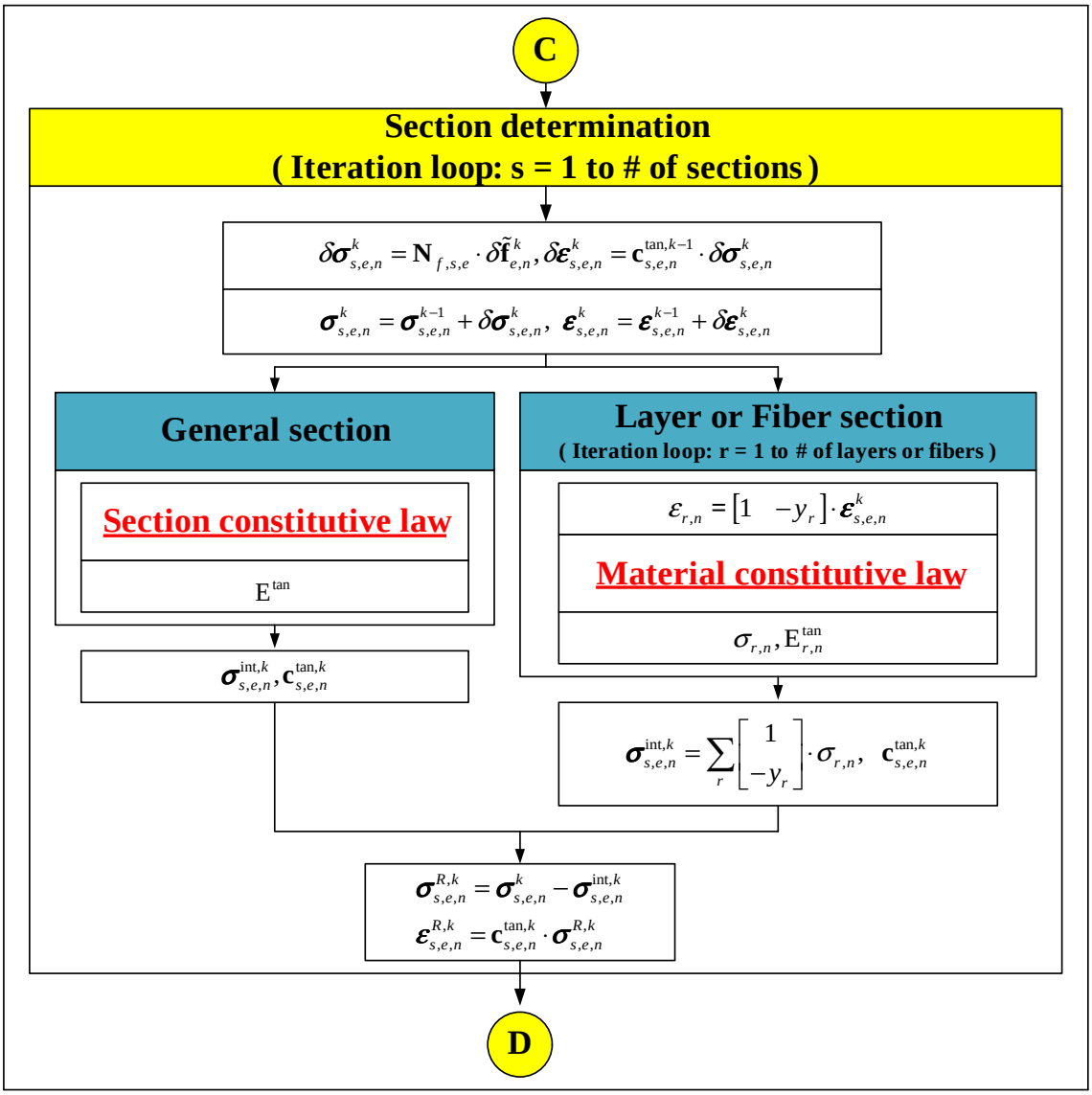


Fig. 2.21 Flow chart of nonlinear analysis using flexibility-based method without iteration in element level (3)

Step 2 Compute the incremental nodal displacement vector $\delta\mathbf{u}_{t,n}$ and total nodal displacement vector $\mathbf{u}_{t,n}$ in structure level. Initially, iteration started from Eq. 2.29 and $k = 1$.

If $k = 1$,

$$\begin{aligned}
\delta\mathbf{u}^k_{u,n} &= \mathbf{u}_{u,n} - \mathbf{u}_{u,n-1} \\
\mathbf{u}_{u,n} &= \mathbf{u}_{u,n-1} + \delta\mathbf{u}^k_{u,n} \\
\delta\mathbf{u}^k_{t,n} &= [\mathbf{K}^{tan,k-1}_{tt,n}]^{-1} \cdot [\mathbf{P}^{ext}_{t,n} - \mathbf{P}^{ext}_{t,n-1} - \mathbf{K}^{tan,k-1}_{tu,n}\delta\mathbf{u}^k_{u,n}] \\
\mathbf{u}^k_{t,n} &= \mathbf{u}^{k-1}_{t,n} + \delta\mathbf{u}^k_{t,n}
\end{aligned} \tag{2.29}$$

If $k \neq 1$,

$$\begin{aligned}
\mathbf{P}^{R,k}_{t,n} &= \mathbf{P}^{ext}_{t,n} - \mathbf{P}^{int,k}_{t,n} \\
\delta\mathbf{u}^{k+1}_{t,n} &= [\mathbf{K}^{tan,k}_{tt,n}]^{-1} \cdot \mathbf{P}^{R,k}_{t,n} \\
\mathbf{u}^{k+1}_{t,n} &= \mathbf{u}_{t,n-1} + \Delta\mathbf{u}^{k+1}_{t,n} = \mathbf{u}^k_{t,n} + \delta\mathbf{u}^{k+1}_{t,n}
\end{aligned} \tag{2.30}$$

where, superscript k is the iteration counter, $\mathbf{P}^R_t$ the residual nodal force vector in structural level, $\Delta\mathbf{u}^{k+1}_{t,n}$ the total incremental displacement vector from the last converged step, and $\delta\mathbf{u}^{k+1}_n$ the last incremental displacement vector.

Step 3 Loop over all the elements and determine their state.

- Determine the element nodal displacement vector in global reference.
  If $k = 1$,
  $$\delta\boldsymbol{\delta}^k_{e,n} = \mathcal{A}_{b,e} \cdot \delta\mathbf{u}^k_{t,n} + \mathcal{A}_{b,e} \cdot \delta\mathbf{u}^k_{u,n}$$
  If $k \neq 1$,
  $$\delta\boldsymbol{\delta}^k_{e,n} = \mathcal{A}_{b,e} \cdot \delta\mathbf{u}^k_{t,n}$$
  where, $\mathcal{A}_{b,e}$ is displacement extracting operator.
  $$\boldsymbol{\delta}^k_{e,n} = \boldsymbol{\delta}^{k-1}_{e,n} + \delta\boldsymbol{\delta}^k_{e,n}$$
  and $\boldsymbol{\delta}^0_{e,n}$ corresponds to $\boldsymbol{\delta}_{e,n-1}$.
- Determine the element nodal displacement vector in local reference.
  $$\begin{aligned}
  \delta\overline{\mathbf{d}}^k_{e,n} &= \boldsymbol{\Gamma}_e \cdot \delta\boldsymbol{\delta}^k_{e,n} \\
  \overline{\mathbf{d}}^k_{e,n} &= \overline{\mathbf{d}}^{k-1}_{e,n} + \delta\overline{\mathbf{d}}^k_{e,n}
  \end{aligned}$$
  and $\overline{\mathbf{d}}^0_{e,n}$ corresponds to $\overline{\mathbf{d}}_{e,n-1}$.
- Determine the element nodal displacement vector without rigid body modes in local

reference.

$$\delta\tilde{\mathbf{d}}^k_{e,n} = \tilde{\mathbf{\Gamma}}_e \cdot \delta\overline{\mathbf{d}}_{e,n}$$
$$\tilde{\mathbf{d}}^k_{e,n} = \tilde{\mathbf{d}}^{k-1}_{e,n} + \delta\tilde{\mathbf{d}}^k_{e,n}$$

and, $\overline{\mathbf{d}}^0_{e,n}$ and $\tilde{\mathbf{d}}^0_{e,n}$ correspond to $\overline{\mathbf{d}}_{e,n-1}$ and $\tilde{\mathbf{d}}_{e,n-1}$.

Step 4 Determine the element nodal force vector without rigid body modes

$$\delta\tilde{\mathbf{f}}^k_{e,n} = \tilde{\mathbf{k}}^{tan,k-1}_{e,n} \cdot \delta\tilde{\mathbf{d}}^k_e$$
$$\tilde{\mathbf{f}}^k_{e,n} = \tilde{\mathbf{f}}^{k-1}_{e,n} + \delta\tilde{\mathbf{f}}^k_{e,n}$$

and, $\tilde{\mathbf{f}}^{k=0}_{e,n}$ and $\tilde{\mathbf{k}}^{tan,k=0}_{e,n}$ correspond to $\tilde{\mathbf{f}}_{e,n-1}$, and $\tilde{\mathbf{k}}^{tan}_{e,n-1}$.

Step 5 Start the section state determination by looping over the element's sections. The total number of section may vary from element to element. Therefore, the total number of sections in an element is nIp_e and it depends on the number of integration points in the Gauss Lobatto quadrature rule.

- Determine the section force vector.

$$\delta\boldsymbol{\sigma}^k_{s,e,n} = \mathbf{N}_{f,s,e} \cdot \delta\tilde{\mathbf{f}}^k_{e,n}$$
$$\boldsymbol{\sigma}^k_{s,e,n} = \boldsymbol{\sigma}^{k-1}_{s,e,n} + \delta\boldsymbol{\sigma}^k_{s,e,n}$$

  where, $\mathbf{N}_{f,s,e}$ is the matrix derived from the force interpolation functions at the s^{th} section of the e^{th} element, and $\boldsymbol{\sigma}^{k=0}_{s,e,n}$ corresponds to $\boldsymbol{\sigma}_{s,e,n-1}$.

Step 6 Determine the section deformation vector. We again assume a general section. The state determination of each section is performed in loop s.

$$\delta\boldsymbol{\varepsilon}^k_{s,e,n} = \mathbf{c}^{tan,k-1}_{s,e,n} \cdot \delta\boldsymbol{\sigma}^k_{s,e,n}$$
$$\boldsymbol{\varepsilon}^k_{s,e,n} = \boldsymbol{\varepsilon}^{k-1}_{s,e,n} + \delta\boldsymbol{\varepsilon}^k_{s,e,n}$$

and, $\mathbf{c}^{tan,k=0}_{s,e,n}$ and $\boldsymbol{\varepsilon}^{k=0}_{s,e,n}$ correspond to $\tilde{\mathbf{c}}^{tan}_{s,e,n-1}$ and $\boldsymbol{\varepsilon}_{s,e,n-1}$.

Step 7 Determine the section tangent flexibility matrix and the internal section force vector. If we assume that the section constitutive law is explicitly known, $\mathbf{c}^{tan,k}_{s,e,n}$ and $\boldsymbol{\sigma}^{int,k}_{s,e,n}$ are both determined from $\boldsymbol{\varepsilon}^k_{s,e,n}$. However, in elastic section, we need not to compute $\mathbf{c}^{tan,k}_{s,e,n}$ again as it is identical to the initial section flexibility matrix $\tilde{\mathbf{c}}_{s,e}$.

- Determine the residual section force vector.

$$\boldsymbol{\sigma}^{R,k}_{s,e,n} = \boldsymbol{\sigma}^k_{s,e,n} - \boldsymbol{\sigma}^{int,k}_{s,e,n}$$

Step 8 Determine the residual section deformation vector.

$$\boldsymbol{\varepsilon}^{R,k}_{s,e,n} = \mathbf{c}^{tan,k}_{s,e,n} \cdot \boldsymbol{\sigma}^{R,k}_{s,e,n}$$

Step 9 Determine updated element flexibility matrix and the residual element nodal displacement vector.

The residual section deformation vectors and the section tangent flexibility matrices are integrated along the element using on the complimentary principle of virtual work to obtain the residual element nodal displacement vector and the element tangent flexibility matrix.

$$\begin{aligned}
\tilde{\mathbf{d}}^{R,k}_{e,n} &= \sum_s \mathbf{N}^T_{f,s,e} \cdot \boldsymbol{\varepsilon}^{R,k}_{s,e,n} \cdot \mathrm{wt}_{s,e} \\
\tilde{\mathbf{c}}^{tan,k}_{e,n} &= \sum_s \mathbf{N}^T_{f,s,e} \cdot \mathbf{c}^{tan,k}_{s,e,n} \cdot \mathbf{N}_{f,s,e} \cdot \mathrm{wt}_{s,e} \\
\tilde{\mathbf{k}}^{tan,k}_{e,n} &= [\tilde{\mathbf{c}}^{tan,k}_{e,n}]^{-1}
\end{aligned}$$

where, $\mathrm{wt}_{s,e}$ is the weight coefficient associated with the Jacobian at the s^{th} section of the e^{th} element.

Step 10 Determine the internal element nodal force vector and the element tangent stiffness matrix.

$$\begin{aligned}
\tilde{\mathbf{f}}^{R,k}_{e,n} &= \tilde{\mathbf{k}}^{tan,k}_{e,n} \cdot \tilde{\mathbf{d}}^{R,k}_{e,n} \\
\tilde{\mathbf{f}}^{int,k}_{e,n} &= \tilde{\mathbf{f}}^{k}_{e,n} - \tilde{\mathbf{f}}^{R,k}_{e,n} \\
\overline{\mathbf{f}}^{int,k}_{e,n} &= \tilde{\boldsymbol{\Gamma}}^T_e \cdot \tilde{\mathbf{f}}^{int,k}_{e,n} \\
\mathbf{F}^{int,k}_{e,n} &= \boldsymbol{\Gamma}_e{}^T \cdot \overline{\mathbf{f}}^{int,k}_{e,n} \\
\overline{\mathbf{k}}^{tan,k}_{e,n} &= \tilde{\boldsymbol{\Gamma}}^T_e \cdot \tilde{\mathbf{k}}^{tan,k}_{e,n} \cdot \tilde{\boldsymbol{\Gamma}}_e \\
\mathbf{K}^{tan,k}_{e,n} &= \boldsymbol{\Gamma}^T_e \cdot \overline{\mathbf{k}}^{tan,k}_{e,n} \cdot \boldsymbol{\Gamma}_e
\end{aligned}$$

Step 11 Determine the internal nodal force vector and the augmented tangent stiffness matrix.

$$\begin{aligned}
\mathbf{P}^{int,k}_{t,n} &= \sum_e \mathcal{A}^T_{b,e} \cdot \mathbf{F}^{int,k}_{e,n} \\
\mathbf{K}^{tan,k}_{S,n} &= \sum_e \mathcal{A}^T_{b,e} \cdot \mathbf{K}^{tan,k}_{e,n} \; \mathcal{A}_{b,e}
\end{aligned}$$

where, $\mathcal{A}^T_{b,e}$ is force assembling operator.

Step 12 Compute the residual nodal force vector at the structural level from Eq. 2.28. We then satisfy convergence with the residual nodal force vector.

Check convergence

- If $\mathbf{P}^{R,k}_{t,n}$ is within the specified tolerance, go to next force increment.

- If $\mathbf{P}_{t,n}^{R,k}$ is not within the specified tolerance, k is updated to $k+1$ and the next Newton-Raphson iteration initiates. Eq. 2.30 in **Step 2** through **Step 12** are repeated until convergence occurs at the structure level.

2.3 Layer/Fiber Section

A major limitation of 1D framework elements is their inability to capture the varying stress distribution across the section for nonlinear material as shown in Fig. 2.22. Whereas this can be readily captured by continuum finite elements, it will come at a very expensive computational price. Hence a natural approach is to adopt layer/fiber sections as shown in Fig. 2.23.

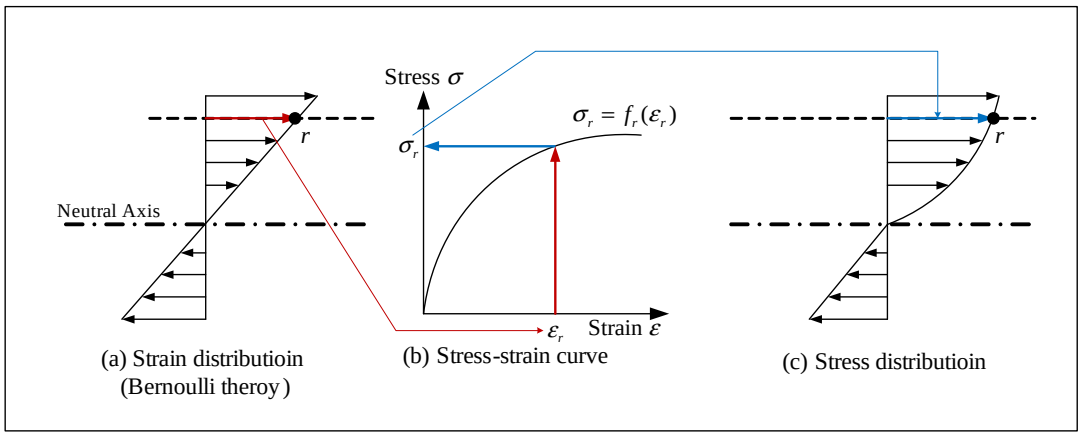


Fig. 2.22 Stress-strain curve of nonliner material

The geometric characteristics of the layers/fibers are its location in the local y or y and z reference system and the layer/fiber area $A_r(x)$, where subscript r is the r^{th} layer/fiber. The constitutive relation of the section does not have to be explicitly specified, but is derived by integration of the response of the layers/fibers, based on the uniaxial stress-strain relation of the particular material, Fig. 2.23. Again, we assume that elements have small displacements and deformations and that plane sections remain plane.

The layer/fiber strains are determined from the layer/fiber stress-strain relations. However, the determination of layer/fiber stresses from section forces is a statically indeterminate problem for a section with more than two layers/fibers. The layer/fiber stresses cannot be determined from the axial force and bending moment at the section, since there are only two equilibrium equations in the uniaxial bending case for three or more unknown stresses. One possible solution is to assume a stress distribution within the section. This solution is described in Fig. 2.24, since layer/fiber stress-strain relations are typically expressed as explicit functions of strain. Layer/fiber stresses and stiffnesses are thus determined from the layer/fiber stress-strain relations. The internal section force vectors $\boldsymbol{\sigma}_{s,e}^{int}(x)$ are computed from the layer/fiber stress distribution and section tangent stiffness matrices $\mathbf{k}_{s,e}^{tan}(x)$ are assembled from the layer/fiber tangent stiffnesses. In the uniaxial bending case, $\mathbf{k}_{s,e}^{tan}(x)$ takes the form

$$\mathbf{k}_{s,e}^{tan}(x) = \sum_r \left[\begin{array}{c|cc} & N_x(x) & M_z(x) \\ \hline \varepsilon(x) & E_r^{tan}(x)\cdot A_r(x) & -E_r^{tan}(x)\cdot A_r(x)\cdot y_r \\ \phi(x) & -E_r^{tan}(x)\cdot A_r(x)\cdot y_r & E_r^{tan}(x)\cdot A_r(x)\cdot y_r^2 \end{array}\right]$$

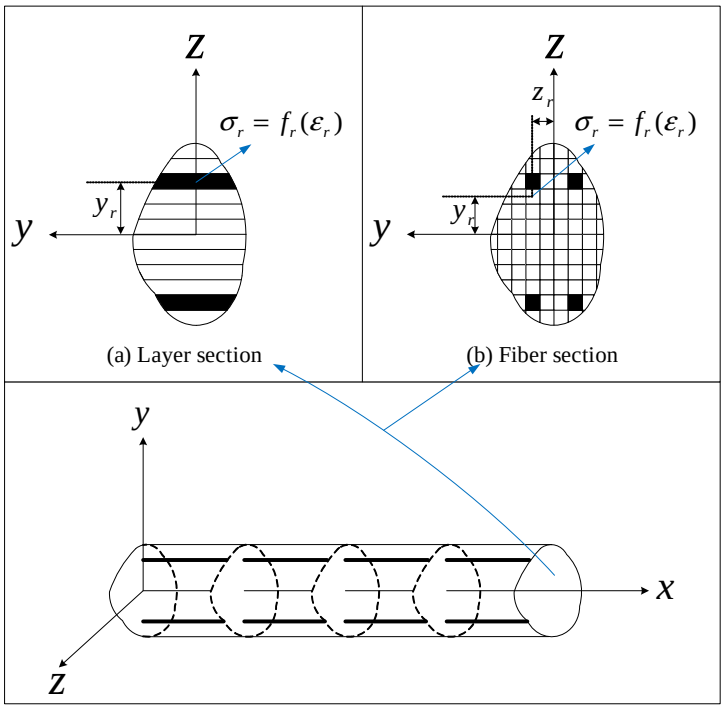


Fig. 2.23 Layer/fiber sections-Distribution of control sections and section subdivision into layers/fibers

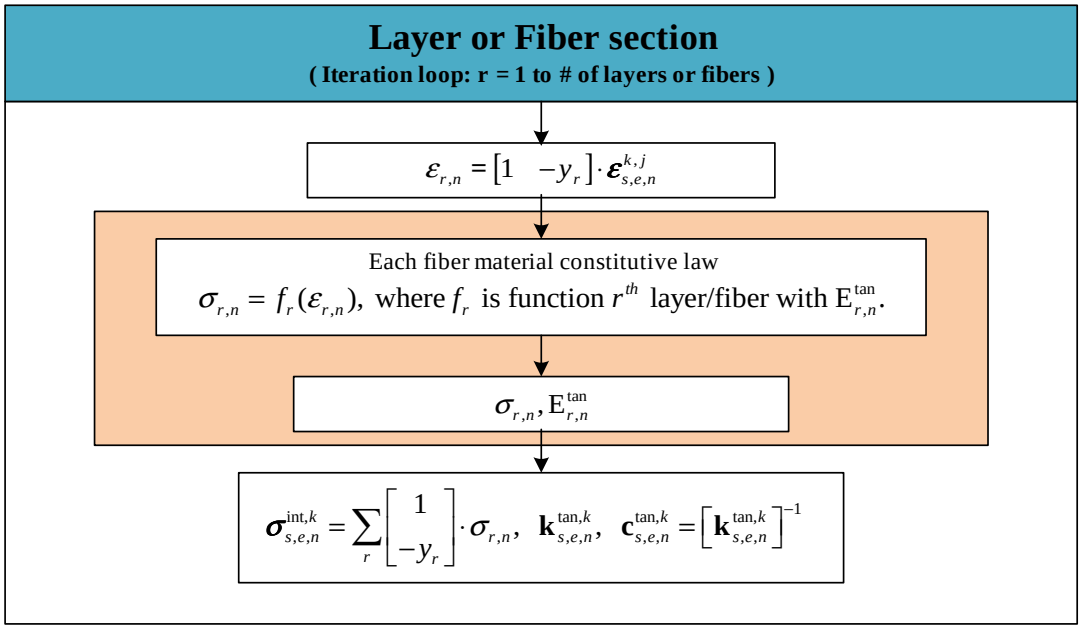


Fig. 2.24 Layer/fiber section state determination

We note that this cross sectional definition allows us to easily specify longitudinal steel reinforcement. Shear reinforcement, on the other hand, can not be explicitly modeled, however, common practice is to assign modified properties to the confined concrete (Taucer et al. 1991).

2.4 Zero-Length 2D Element Formulation

As discussed in Sec. 1.1.1.1 a methodology to account for nonlinearities in frame analysis is lumped plasticity. In this approach we consider that plastic hinges (or their formulation) contribute for the structural nonlinearity. Those hinges typically form at the end node regions. Therefore, zero-length element can be used at the end of beam or beam-column elements.

2.4.1 Formulation

The element end deformations in the reinforced concrete are composed of two types:

- flexural deformation that causes inelastic strains
- element end rotation which may be caused be the slip of longitudinal reinforcement in reinforced concrete or plastic hinges in steel members.

Fig. 2.25 describes zero-length 2D element and examples for usage. Its formulation does not account for coupling of the three possible degrees of freedom.

Constitutive law Section constitutive law is expressed as

$$\underbrace{\begin{Bmatrix} N_x \\ V_y \\ M_z \end{Bmatrix}}_{\boldsymbol{\sigma}_s} = \underbrace{\begin{bmatrix} [EA]^{tan} & 0 & 0 \\ 0 & [GA]^{tan} & 0 \\ 0 & 0 & [EI_z]^{tan} \end{bmatrix}}_{\mathbf{k}_s^{tan}} \underbrace{\begin{Bmatrix} \overline{u}_{x2} - \overline{u}_{x1} \\ \overline{v}_{y2} - \overline{v}_{y1} \\ \overline{\theta}_{z2} - \overline{\theta}_{z1} \end{Bmatrix}}_{\boldsymbol{\varepsilon}_s}$$

where, $[EA]^{tan}$, $[GA]^{tan}$ and $[EI_z]^{tan}$ are tangent stiffnesses associated with axial, shear and moment.

Equilibrium

Composing equilibrium equations between point A and point B in Fig. 2.26,

$$\begin{aligned} \overline{N}_{x1} &= [EA]^{tan} \cdot (\overline{u}_{x1} - \overline{u}_{x2}) \\ \overline{V}_{y1} &= [GA]^{tan} \cdot (\overline{v}_{y1} - \overline{v}_{y2}) \\ \overline{M}_{z1} &= [EI_z]^{tan} \cdot (\overline{\theta}_{z1} - \overline{\theta}_{z2}) \end{aligned} \tag{2.31}$$

Likewise between point B and point C,

$$\begin{aligned} \overline{N}_{x2} &= [EA]^{tan} \cdot (\overline{u}_{x2} - \overline{u}_{x1}) \\ \overline{V}_{y2} &= [GA]^{tan} \cdot (\overline{v}_{y2} - \overline{v}_{y1}) \\ \overline{M}_{z2} &= [EI_z]^{tan} \cdot (\overline{\theta}_{z2} - \overline{\theta}_{z1}) \end{aligned} \tag{2.32}$$

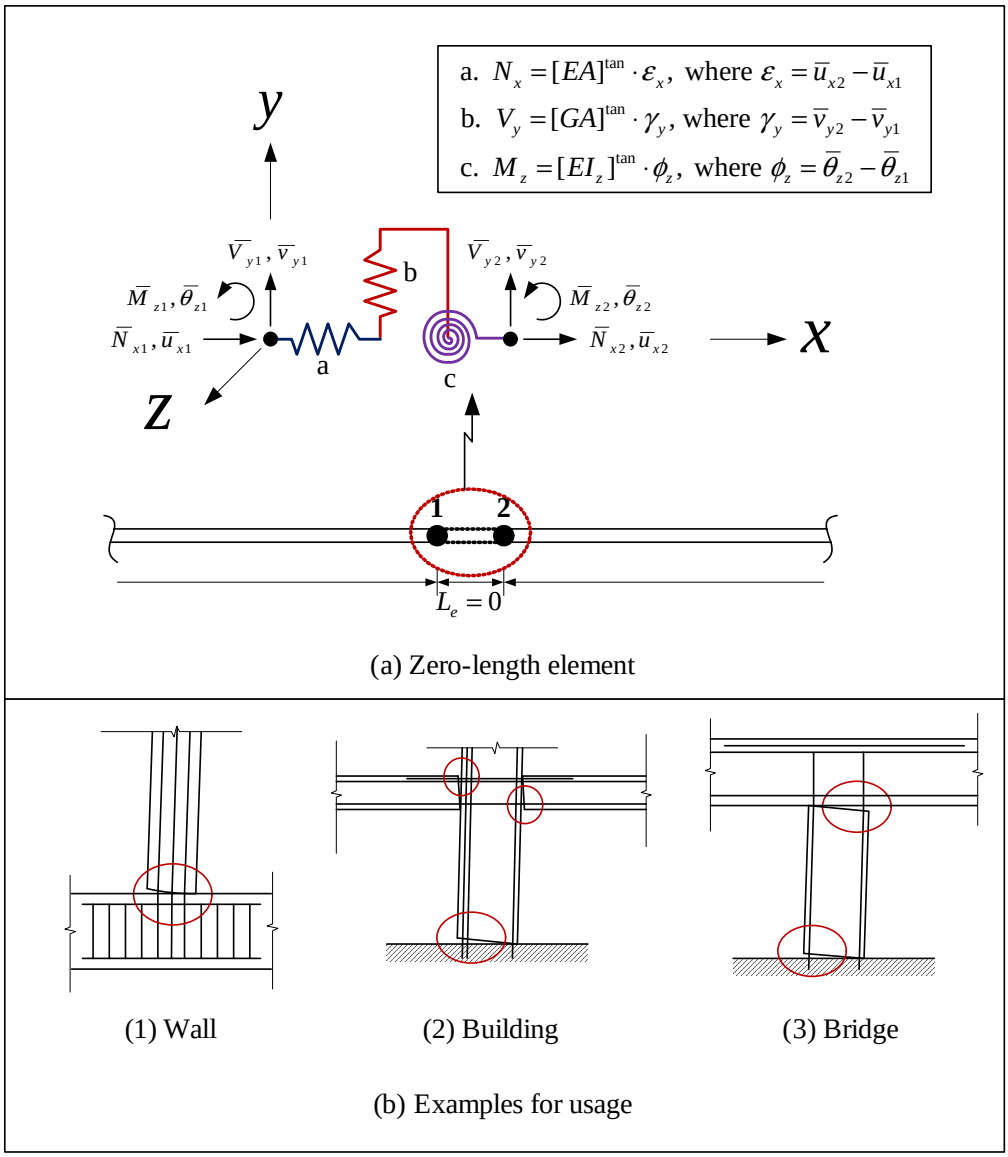


Fig. 2.25 Zero-length 2D element(1)

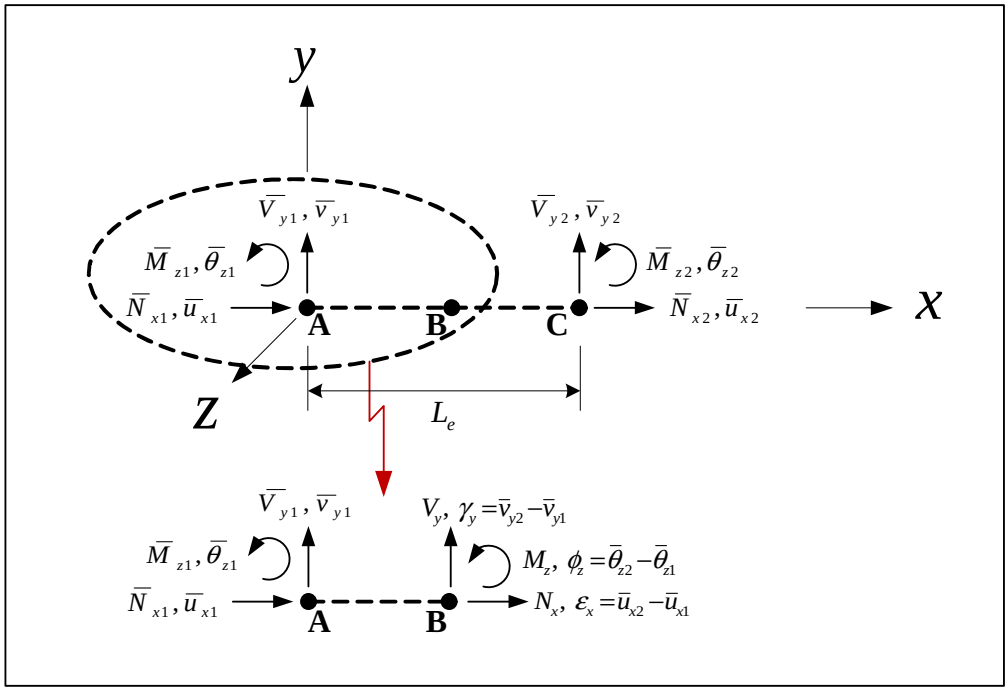


Fig. 2.26 Zero-length 2D element(2)

Rewriting Eq. 2.31 and 2.32 to matrix form, the relationship between element nodal force and displacement vector is given by

$$\underbrace{\begin{Bmatrix} \overline{N}_{x1} \\ \overline{V}_{y1} \\ \overline{M}_{z1} \\ \overline{N}_{x2} \\ \overline{V}_{y2} \\ \overline{M}_{z2} \end{Bmatrix}}_{\overline{\mathbf{f}}_e} = \overline{\mathbf{k}}_e^{tan} \underbrace{\begin{Bmatrix} \overline{u}_{x1} \\ \overline{v}_{y1} \\ \overline{\theta}_{z1} \\ \overline{u}_{x2} \\ \overline{v}_{y2} \\ \overline{\theta}_{z2} \end{Bmatrix}}_{\overline{\mathbf{d}}_e}$$

where, $\overline{\mathbf{k}}_e^{tan}$ is the element stiffness matrix in local reference.

$$\overline{\mathbf{k}}_e^{tan} = \begin{bmatrix} [EA]^{tan} & 0 & 0 & -[EA]^{tan} & 0 & 0 \\ 0 & [GA]^{tan} & 0 & 0 & -[GA]^{tan} & 0 \\ 0 & 0 & [EI_z]^{tan} & 0 & 0 & -[EI_z]^{tan} \\ -[EA]^{tan} & 0 & 0 & [EA]^{tan} & 0 & 0 \\ 0 & -[GA]^{tan} & 0 & 0 & [GA]^{tan} & 0 \\ 0 & 0 & -[EI_z]^{tan} & 0 & 0 & [EI_z]^{tan} \end{bmatrix} \quad (2.33)$$

2.4.2 Coordinate system in zero-length 2D element

Coordinate system in zero-length 2D element is same as in Sec. 2.2.1.2, Fig. 2.2.

2.4.3 Element state determination with 2 Dimension and 3 degrees of freedom per node

With reference Fig. 2.27, we will examine one single step of zero-length 2D element for nonlinear analysis.

Step 1: Determine the section deformation vector, axial deformation, shear deformation and curvature.

For each deformation, we extract the associated components from $\overline{\mathbf{d}}^k_{e,n}$.

$$\begin{aligned}
\overline{\mathbf{d}}^k_{e,n} &= \lfloor \overline{u}^k_{x1,e,n}\ \overline{v}^k_{y1,e,n}\ \overline{\theta}^k_{z1,e,n}\ \overline{u}^k_{x2,e,n}\ \overline{v}^k_{y2,e,n}\ \overline{\theta}^k_{z2,e,n} \rfloor^T \\
\boldsymbol{\varepsilon}^k_{s,e,n} &= \lfloor \varepsilon^k_{x,e,n},\ \gamma^k_{y,e,n},\ \phi^k_{z,e,n} \rfloor^T \\
\varepsilon^k_{x,e,n} &= \overline{u}^k_{x2,e,n} - \overline{u}^k_{x2,e,n} \qquad (2.34) \\
\gamma^k_{y,e,n} &= \overline{v}^k_{y2,e,n} - \overline{v}^k_{y2,e,n} \qquad (2.35) \\
\phi^k_{z,e,n} &= \overline{\theta}^k_{z2,e,n} - \overline{\theta}^k_{z2,e,n} \qquad (2.36)
\end{aligned}$$

where, Eq. 2.34 defines axial section deformation, Eq. 2.35 the shear deformation, and Eq. 2.36 the curvature.

Step 2: Determine the section tangent stiffness associated with axial force-deformation, shear force-deformation, and moment-curvature in the section constitutive laws. Section constitutive laws modified with several variables in function of material constitutive law associated with uniaxial stress-strain relationship can be used (Liel 2008). The internal section force vector is determined next. If we assume that the section constitutive law is explicitly known, $\mathbf{k}^{tan,k}_{s,e,n}$ and $\boldsymbol{\sigma}^{int,k}_{s,e,n}$ are determined from $\boldsymbol{\varepsilon}^k_{s,e,n}$. However, in elastic section, we need not to compute $\mathbf{k}^{tan,k}_{s,e,n}$ again as it is identical to the initial section stiffness matrix $\mathbf{k}_{s,e}$.

For an elastic section,

$$\begin{aligned}
\mathbf{k}^{tan}_{s,e,n} &= \mathbf{k}_{s,e} \\
\underbrace{\begin{Bmatrix} N^{int,k}_{x,e,n} \\ V^{int,k}_{y,e,n} \\ M^{int,k}_{z,e,n} \end{Bmatrix}}_{\boldsymbol{\sigma}^{int,k}_{s,e,n}} &= \mathbf{k}^{tan}_{s,e,n} \underbrace{\begin{Bmatrix} \varepsilon^k_{x,e,n} \\ \gamma^k_{y,e,n} \\ \phi^k_{z,e,n} \end{Bmatrix}}_{\boldsymbol{\varepsilon}^k_{s,e,n}}
\end{aligned}$$

where, $\mathbf{k}^{tan,k}_{s,e,n}$ is the section tangent stiffness matrix at k^{th} iteration.

Step 3: Determine the internal element nodal force vector and the element tangent stiffness matrix from Eq. 2.33.

$$\overline{\mathbf{f}}^{int,k}_{e,n} = \lfloor N^{int,k}_{x,e,n}, V^{int,k}_{y,e,n}, M^{int,k}_{z,e,n}, -N^{int,k}_{x,e,n}, -V^{int,k}_{y,e,n}, -M^{int,k}_{z,e,n} \rfloor^T$$

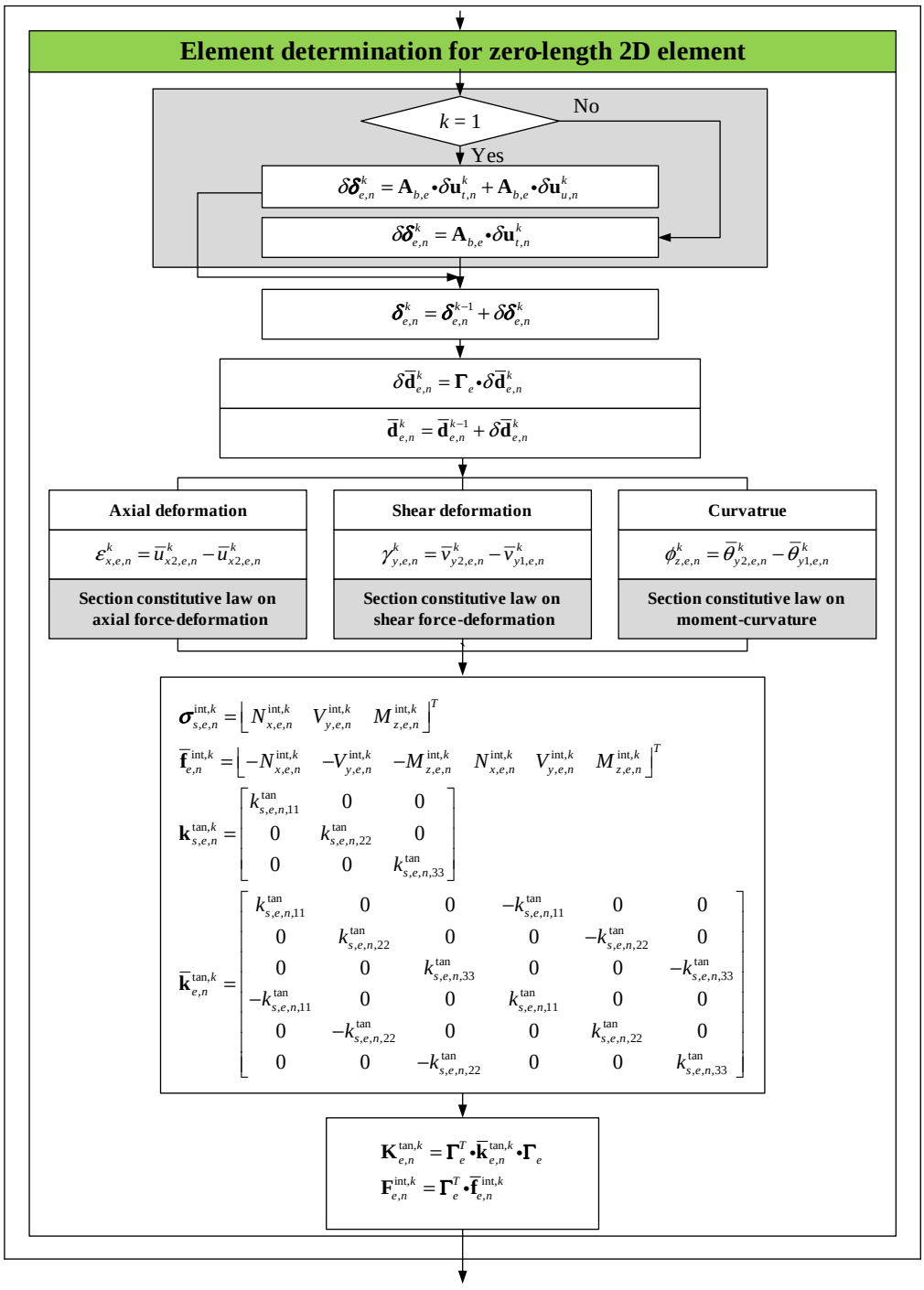


Fig. 2.27 Flow chart of zero-length 2D element for element state determination

$$\overline{\mathbf{k}}_e^{tan,k} = \begin{bmatrix} EA_{e,n}^{tan,k} & 0 & 0 & -EA_{e,n}^{tan,k} & 0 & 0 \\ 0 & GA_{e,n}^{tan,k} & 0 & 0 & -GA_{e,n}^{tan,k} & 0 \\ 0 & 0 & EI_{z,e,n}^{tan,k} & 0 & 0 & -EI_{z,e,n}^{tan,k} \\ -EA_{e,n}^{tan,k} & 0 & 0 & EA_{e,n}^{tan,k} & 0 & 0 \\ 0 & -GA_{e,n}^{tan,k} & 0 & 0 & GA_{e,n}^{tan,k} & 0 \\ 0 & 0 & -EI_{z,e,n}^{tan,k} & 0 & 0 & EI_{z,e,n}^{tan,k} \end{bmatrix}$$

where, $\overline{\mathbf{k}}_{e,n}^{tan,k}$ is the element tangent stiffness matrix in local reference.

We determine $\mathbf{F}_{e,n}^{int,k}$ and $\mathbf{K}_{e,n}^{tan,k}$.

$$\begin{aligned} \mathbf{F}_{e,n}^{int,k} &= \mathbf{\Gamma}_e{}^T \cdot \overline{\mathbf{f}}_{e,n}^{int,k} \\ \mathbf{K}_{e,n}^{tan,k} &= \mathbf{\Gamma}_e{}^T \cdot \overline{\mathbf{k}}_{e,n}^{tan,k} \cdot \mathbf{\Gamma}_e \end{aligned}$$

2.5 Zero-Length Section Element Formulation

Zero-length section element is analogous to the zero length element, however, it uses layer/fiber. This element enables us to model the shift in center of section rotation which may occur (in bar-slip for example). The element is formulated on the basis of coupled axial force and moment.

Fig. 2.28 describes zero-length 2D section element.

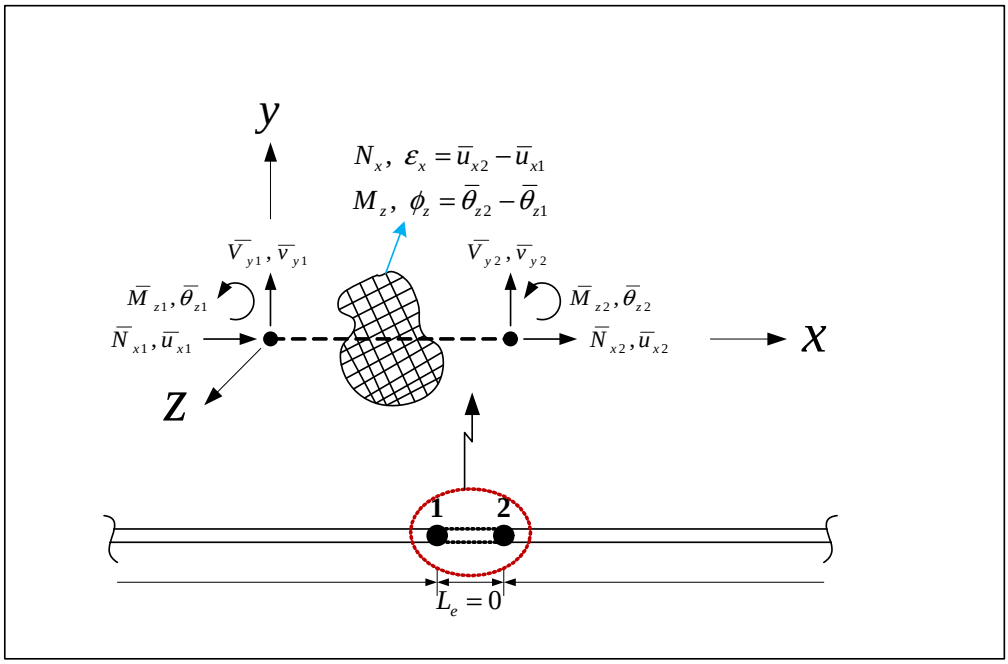


Fig. 2.28 Zero-length 2D section element(1)

2.5.1 Formulation

Constitutive law Section constitutive law is expressed as

$$\underbrace{\left\{ \begin{array}{c} N_x \\ M_z \end{array} \right\}}_{\boldsymbol{\sigma}_s} = \underbrace{\left[\begin{array}{cc} \mathrm{k}_{s,11}^{tan} & \mathrm{k}_{s,12}^{tan} \\ \mathrm{k}_{s,21}^{tan} & \mathrm{k}_{s,22}^{tan} \end{array} \right]}_{\mathbf{k}_s^{tan}} \cdot \underbrace{\left\{ \begin{array}{c} \overline{u}_{x2} - \overline{u}_{x1} \\ \overline{\theta}_{z2} - \overline{\theta}_{z1} \end{array} \right\}}_{\boldsymbol{\varepsilon}_s}$$

where, $\mathbf{k}_s^{tan}$ is the section tangent stiffness matrix obtained from layer/fiber state determination.

Equilibrium Zero-length section element is based on Bernoulli beam theory.

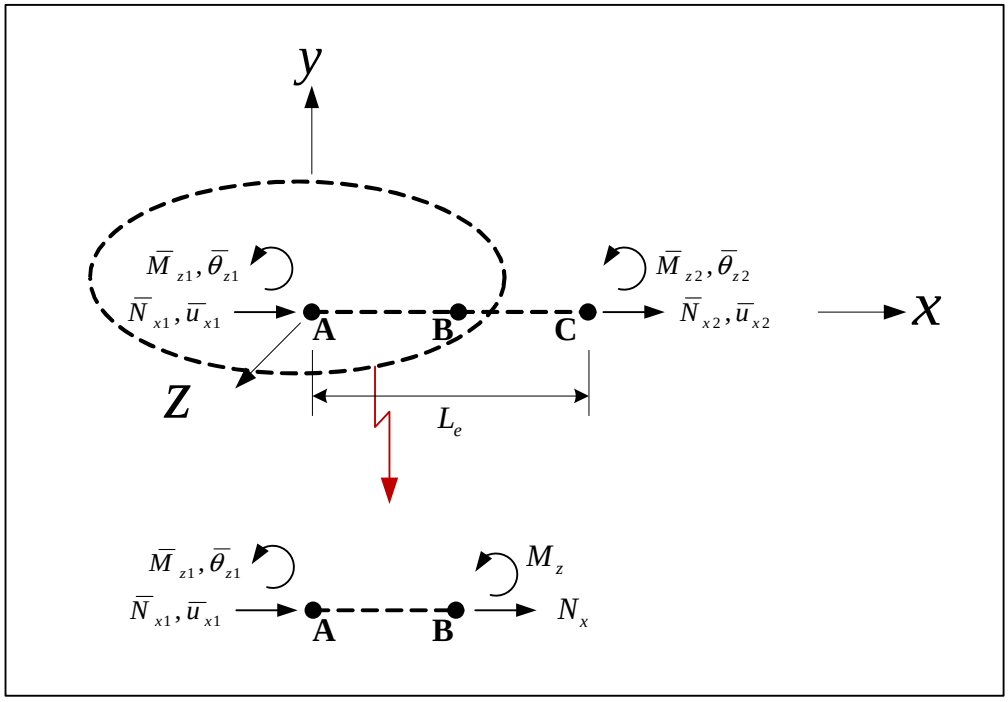


Fig. 2.29 Zero-length 2D section element(2)

Composing equilibrium equations between point A and point B in Fig. 2.29,

$$\begin{aligned} \overline{N}_{x1} &= k_{s,11}^{tan} \cdot (\overline{u}_{x1} - \overline{u}_{x2}) + k_{s,12}^{tan} \cdot (\overline{\theta}_{z1} - \overline{\theta}_{z2}) \\ \overline{M}_{z1} &= k_{s,21}^{tan} \cdot (\overline{u}_{x1} - \overline{u}_{x2}) + k_{s,22}^{tan} \cdot (\overline{\theta}_{z1} - \overline{\theta}_{z2}) \end{aligned} \tag{2.37}$$

Likewise between point B and point C,

$$\begin{aligned} \overline{N}_{x2} &= k_{s,11}^{tan} \cdot (\overline{u}_{x2} - \overline{u}_{x1}) + k_{s,12}^{tan} \cdot (\overline{\theta}_{z2} - \overline{\theta}_{z1}) \\ \overline{M}_{z2} &= k_{s,21}^{tan} \cdot (\overline{u}_{x2} - \overline{u}_{x1}) + k_{s,22}^{tan} \cdot (\overline{\theta}_{z2} - \overline{\theta}_{z1}) \end{aligned} \tag{2.38}$$

Rewriting Eq. 2.37 and 2.38 to matrix form, the relationship between element nodal force

and displacement vector is given by

$$\underbrace{\begin{Bmatrix} \overline{N}_{x1} \\ 0 \\ \overline{M}_{z1} \\ \overline{N}_{x2} \\ 0 \\ \overline{M}_{z2} \end{Bmatrix}}_{\overline{\mathbf{f}}_e} = \overline{\mathbf{k}}_e^{tan} \underbrace{\begin{Bmatrix} \overline{u}_{x1} \\ 0 \\ \overline{\theta}_{z1} \\ \overline{u}_{x2} \\ 0 \\ \overline{\theta}_{z2} \end{Bmatrix}}_{\overline{\mathbf{d}}_e}$$

where, $\overline{\mathbf{k}}_e^{tan}$ is the element stiffness matrix in local reference.

$$\overline{\mathbf{k}}_e^{tan} = \begin{bmatrix} \mathrm{k}_{s,11}^{tan} & 0 & \mathrm{k}_{s,12}^{tan} & -\mathrm{k}_{s,11}^{tan} & 0 & -\mathrm{k}_{s,12}^{tan} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \mathrm{k}_{s,21}^{tan} & 0 & \mathrm{k}_{s,22}^{tan} & -\mathrm{k}_{s,21}^{tan} & 0 & -\mathrm{k}_{s,22}^{tan} \\ -\mathrm{k}_{s,11}^{tan} & 0 & -\mathrm{k}_{s,12}^{tan} & \mathrm{k}_{s,11}^{tan} & 0 & \mathrm{k}_{s,12}^{tan} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -\mathrm{k}_{s,21}^{tan} & 0 & -\mathrm{k}_{s,22}^{tan} & \mathrm{k}_{s,21}^{tan} & 0 & \mathrm{k}_{s,22}^{tan} \end{bmatrix} \tag{2.39}$$

2.5.2 Coordinate system in zero-length 2D section element

Coordinate system in zero-length 2D element is same as in Sec. 2.2.1.2, Fig. 2.2.

2.5.3 Element state determination with 2 Dimension and 3 degrees of freedom per node

With reference Fig. 2.30, we will examine one single step of zero-length 2D section element for nonlinear analysis.

Step 1: Determine the section deformation vector, axial deformation and curvature.

For each deformation, we extracts the associated components from $\overline{\mathbf{d}}_{e,n}^k$.

$$\begin{aligned} \overline{\mathbf{d}}_{e,n}^k &= \lfloor \overline{u}_{x1,e,n}^k \; 0 \; \overline{\theta}_{z1,e,n}^k \; \overline{u}_{x2,e,n}^k \; 0 \; \overline{\theta}_{z2,e,n}^k \rfloor^T \\ \boldsymbol{\varepsilon}_{s,e,n}^k &= \lfloor \varepsilon_{x,e,n}^k, \; \phi_{z,e,n}^k \rfloor^T \\ \varepsilon_{x,e,n}^k &= \overline{u}_{x2,e,n}^k - \overline{u}_{x2,e,n}^k \\ \phi_{z,e,n}^k &= \overline{\theta}_{z2,e,n}^k - \overline{\theta}_{z2,e,n}^k \end{aligned}$$

Step 2: Determine the section tangent stiffness associated with axial force-deformation and moment-curvature using layer/fiber state determination in Sec. 2.3. Determine next the internal section force vector. If we assume that the material constitutive law is explicitly known, $\mathbf{k}_{s,e,n}^{tan,k}$ and $\boldsymbol{\sigma}_{s,e,n}^{int,k}$ are determined from $\boldsymbol{\varepsilon}_{s,e,n}^k$. However, in the section with elastic material, we need not to compute $\mathbf{k}_{s,e,n}^{tan,k}$ again as it is identical to the initial section stiffness matrix $\mathbf{k}_{s,e}$.

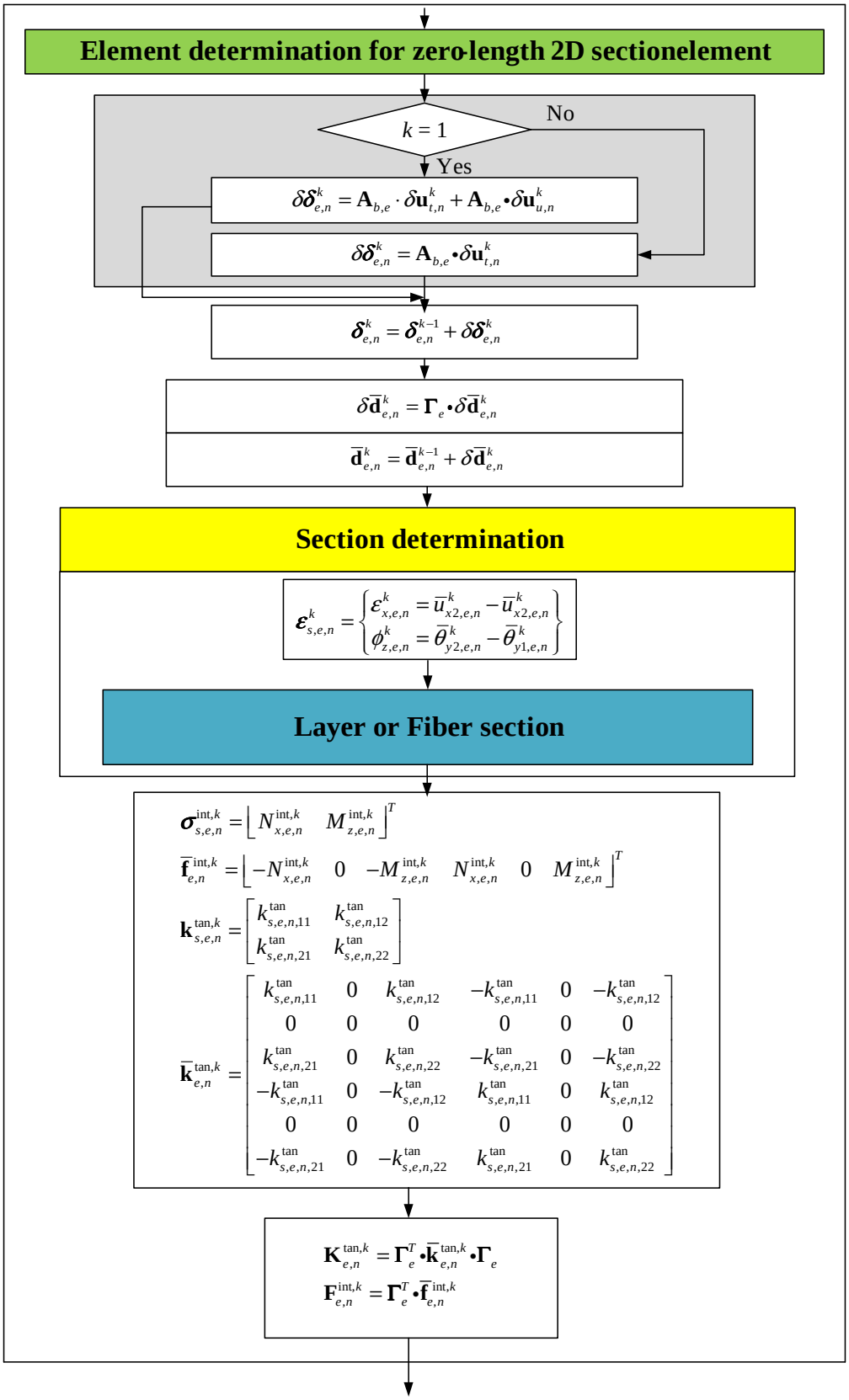


Fig. 2.30 Flow chart of zero-length 2D section element for element state determination

If we have a section with elastic material, then

$$\begin{aligned} \mathbf{k}^{tan}_{s,e,n} &= \mathbf{k}_{s,e} \\ \underbrace{\left\{ \begin{array}{c} N^{int,k}_{x,e,n} \\ M^{int,k}_{z,e,n} \end{array} \right\}}_{\boldsymbol{\sigma}^{int,k}_{s,e,n}} &= \mathbf{k}^{tan}_{s,e,n} \underbrace{\left\{ \begin{array}{c} \varepsilon^{k}_{x,e,n} \\ \phi^{k}_{z,e,n} \end{array} \right\}}_{\boldsymbol{\varepsilon}^{k}_{s,e,n}} \end{aligned}$$

where, $\mathbf{k}^{tan,k}_{s,e,n}$ is the section tangent stiffness matrix at k^{th} iteration.

Step 3: Determine the internal element nodal force vector and the element tangent stiffness matrix from Eq. 2.39.

$$\overline{\mathbf{f}}^{int,k}_{e,n} = \lfloor N^{int,k}_{x,e,n}, 0, M^{int,k}_{z,e,n}, -N^{int,k}_{x,e,n}, 0, -M^{int,k}_{z,e,n} \rfloor^T$$

$$\overline{\mathbf{k}}^{tan,k}_{e,n} = \left[\begin{array}{cccccc} \mathrm{k}^{tan,k}_{s,e,n,11} & 0 & \mathrm{k}^{tan,k}_{s,e,n,12} & -\mathrm{k}^{tan,k}_{s,e,n,11} & 0 & -\mathrm{k}^{tan,k}_{s,e,n,12} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \mathrm{k}^{tan,k}_{s,e,n,21} & 0 & \mathrm{k}^{tan,k}_{s,e,n,22} & -\mathrm{k}^{tan,k}_{s,e,n,21} & 0 & -\mathrm{k}^{tan,k}_{s,e,n,22} \\ -\mathrm{k}^{tan,k}_{s,e,n,11} & 0 & -\mathrm{k}^{tan,k}_{s,12e,n,} & \mathrm{k}^{tan,k}_{s,e,n,11} & 0 & \mathrm{k}^{tan,k}_{s,e,n,12} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -\mathrm{k}^{tan,k}_{s,e,n,21} & 0 & -\mathrm{k}^{tan,k}_{s,e,n,22} & \mathrm{k}^{tan,k}_{s,e,n,21} & 0 & \mathrm{k}^{tan,k}_{s,e,n,22} \end{array} \right]$$

where, $\overline{\mathbf{k}}^{tan,k}_{e,n}$ is the element tangent stiffness matrix in local reference.

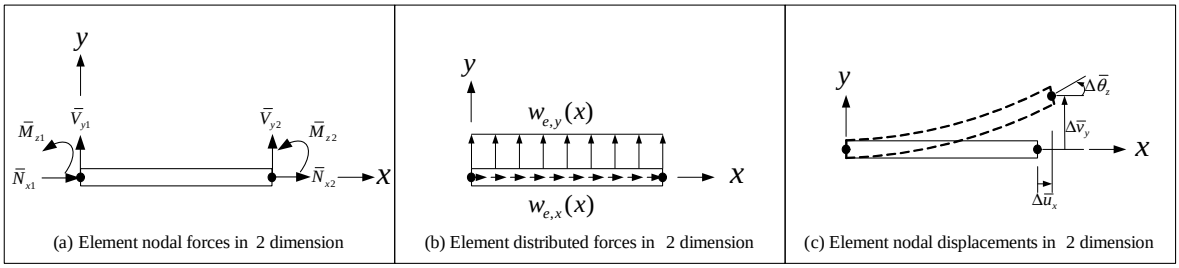
We determine $\mathbf{F}^{int,k}_{e,n}$ and $\mathbf{K}^{tan,k}_{e,n}$.

$$\begin{aligned} \mathbf{F}^{int,k}_{e,n} &= {\boldsymbol{\Gamma}_e}^T \cdot \overline{\mathbf{f}}^{int,k}_{e,n} \\ \mathbf{K}^{tan,k}_{e,n} &= {\boldsymbol{\Gamma}_e}^T \cdot \overline{\mathbf{k}}^{tan,k}_{e,n} \cdot \boldsymbol{\Gamma}_e \end{aligned}$$

2.6 Element Force

Structures should resist applied external forces. Applicable external forces of an element basically are divided into three forces; element nodal forces, element distributed forces and element nodal displacement such as settlements. These external forces are shown in Fig. 2.31.

Element nodal forces shown in Fig. 2.31 (a) are applied directly at the corresponding degrees of freedom. However, so far we have not accounted for the element forces $\mathbf{w}_e(x)$ (as defined in Eq. 2.14) shown in Fig. 2.31 (b). Those can be lumped as concentrated forces at suitably selected arbitrary nodes, and the degrees of freedom at these and the actual joints are treated as the unknowns. For element distributed forces, we should consider the equivalent fixed end action $\overline{\mathbf{f}}^{fea}_e$ of element distributed forces in structural level irrespective of whether we use the stiffness-based or the mixed stiffness-based and flexibility-based method. $\overline{\mathbf{f}}^{fea}_e$ on element distributed force of an element is based on stiffness-based formulation.



(a) Element nodal forces in 2 dimension (b) Element distributed forces in 2 dimension (c) Element nodal displacements in 2 dimension

Fig. 2.31 Forces of an element

$$\begin{aligned}
\mathbf{N}_d(x) &= \begin{bmatrix} 1-\frac{x}{L_e} & 0 & 0 & \frac{x}{L_e} & 0 & 0 \\ 0 & \frac{2x^3}{L_e^3}-\frac{3x^2}{L_e^2}+1 & \frac{x^3}{L_e^2}-\frac{2x^2}{L_e}+x & 0 & \frac{3x^2}{L_e^2}-\frac{2x^3}{L_e^3} & \frac{x^3}{L_e^2}-\frac{x^2}{L_e} \end{bmatrix} \\
\mathbf{w}_e(x) &= \begin{Bmatrix} w_{e,x}(x) \\ w_{e,y}(x) \end{Bmatrix} \\
\overline{\mathbf{f}}_e^{fea} &= \int_o^{L_e} \mathbf{N}_d(x)\cdot\mathbf{w}_e(x)\mathrm{d}x
\end{aligned} \tag{2.40}$$

For a uniformly distributed force $\mathbf{w}_e(x) = \begin{Bmatrix} w_x \\ w_y \end{Bmatrix}$, then Eq. 2.40 reduces to the classical

$$\overline{\mathbf{f}}_e^{fea} = \lfloor \frac{w_x\cdot L_e}{2}, \frac{w_y\cdot L_e}{2}, \frac{w_y\cdot L_e^2}{12}, \frac{w_x\cdot L_e}{2}, \frac{w_y\cdot L_e}{2}, -\frac{w_y\cdot L_e^2}{12} \rfloor^T \tag{2.41}$$

which are shown in Fig. 2.32.

Element nodal displacements considered as external forces are shown in Fig. 2.31 (c). Those are applied as constraint degrees of freedom.

Let us consider the equilibrium equation considered by element nodal forces, element distributed forces, and element nodal displacements in an element shown in Fig. 2.33.

If the element has elastic section, we obtain equilibrium equation, Eq. 2.42.

$$\underbrace{\begin{Bmatrix} \overline{N}_{x1}^{?} \\ \overline{V}_{y1}^{?} \\ \overline{M}_{z1}^{\checkmark} \\ \overline{N}_{x2}^{\checkmark} \\ \overline{V}_{y2}^{?} \\ \overline{M}_{z2}^{\checkmark} \end{Bmatrix}}_{\overline{\mathbf{f}}_e} = \underbrace{\begin{bmatrix} \frac{EA}{L_e} & 0 & 0 & -\frac{EA}{L_e} & 0 & 0 \\ 0 & \frac{12EI_z}{L_e^3} & \frac{6EI_z}{L_e^2} & 0 & -\frac{12EI_z}{L_e^3} & \frac{6EI_z}{L_e^2} \\ 0 & \frac{6EI_z}{L_e^2} & \frac{4EI_z}{L_e} & 0 & -\frac{6EI_z}{L_e^2} & \frac{2EI_z}{L_e} \\ -\frac{EA}{L_e} & 0 & 0 & \frac{EA}{L_e} & 0 & 0 \\ 0 & -\frac{12EI_z}{L_e^3} & -\frac{6EI_z}{L_e^2} & 0 & \frac{12EI_z}{L_e^3} & -\frac{6EI_z}{L_e^2} \\ 0 & \frac{6EI_z}{L_e^2} & \frac{2EI_z}{L_e} & 0 & -\frac{6EI_z}{L_e^2} & \frac{4EI_z}{L_e} \end{bmatrix}}_{\overline{\mathbf{k}}_e} \cdot \underbrace{\begin{Bmatrix} 0^{\checkmark} \\ 0^{\checkmark} \\ \overline{\theta}_{z1}^{?} \\ \overline{u}_{x2}^{?} \\ \Delta\overline{v}_y^{\checkmark} \\ \overline{\theta}_{z2}^{?} \end{Bmatrix}}_{\overline{\mathbf{d}}_e} - \underbrace{\begin{Bmatrix} \frac{w_x\cdot L_e}{2} \\ \frac{w_y\cdot L_e}{2} \\ \frac{w_y\cdot L_e^2}{12} \\ \frac{w_x\cdot L_e}{2} \\ \frac{w_y\cdot L_e}{2} \\ -\frac{w_y\cdot L_e^2}{12} \end{Bmatrix}}_{\overline{\mathbf{f}}_e^{fea}} \tag{2.42}$$

where, superscript ? indicates unknown forces and displacements, and ✓ known forces and displacements.

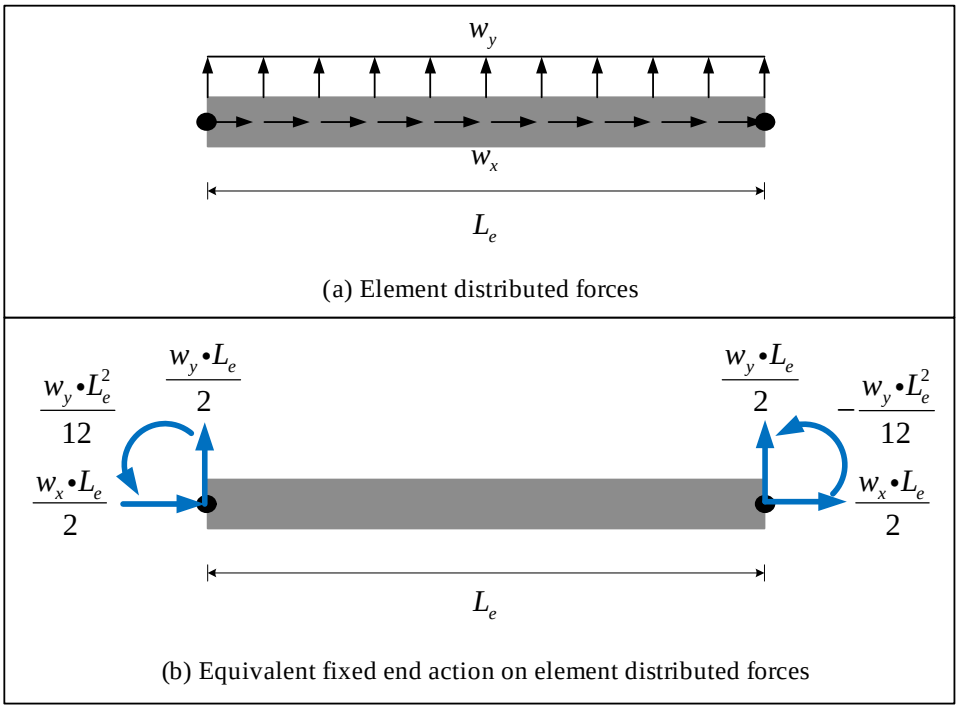


Fig. 2.32 Fixed end action of an element

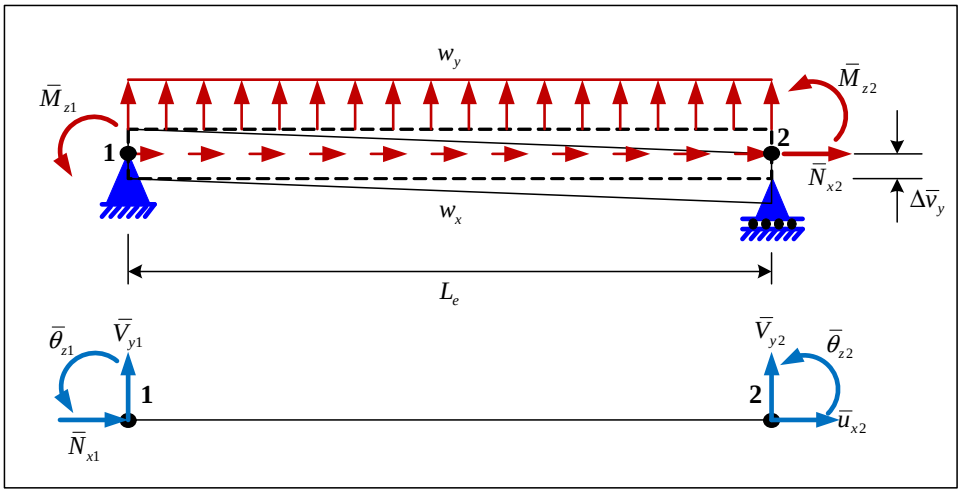


Fig. 2.33 Example for equivalent fixed end action

Table 2.1 Comparison on maximum excitation force of Y direction at point B (F: Flexibility-based with iterations, FNI: Flexibility-based without iterations, NR: Newton Raphson, IS: Initial stiffness)

	Experiment	F, NR	F, IS	FNI, NR	FNI, IS	Castem
Maximum force	235,000	222,232.7	221,958.2	234,852.6	239,021.9	219,469
Error on maximum force	-	5.43 %	5.55 %	0.06%	1.71%	6.61 %

2.7 Comparion

Comparing flexiblity-based beam column with element iterations and without element iteation in Sec. 2.2.2, here are examples.

2.7.1 Example 1

Example 1 in Fig. 2.34 has layer sections and elastic sections.

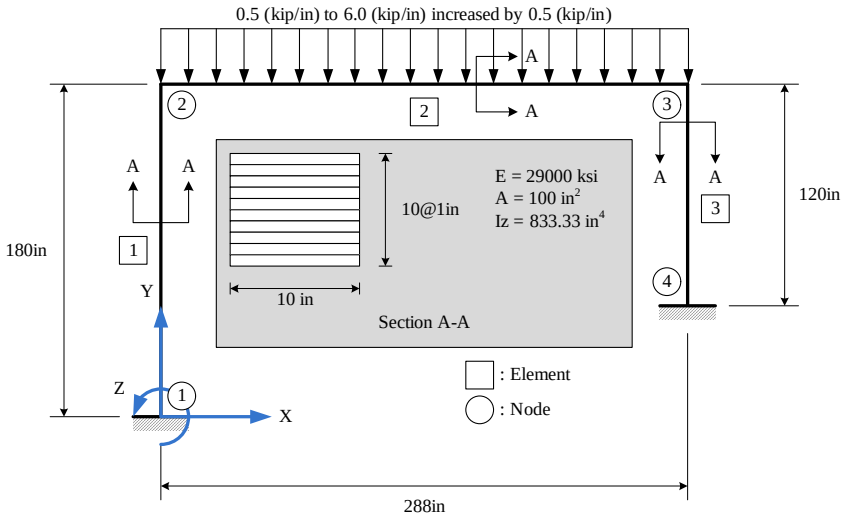


Fig. 2.34 Example 1: Frame with layer sections and elastic sections

2.7.2 Example 2

Example 2 in Fig. 2.37 has layer sections and anisotropic 1D damage material for concrete and hardening material for steel.

Table. 2.1 describes the error on maximum excitation force of Y direction at point B.

2.7.3 Example 3

Example 3 in Fig. 2.39 and 2.39 has layer sections and modified Kent-Park material for concrete and modified Giuffre-Menegotto-Pinto material for steel.

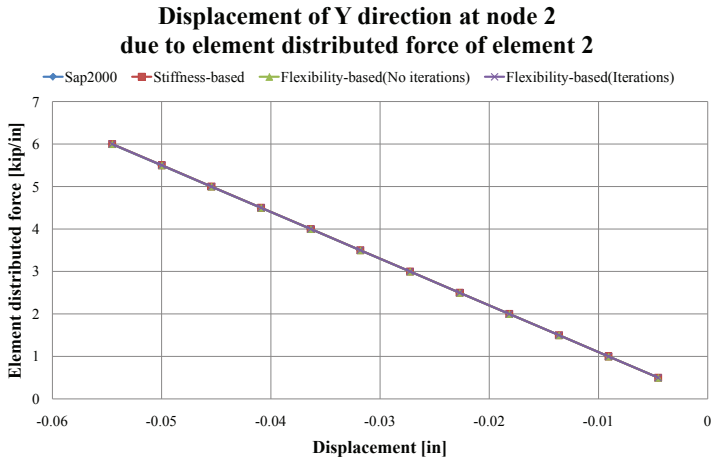


Fig. 2.35 Example 1: Displacement of Y direction at node 2 due to element distributed force of element 2

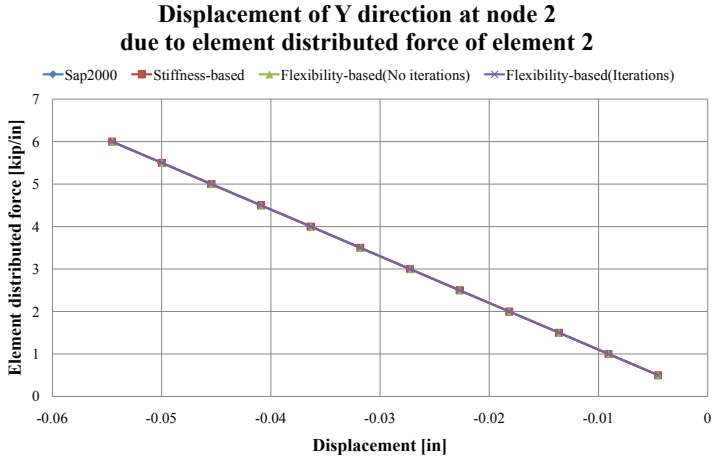
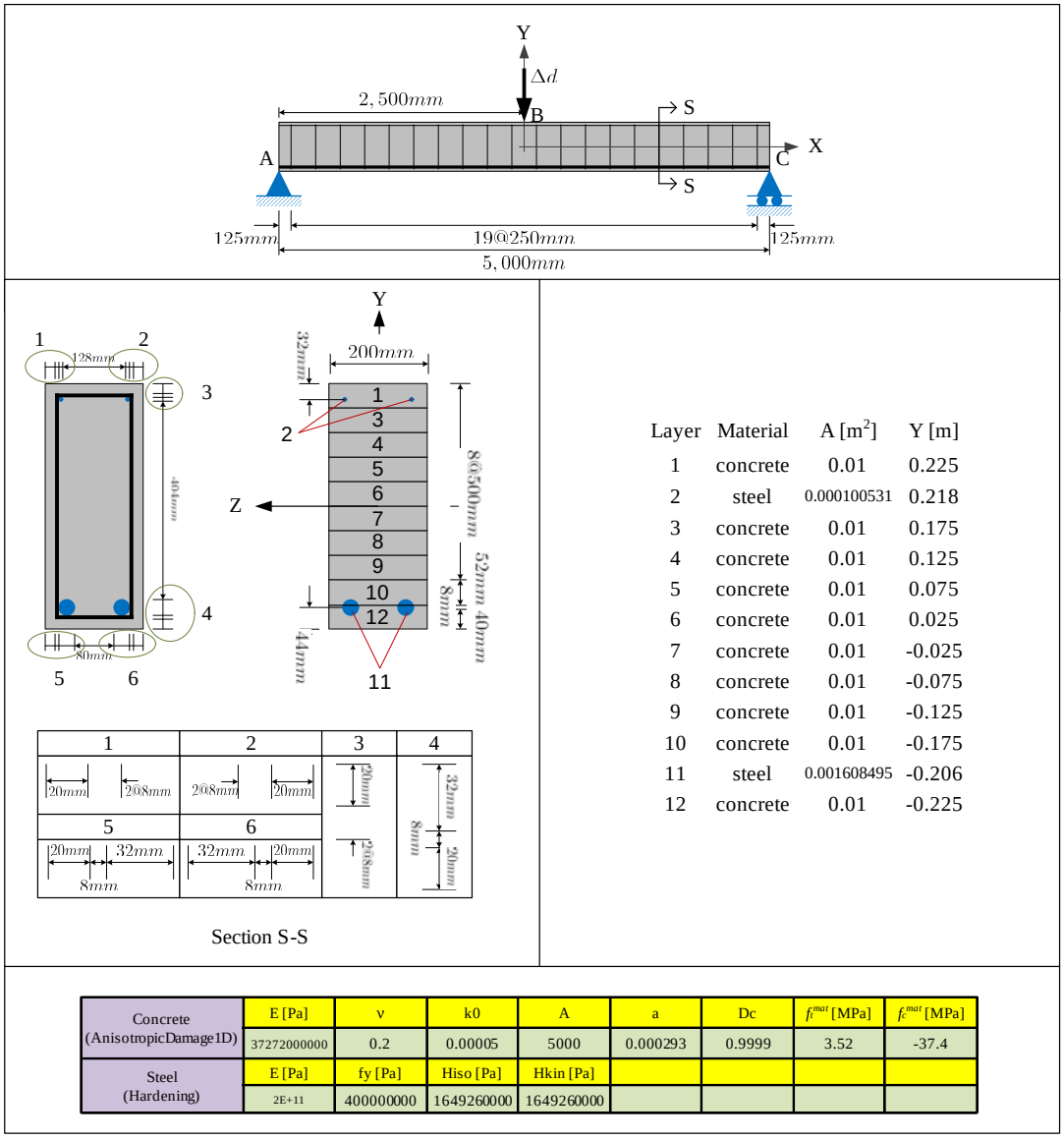


Fig. 2.36 Example 1: Reaction of Z direction at node 1 due to element distributed force of element 2



Layer	Material	A [m^2]	Y [m]
1	concrete	0.01	0.225
2	steel	0.000100531	0.218
3	concrete	0.01	0.175
4	concrete	0.01	0.125
5	concrete	0.01	0.075
6	concrete	0.01	0.025
7	concrete	0.01	-0.025
8	concrete	0.01	-0.075
9	concrete	0.01	-0.125
10	concrete	0.01	-0.175
11	steel	0.001608495	-0.206
12	concrete	0.01	-0.225

Concrete (AnisotropicDamage1D)	E [Pa]	ν	k0	A	a	Dc	f_t^{mat} [MPa]	f_c^{mat} [MPa]
	37272000000	0.2	0.00005	5000	0.000293	0.9999	3.52	-37.4
Steel (Hardening)	E [Pa]	fy [Pa]	Hiso [Pa]	Hkin [Pa]				
	2E+11	400000000	1649260000	1649260000				

Fig. 2.37 Example 2: Beam with layer sections, and anisotropic 1D damage and hardening material

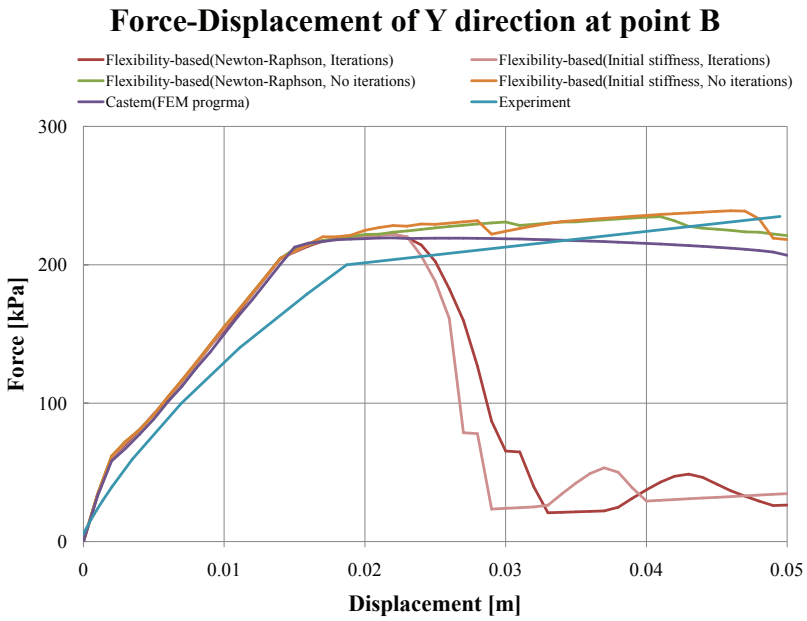
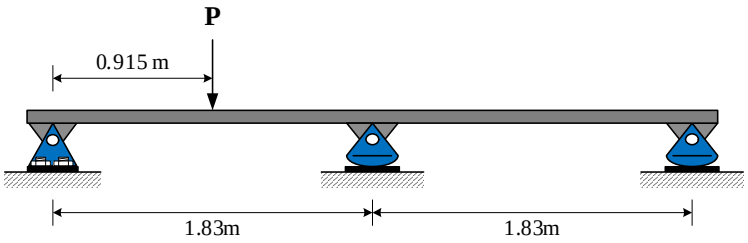
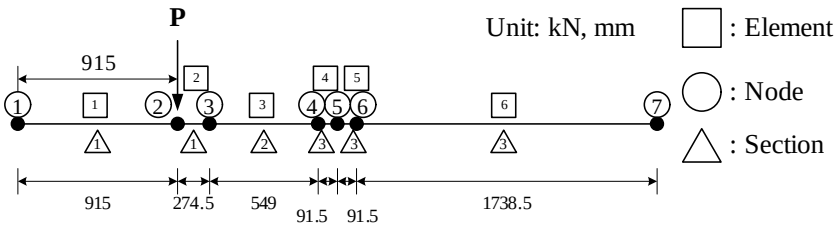
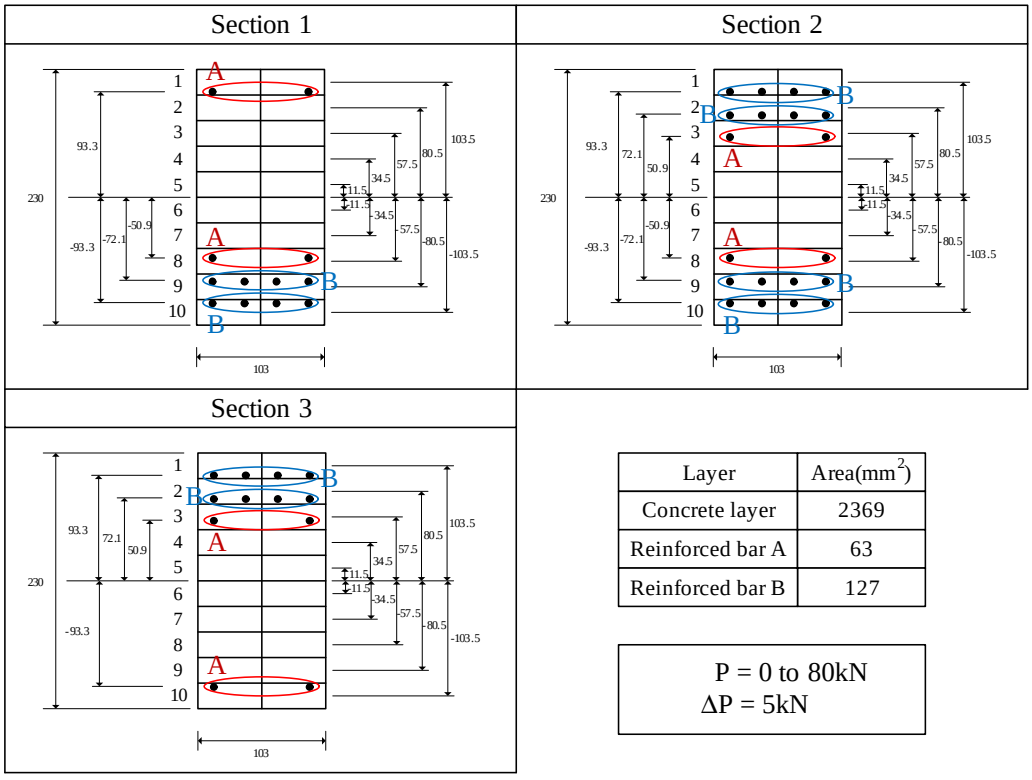


Fig. 2.38 Example 2: Force-displacement curve of Y direction at point B





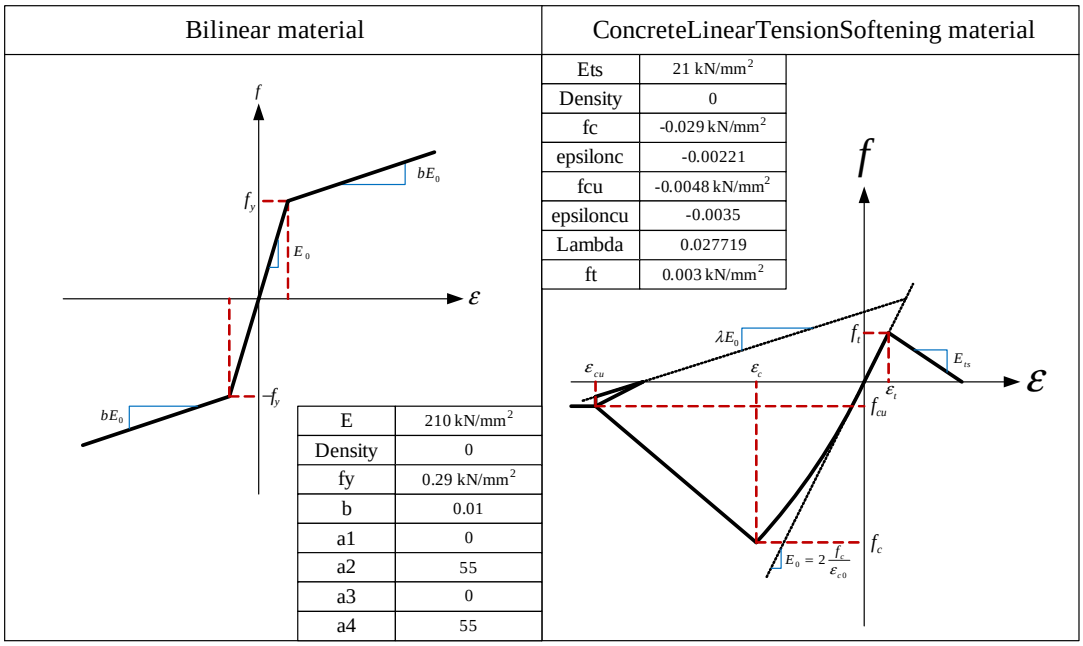
Element	inode	jnode	Section	# of integraion points
1	1	2	1	5
2	2	3	1	3
3	3	4	2	3
4	4	5	3	3
5	5	6	3	3
6	6	7	3	9



Layer	Area(mm^2)
Concrete layer	2369
Reinforced bar A	63
Reinforced bar B	127

P = 0 to 80kN
ΔP = 5kN

Fig. 2.39 Example 3: Beam with layer sections, and modified Kent-Park and modified Giuffre-Menegotto-Pinto material



E	210 kN/mm^2
Density	0
fy	0.29 kN/mm^2
b	0.01
a1	0
a2	55
a3	0
a4	55

Ets	21 kN/mm^2
Density	0
fc	-0.029 kN/mm^2
epsilonc	-0.00221
fcu	-0.0048 kN/mm^2
epsiloncu	-0.0035
Lambda	0.027719
ft	0.003 kN/mm^2

Fig. 2.40 Example 3: Material properties on modified Kent-Park and modified Giuffre-Menegotto-Pinto material

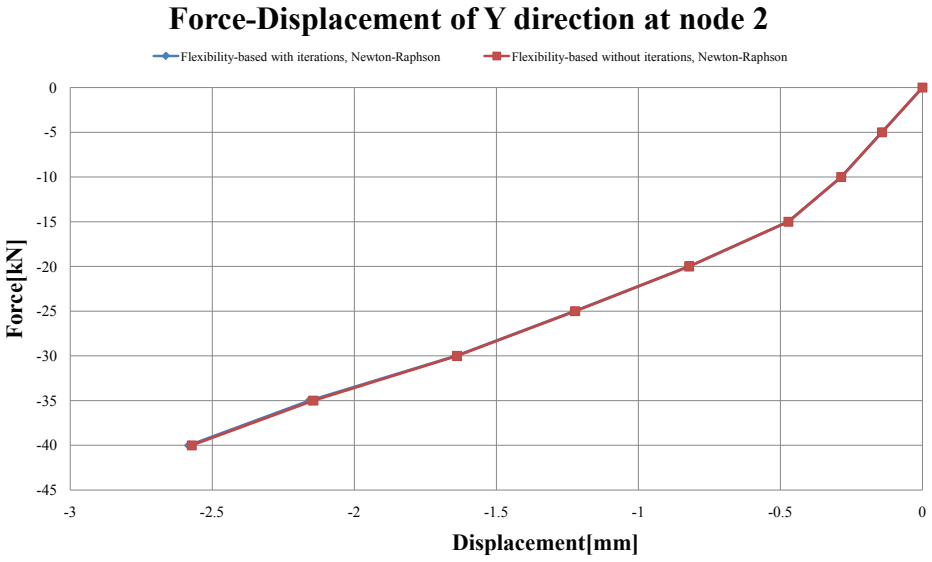


Fig. 2.41 Example 3: Force-displacement curve of Y direction at node 2

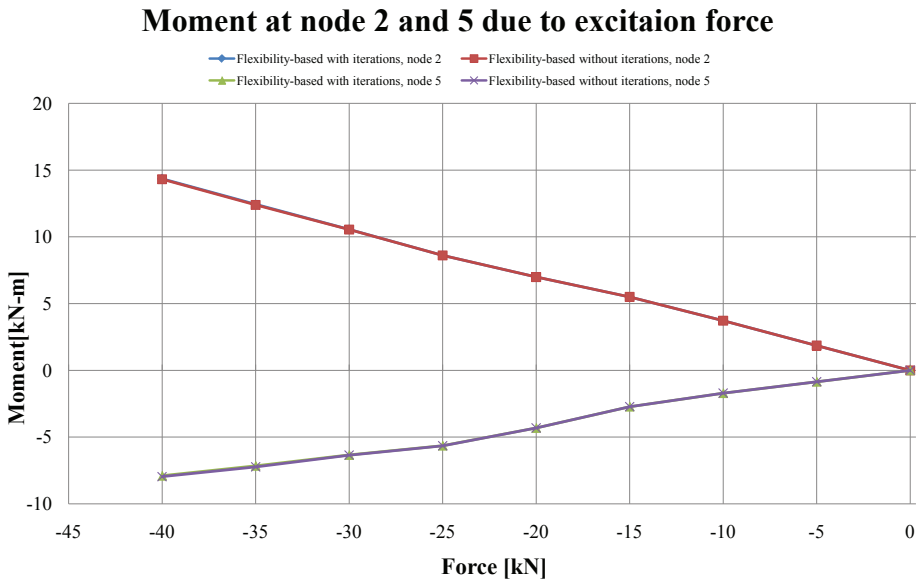


Fig. 2.42 Example 3: Moment at node 2 and 5 due to exciation force

Chapter 3

CONSTITUTIVE MODELS

This chapter will describe the various constitutive models used in Mercury. All of them are limited to uniaxial stress-strain relations as they will be used in conjunction with the uniaxial elements described in the previous chapter. Most of these models are well established yet some of them being poorly documented in recent literature, this chapter will clearly describe them. Hence particular attention is given to a rigorous description and validation problems will be presented in chapter **??**. One relatively new model for concrete is presented, it is an anisotropic damage mechanics model particularly well suited for real time hybrid simulation.

3.1 Steel Models

This section describes various steel models shown in Table. 3.1.

3.1.1 Classical plasticity

Most metallic materials, when subjected to high stress level, exhibit plasticity behavior, i.e. when force is removed, the body does not return to its original shape, but has some permanent plastic deformation associated with it.

Materials such as steel or concrete, commonly used in structures, show yielding and plastic deformation. A typical uniaxial stress-strain curve for steel is shown in Fig. 3.1. Following a linear response, steel exhibits an abrupt change in stiffness which occurs at the yield point. Beyond yielding, a permanent deformation is introduced upon unloading. This behavior can be idealized by a bilinear stress-strain relationship with two slopes, the second being called the tangent modulus E^{tan}. After reaching the yield point, the slope could be smaller, equal, or greater than zero as shown in Fig. 3.2(a).

Fig. 3.2(b) illustrates on elastic-perfectly plastic situation, that is after reaching the yield stress, the material starts flowing plastically without any further increase or decrease. The specimen is first loaded to point A, reaches its yield stress, and then flows plastically at point B, we unload the specimen until we reach a zero-load condition at point C. However, since we have loaded the specimen beyond the yield point, we observe that a permanent deformation or a permanent strain has been introduced in the material. We denote this permanent strain in one dimension the plastic strain ε^p which is permanent deformation.

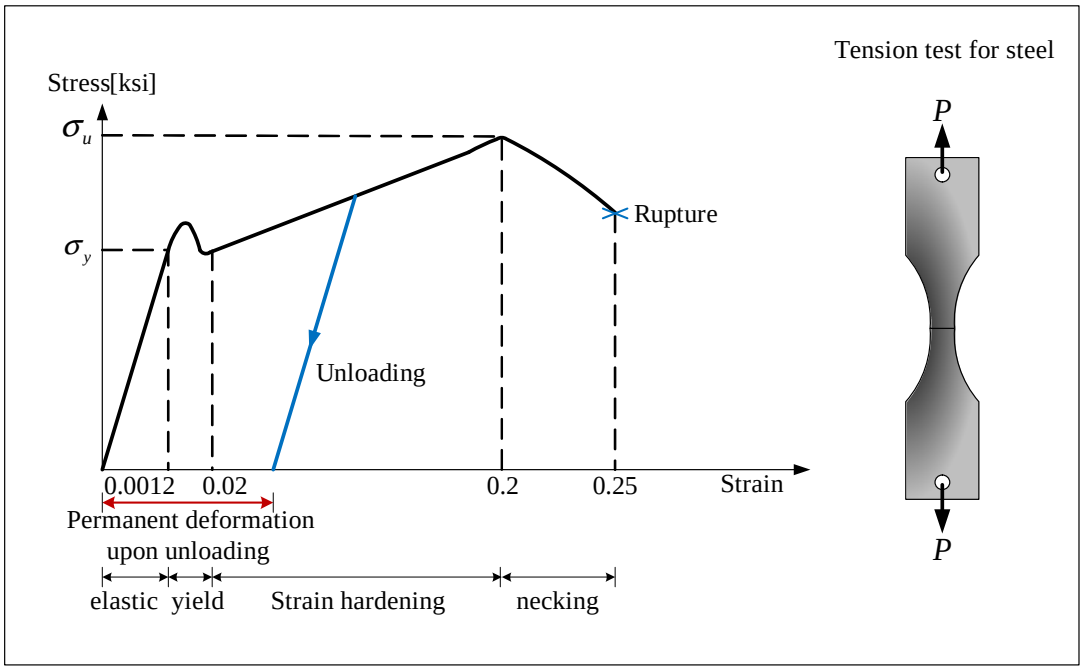


Fig. 3.1 Uniaxial stress-strain curve for steel

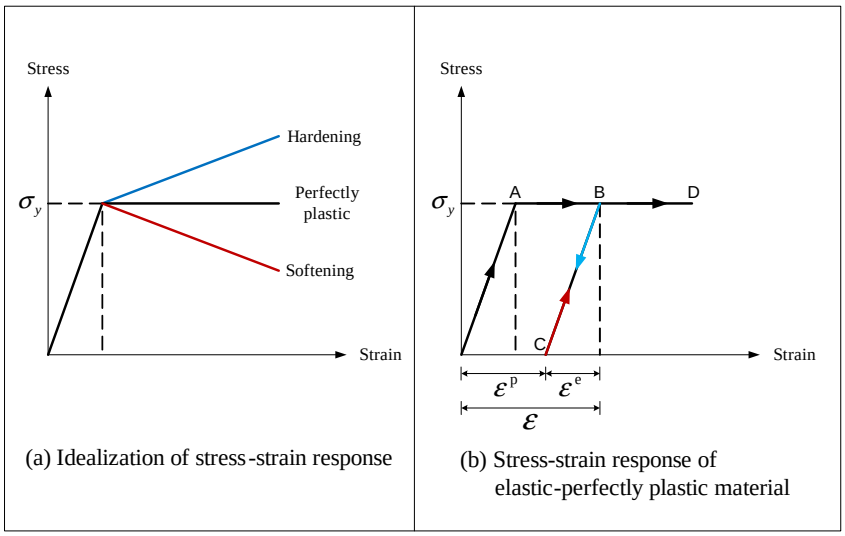


Fig. 3.2 Idealized stress-strain response hardening material

Table 3.1 Steel models

Steel Models	Hardening		Heuristic
	Isotropic	Kinematic	
Classical Plasticity	O	O	
Bilinear	O	X	
GMP	O	O	O

where GMP is Giuffre-Menegotto-Pinto model modified by Fillipou et al.

The total strain ε can thus be decomposed into an elastic strain ε^e and a plastic one.

$$\begin{aligned} \varepsilon &= \varepsilon^e + \varepsilon^p \\ \sigma &= E \cdot \varepsilon^e \\ \sigma &= E \cdot (\varepsilon - \varepsilon^p) \end{aligned} \tag{3.1}$$

Eq. 3.1 describes the stress-strain relation with plastic behavior for the elastic-perfectly plastic uniaxial loading conditions.

Next we will present the mathematical structure of two classical phenomenological isotropic hardening model (Regueiro 2007).

3.1.1.1 Isotropic hardening model

Formulation of this constitutive law is based on classical theory of plasticity and is thus based on the following considerations:

Helmholtz free energy The general format of associated dissipative models starts from the Helmholtz free energy expansion and the dissipation inequality. Expanding the free energy into two terms, an elastic and a plastic one, we have

$$\rho \cdot \phi(\varepsilon^e, \xi) =: \underbrace{\frac{1}{2}\varepsilon^e \cdot E \cdot \varepsilon^e}_{\text{Elastic}} + \underbrace{\frac{1}{2} \cdot \xi \cdot H \cdot \xi}_{\text{Plastic}}$$

where ρ is density, $\phi(\varepsilon^e, \xi)$ the Helmholtz free energy function in terms of elastic strain ε^e and a strain-like internal variable ξ, E elastic modulus, and H plastic modulus.

Fig. 3.3 describe the elastic modulus E and plastic modulus H where the tangent modulus is

$$E^{tan} = \frac{E \cdot H}{E + H}$$

The elastic and plastic stress, σ and q respectively, are give by Eq. 3.2 and Eq. 3.3.

$$\text{Linear elasticity} \quad : \sigma \quad = \rho \cdot \frac{\partial \phi}{\partial \varepsilon^e} = E \cdot \varepsilon^e \tag{3.2}$$

$$\text{Plasticity} \quad : \text{q} \quad = \rho \cdot \frac{\partial \phi}{\partial \xi} = H \cdot \xi \tag{3.3}$$

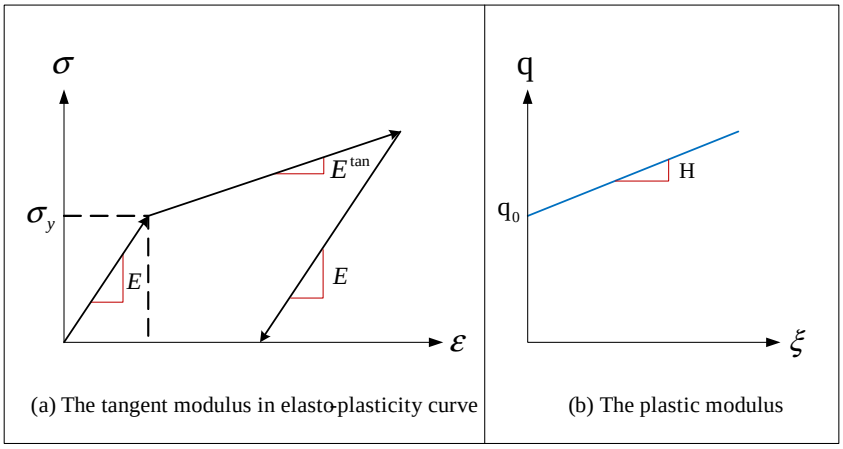


Fig. 3.3 The tangent modulus for isotropic hardening model

where, q is identical to the initial yield stress σ_y at the initial state.

Assuming E and H constant, we can rewrite Eq. 3.2 and Eq. 3.3 in rate form as

$$\dot{\sigma} = E \cdot \dot{\varepsilon}^e = E \cdot (\dot{\varepsilon} - \dot{\varepsilon}^p) \tag{3.4}$$

$$\dot{\mathrm{q}} = H \cdot \dot{\xi} \tag{3.5}$$

Yield function The yield function is defined as

$$f(\sigma, \kappa) := |\sigma| - \mathrm{q}$$

Thus if $f(\sigma,\ \mathrm{q}) < 0$, then the stress is in the elastic domain. Alternatively, if $f(\sigma,\ \mathrm{q}) = 0$, the stress reaches its plastic limit.

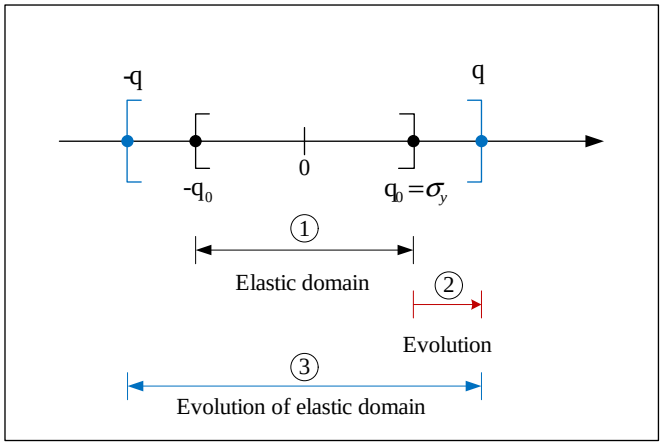


Fig. 3.4 The evolution of elastic domain in isotropic hardening model

Fig. 3.4 shows the evolution of the elastic domain. First the strain reaches yielding

($q_0 = \sigma_y$), and then at that point further increase in strain results in an expansion of q this will in turn expand the elastic domain.

Evolution equation We define plastic a flow rule by

$$\dot{\varepsilon}^p := \dot{\gamma} \cdot \frac{\partial g}{\partial \sigma} \tag{3.6}$$

where, $\dot{\gamma}$ is plastic multiplier or consistency parameter, and g is the plastic potential function in terms of σ and q.

Assuming $g(\sigma,\ \mathrm{q}) = f(\sigma, then\ \mathrm{q})$, $\frac{\partial g}{\partial \sigma} = \mathrm{sign}(\sigma)$ where $\mathrm{sign}(\sigma)$ is 1 if $\sigma \geq 0$ or -1 if $\sigma < 0$. Then, we rewrite Eq. 3.6 as

$$\dot{\varepsilon}^p = \dot{\gamma} \cdot \mathrm{sign}(\sigma)$$

or

$$\dot{\gamma} = |\dot{\varepsilon}^p| \geq 0$$

Next, we consider the evolution of the strain-like internal state variable associated with plasticity.

$$\dot{\xi} := \dot{\gamma} \cdot h(\sigma,\ \mathrm{q})$$

where, $h(\sigma,\ \mathrm{q})$ is a dimensionless hardening function. For simplicity and under the assumption of elasto-plastic hardening material with bilinear curve, we assume that

$$h(\sigma,\ \mathrm{q}) = -\frac{\partial g}{\partial \mathrm{q}} = 1$$

Therefore, we can rewrite Eq. 3.4 and Eq. 3.5 as follows

$$\begin{aligned} \dot{\sigma} &= E \cdot (\dot{\varepsilon} - \dot{\gamma} \cdot \frac{\partial g}{\partial \sigma}) \\ &= E \cdot (\dot{\varepsilon} - \dot{\gamma} \cdot \mathrm{sign}(\sigma)) \end{aligned} \tag{3.7}$$

and

$$\begin{aligned} \dot{\mathrm{q}} &= H \cdot \dot{\gamma} \cdot h(\sigma,\ \mathrm{q}) \\ &= H \cdot \dot{\gamma} \end{aligned} \tag{3.8}$$

Kuhn-Tucker conditions and consistency condition In the elastic regime, the yield function f must remain negative and the plastic multiplier is zero. On the other hand, during plastic flow the yield function f must be zero while plastic multiplier is positive. This is succinctly expressed by the so-called Kuhn-Tucker condition,

$$\text{Elastic loading and unloading} \quad : f < 0, \quad \dot{\gamma} = 0 \tag{3.9}$$

$$\text{Plastic loading} \quad : f = 0, \quad \dot{\gamma} > 0 \tag{3.10}$$

Eq. 3.9 and Eq. 3.10 can be put together.

$$f \leq 0,\ \dot{\gamma} \geq 0,\ f \cdot \dot{\gamma} = 0 \tag{3.11}$$

The consistency condition precludes us from going beyond the yield surface as we must stay on it, thus

$$\begin{aligned} \dot{f}(\sigma,\ \mathrm{q}) &= \frac{\partial f}{\partial \sigma} \cdot \frac{\partial \sigma}{\partial t} + \frac{\partial f}{\partial \mathrm{q}} \cdot \frac{\partial \mathrm{q}}{\partial t} \\ &= \frac{\partial f}{\partial \sigma} \cdot \dot{\sigma} + \frac{\partial f}{\partial \mathrm{q}} \cdot \dot{\mathrm{q}} = 0 \end{aligned} \tag{3.12}$$

Finally, we need to evaluate tangent modulus E^{tan}. Substituting Eq. 3.7 and Eq. 3.8 into Eq. 3.12, we obtain

$$\frac{\partial f}{\partial \sigma} \cdot E \cdot \left(\dot{\varepsilon} - \dot{\gamma} \cdot \frac{\partial g}{\partial \sigma} \right) + \frac{\partial f}{\partial \mathrm{q}} \cdot H \cdot \dot{\gamma} \cdot h(\sigma,\ \mathrm{q}) = 0$$

Therefore,

$$\dot{\gamma} = \frac{\frac{\partial f}{\partial \sigma} \cdot E \cdot \dot{\varepsilon}}{\frac{\partial f}{\partial \sigma} \cdot E \cdot \frac{\partial g}{\partial \sigma} - \frac{\partial f}{\partial \mathrm{q}} \cdot H \cdot h(\sigma,\ \mathrm{q})} \tag{3.13}$$

Substituting Eq. 3.13 into Eq. 3.7, we obtain an explicit expression for the incremental stress,

$$\begin{aligned} \dot{\sigma} &= E \cdot \left(\dot{\varepsilon} - \dot{\gamma} \cdot \frac{\partial g}{\partial \sigma} \right) \\ &= \left(E - \frac{\frac{\partial f}{\partial \sigma} \cdot E^2 \cdot \frac{\partial g}{\partial \sigma}}{\frac{\partial f}{\partial \sigma} \cdot E \cdot \frac{\partial g}{\partial \sigma} - \frac{\partial f}{\partial \mathrm{q}} \cdot H \cdot h(\sigma,\ \mathrm{q})} \right) \cdot \dot{\varepsilon} \\ &= E^{tan} \cdot \dot{\varepsilon} \end{aligned}$$

For elasto-plastic hardening material with bilinear curve in one dimension,

$$\dot{\gamma} = \frac{\mathrm{sign}(\sigma) \cdot E \cdot \dot{\varepsilon}}{E + H}$$

the tangent modulus reduces to

$$\begin{aligned} E^{tan} &= E - \frac{\mathrm{sign}(\sigma) \cdot E^2 \cdot \mathrm{sign}(\sigma)}{\mathrm{sign}(\sigma) \cdot E \cdot \mathrm{sign}(\sigma) + (-1) \cdot H \cdot (1)} \\ &= E - \frac{E^2}{E + H} \\ &= \frac{E \cdot H}{E + H} \end{aligned}$$

3.1.1.2 Combined isotropic and kinematic hardening model

We describe isotropic and kinematic hardening model. Finally, we summarize the major equations governing plasticity.

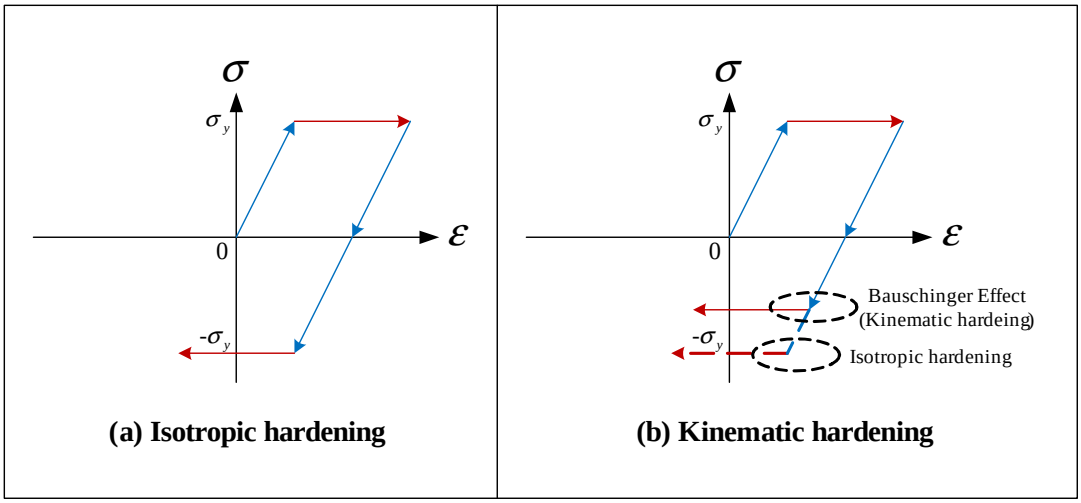


Fig. 3.5 Isotropic and kinematic hardening plasticity

Yield surface of isotopic hardening plasticity is symmetric about the origin at $\sigma = 0$, Fig. 3.5(a), whereas for kinematic hardening plasticity the yield surface is unsymmetric, Fig. 3.5(b). This lack of symmetry is caused by the classical Bauschinger effect. Hence, we will thus enrich the helmholtz free energy function with strain-like internal state variables for combined isotropic and kinematic hardening.

$$\rho \cdot \phi(\varepsilon^e, \boldsymbol{\xi}) = \underbrace{\frac{1}{2} \cdot \varepsilon^e \cdot E \cdot \varepsilon^e}_{\text{Elastic}} + \underbrace{\frac{1}{2} \cdot \boldsymbol{\xi}^T \cdot \mathbf{H} \cdot \boldsymbol{\xi}}_{\text{Plastic}}$$

where, $\boldsymbol{\xi}$ is a strain-like vector of internal state variables, and $\mathbf{H}$ is the hardening matrix.

$$\boldsymbol{\xi} = \lfloor \xi^{iso},\ \xi^{kin} \rfloor^T$$
$$\mathbf{H} = \begin{bmatrix} H^{iso} & 0 \\ 0 & H^{kin} \end{bmatrix}$$

where, ξ^{iso} is a strain-like internal state variable of isotropic plasticity, ξ^{kin} a strain-like internal state variable of kinematic plasticity, H^{iso} the isotropic plastic modulus, and H^{kin} the kinematic plastic modulus. The thermodynamically stress-like vector of internal state variables is thus given by

$$\begin{aligned} \mathbf{q}^{\xi} &= \mathbf{H} \cdot \boldsymbol{\xi} \\ &= \lfloor H^{iso} \cdot \xi^{iso},\ H^{kin} \cdot \xi^{kin} \rfloor^T \\ &= \lfloor \mathrm{q}^{iso},\ \mathrm{q}^{kin} \rfloor^T \end{aligned}$$

where, q^{iso} is an isotropic stress-like internal state variable and q^{kin} a kinematic stress-like internal state variable. Therefore, the rate form of stress-like vector of internal state variables becomes

$$\begin{aligned}\dot{\mathbf{q}}^{\xi} &= \mathbf{H} \cdot \dot{\boldsymbol{\xi}} \\ &= \lfloor H^{iso} \cdot \dot{\xi}^{iso},\ H^{kin} \cdot \dot{\xi}^{kin} \rfloor^T \\ &= \lfloor \dot{\mathrm{q}}^{iso},\ \dot{\mathrm{q}}^{kin} \rfloor^T\end{aligned} \tag{3.14}$$

Having stress-like internal state variable q^{iso}, we redefine the yield function as

$$f(\sigma, \mathbf{q}^{\xi}) := |\sigma - \mathrm{q}^{kin}| - \mathrm{q}^{iso}$$

As mentioned above, if $f(\sigma,\ \mathbf{q}^{\xi}) < 0$, the stress is in the elastic domain, otherwise, if $f(\sigma,\ \mathbf{q}^{\xi}) = 0$, the stress reaches plasticity. In one dimension, we can assume $g(\sigma,\ \mathrm{q}^{iso})$ to be the same as $f(\sigma,\ \mathrm{q}^{iso})$.

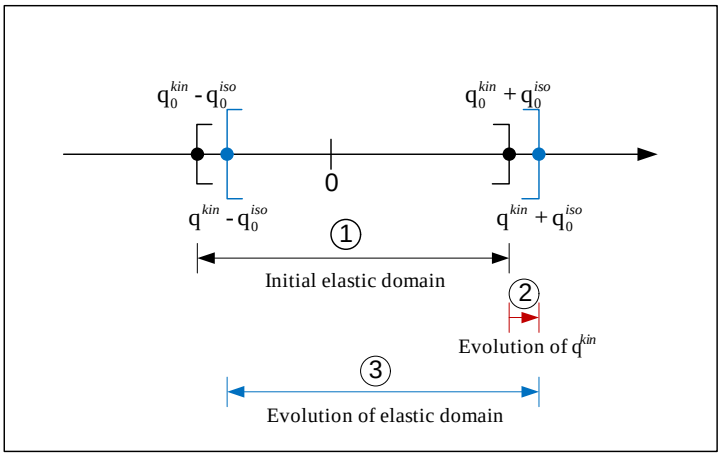


Fig. 3.6 The evolution of elastic domain in isotropic and kinematic hardening model for $H^{iso} = 0$ (isotropic perfectly plastic) and $H^{kin} > 0$

Fig. 3.6 depicts the evolution of the elastic domain for $H^{iso} = 0$ (isotropic perfectly plastic) and $H^{kin} > 0$. The size of elastic domains is the same after evolution.

Finally, we will define the evolution of $\dot{\boldsymbol{\xi}}$ as

$$\dot{\boldsymbol{\xi}} := \dot{\gamma} \cdot \mathbf{h}(\sigma,\ \mathbf{q}^{\xi}) \tag{3.15}$$

where, $\mathbf{h}(\sigma,\ \mathbf{q}^{\xi})$ is a dimensionless hardening function.

$$\begin{aligned}\mathbf{h}(\sigma,\ \mathbf{q}^{\xi}) &= -\frac{\partial g}{\partial \mathbf{q}^{\xi}} \\ &= \lfloor -\frac{\partial g}{\partial \mathrm{q}^{iso}},\ -\frac{\partial g}{\partial \mathrm{q}^{kin}} \rfloor^{T} \\ &= \lfloor 1,\ \mathrm{sign}(\sigma - \mathrm{q}^{kin}) \rfloor^{T}\end{aligned}$$

Combining Eq. 3.14 and Eq. 3.15,

$$\dot{\mathbf{q}}^{\xi} = \mathbf{H} \cdot \dot{\boldsymbol{\xi}} = \mathbf{H} \cdot \dot{\gamma} \cdot \mathbf{h}(\sigma,\ \mathbf{q}^{\xi}) \tag{3.16}$$

To determine E^{tan}, we consider the consistency condition given by Eq. 3.12 and where $\dot{\mathrm{q}}$ has now been replaced by $\dot{\mathbf{q}}^{\xi}$

$$\dot{f}(\sigma,\ \mathbf{q}^{\xi}) = \frac{\partial f}{\partial \sigma} \cdot \dot{\sigma} + \frac{\partial f}{\partial \mathbf{q}^{\xi}} \cdot \dot{\mathbf{q}}^{\xi} = 0 \tag{3.17}$$

Substituting Eq. 3.7 and Eq. 3.16 into Eq. 3.17,

$$\frac{\partial f}{\partial \sigma} \cdot E \cdot \left(\dot{\varepsilon} - \dot{\gamma} \cdot \frac{\partial g}{\partial \sigma} \right) \ + \ \frac{\partial f}{\partial \mathbf{q}^{\xi}} \cdot \mathbf{H} \cdot \dot{\gamma} \cdot \mathbf{h}(\sigma,\ \mathbf{q}^{\xi}) = 0 \tag{3.18}$$

$\dot{\gamma}$ is determined from Eq. 3.18

$$\dot{\gamma} = \frac{\frac{\partial f}{\partial \sigma} \cdot E \cdot \dot{\varepsilon}}{\frac{\partial f}{\partial \sigma} \cdot E \cdot \frac{\partial g}{\partial \sigma} - \frac{\partial f}{\partial \mathbf{q}^{\xi}} \cdot \mathbf{H} \cdot \mathbf{h}(\sigma,\ \mathbf{q}^{\xi})} \tag{3.19}$$

From Eq. 3.7,

$$\begin{aligned}\dot{\sigma} &= E \cdot \left(\dot{\varepsilon} - \dot{\gamma} \cdot \frac{\partial g}{\partial \sigma} \right) \\ &= \left(E - \frac{\frac{\partial f}{\partial \sigma} \cdot E^{2} \cdot \frac{\partial g}{\partial \sigma}}{\frac{\partial f}{\partial \sigma} \cdot E \cdot \frac{\partial g}{\partial \sigma} - \frac{\partial f}{\partial \mathbf{q}^{\xi}} \cdot \mathbf{H} \cdot \mathbf{h}(\sigma, \mathbf{q}^{\xi})} \right) \cdot \dot{\varepsilon} \\ &= E^{tan} \cdot \dot{\varepsilon}\end{aligned}$$

Thus, for elasto-plastic hardening material with bilinear curve, Eq. 3.19 reduces to:

$$\dot{\gamma} = \frac{\mathrm{sign}(\sigma - \mathrm{q}^{kin}) \cdot E \cdot \dot{\varepsilon}}{E + H^{iso} + H^{kin}}$$

Finally, we determine the tangent modulus, E^{tan},

$$\begin{aligned} E^{tan} &= E - \frac{\frac{\partial f}{\partial \sigma} \cdot E^2 \cdot \frac{\partial g}{\partial \sigma}}{\frac{\partial f}{\partial \sigma} \cdot E \cdot \frac{\partial g}{\partial \sigma} - \frac{\partial f}{\partial \mathbf{q}^{\xi}} \cdot \mathbf{H} \cdot \mathbf{h}(\sigma,\ \mathbf{q}^{\xi})} \\ &= E - \frac{\text{sign}(\sigma - \text{q}^{kin})^2 \cdot E^2}{\text{sign}(\sigma - \text{q}^{kin})^2 \cdot E - \lfloor -1,\ -\text{sign}(\sigma - \text{q}^{kin}) \rfloor \begin{bmatrix} H^{iso} & 0 \\ 0 & H^{kin} \end{bmatrix} \begin{Bmatrix} 1 \\ \text{sign}(\sigma - \text{q}^{kin}) \end{Bmatrix}} \\ &= \frac{E \cdot (H^{iso} + H^{kin})}{E + H^{iso} + H^{kin}} \end{aligned}$$

3.1.1.3 Determination for isotropic and kinematic harding parameters

Fig. 3.7 illustrates the procedure to determine uniaxial stress σ and tangent modulus E^{tan} from uniaxial strain ε.

3.1.2 Bilinear model with isotropic hardening

Whereas the proceeding model is rooted in plasticity theory, its implementation may be problematic. Alternatively, a phenomenologically similar model can be derived based on (Filippou, Popov and Bertero 1983) and as implemented in (Mazzoni, McKenna, Scott, Fenves and et al. 2006).

3.1.2.1 Stress-strain relation

Instead of determining the E^{tan} with H in Sec. 3.1.1.1, the bilinear model computes it through a strain-hardening coefficient b which is the ratio of the post-yield tangent modulus E^{tan} and the initial elastic modulus E, and only considers isotropic hardening with Eq. 3.20 and Eq. 3.21 (Filippou et al. 1983). Fig. 3.8 describes this bilinear model.

$$E^{tan} = b \cdot E$$

To account for the evolution of elastic domain in isotropic harding, a stress shift σ_{Δ} is determined as follow:

- If the incremental strain $\Delta\varepsilon$ changes a positive value into a negative one:

$$\begin{aligned} \Delta^N &= 1 + a_1 \cdot \left(\frac{\varepsilon^{max} - \varepsilon^{min}}{2 \cdot a_2 \cdot \cdot \varepsilon_y} \right)^{0.8} \\ \sigma_{\Delta} &= \Delta^N \cdot \sigma_y \cdot (1 - b) \end{aligned} \tag{3.20}$$

- If the incremental strain $\Delta\varepsilon$ changes a negative value into a positive one:

$$\begin{aligned} \Delta^P &= 1 + a_3 \cdot \left(\frac{\varepsilon^{max} - \varepsilon^{min}}{2 \cdot a_4 \cdot \cdot \varepsilon_y} \right)^{0.8} \\ \sigma_{\Delta} &= \Delta^P \cdot \sigma_y \cdot (1 - b) \end{aligned} \tag{3.21}$$

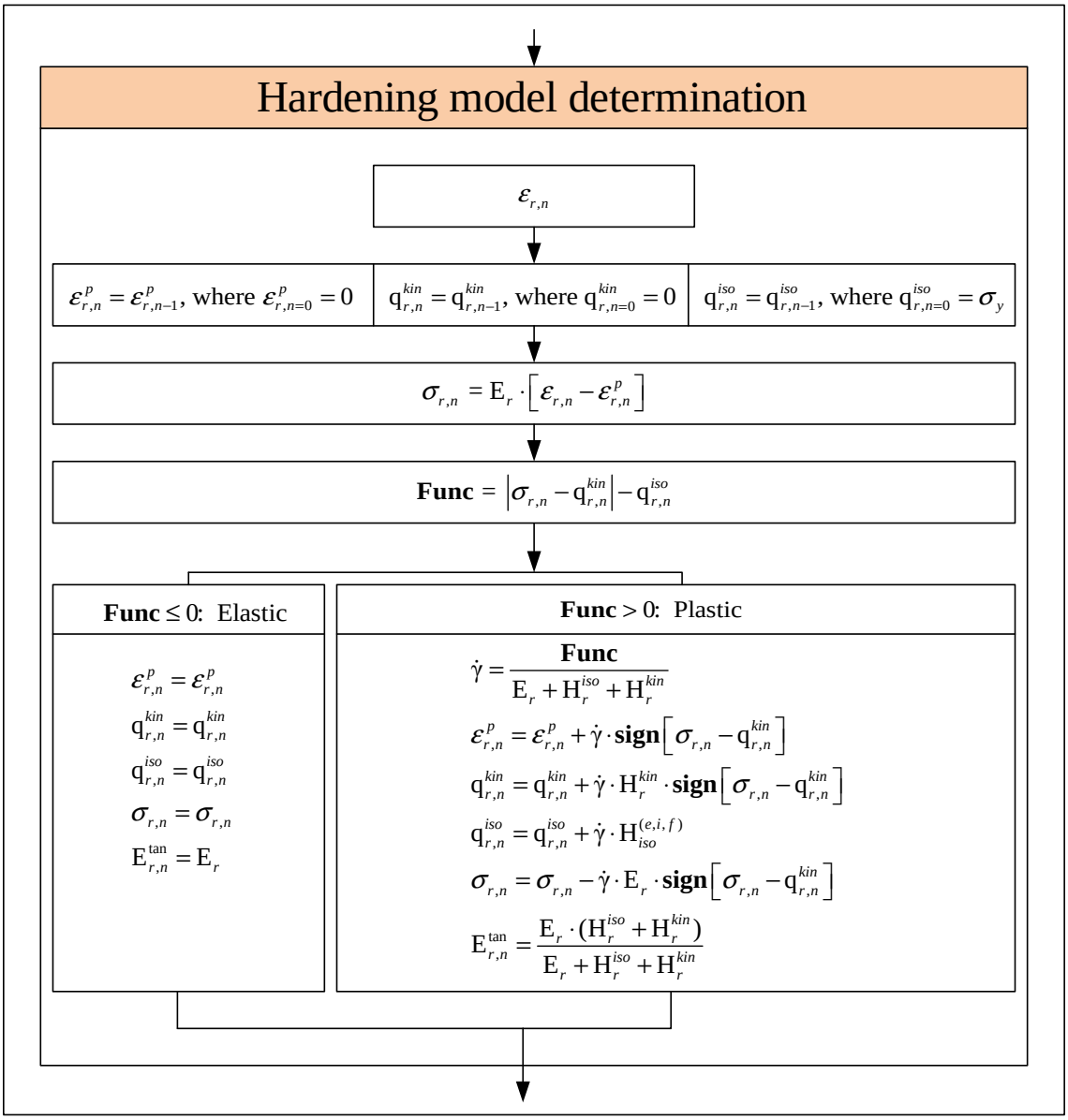


Fig. 3.7 Determination for isotropic and kinematic hardening model

where, a_1 and a_3 are isotropic hardening parameter which reflect an increase of the compression yield envelope through a fraction of the yield strength after a plastic strain $a_2 \cdot \frac{\sigma_y}{E}$, and tension yield envelope as a fraction of the yield strength after a plastic strain of $a_4 \cdot \frac{\sigma_y}{E}$. a_2 and a_4 are isotropic hardening parameter with respect to a_1 and a_3, and ε_{max} and ε_{min} are the strain at the maximum and minimum strain reversal point. Limiting factor of this model is that a_1, a_2, a_3 and a_4 must be determined through curve fitting of the model with experimental results. Default values are $a_1 = 0$, $a_2 = 55$, $a_3 = 0$, and $a_4 = 55$ in (Mazzoni et al. 2006).

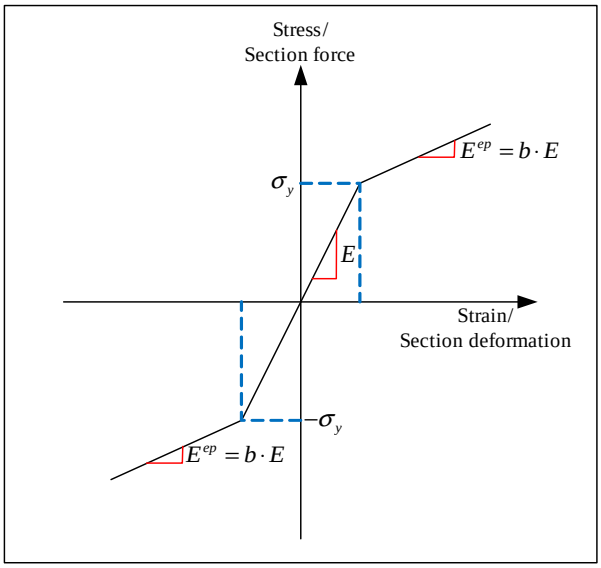


Fig. 3.8 Bilinear model

3.1.2.2 Determination for bilinear model

Fig. 3.9 shows the procedure to obtain uniaxial stress σ and tangent modulus E^{tan} from uniaxial strain ε.

3.1.3 Giuffre-Menegotto-Pinto Model Modified by Filippou et al.

This section is based on doctoral dissertation of Yassin (1994).

3.1.3.1 Stress-strain relationship

The reinforcing steel stress-strain behavior is described by the nonlinear model of Menegotto and Pinto (1973), as modified by Filippou et al. (1983), to include isotropic strain hardening. This model introduces smooth curve to describe similar behavior to experimental one instead of bilinear curve.

The model presented in Menegotto and Pinto (1973) starts from the following emperical form of the stress-strain relation which is found to be close enough to experimentally determined ones.

$$\sigma^* = b \cdot \varepsilon^* + \frac{(1-b) \cdot \varepsilon^*}{(1+\varepsilon^{*R})^{1/R}} \tag{3.22}$$

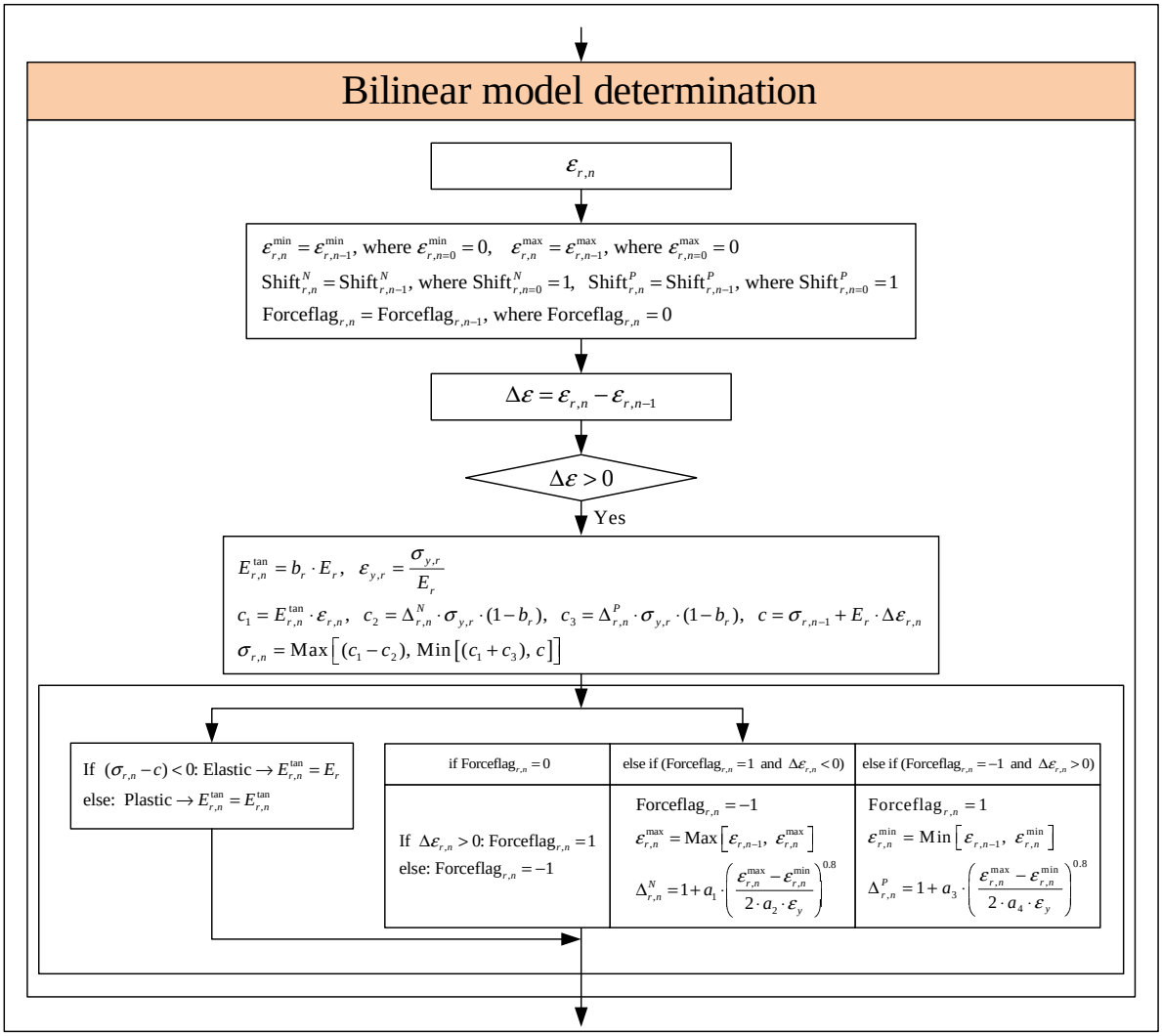


Fig. 3.9 Determination for bilinear model

where,

$$\varepsilon^* = \frac{\varepsilon - \varepsilon_{rev}}{\varepsilon_0 - \varepsilon_{rev}}$$
$$\sigma^* = \frac{\sigma - \sigma_{rev}}{\sigma_0 - \sigma_{rev}} \tag{3.23}$$

The tangent modulus E^{tan} is obtained by differentiating Eq. 3.22 and 3.23,

$$E^{tan} = \frac{\mathrm{d}\sigma}{\mathrm{d}\varepsilon} = \frac{\sigma_0 - \sigma_{rev}}{\varepsilon_0 - \varepsilon_{rev}} \cdot \frac{\mathrm{d}\sigma^*}{\mathrm{d}\varepsilon^*} \tag{3.24}$$

where,

$$\frac{\mathrm{d}\sigma^*}{\mathrm{d}\varepsilon^*} = b + \left[\frac{1-b}{(1+\varepsilon^{*R})^{1/R}}\right] \cdot \left[1 - \frac{\varepsilon^{*R}}{1+\varepsilon^{*R}}\right] \tag{3.25}$$

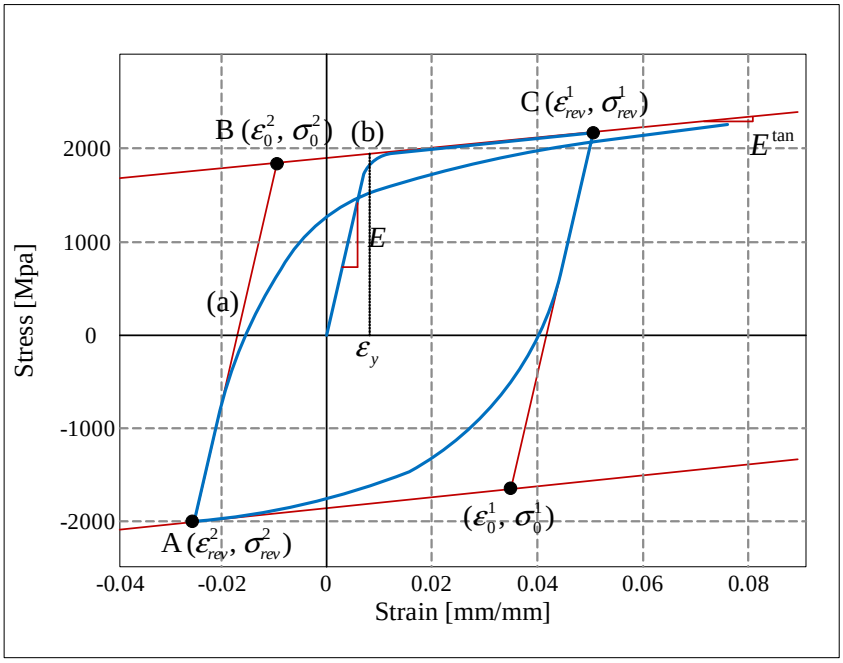


Fig. 3.10 Menegotto-Pinto steel model
(Yassin 1994)

In Fig. 3.10, Eq. 3.23 represents a curved transition from a straight line asymptote with slope E (a) to another asymptote with slope E^{tan} (b), σ_{rev} and ε_{rev} are the stress and strain at the point of strain reversal (point A), which also forms the origin of the asymptote with slope E (a), and σ_0 and ε_0 are the stress and strain at the point of intersection of the two asymptotes (point B).

b is the strain hardening ratio between slope E^{tan} and E, and R is a parameter that influences the curvature of the transition curve between the two asymptotes and permits a good representation of the Bauschinger effect. As indicated in Fig. 3.10, σ_0, ε_0, σ_{rev} and ε_{rev} are

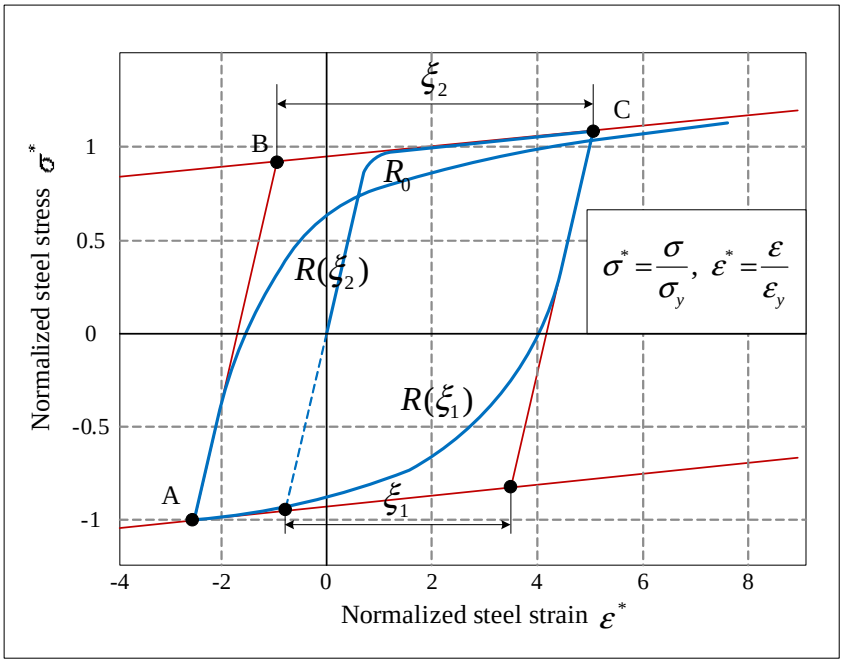


Fig. 3.11 Definition of curvature parameter R in Menegotto-Pinto steel model (Yassin 1994)

updated after each strain reversal.

In Fig. 3.11, R is dependent on the absolute strain difference between the current asymptote intersection point (point B) and the previous maximum or minimum strain reversal point (point C) depending on whether the current strain is increasing or decreasing, respectively. There are two reported expression for $R(\xi)$:

- Menegotto-Pinto original model (Menegotto and Pinto 1973),

$$R(\xi) = R_0 - \frac{cR_1 \cdot \xi}{cR_2 + \xi} \tag{3.26}$$

- `Steel2` in OpenSees (Mazzoni et al. 2006),

$$R(\xi) = R_0 \left(1 - \frac{cR_1 \cdot \xi}{cR_2 + \xi}\right) \tag{3.27}$$

where, R_0 is the value of the parameter R during first loading, and cR_1 and cR_2 are experimentally determined parameters to be defined together with R_0. ξ can be expressed as

$$\xi = \left| \frac{\varepsilon^m - \varepsilon_0}{\varepsilon_y} \right| \tag{3.28}$$

where, ε^m is the strain at the previous maximum or minimum strain reversal point depending on whether the current strain is increasing or decreasing, respectively. ε_0 is the strain at the

current intersection point of the two asymptotes.

As shown in Fig. 3.10, both ε^m and ε_0 lie along the same asymptote and ε_y is the initial yield strain. Fig. 3.11 shows how ξ is updated following a strain reversal.

Some clarification is needed in connection with the set of rules for unloading and reloading which are implied by Eq. 3.22 to 3.28, allowing for a generalized load history. If the analytical model had a memory extending over all previous branches of the stress-strain history, then it would allow for the resumption of the previous reloading branch, as soon as the new reloading curve reached it. However, this would require that the model store all necessary information to retrace all previous incomplete reloading curves, and this is clearly impractical from a computational standpoint. Memory of the past stress-strain history is therefore limited to a predefined number of parameters, which in the present model are:

1. stress and strain at the last state of the model
2. stress and strain at the last state reversal point
3. stress and strain at the last asymptote intersection point
4. flag indication whether the last branch is ascending or descending
5. strain at the previous minimum strain reversal point
6. strain at the previous maximum strain reversal point

As a result of these restrictions reloading following partial unloading does not hit the original curve following unloading started, but, instead, continues on the new reloading curve until reaching the monotonic envelope. However, the discrepancy between the analytical model and the actual behavior is typically very small, Filippou et al. (1983).

The above implementation of the model corresponds to its simplest form, as proposed by Menegotto and Pinto (1973): elastic and yield asymptotes are assumed to be straight lines, the position of the limiting asymptotes corresponding to the yield surface is assumed to be fixed at all times and the slope E remains constant, Fig. 3.10.

In spite of the simplicity in formulation, the model is capable of reproducing well experimental results and its major drawback stems from its failure to allow for isotropic hardening. To account for this effect Filippou et al. (1983) proposed a shift of σ_0 and ε_0 in the linearly yield asymptote as follows:

- If the incremental strain $\Delta\varepsilon$ changes a positive value to a negative value:

$$\begin{aligned} \Delta^N &= 1 + a_1 \cdot \left(\frac{\varepsilon^{max} - \varepsilon^{min}}{2 \cdot a_2 \cdot \cdot \varepsilon_y} \right)^{0.8} \\ \varepsilon_0 &= \frac{-\sigma_y \cdot \Delta^N + E^{tan} \cdot \varepsilon_y \cdot \Delta^N - \sigma_{rev} + E \cdot \varepsilon_{rev}}{E - E^{tan}} \\ \sigma_0 &= -\sigma_y \cdot \Delta^N + E^{tan} \cdot (\varepsilon_o + \varepsilon_y \cdot \Delta^N) \end{aligned} \tag{3.29}$$

- If the incremental strain $\Delta\varepsilon$ changes a negative value to a positive value,

$$\begin{aligned} \Delta^P &= 1 + a_3 \cdot \left(\frac{\varepsilon^{max} - \varepsilon^{min}}{2 \cdot a_4 \cdot \cdot \varepsilon_y} \right)^{0.8} \\ \varepsilon_0 &= \frac{\sigma_y \cdot \Delta^P - E^{tan} \cdot \varepsilon_y \cdot \Delta^P - \sigma_{rev} + E \cdot \varepsilon_{rev}}{E - E^{tan}} \\ \sigma_0 &= \sigma_y \cdot \Delta^P + E^{tan} \cdot (\varepsilon_o - \varepsilon_y \cdot \Delta^P) \end{aligned} \tag{3.30}$$

where, a_1 and a_3 are isotropic hardening parameter which reflect an increase of the compression yield envelope through a fraction of the yield strength after a plastic strain $a_2 \cdot \frac{\sigma_y}{E}$, and tension yield envelope as a fraction of the yield strength after a plastic strain of $a_4 \cdot \frac{\sigma_y}{E}$. a_2 and a_4 are isotropic hardening parameter with respect to a_1 and a_3, and ε_{max} and ε_{min} are the strain at the maximum and minimum strain reversal point. Limiting factor of this model is that a_1, a_2, a_3 and a_4 must be determined through curve fitting of the model with experimental results. Default values are $a_1 = 0$, $a_2 = 55$, $a_3 = 0$, and $a_4 = 55$ in (Mazzoni et al. 2006).

3.1.3.2 Determination for modified Giuffre-Menegotto-Pinto Model

Fig. 3.12 to 3.15 show the procedure to obtain uniaxial stress σ and tangent modulus E^{tan} from uniaxial strain ε.

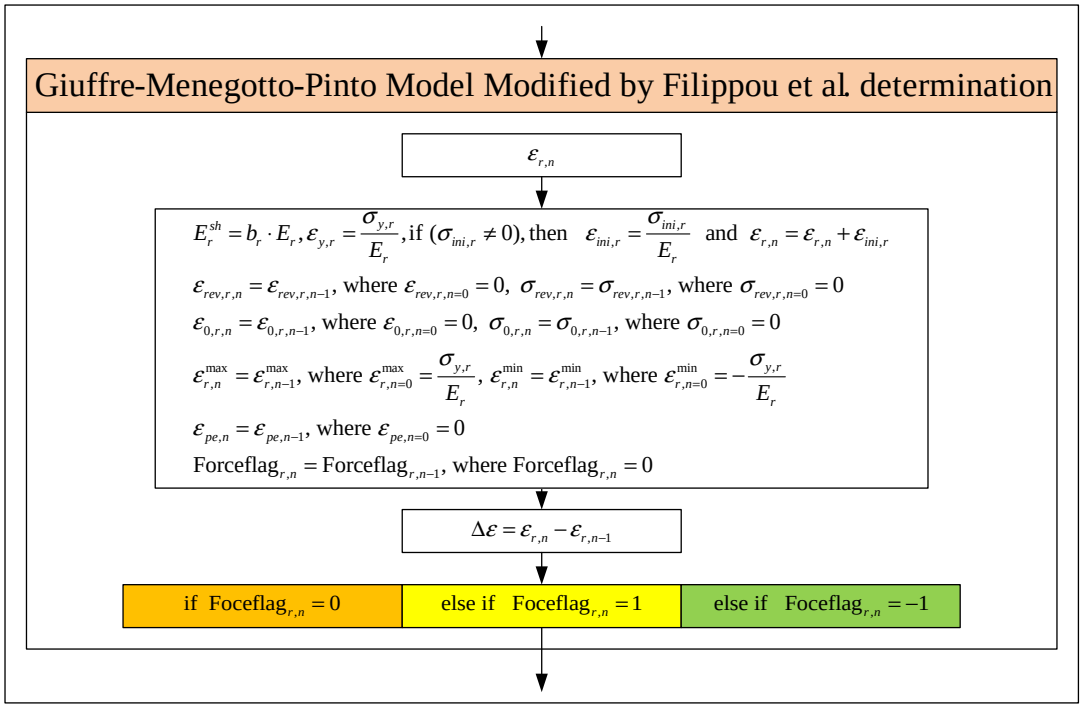


Fig. 3.12 Determination (1) for Giuffre-Menegotto-Pinto Model Modified by Filippou et al.

if $\text{Foceflag}_{r,n} = 0$

if $\Delta\varepsilon = 0$	else if $\Delta\varepsilon < 0$	else if $\Delta\varepsilon > 0$
$\varepsilon_{rev,r,n} = 0$	$\varepsilon_{rev,r,n} = 0$	$\varepsilon_{rev,r,n} = 0$
$\sigma_{rev,r,n} = 0$	$\sigma_{rev,r,n} = 0$	$\sigma_{rev,r,n} = 0$
$\varepsilon_{0,r,n} = 0$	$\varepsilon_{0,r,n} = \sigma_{y,r} / E_r$	$\varepsilon_{0,r,n} = -\sigma_{y,r} / E_r$
$\sigma_{0,r,n} = 0$	$\sigma_{0,r,n} = \sigma_{y,r}$	$\sigma_{0,r,n} = -\sigma_{y,r}$
$\varepsilon_{r,n}^{\max} = \sigma_{y,r} / E_r$	$\varepsilon_{r,n}^{\max} = \sigma_{y,r} / E_r$	$\varepsilon_{r,n}^{\max} = \sigma_{y,r} / E_r$
$\varepsilon_{r,n}^{\min} = -\sigma_{y,r} / E_r$	$\varepsilon_{r,n}^{\min} = -\sigma_{y,r} / E_r$	$\varepsilon_{r,n}^{\min} = -\sigma_{y,r} / E_r$
$\text{Forceflag}_{r,n} = 0$	$\varepsilon_{pe,n} = \varepsilon_{r,n}^{\min}$	$\varepsilon_{pe,n} = \varepsilon_{r,n}^{\max}$
$\sigma_{r,n} = \sigma_{ini,r}$	$\text{Forceflag}_{r,n} = 1$	$\text{Forceflag}_{r,n} = -1$
$E_{r,n}^{\tan} = E_r$	$\xi = \left\lvert \frac{\varepsilon_{pe,n} - \varepsilon_{0,r,n}}{\varepsilon_{y,r}} \right\rvert$	$\xi = \left\lvert \frac{\varepsilon_{pe,n} - \varepsilon_{0,r,n}}{\varepsilon_{y,r}} \right\rvert$
	$R = R_0 - R_0^{nratio} \cdot \frac{cR_1 \cdot \xi}{cR_2 + \xi}$	$R = R_0 - R_0^{nratio} \cdot \frac{cR_1 \cdot \xi}{cR_2 + \xi}$
	$\dot{\varepsilon} = \frac{\varepsilon_{r,n} - \varepsilon_{rev,r,n}}{\varepsilon_{0,r,n} - \varepsilon_{rev,r,n}}$	$\dot{\varepsilon} = \frac{\varepsilon_{r,n} - \varepsilon_{rev,r,n}}{\varepsilon_{0,r,n} - \varepsilon_{rev,r,n}}$
	$c_1 = 1 + \lvert\dot{\varepsilon}\rvert^R$	$c_1 = 1 + \lvert\dot{\varepsilon}\rvert^R$
	$c_2 = c_1^{1/R}$	$c_2 = c_1^{1/R}$
	$\sigma_{r,n} = b_r \cdot \dot{\varepsilon} + (1 - b_r) \cdot \frac{\dot{\varepsilon}}{c_2}$	$\sigma_{r,n} = b_r \cdot \dot{\varepsilon} + (1 - b_r) \cdot \frac{\dot{\varepsilon}}{c_2}$
	$\sigma_{r,n} = \sigma_{r,n} \cdot (\sigma_{0,r,n} - \sigma_{rev,r,n}) + \sigma_{rev,r,n}$	$\sigma_{r,n} = \sigma_{r,n} \cdot (\sigma_{0,r,n} - \sigma_{rev,r,n}) + \sigma_{rev,r,n}$
	$E_{r,n}^{\tan} = b_r + \frac{(1 - b_r)}{c_1 \cdot c_2}$	$E_{r,n}^{\tan} = b_r + \frac{(1 - b_r)}{c_1 \cdot c_2}$
	$E_{r,n}^{\tan} = E_{r,n}^{\tan} \cdot \frac{\sigma_{0,r,n} - \sigma_{rev,r,n}}{\varepsilon_{0,r,n} - \varepsilon_{rev,r,n}}$	$E_{r,n}^{\tan} = E_{r,n}^{\tan} \cdot \frac{\sigma_{0,r,n} - \sigma_{rev,r,n}}{\varepsilon_{0,r,n} - \varepsilon_{rev,r,n}}$

Fig. 3.13 Determination (2) for Giuffre-Menegotto-Pinto Model Modified by Filippou et al.

else if $\text{Foceflag}_{r,n} = 1$

if $\Delta\varepsilon < 0$	else if $\Delta\varepsilon > 0$
$\varepsilon_{rev,r,n}$, $\sigma_{rev,r,n}$	$\varepsilon_{rev,r,n}$, $\sigma_{rev,r,n}$, $\varepsilon_{0,r,n}$, $\sigma_{0,r,n}$
$\Delta^N = 1 + a_1 \cdot \left(\dfrac{\varepsilon_{r,n}^{\max} - \varepsilon_{r,n}^{\min}}{2 \cdot a_2 \cdot \varepsilon_y} \right)^{0.8}$	$\varepsilon_{r,n}^{\max} = \text{Max}\left[\varepsilon_{r,n}^{\max}, \varepsilon_{r,n-1} \right]$
$\varepsilon_{0,r,n} = \dfrac{-\sigma_{y,r} \cdot \Delta^N + E_r^{sh} \cdot \varepsilon_{y,r} \cdot \Delta^N - \sigma_{rev,r,n} + E_r \cdot \varepsilon_{rev,r,n}}{E_r - E_r^{sh}}$	$\varepsilon_{r,n}^{\min} = \text{Min}\left[\varepsilon_{r,n}^{\min}, \varepsilon_{r,n-1} \right]$
$\sigma_{0,r,n} = -\sigma_{y,r} \cdot \Delta^N + E_r^{sh} \cdot (\varepsilon_{0,r,n} + \varepsilon_{y,r} \cdot \Delta^N)$	$\varepsilon_{pe,n} = \varepsilon_{r,n}^{\max}$
$\varepsilon_{r,n}^{\max} = \text{Max}\left[\varepsilon_{r,n}^{\max}, \varepsilon_{r,n-1} \right]$	$\text{Forceflag}_{r,n} = 1$
$\varepsilon_{r,n}^{\min} = \text{Min}\left[\varepsilon_{r,n}^{\min}, \varepsilon_{r,n-1} \right]$	$\xi = \left\lvert \dfrac{\varepsilon_{pe,n} - \varepsilon_{0,r,n}}{\varepsilon_{y,r}} \right\rvert$
$\varepsilon_{pe,n} = \varepsilon_{r,n}^{\min}$	$R = R_0 - R_0^{nratio} \cdot \dfrac{cR_1 \cdot \xi}{cR_2 + \xi}$
$\text{Forceflag}_{r,n} = -1$	$\dot{\varepsilon} = \dfrac{\varepsilon_{r,n} - \varepsilon_{rev,r,n}}{\varepsilon_{0,r,n} - \varepsilon_{rev,r,n}}$
$\xi = \left\lvert \dfrac{\varepsilon_{pe,n} - \varepsilon_{0,r,n}}{\varepsilon_{y,r}} \right\rvert$	$c_1 = 1 + \lvert \dot{\varepsilon} \rvert^R$
$R = R_0 - R_0^{nratio} \cdot \dfrac{cR_1 \cdot \xi}{cR_2 + \xi}$	$c_2 = c_1^{1/R}$
$\dot{\varepsilon} = \dfrac{\varepsilon_{r,n} - \varepsilon_{rev,r,n}}{\varepsilon_{0,r,n} - \varepsilon_{rev,r,n}}$	$\sigma_{r,n} = b_r \cdot \dot{\varepsilon} + (1 - b_r) \cdot \dfrac{\dot{\varepsilon}}{c_2}$
$c_1 = 1 + \lvert \dot{\varepsilon} \rvert^R$	$\sigma_{r,n} = \sigma_{r,n} \cdot (\sigma_{0,r,n} - \sigma_{rev,r,n}) + \sigma_{rev,r,n}$
$c_2 = c_1^{1/R}$	$E_{r,n}^{\tan} = b_r + \dfrac{(1 - b_r)}{c_1 \cdot c_2}$
$\sigma_{r,n} = b_r \cdot \dot{\varepsilon} + (1 - b_r) \cdot \dfrac{\dot{\varepsilon}}{c_2}$	$E_{r,n}^{\tan} = E_{r,n}^{\tan} \cdot \dfrac{\sigma_{0,r,n} - \sigma_{rev,r,n}}{\varepsilon_{0,r,n} - \varepsilon_{rev,r,n}}$
$\sigma_{r,n} = \sigma_{r,n} \cdot (\sigma_{0,r,n} - \sigma_{rev,r,n}) + \sigma_{rev,r,n}$	
$E_{r,n}^{\tan} = b_r + \dfrac{(1 - b_r)}{c_1 \cdot c_2}$	
$E_{r,n}^{\tan} = E_{r,n}^{\tan} \cdot \dfrac{\sigma_{0,r,n} - \sigma_{rev,r,n}}{\varepsilon_{0,r,n} - \varepsilon_{rev,r,n}}$	

Fig. 3.14 Determination (3) for Giuffre-Menegotto-Pinto Model Modified by Filippou et al.

else if $\text{Foceflag}_{r,n} = -1$

if $\Delta\varepsilon > 0$	else if $\Delta\varepsilon < 0$
$\varepsilon_{rev,r,n}$, $\sigma_{rev,r,n}$	$\varepsilon_{rev,r,n}$, $\sigma_{rev,r,n}$, $\varepsilon_{0,r,n}$, $\sigma_{0,r,n}$
$\Delta^P = 1 + a_3 \cdot \left(\frac{\varepsilon_{r,n}^{\max} - \varepsilon_{r,n}^{\min}}{2 \cdot a_4 \cdot \varepsilon_y} \right)^{0.8}$	$\varepsilon_{r,n}^{\max} = \text{Max}\left[\varepsilon_{r,n}^{\max}, \varepsilon_{r,n-1} \right]$
$\varepsilon_{0,r,n} = \frac{\sigma_{y,r} \cdot \Delta^P - E_r^{sh} \cdot \varepsilon_{y,r} \cdot \Delta^P - \sigma_{rev,r,n} + E_r \cdot \varepsilon_{rev,r,n}}{E_r - E_r^{sh}}$	$\varepsilon_{r,n}^{\min} = \text{Min}\left[\varepsilon_{r,n}^{\min}, \varepsilon_{r,n-1} \right]$
$\sigma_{0,r,n} = \sigma_{y,r} \cdot \Delta^P + E_r^{sh} \cdot (\varepsilon_{0,r,n} - \varepsilon_{y,r} \cdot \Delta^P)$	$\varepsilon_{pe,n} = \varepsilon_{r,n}^{\min}$
$\varepsilon_{r,n}^{\max} = \text{Max}\left[\varepsilon_{r,n}^{\max}, \varepsilon_{r,n-1} \right]$	$\text{Forceflag}_{r,n} = -1$
$\varepsilon_{r,n}^{\min} = \text{Min}\left[\varepsilon_{r,n}^{\min}, \varepsilon_{r,n-1} \right]$	$\xi = \left\lvert \frac{\varepsilon_{pe,n} - \varepsilon_{0,r,n}}{\varepsilon_{y,r}} \right\rvert$
$\varepsilon_{pe,n} = \varepsilon_{r,n}^{\max}$	$R = R_0 - R_0^{nratio} \cdot \frac{cR_1 \cdot \xi}{cR_2 + \xi}$
$\text{Forceflag}_{r,n} = 1$	$\dot{\varepsilon} = \frac{\varepsilon_{r,n} - \varepsilon_{rev,r,n}}{\varepsilon_{0,r,n} - \varepsilon_{rev,r,n}}$
$\xi = \left\lvert \frac{\varepsilon_{pe,n} - \varepsilon_{0,r,n}}{\varepsilon_{y,r}} \right\rvert$	$\text{c}_1 = 1 + \lvert \dot{\varepsilon} \rvert^R$
$R = R_0 - R_0^{nratio} \cdot \frac{cR_1 \cdot \xi}{cR_2 + \xi}$	$\text{c}_2 = \text{c}_1^{1/R}$
$\dot{\varepsilon} = \frac{\varepsilon_{r,n} - \varepsilon_{rev,r,n}}{\varepsilon_{0,r,n} - \varepsilon_{rev,r,n}}$	$\sigma_{r,n} = b_r \cdot \dot{\varepsilon} + (1 - b_r) \cdot \frac{\dot{\varepsilon}}{\text{c}_2}$
$\text{c}_1 = 1 + \lvert \dot{\varepsilon} \rvert^R$	$\sigma_{r,n} = \sigma_{r,n} \cdot (\sigma_{0,r,n} - \sigma_{rev,r,n}) + \sigma_{rev,r,n}$
$\text{c}_2 = \text{c}_1^{1/R}$	$E_{r,n}^{\tan} = b_r + \frac{(1 - b_r)}{\text{c}_1 \cdot \text{c}_2}$
$\sigma_{r,n} = b_r \cdot \dot{\varepsilon} + (1 - b_r) \cdot \frac{\dot{\varepsilon}}{\text{c}_2}$	$E_{r,n}^{\tan} = E_{r,n}^{\tan} \cdot \frac{\sigma_{0,r,n} - \sigma_{rev,r,n}}{\varepsilon_{0,r,n} - \varepsilon_{rev,r,n}}$
$\sigma_{r,n} = \sigma_{r,n} \cdot (\sigma_{0,r,n} - \sigma_{rev,r,n}) + \sigma_{rev,r,n}$	
$E_{r,n}^{\tan} = b_r + \frac{(1 - b_r)}{\text{c}_1 \cdot \text{c}_2}$	
$E_{r,n}^{\tan} = E_{r,n}^{\tan} \cdot \frac{\sigma_{0,r,n} - \sigma_{rev,r,n}}{\varepsilon_{0,r,n} - \varepsilon_{rev,r,n}}$	

Fig. 3.15 Determination (4) for Giuffre-Menegotto-Pinto Model Modified by Filippou et al.

3.2 Concrete Models

3.2.1 Modified Kent And Park Model

This section is based on doctoral dissertation of (Yassin 1994).

3.2.1.1 Stress-strain relation

The concrete model describes the concrete stress-strain relation under an arbitrary cyclic strain history. In particular, the model implemented takes into account four important factors:

1. effect of concrete confinement on the monotonic envelope curve in compression
2. successive degradation of stiffness of both the unloading and reloading curves, for increasing values of compressive strain
3. effect of tension stiffening
4. hysteretic response under cyclic loading in compression

The monotonic envelope curve of concrete in compression follows the original model of Kent and Park (1971) and extended by Scott, Park and Priestley (1982). Even though more accurate and complete models have been published since, the so-called modified Kent and Park model offers a good balance between simplicity and accuracy.

Tension stiffening is the ability of concrete between cracks to resist tensile stress and contribute to the flexural stiffness of the member. Due to the discrete nature of the cracks, concrete in between cracks remains bonded to the reinforcement and, thus, contributes to the stiffness of the member. However, as the magnitude of load increases, additional cracks form at closer intervals, hence reducing the tensile stress that can be developed in the concrete. Therefore tension stiffening is gradually reduced as load is increased in the post-cracking stage. Past investigators have taken tension stiffening into account by modifying the concrete stress-strain relation such that, after reaching the tensile strength (cracking), the tensile stress reduces gradually to zero as tensile strain is increased. The gradual reduction of tensile strength is often approximated as linear, multi-linear or exponential. A similar approach with a linear rate of reduction is adopted. However, the results indicate that the application of tension stiffening over all fibers of a member can lead to significant overestimation of the ultimate strength. Rather tension stiffening is a localized phenomenon that affects the concrete in the immediate vicinity of the reinforcement. In analytical studies, only the concrete fibers within an effective area around the reinforcements are assigned tension softening. Clearly, the size of the effective area has a significant effect on tension stiffening behavior of the member.

In the modified Kent and Park model (1971) shown in Fig. 3.16, the monotonic concrete stress-strain relation in compression is empirically defined by three regions. Adopting the convention that compression is positive, the three regions are,

- Region OA: $\varepsilon_c \leq \varepsilon_0$

$$\sigma_c = K \cdot f_c \cdot \left[2 \cdot \frac{\varepsilon_c}{\varepsilon_0} - \left(\frac{\varepsilon_c}{\varepsilon_0} \right)^2 \right]$$

- Region AB: $\varepsilon_0 < \varepsilon_c \leq \varepsilon_{20}$

$$\sigma_c = K \cdot f_c \cdot [1 - Z(\varepsilon_c - \varepsilon_0)]$$

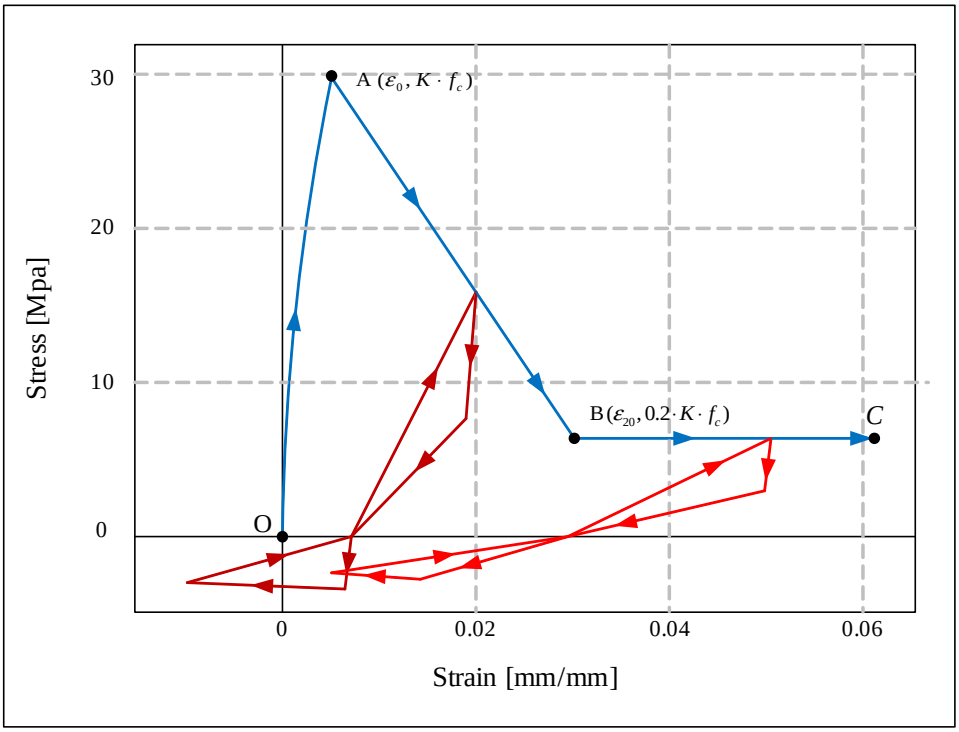


Fig. 3.16 Concrete material model in compression (Yassin 1994)

- Region BC: $\varepsilon_c > \varepsilon_{20}$

$$\sigma_c = 0.2 \cdot K \cdot f_c$$

The corresponding tangent moduli are given by the following equations

- Region OA: $\varepsilon_c \leq \varepsilon_0$

$$E^{tan} = \frac{2 \cdot K \cdot f_c}{\varepsilon_0} \cdot \left(1 - \frac{\varepsilon_c}{\varepsilon_0}\right)$$

- Region AB: $\varepsilon_0 < \varepsilon_c \leq \varepsilon_{20}$

$$E^{tan} = -Z \cdot K \cdot f_c$$

- Region BC: $\varepsilon_c > \varepsilon_{20}$

$$E^{tan} = 0$$

where,

$$\varepsilon_0 = 0.002 \cdot K$$

$$K = 1 + \frac{\rho_s \cdot f_{ys}}{f_c}$$

$$Z = \frac{0.5}{\frac{3+0.29 \cdot f_c}{145 \cdot f_c - 1000} + 0.75 \cdot \rho_s \cdot \sqrt{\frac{h}{s_h}} - 0.002 \cdot K}$$

ε_0 is the concrete strain corresponding maximum stress, ε_{20} the concrete strain at 20 percent of maximum stress, K a factor which accounts for the strength increase due to confinement, Z the strain softening slope, f_c the concrete compressive cylinder strength in MPa (1 MPa = 145 psi), f_{ys} the yield strength of stirrups in MPa, ρ_s the ratio of the volume of hoop reinforcement to the volume of concrete core measured to outside of stirrups, h the width of concrete core measured to outside of stirrups, and s_h the center to center spacing of stirrups or hoop sets.

The cyclic unloading and reloading behavior is represented by a set of straight lines. Fig. 3.16 shows that hysteretic behavior occurs under, both, tensile and compressive stress. Although the compressive and tensile hysteresis loops are continuous, they will be discussed separately for the sake of clarity.

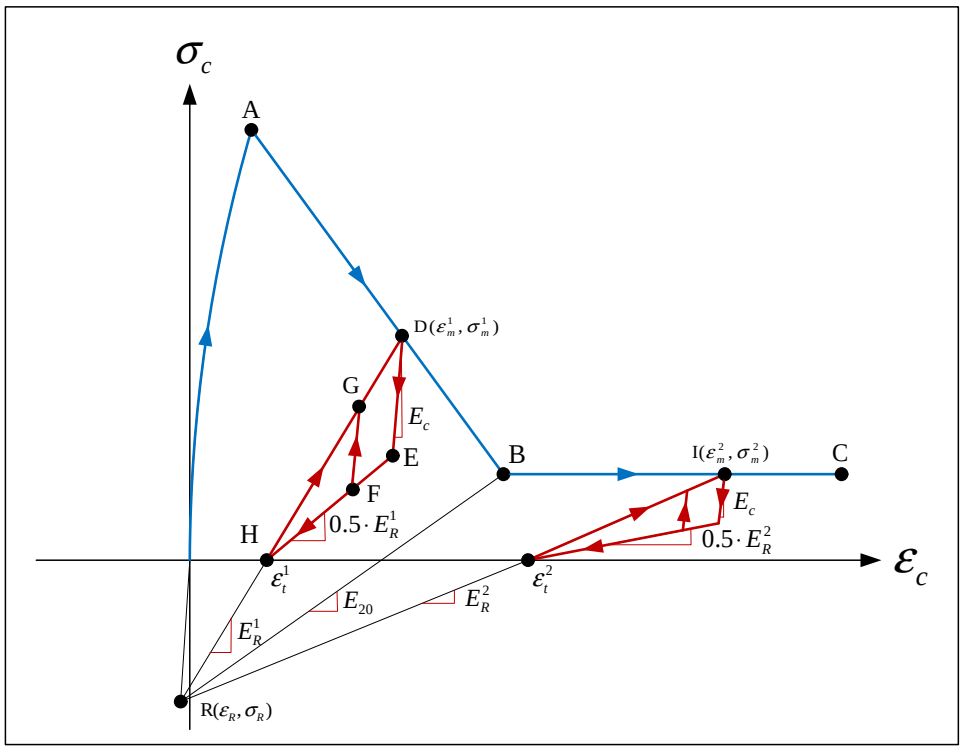


Fig. 3.17 Concrete material model under cyclic loading in compression (Yassin 1994)

On the compressive side of the model, there is a successive degradation of stiffness of both the unloading and reloading lines for increasing values of maximum strain, as shown in Fig. 3.16. The degradation of stiffness is such that the projections of all reloading lines intersect at a common point R in Fig. 3.17. Point R is determined by the intersection of the tangent to the monotonic envelope curve at the origin and the projection of the unloading line from point B that corresponds to concrete strength of $0.2 \cdot f_c$ (Fig. 3.17). The strain and stress at the intersection point are given by the following expressions

$$\begin{aligned} \varepsilon_R &= \frac{0.2 \cdot K \cdot f_c - E_{20} \cdot \varepsilon_{20}}{E_c - E_{20}} \\ \sigma_R &= E_c \cdot \varepsilon_R \end{aligned} \tag{3.31}$$

where E_c is the tangent modulus of the monotonic envelope curve at the origin, E_{20} is the unloading modulus at point B of the monotonic envelope curve with a strength of $0.2 \cdot f_c$. The magnitude of E_{20} has to be determined experimentally.

After unloading from a point on the compressive monotonic envelope (point D in Fig. 3.17), and before reaching the zero stress axis (point H in Fig. 3.17), the model response follows two smaller envelopes that are defined by the following equations,

- Maximum envelope (line HD)

$$\sigma_{max} = \sigma_m + E_R \cdot (\varepsilon_c - \varepsilon_m) \tag{3.32}$$

- Minimum envelope (line HE)

$$\sigma_{min} = 0.5 \cdot E_R \cdot (\varepsilon_c - \varepsilon_t) \tag{3.33}$$

where,

$$\begin{aligned} E_R &= \frac{\sigma_m - \sigma_R}{\varepsilon_m - \varepsilon_R} \\ \varepsilon_t &= \varepsilon_m - \frac{\sigma_m}{E_R} \end{aligned} \tag{3.34}$$

σ_m and ε_m are the stress and strain at the unloading point on the compressive monotonic envelope, respectively. Therefore, the positions of the two smaller envelopes depend on the position of the unloading point. For partial loading and unloading cycles within the smaller envelopes the model follows straight line with modulus E_c.

In the numerical implementation a trial stress and tangent modulus are assumed based on linear elastic behavior with slope E_c,

$$\sigma_{c,n}^{tr} = \sigma_{c,n-1} + E_c \cdot \Delta\varepsilon_{c,n} \tag{3.35}$$

where $\sigma_{c,n}^{tr}$ is the updated trial stress, $\sigma_{c,n-1}$ is the previous stress state and $\Delta\varepsilon_{c,n}$ is the strain increment. The following rules are then used to determine actual stress and modulus of the model

$$\begin{aligned} &\text{if} \quad \sigma_{min} \leq \sigma_{c,n}^{tr} \leq \sigma_{max} \quad \text{then} \quad \sigma_{c,n} = \sigma_{c,n}^{tr} \quad \text{and} \quad E^{tan} = E_c \\ &\text{if} \quad \sigma_{c,n}^{tr} < \sigma_{min} \quad \text{then} \quad \sigma_{c,n} = \sigma_{min} \quad \text{and} \quad E^{tan} = 0.5 \cdot E_r \\ &\text{if} \quad \sigma_{c,n}^{tr} > \sigma_{max} \quad \text{then} \quad \sigma_{c,n} = \sigma_{max} \quad \text{and} \quad E^{tan} = E_r \end{aligned} \tag{3.36}$$

The rules governing the hysteretic behavior of the model in compression according to Eq. 3.31 to 3.36 are illustrated by a sample history in Fig. 3.17. If unloading occurs from point D to point E, reloading will be on the same path back to D. If unloading reaches point F, the hysteresis loop DEFGD will result upon reloading. If complete unloading to point H occurs, reloading will result in the hysteresis loop DEHD. It is important to note that the reloading line will always rejoin the compression monotonic envelope at the point of initial unloading. For the case when unloading continues past point H and the model starts reloading in tension, a

different set of rules govern the hysteretic behavior. However upon reloading in compression, the model will reenter the compression region at the point on the zero stress axis at the completion of unloading (point H) and whatever happens in the tension region will not affect the behavior of the model, once it returns to the compression region.

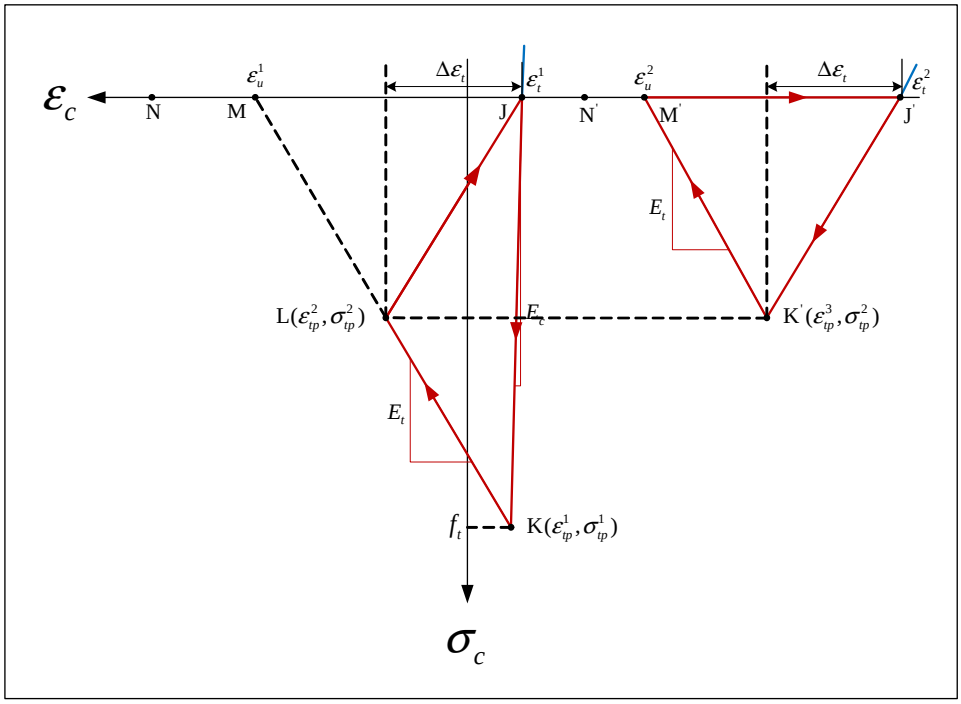


Fig. 3.18 Concrete material model under cyclic loading in tension (Yassin 1994)

The tensile behavior of the model, as shown in Fig. 3.18, takes into account tension stiffening and the degradation of the unloading and reloading stiffness for increasing values of maximum tensile strain after initial cracking. The maximum tensile strength of the concrete (modulus of rupture) is assumed equal to,

$$f_t = 0.6228\sqrt{f_c} \tag{3.37}$$

where f_t and f_c are expressed in MPa.

Fig. 3.18 shows two consecutive tensile hysteresis loops which are part of a sample cyclic history that also include compressive stresses. The model assumes that tensile stress can occur anywhere along the strain axis, either as a result of initial tensile loading or as a result of unloading from a compressive state. In the latter case a tensile stress occurs under a compressive strain. The tensile stress-strain relation is defined by three points with coordinates $(\varepsilon_t, 0)$, $(\varepsilon_{tp}, \sigma_{tp})$ and $(\varepsilon_u, 0)$, as represented by points J, K and M in Fig. 3.18, respectively. ε_t is the strain at the point where the unloading line from the compressive stress region crosses the strain axis. ε_t is given by Eq. 3.34 and changes with maximum compressive strain. ε_{tp} and σ_{tp} are the strain and stress at the peak of the tensile stress-strain relation and are given by the

following expressions,

$$\begin{aligned} \varepsilon_{tp} &= \varepsilon_t + \Delta\varepsilon_t \\ \sigma_{tp} &= f_t \cdot \left(1 + \frac{E_t}{E_c}\right) - E_t \cdot \Delta\varepsilon_t \end{aligned} \tag{3.38}$$

where $\Delta\varepsilon_t$ is the previous maximum differential between tensile strain and ε_t as shown in Fig. 3.18. Before initial cracking, $\Delta\varepsilon_t$ is equal to f_t/E_c. E_t is the tension stiffening modulus that depends on numerical and physical parameters, ε_u is the strain at the point where the tensile stress is reduced to zero and is given by the expression,

$$\varepsilon_u = \varepsilon_t + f_t \cdot \left(\frac{1}{E_t} + \frac{1}{E_c}\right) \tag{3.39}$$

Given these three control points, the tensile stress-strain relation and tangent moduli are defined by the following equations (tension is positive),

- Region JK: $\varepsilon_t < \varepsilon_c \leq \varepsilon_{tp}$

$$\sigma_c = E^{tan} \cdot (\varepsilon_c - \varepsilon_t), \quad E^{tan} = \frac{\sigma_{tp}}{\varepsilon_{tp} - \varepsilon_t} \tag{3.40}$$

- Region KM: $\varepsilon_{tp} < \varepsilon_c \leq \varepsilon_u$

$$\sigma_c = \sigma_{tp} + E^{tan} \cdot (\varepsilon_c - \varepsilon_{tp}), \quad E^{tan} = -E_t \tag{3.41}$$

- Region MN: $\varepsilon_c > \varepsilon_u$

$$\sigma_c = 0, \quad E^{tan} = 0 \tag{3.42}$$

If $\varepsilon_{tp} \geq \varepsilon_u$ then σ_{tp}, σ_c and E^{tan} are all assumed to be zero. The modulus E_t controls the degree of tension stiffening by controlling the slope of Eq. 3.41. The steeper the slope, the smaller will be the effect of tension stiffening. Tensile unloading and reloading are governed by Eq. 3.40 which also includes stiffness degradation for increasing values of strain differential $\Delta\varepsilon_t$. The value of $\Delta\varepsilon_t$ changes whenever $\varepsilon_c - \varepsilon_{tp}$.

The tensile behavior of the model, as characterized by Eq. 3.37 to 3.42, can be better understood by following the example load paths in Fig. 3.18. As the model unloads from compression, it crosses the strain axis at the point J. It then loads in tension until initial cracking occurs at point K. Beyond point K softening commences until the strain reversal point L. The unloading path follows a straight line from point L to point J where the model reloads in compression. The second time the model goes into tension is at point J'. The reloading path J'K' is exactly the duplication of the previous unloading path LJ that has been shifted a distance JJ' along the strain axis. At point K' the model rejoins the softening branch which continues until the tensile stress is reduced to zero at point M'. The stress remains zero through the strain reversal point N' until the model reloads in compression at point J'. Henceforth, the tensile stress capacity of the model is reduced to zero.

The present concrete model is relatively economical in terms of the amount of memory required of the past stress-strain history. The parameters that are used as memory can be

listed as follows:

1. the stress and strain at the point corresponding to the last model state
2. the strain at the last unloading point on the compressive monotonic envelope, ε_m
3. The differential $\Delta\varepsilon_t$ between maximum previous tensile strain and ε_t

The concrete damage considered in the present model is in the form of unloading and reloading stiffness degradation for increasing values of maximum strain. But actual concrete damage also includes the reduction of the monotonic envelopes under cyclic loading.

3.2.1.2 Determination for modified Kent and Park model

Fig. 3.19 and 3.20 show the procedure to obtain uniaxial stress σ and tangent modulus E^{tan} from uniaxial strain ε.

3.2.2 Anisotropic damage model with effective damage and stiffness recovery

For real time hybrid simulation of reinforced concrete structures, it is of paramount importance that the concrete model does not require heavy computations, yet be accurate enough to be realistic. Such a model is developed by Ragueneau, Souid, Delaplace and Desmorat (2006).

3.2.2.1 Constitutive model

Thermodynamic potential The Gibbs potential $\rho \cdot \phi^*$ is based on the Ladeveze's framework (Ladeveze 1983) for anisotropic damage model. The splitting of the stress tensor into deviatoric and hydrostatic components allows us to consider seperately the shear and the hydrostatic parts. Furthemore, each component is again splitted into two additional ones: the positive and the negative in order to model the unilateral effect, Fig. 3.21. The damage influences only the positive part through the damage tensor $\mathbf{D}$.

$$\rho \cdot \phi^* = \frac{1+\nu}{2 \cdot E}\left[\mathrm{Tr}\left(\mathbf{H} \cdot \boldsymbol{\sigma}_+^D \mathbf{H} \cdot \boldsymbol{\sigma}_+^D\right) + \mathrm{Tr}\left(\langle \boldsymbol{\sigma}^D \rangle_- \langle \boldsymbol{\sigma}^D \rangle_-\right)\right] + \frac{1-2\nu}{6 \cdot E}\left[\frac{\langle \mathrm{Tr}\boldsymbol{\sigma}\rangle_+^2}{1-D_H} + \langle \mathrm{Tr}\boldsymbol{\sigma}\rangle_-^2\right] \tag{3.43}$$

where ρ, ν and E are respectively the density, the Poisson's ratio and Young modulus. The damage tensor is defined by

$$\mathbf{H} = (\mathbf{1} - \mathbf{D})^{\frac{1}{2}}$$

where $(\cdot)^D$ is the deviatoric of $(\cdot)$,

$$(\cdot)^D = (\cdot) - \frac{1}{3}\mathrm{Tr}(\cdot) \cdot \mathbf{1}$$

$\langle \cdot \rangle_{+,-}$ is the positive or negative part of $(\cdot)$, and D_H is the hydrostatic part of $\mathbf{D}$,

$$D_H = \frac{1}{3}\mathrm{Tr}\mathbf{D}$$

In Ladeveze's framework (Ladeveze 1983), $\boldsymbol{\sigma}_+^D$ is an unusual positive part of $\boldsymbol{\sigma}^D$: if $\boldsymbol{\lambda}_I$ are the eigenvalues of $\mathbf{H}\boldsymbol{\sigma}^D$ and $\mathbf{T}^I$ the corresponding eigenvectors, $\boldsymbol{\sigma}_+^D$ is

$$\boldsymbol{\sigma}_+^D = \sum_I \langle \boldsymbol{\lambda}_I \rangle_+ \left(\mathbf{H}^{-1} \cdot \mathbf{T}^I\right) \left(\mathbf{H}^{-1} \cdot \mathbf{T}^I\right)^T$$

Constitutive law The constitutive law is derived from Gibbs potential using Eq. 3.43:

$$\begin{aligned}
\boldsymbol{\varepsilon} &= \rho \cdot \frac{\partial \phi^*}{\partial \boldsymbol{\sigma}} \\
&= \frac{1+\nu}{E}\left[\left(\mathbf{H} \cdot \boldsymbol{\sigma}_+^D \mathbf{H}\right)^D + \langle \boldsymbol{\sigma}^D \rangle_-^D\right] + \frac{1-2\nu}{3 \cdot E}\left[\frac{\langle \mathrm{Tr}\left(\boldsymbol{\sigma}\right)\rangle_+}{1-D_H} + \langle \mathrm{Tr}\left(\boldsymbol{\sigma}\right)\rangle_-\right] \cdot \mathbf{I} \\
&= \frac{1+\nu}{E} \cdot \tilde{\boldsymbol{\sigma}} - \frac{\nu}{E} \cdot \mathrm{Tr}\left(\tilde{\boldsymbol{\sigma}}\right) \cdot \mathbf{I}
\end{aligned}$$

where $\tilde{\boldsymbol{\sigma}}$ is the effective stress defined by

$$\tilde{\boldsymbol{\sigma}} = \left[\left(\mathbf{H} \cdot \boldsymbol{\sigma}_+^D \mathbf{H}\right)^D + \langle \boldsymbol{\sigma}^D \rangle_-^D\right] + \left[\frac{\langle \mathrm{Tr}\left(\boldsymbol{\sigma}\right)\rangle_+}{1-D_H} + \langle \mathrm{Tr}\left(\boldsymbol{\sigma}\right)\rangle_-\right] \cdot \mathbf{I}$$

Damage evolution law The damage evolution law should account for cyclic loading. Hence, the damage criterion f should depend on an effective damage d_ε (Ragueneau et al. 2006) such that

$$\begin{aligned}
f &= \hat{\varepsilon} - \kappa\left(d_\varepsilon\right) \\
d_\varepsilon &= \frac{\mathbf{D} : \langle \boldsymbol{\varepsilon} \rangle_+}{\max\left(\boldsymbol{\varepsilon}_I\right)}
\end{aligned}$$

where $\hat{\varepsilon}$ is Mazars strain defined by

$$\hat{\varepsilon} = \sqrt{\langle \boldsymbol{\varepsilon} \rangle_+ : \langle \boldsymbol{\varepsilon} \rangle_+}$$

and κ the consolidation function,

$$\kappa\left(d_\varepsilon\right) = a \cdot \tan\left[\frac{d_\varepsilon}{aA} + \arctan\left(\frac{\kappa_0}{a}\right)\right]$$

where A and a are two damage coefficients, and κ_0 is initial elasticity threshold.

Thus, if the damage criterion is negative, the material is in the linear elastic range, and if it is positive, the damage increases along the positive strain:

$$\dot{\mathbf{D}} = \dot{\lambda} \langle \boldsymbol{\varepsilon} \rangle_+$$

where the damage Lagrange multiplier $\dot{\lambda}$ is determined from the Kuhn-Tucker condition ($f = 0$ and $f' = 0$).

3.2.2.2 Uniaxial multi-fiber formulation

Multi-fiber formulation based on Euler-Bernoulli beam theory is equivalent to a uniaxial loading. Hence, the stress tensor and the strain tensor are reduced to

$$\boldsymbol{\sigma} = \begin{pmatrix} \sigma_{11} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

and,

$$\boldsymbol{\varepsilon} = \begin{pmatrix} \varepsilon_{11} & 0 & 0 \\ 0 & \varepsilon_{22} & 0 \\ 0 & 0 & \varepsilon_{33} \end{pmatrix}$$

The damage tensor becomes diagonal and the constitutive law can be rewritten as (Desmorat, Ragueneau, Souid and Delaplace 2008),

$$\boldsymbol{\varepsilon} = \mathbf{B}\left(D_1, D_2\right) \cdot \frac{\boldsymbol{\sigma}}{E}$$

where $\mathbf{B}$ is a diagonal tensor which depends on the damage variable and loading, and D_1 and D_2 are associated with tension and compression damage respectively.

For tension, we assume that only D_1 increases.

$$\begin{aligned} B_{11}^t &= \frac{1+\nu}{9} \cdot \left(\frac{4}{1-D_1} + 2\right) + \frac{1-2\nu}{3 \cdot \left[1 - \frac{\eta_t}{3} \cdot (D_1 + 2 \cdot D_2)\right]} \\ B_{22}^t = B_{33}^t &= \frac{1+\nu}{9} \cdot \left(\frac{2}{1-D_1} + 1\right) + \frac{1-2\nu}{3 \cdot \left[1 - \frac{\eta_t}{3} \cdot (D_1 + 2 \cdot D_2)\right]} \end{aligned}$$

For compression, we assume that only D_2 increases.

$$\begin{aligned} B_{11}^c &= \frac{1+\nu}{9} \cdot \left(4 + \frac{2}{1-D_2}\right) + \frac{1-2\nu}{3} \\ B_{22}^c = B_{33}^c &= -\frac{1+\nu}{9} \cdot \left(2 + \frac{1}{1-D_2}\right) + \frac{1-2\nu}{3} \end{aligned} \tag{3.44}$$

3.2.2.3 Determination for anisotropic damage model

3.2.2.3.1 Tension loading: $\varepsilon_{11} > 0$

- Constitutive equations

  If $f = \hat{\varepsilon} - \kappa_0 \geq 0$ and $\eta_t = 1$ then:

$$\begin{aligned} \sigma &= \frac{E}{\frac{1+\nu}{9} \cdot \left(\frac{4}{1-D_1} + 2\right) + \frac{1-2\nu}{3\left(1 - \frac{D_1 + 2D_2}{3}\right)}} \cdot \varepsilon \\ D_1 &= a \cdot A \left(\texttt{atan}\left(\frac{\varepsilon_{11}}{a}\right) - \texttt{atan}\left(\frac{\kappa_0}{a}\right)\right) \end{aligned} \tag{3.45}$$

where, $\hat{\varepsilon} = \sqrt{\langle\boldsymbol{\varepsilon}\rangle_+ : \langle\boldsymbol{\varepsilon}\rangle_+} = \varepsilon_{11} \geq 0$,

$$\langle\boldsymbol{\varepsilon}\rangle_+ = \begin{pmatrix} \varepsilon_{11} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

D_2 is constant and tangent modulus is determined with Eq. 3.45:

$$\dot{\sigma} = E^{tan} \cdot \dot{\varepsilon} \tag{3.46}$$

However, if $f = \hat{\varepsilon} - \kappa_0 < 0$, the tangent modulus is equal to the secant one.

$$E^{tan} = \frac{E}{B^t_{11}}$$

- Coherent tangent stiffness modulus in numerical implementation

  If $f = \hat{\varepsilon} - \kappa_0 \geq 0$ and $\eta_t = 1$, tangent modulus is determined with Eq. 3.45 and 3.46:

$$\begin{aligned} E^{tan}_n &= \frac{\partial \sigma_n}{\partial \varepsilon_n} \\ B^t_{11,n} &= \frac{1+\nu}{9} \cdot \left(\frac{4}{1 - D_{1,n}} + 2 \right) + \frac{1 - 2\nu}{3 \cdot \left(1 - \frac{(D_{1,n} + 2 \cdot D_{2,n-1})}{3}\right)} \\ D_{1,n} &= a \cdot A \left(\mathtt{atan}\left(\frac{\varepsilon_{11,n}}{a}\right) - \mathtt{atan}\left(\frac{\kappa_0}{a}\right)\right) \end{aligned}$$

  where n is current excitation step.

$$\frac{\partial \sigma_n}{\partial \varepsilon_n} = \frac{\partial}{\partial \varepsilon_n}\left(\frac{E}{B^t_{11,n}} \cdot \varepsilon_n\right) = \frac{E}{B^t_{11,n}} - E \cdot \varepsilon_n \frac{1}{(B^t_{11,n})^2} \cdot \frac{\partial B^t_{11,n}}{\partial \varepsilon_n}$$

  where,

$$\begin{aligned} \frac{\partial B^t_{11,n}}{\partial \varepsilon_n} &= \frac{\partial B^t_{11,n}}{\partial D_{1,n}} \cdot \frac{\partial D_{1,n}}{\partial \varepsilon_n} \\ C_1 = \frac{\partial B^t_{11,n}}{\partial D_{1,n}} &= \frac{1+\nu}{9} \cdot \frac{4}{(1 - D_{1,n})^2} + \frac{1 - 2\nu}{9\left(1 - \frac{D_{1,n} + 2D_{2,n-1}}{3}\right)} \\ C_2 = \frac{\partial D_{1,n}}{\partial \varepsilon_n} &= \frac{\partial}{\partial \varepsilon_n}\left(a \cdot A \left(\mathtt{atan}\left(\frac{\varepsilon_n}{a}\right) - \mathtt{atan}\left(\frac{\kappa_0}{a}\right)\right)\right) = \frac{A}{1 + \left(\frac{\varepsilon_n}{a}\right)^2} \end{aligned}$$

Finally, we obtain tangent modulus:

$$E_n^{tan} = \frac{E}{B_{11,n}^t} - E \cdot \varepsilon_n \cdot \frac{1}{(B_{11,n}^t)^2} \cdot C_1 \cdot C_2$$

3.2.2.3.2 Compressive Tension loading: $\varepsilon_{11} < 0$

- Constitutive equations

  If $f = \hat{\varepsilon} - \kappa_0 \geq 0$ and $\eta_t = 1$ then:

$$\begin{aligned} \sigma &= \frac{E}{\frac{1+\nu}{9} \cdot \left(\frac{2}{1-D_2} + 4\right) + \frac{1-2\nu}{3}} \cdot \varepsilon \\ D_2 &= \frac{a \cdot A}{2} \left(\texttt{atan}\left(\frac{\sqrt{2} \cdot \varepsilon_{22}}{a}\right) - \texttt{atan}\left(\frac{\kappa_0}{a}\right) \right) \\ \varepsilon_{22} &= \frac{B_{22}^c}{B_{11}^c} \cdot \varepsilon_{11} \end{aligned} \tag{3.47}$$

  where, $\hat{\varepsilon} = \sqrt{\langle \boldsymbol{\varepsilon} \rangle_+ : \langle \boldsymbol{\varepsilon} \rangle_+} = \sqrt{2 \cdot \varepsilon_{22}^2} \geq 0$,

$$\langle \boldsymbol{\varepsilon} \rangle_+ = \begin{pmatrix} 0 & 0 & 0 \\ 0 & \varepsilon_{22} & 0 \\ 0 & 0 & \varepsilon_{33} = \varepsilon_{22} \end{pmatrix}$$

  D_1 is constant and tangent modulus is determined with Eq. 3.47:

$$\dot{\sigma} = E^{tan} \cdot \dot{\varepsilon} \tag{3.48}$$

  However, if $f = \hat{\varepsilon} - \kappa_0 < 0$, the tangent modulus is equal to the secant one.

$$E^{tan} = \frac{E}{B_{11}^c}$$

- Coherent tangent stiffness modulus in numerical implementation

  If $f = \hat{\varepsilon} - \kappa_0 \geq 0$ and $\eta_t = 1$, tangent modulus is determined with Eq. 3.47 and 3.48:

$$
\begin{aligned}
E_n^{tan} &= \frac{\partial \sigma_n}{\partial \varepsilon_n} \\
B_{11,n-1}^c &= \frac{1+\nu}{9} \cdot \left(4 + \frac{2}{1-D_{2,n-1}}\right) + \frac{1-2\nu}{3} \\
B_{22,n-1}^c &= -\frac{1+\nu}{9} \cdot \left(2 + \frac{1}{1-D_{2,n-1}}\right) + \frac{1-2\nu}{3} \\
B_{11,n}^c &= \frac{1+\nu}{9} \cdot \left(4 + \frac{2}{1-D_{2,n}}\right) + \frac{1-2\nu}{3} \\
B_{22,n}^c &= -\frac{1+\nu}{9} \cdot \left(2 + \frac{1}{1-D_{2,n}}\right) + \frac{1-2\nu}{3}
\end{aligned}
$$

where n is current excitation step.

$$
\begin{aligned}
\tilde{\nu}_{n-1} = -\frac{B_{22,n-1}^c}{B_{11,n-1}^c}, \quad \varepsilon_{22,n}^{tr} = -\tilde{\nu}_{n-1} \cdot \varepsilon_{11,n} \\
\tilde{\nu}_{n} = -\frac{B_{22,n}^c}{B_{11,n}^c}, \quad \varepsilon_{22,n} = -\tilde{\nu}_{n} \cdot \varepsilon_{11,n}
\end{aligned}
$$

where, $\varepsilon_{22,n}^{tr}$ is trial strain on ε_{22}.

$$
\frac{\partial \sigma_n}{\partial \varepsilon_n} = \frac{\partial}{\partial \varepsilon_n}\left(\frac{E}{B_{11,n}^c} \cdot \varepsilon_n\right) = \frac{E}{B_{11,n}^c} - E \cdot \varepsilon_n \frac{1}{(B_{11,n}^c)^2} \cdot \frac{\partial B_{11,n}^c}{\partial \varepsilon_n}
$$

where,

$$
\begin{aligned}
\frac{\partial B_{11,n}^c}{\partial \varepsilon_n} &= \frac{\partial B_{11,n}^c}{\partial D_{2,n}} \cdot \frac{\partial D_{2,n}}{\partial \varepsilon_n} \\
C_1 = \frac{\partial B_{11,n}^c}{\partial D_{2,n}} &= \frac{2(1+\nu)}{9(1-D_{2,n})^2} \\
C_2 = \frac{\partial D_{2,n}}{\partial \varepsilon_n} &= \frac{\partial}{\partial \varepsilon_n}\left(a \cdot A\left(\texttt{atan}\left(\frac{-\tilde{\nu}_{n-1} \cdot \sqrt{2} \cdot \varepsilon_{11,n}}{a}\right) - \texttt{atan}\left(\frac{\kappa_0}{a}\right)\right)\right) \\
&= -\tilde{\nu}_{n-1}\frac{A}{\sqrt{2}} \cdot \frac{1}{1 + \left(\frac{-\tilde{\nu}_{n-1} \cdot \sqrt{2} \cdot \varepsilon_{11,n}}{a}\right)^2}
\end{aligned}
$$

Finally, we obtain tangent modulus:

$$
E_n^{tan} = \frac{E}{B_{11,n}^c} - E \cdot \varepsilon_n \cdot \frac{1}{(B_{11,n}^c)^2} \cdot C_1 \cdot C_2
$$

Fig 3.22 to 3.24 shows the procedure to obtain uniaxial stress σ and tangent modulus E^{tan} from uniaxial strain ε.

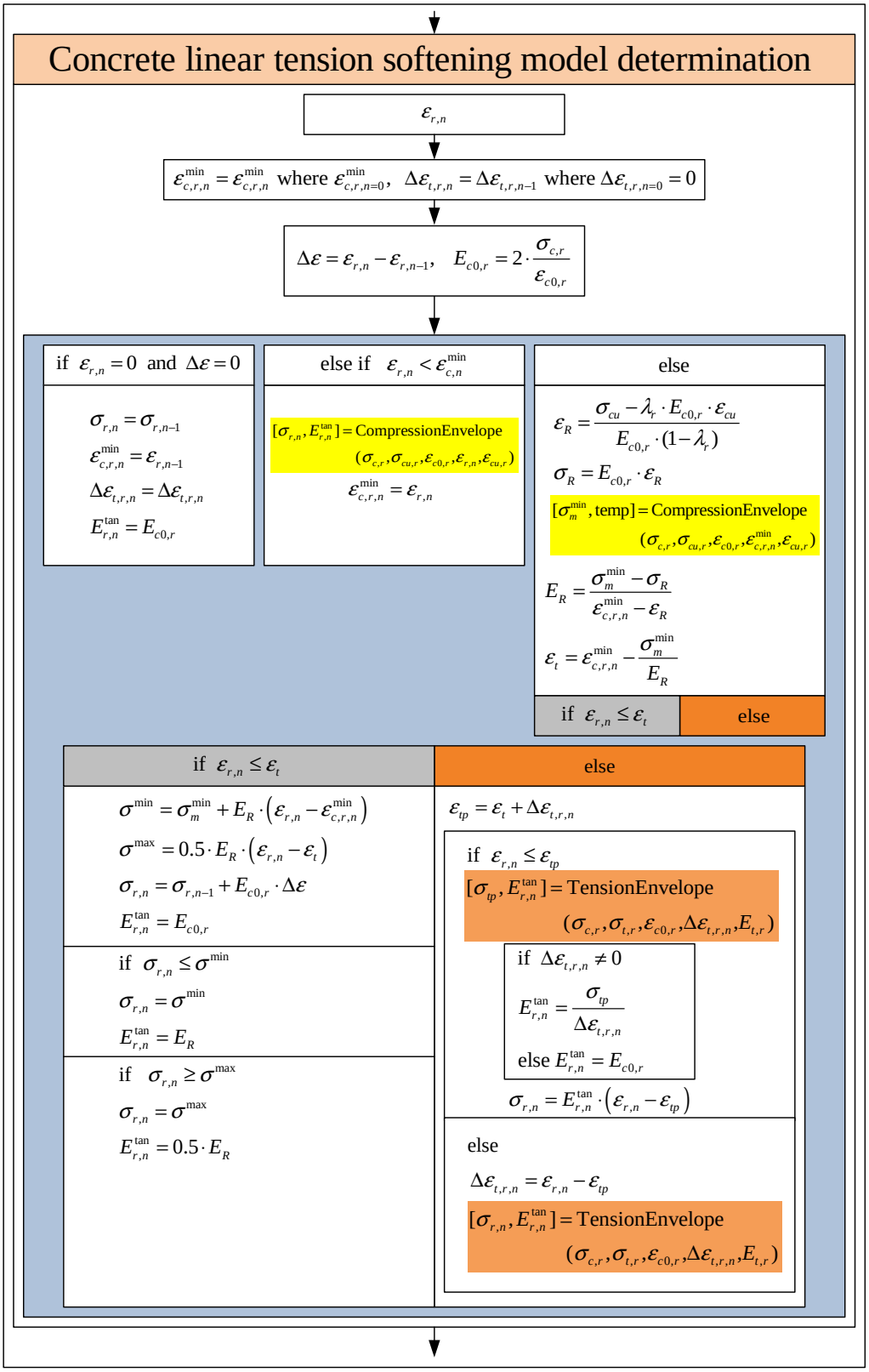


Fig. 3.19 Determination (1) for modified Kent and Park model

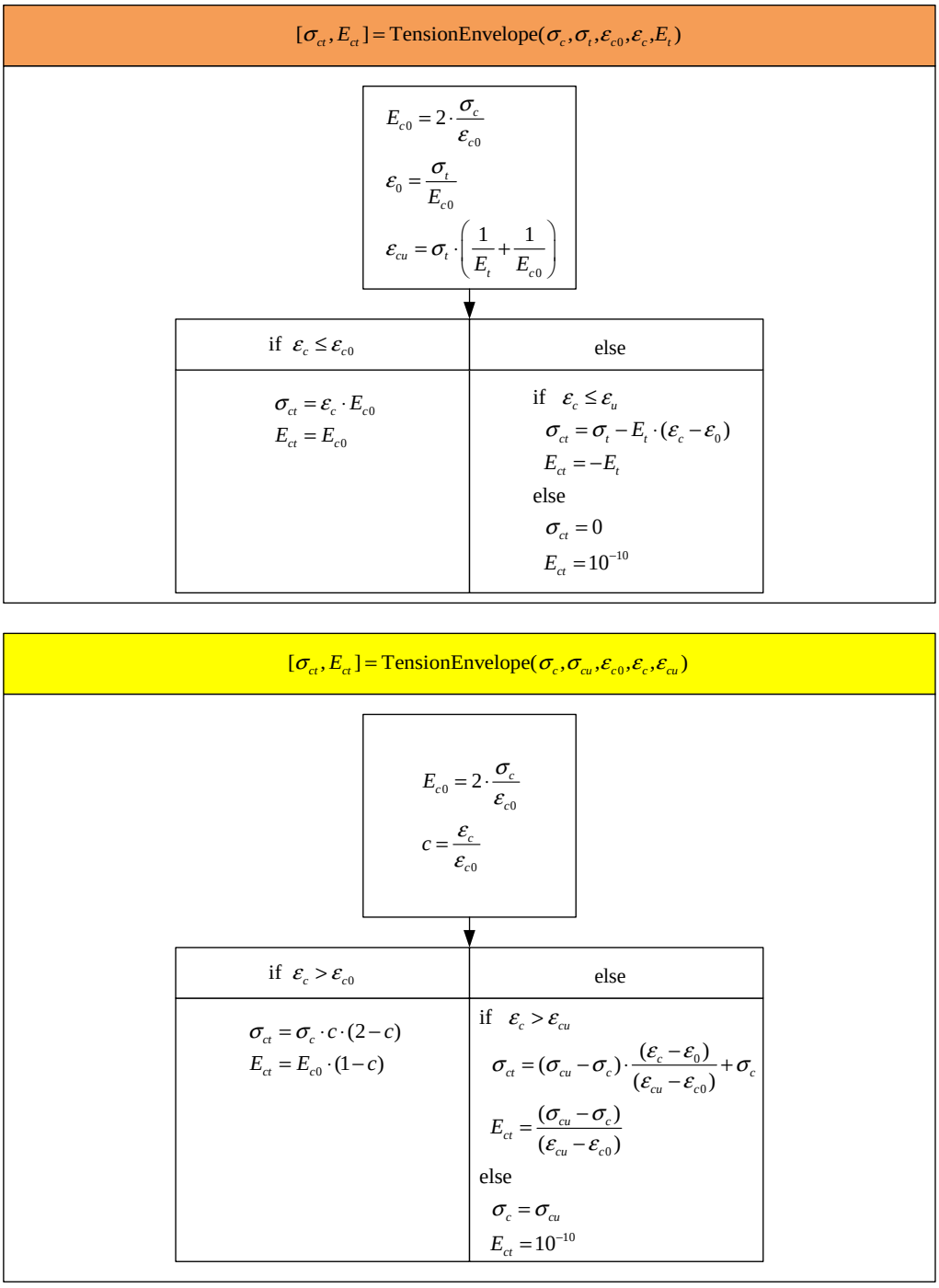


Fig. 3.20 Determination (2) for modified Kent and Park model

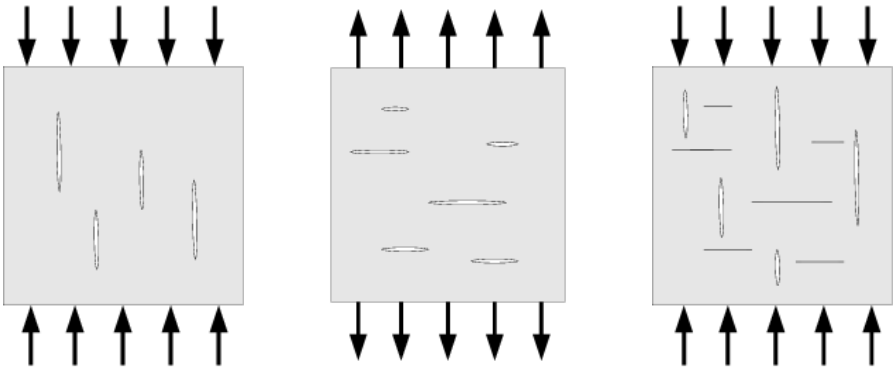


Fig. 3.21 Cracks induced by compression, tension, tension and compression

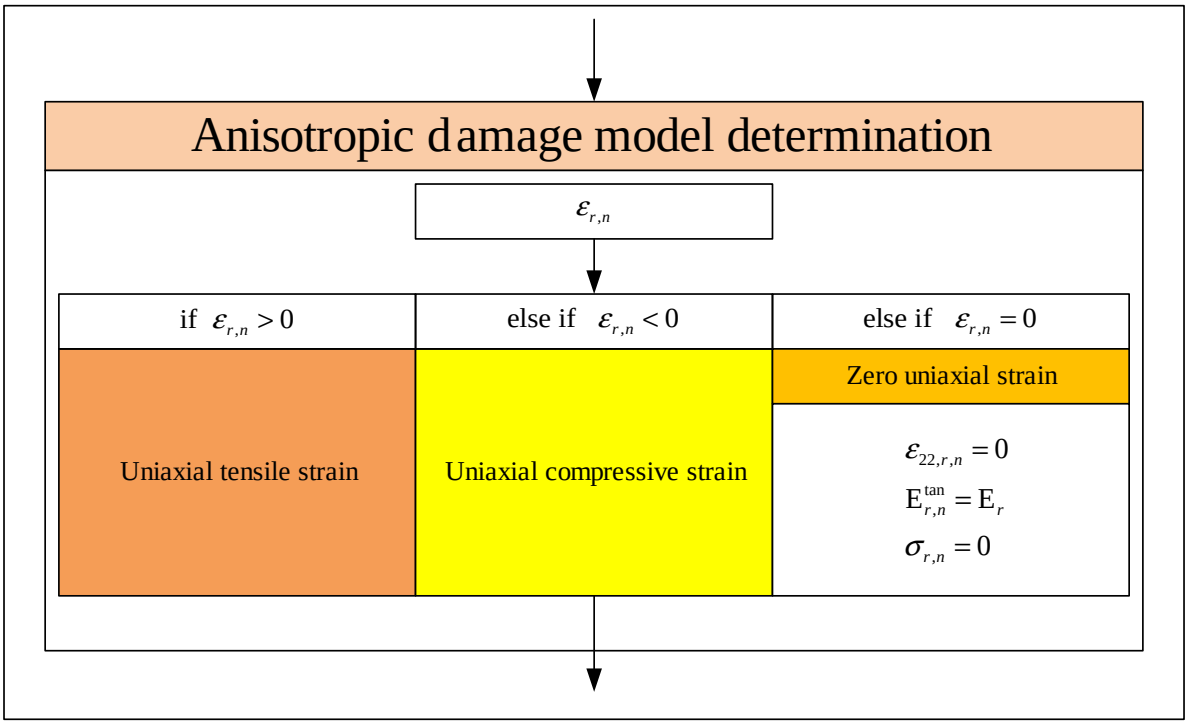


Fig. 3.22 Determination (1) for the anisotropic damage model

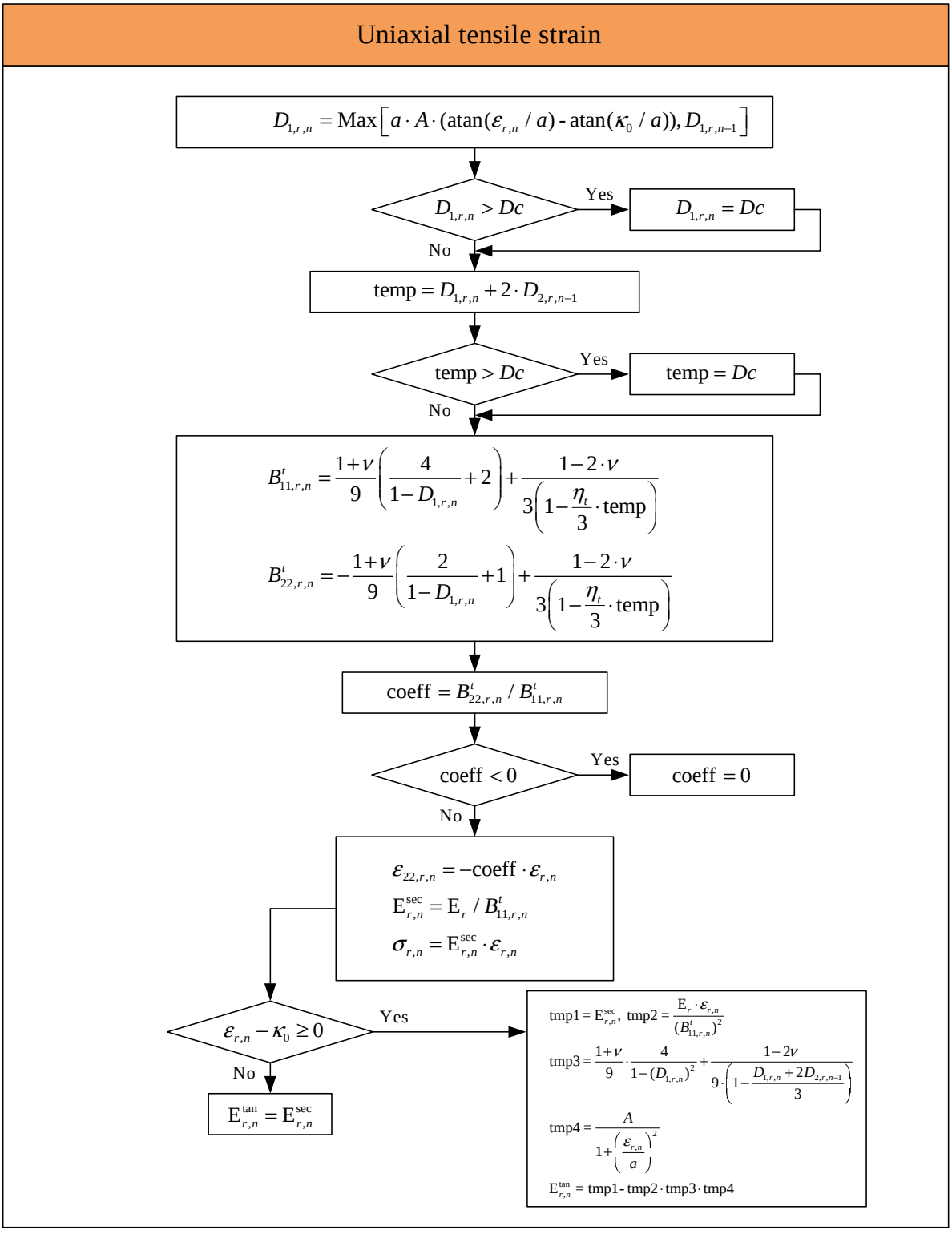


Fig. 3.23 Determination (2) for the anisotropic damage model

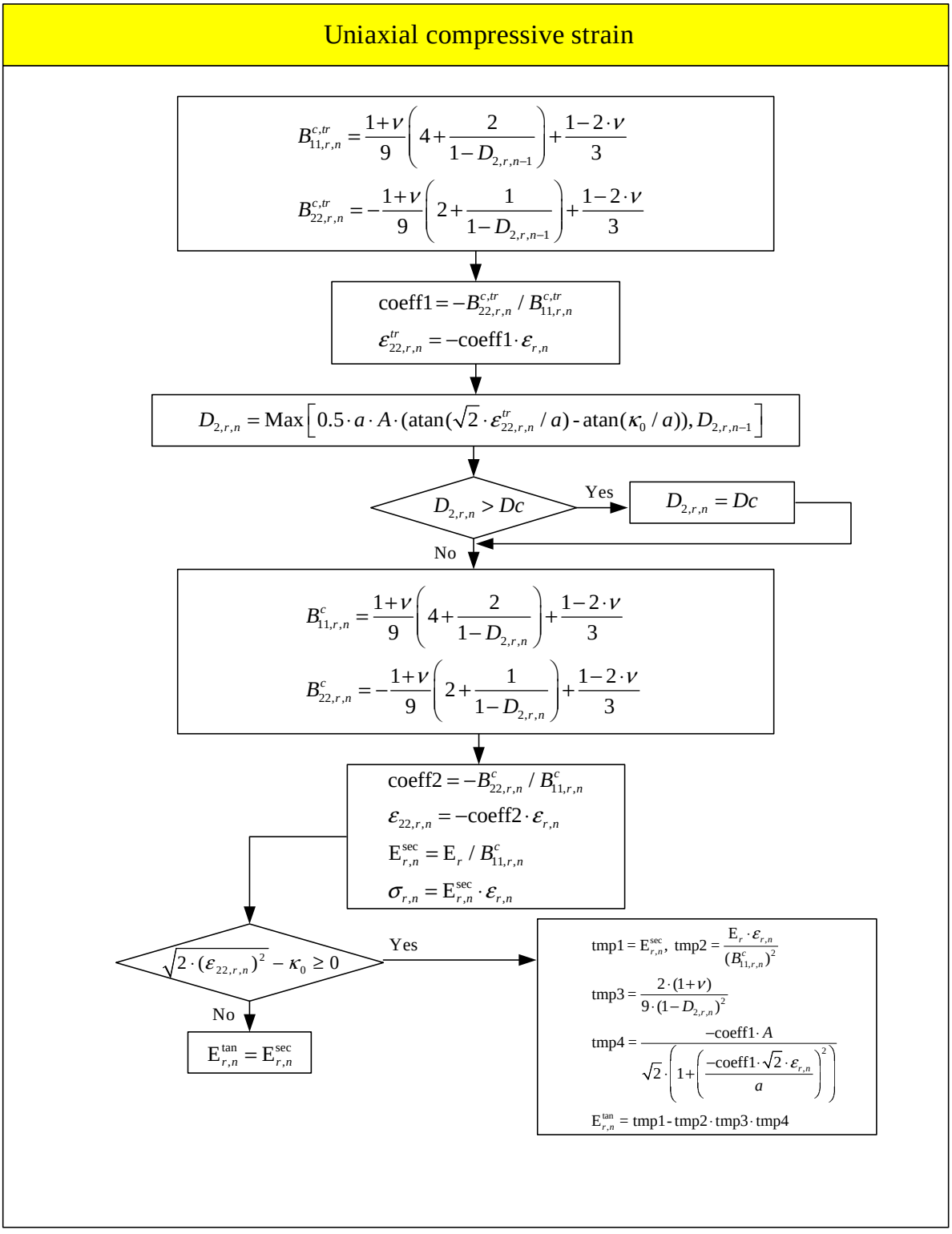


Fig. 3.24 Determination (3) for the anisotropic damage model

Chapter 4

NONLINEAR SOLUTION FOR ELLIPTIC PROBLEMS

This chapter describes the nonlinear solver for static analysis in Mercury.

4.1 Force And Displacement Control

The governing equilibrium equation of a nonlinear system can be expressed as

$$\underbrace{\left\{ \begin{array}{c} \mathbf{P}_t^{ext} \\ \mathbf{P}_u^{ext} \end{array} \right\} + \left\{ \begin{array}{c} \mathbf{P}_t^{fea} \\ \mathbf{P}_u^{fea} \end{array} \right\}}_{\mathbf{P}_S} = \underbrace{\left[\begin{array}{cc} \mathbf{K}_{tt} & \mathbf{K}_{tu} \\ \mathbf{K}_{ut} & \mathbf{K}_{uu} \end{array} \right]}_{\mathbf{K}_S} \underbrace{\left\{ \begin{array}{c} \mathbf{u}_t \\ \mathbf{u}_u \end{array} \right\}}_{\mathbf{u}_S} \tag{4.1}$$

where $\mathbf{P}$, $\mathbf{u}$ and $\mathbf{K}$ are the nodal force, nodal displacement and stiffness matrix expressed in the structural level, and subscripts 'u', 't' and 'S' refer to known displacement or displacement, known force, and structural level respectively, superscripts 'ext' and 'fea' refer to external and fixed end action caused by element distributed load, Fig 2.32. Equilibrium is satisfied when external nodal forces are equal to the internal ones,

$$\mathbf{P}_S^{ext} - \mathbf{P}_S^{int} = 0 \tag{4.2}$$

where $\mathbf{P}_S^{int}$ is determined from state determination in chapter 2.

For nonlinear system, the incremental displacement vector between the previous step $n-1$ and the current one n is

$$\Delta \mathbf{u}_{S,n} = \mathbf{u}_{S,n} - \mathbf{u}_{S,n-1}$$

During iteration, the nodal displacement vector at the k^{th} iteration is given by:

$$\mathbf{u}_{S,n}^k = \mathbf{u}_{S,n}^{k-1} + \delta \mathbf{u}_{S,n}^k$$

where $\delta \mathbf{u}_{S,n}^{k} = \mathbf{u}_{S,n}^{k} - \mathbf{u}_{S,n}^{k-1}$

We can rewrite Eq. 4.1 in incremental form for the k^{th} iteration of force or displacement control within a Newton-Raphson iteration:

$$\underbrace{\left\{ \begin{array}{l} \mathbf{P}_{t,n}^{R,k} = \mathbf{P}_{t,n}^{ext\checkmark} + \mathbf{P}_{t,n}^{fea\checkmark} - \mathbf{P}_{t,n}^{int,k\checkmark} \\ \mathbf{P}_{u,n}^{R,k} = \mathbf{P}_{u,n}^{ext,k?} + \mathbf{P}_{u,n}^{fea\checkmark} - \mathbf{P}_{u,n}^{int,k\checkmark} \end{array} \right\}}_{\mathbf{P}_{S,n}^{R,k} = \delta \mathbf{P}_{S,n}^{k}} = \underbrace{\left[\begin{array}{cc} \mathbf{K}_{tt,n}^{k-1} & \mathbf{K}_{tu,n}^{k-1} \\ \mathbf{K}_{ut,n}^{k-1} & \mathbf{K}_{uu,n}^{k-1} \end{array} \right]}_{\mathbf{K}_{S,n}^{k-1}} \underbrace{\left\{ \begin{array}{c} \delta \mathbf{u}_{t,n}^{k\ ?} \\ \delta \mathbf{u}_{u,n}{}^{\checkmark} \end{array} \right\}}_{\delta \mathbf{u}_{S,n}^{k}} \tag{4.3}$$

where, superscript '✓' and '?' refer to known and unknown quantities.

For convenience, we will consider the vector of external forces to also include the contribution of the element load through the nodal fixed end actions:

$$\mathbf{P}_{t,n}^{ext\checkmark} = \mathbf{P}_{t,n}^{ext\checkmark} + \mathbf{P}_{t,n}^{fea\checkmark}$$
$$\mathbf{P}_{u,n}^{ext,k?} = \mathbf{P}_{u,n}^{ext,k?} + \mathbf{P}_{u,n}^{fea\checkmark}$$

,and thus the incremental displacements can be expressed as:

$$\delta \mathbf{u}_{t,n}^{k\ ?} = \mathbf{u}_{t,n}^{k\ ?} - \mathbf{u}_{t,n}^{k-1\checkmark}$$
$$\delta \mathbf{u}_{u,n}{}^{\checkmark} = \mathbf{u}_{u,n}{}^{\checkmark} - \mathbf{u}_{u,n-1}{}^{\checkmark}$$

The essence of the finite element analysis is to first determine the displacement $\delta \mathbf{u}_{t,n}^{k\ ?}$, and then $\mathbf{P}_{u,n}^{ext,k?}$ from Eq. 4.3.

4.2 Nolinear Solution

The fundamental equation to be solved in nonlinear analysis is Eq. 4.2 which is rewritten in incremental form at step n (Bathe 1996),

$$\mathbf{P}_{t,n}^{R} = \mathbf{P}_{t,n}^{ext} - \mathbf{P}_{t,n}^{int} = 0 \tag{4.4}$$

where superscript R refers to residual.

Both $\mathbf{P}_{t,n}^{ext}$ and $\mathbf{P}_{t,n}^{int}$ are determined from the principle of virtual displacement. Since the nodal force vector $\mathbf{P}_{t,n}^{ext}$ is a function of the nodal displacement vector $\mathbf{u}_{t,n}$ and the problem is nonlinear, an iterative approach is followed to solve Eq. 4.4. Assuming that forces are independent of the displacements (conservative system and ignoring $P-\Delta$ effects), we solve Eq. 4.4 by the Newton-Raphson method which will be described in more detail later.

Linearization of the system yields

$$\mathbf{P}_{t,n}^{R,k} = \mathbf{P}_{t,n}^{ext} - \mathbf{P}_{t,n}^{int,k} \tag{4.5}$$

$$\delta\mathbf{u}_{t,n}^{k} = [\mathbf{K}_{tt,n}^{k-1}]^{-1} \cdot \mathbf{P}_{t,n}^{R,k} \tag{4.6}$$

$$\mathbf{u}_{t,n}^{k} = \mathbf{u}_{t,n}^{k-1} + \delta\mathbf{u}_{t,n}^{k} \tag{4.7}$$

where,

$$\mathbf{u}_{t,n}^{k=0} = \mathbf{u}_{t,n-1}$$
$$\mathbf{P}_{t,n}^{int,k=0} = \mathbf{P}_{t,n-1}^{int}$$

These equations are obtained by linearizing the structural response of structure at the current step n and k^{th} iteration. Within each iteration we determine the residual nodal force vector, Eq. 4.5, that yields an incremental nodal displacement vector obtained from Eq. 4.6, and we continue the iteration until the residual nodal force vector or the incremental nodal displacement vector is sufficiently small.

We will next discuss several iterative methods in more detail. At the heart of all these, solution is the determination of the internal nodal force vector $\mathbf{P}_{t,n}^{int,k}$, and the tangent stiffness matrix $\mathbf{K}_{tt,n}^{k-1}$. Solution of Eq. 4.6 will be described in chapter. 2. Hence, we next focus on the iterative methods to solve Eq. 4.5 to 4.7.

4.2.1 Newton-Raphson iterative method

The most frequently used iterative method for nonlinear problems is the Newton-Raphson iterative method given as expressed by Eq. 4.5 to 4.7. Let us derive the procedure in a more formal manner.

The fundamental equilibrium equation to be satisfied is

$$\mathbf{f}(\mathbf{u}^*) = \mathbf{P}_{t,n}^{ext}(\mathbf{u}^*) - \mathbf{P}_{t,n}^{int}(\mathbf{u}^*) = 0 \tag{4.8}$$

where, $\mathbf{f}(\cdot)$ is the function of internal state value $(\cdot)$. In the preceding equation it is often, but not exclusively, the vector of nodal displacement $\mathbf{u}$, which is why we express it by $\mathbf{u}^*$ as other variables may be state present.

Assuming that $\mathbf{u}_{t,n}^{k-1}$ is known, then a Taylor series expansion gives

$$\mathbf{f}(\mathbf{u}^*) = f(\mathbf{u}_{t,n}^{k-1}) + \frac{\partial \mathbf{f}}{\partial \mathbf{u}}|_{\mathbf{u}_{t,n}^{k-1}} \cdot (\mathbf{u}^* - \mathbf{u}_{t,n}^{k-1}) + \text{High-order terms} \tag{4.9}$$

Substituting Eq. 4.8 into Eq. 4.9, we obtain

$$\frac{\partial \mathbf{P}_{t}^{int}}{\partial \mathbf{u}}|_{\mathbf{u}_{t,n}^{k-1}} \cdot (\mathbf{u}^* - \mathbf{u}_{t,n}^{k-1}) + \text{High-order terms} = \mathbf{P}_{t,n}^{ext} - \mathbf{P}_{t,n}^{int,k-1} = \mathbf{P}_{t,n}^{R,k} \tag{4.10}$$

where we assume that the external nodal forces are again displacement-independent.

Neglecting the high-order terms in Eq. 4.10, we can determine the incremental displacement,

$$\delta \mathbf{u}_{t,n}^{k} = [\mathbf{K}_{tt,n}^{k-1}]^{-1} \cdot \mathbf{P}_{t,n}^{R,k} \tag{4.11}$$

where $\mathbf{K}_{tt,n}^{k-1}$ is current tangent stiffness matrix

$$\mathbf{K}_{tt,n}^{k-1} = \frac{\partial \mathbf{P}_{t}^{int}}{\partial \mathbf{u}}|_{\mathbf{u}_{t,n}^{k-1}} \tag{4.12}$$

and the improved displacement solution is given by

$$\mathbf{u}_{t,n}^{k} = \mathbf{u}_{t,n}^{k-1} + \delta \mathbf{u}_{t,n}^{k} \tag{4.13}$$

Equations 4.11 and 4.13 constitute the Newton-Raphson solution of Eq. 4.4. Since an incremental analysis is performed with external force steps (or time steps Δt), the initial conditions in this iterative method are

$$\begin{aligned} \mathbf{K}_{tt,n}^{k=0} &= \mathbf{K}_{tt,n-1} \\ \mathbf{u}_{t,n}^{k=0} &= \mathbf{u}_{t,n-1} \\ \mathbf{P}_{t,n}^{int,k=0} &= \mathbf{P}_{t,n-1}^{int} \end{aligned}$$

The iterations proceed until an appropriate convergence criteria is satisfied.

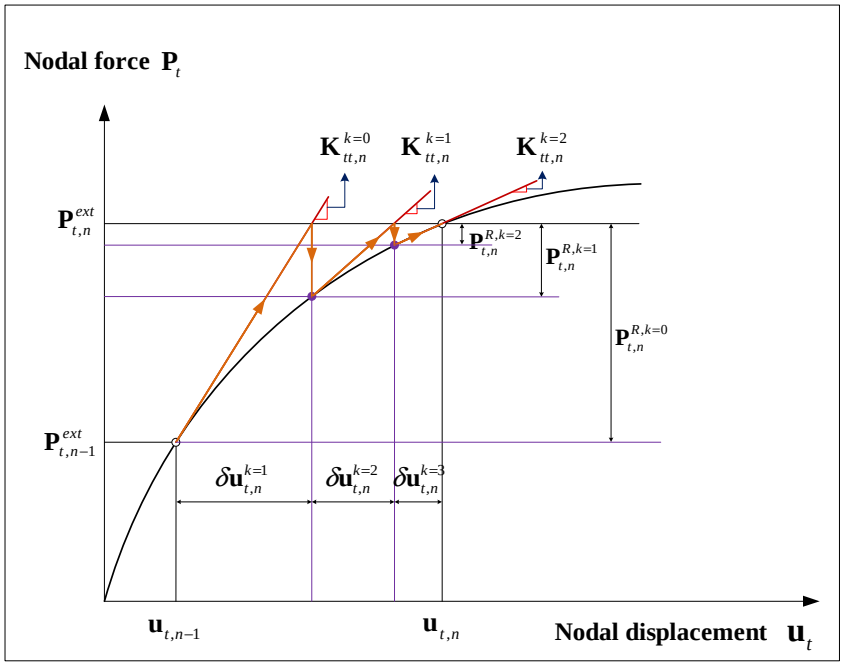


Fig. 4.1 Illustration of Newton-Raphson iterative method (Bathe 1996)

A characteristic of this iterative method is that a new tangent stiffness matrix is determined

in each iteration, which is why this method is also referred to as the full Newton-Raphson iterative method. Fig. 4.1 illustrates the process of solution when used for a single degree of freedom system. The nonlinear response characteristics are such that convergence is rapidly obtained. However, we can imagine a more complex response characteristic with a starting point of iteration for which the procedure does not converge. Thus, the representation in Fig. 4.1 is rather simplistic because a very special case is considered - that of a well-behaved single degree of freedom system. In the solution of systems with many degrees of freedom (and possibly including concrete softening), the response curves will in general be rather non-smooth and complex.

4.2.2 Initial stiffness iterative method

Considering the Newton-Raphson iterative method it is recognized that in general the major computational cost per iteration lies in the calculation and factorization of the tangent stiffness matrix. Since these calculations can be quite expensive for large systems, a modificaion of the Newton-Raphson algorithm can be effective. First we will consider the initial stiffness iterative method

This method always uses the initial stiffness matrix $\mathbf{K}_{tt}$ in Eq. 4.11. Therefore,

$$\delta \mathbf{u}_{t,n}^{k} = [\mathbf{K}_{tt}]^{-1} \cdot \mathbf{P}_{t,n}^{R,k} \tag{4.14}$$

with the initial conditions

$$\begin{aligned} \mathbf{u}_{t,n}^{k=0} &= \mathbf{u}_{t,n-1} \\ \mathbf{P}_{t,n}^{int,k=0} &= \mathbf{P}_{t,n-1}^{int} \end{aligned}$$

As such only $\mathbf{K}_{tt,n=0}^{k=0}$ needs to be factorized, thus avoiding the expense of recalculating and factorizing many times the tangent stiffness matrix is Eq. 4.11. This initial stiffness iterative method corresponds to a linearization of the response about the initial configuration of the finite element system and will converge very slowly and may even diverge. It is illustrated in Fig. 4.2 for a single degree of freedom system.

4.2.3 Modified Newton-Raphson iterative mathod

In the modified Newton-Raphson iterative method an approach somewhat in between Newton-Raphson iterative method and the initial stiffness iterative method is adopted. In this iterative method we use

$$\delta \mathbf{u}_{t,n}^{k} = [\mathbf{K}_{tt,n-1}]^{-1} \cdot \mathbf{P}_{t,n}^{R,k} \tag{4.15}$$

with the initial conditions

$$\begin{aligned} \mathbf{u}_{t,n}^{k=0} &= \mathbf{u}_{t,n-1} \\ \mathbf{P}_{t,n}^{int,k=0} &= \mathbf{P}_{t,n-1}^{int} \end{aligned}$$

The modified Newton-Raphson iterative method involves fewer stiffness decompositions than the Newton-Raphson iterative method and bases the stiffness matrix update on an accepted

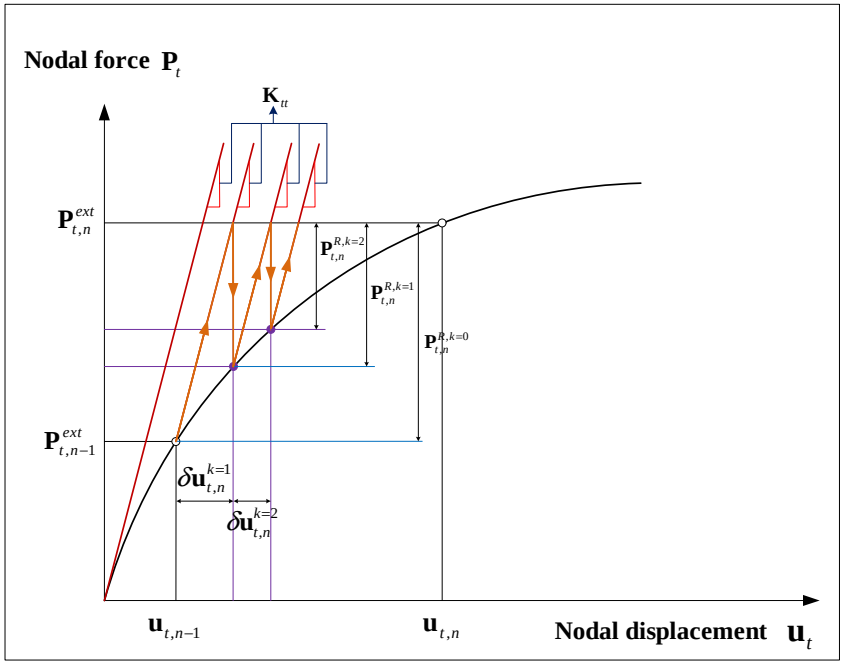


Fig. 4.2 Illustration of initial stiffness iterative method (Bathe 1996)

equilibrium configuration. The choice of external force steps or time steps when the stiffness matrix should be updated depends on the degree of nonlinearity in the system response; i.e. the more nonlinear the response, the more often the updating should be performed. Fig. 4.3 illustrates the performance of the modified Newton-Raphson iterative method for a single degree of freedom system.

4.3 Convergence criteria

An appropriate termination criteria of the iteration should be adopted for any incremental solution strategy based on iterative methods. At the end of each iteration, the solution obtained should be checked to see whether it has converged within defined tolerances or whether the iteration is diverging. If the convergence tolerances are too loose, inaccurate results are obtained, and if the tolerances are too tight, much computational effort is spent to obtain needless accuracy. Similarly, an ineffective divergence check can terminate the iteration when the solution is not actually diverging or force the iteration to search for an unattainable solution. Here now describe briefly some convergence criteria.

Displacement criteria The residual nodal displacement vector $\delta\mathbf{u}^k_n$ at the end of iteration should be within a certain tolerance. Hence, a realistic convergence criterion is

$$\|\delta\mathbf{u}^k_n\| < \epsilon_D \tag{4.16}$$

where ϵ_D is a displacement convergence tolerance and $\|\cdot\|$ is the Euclidian norm defined as the square root of the sum of the vector components squared.

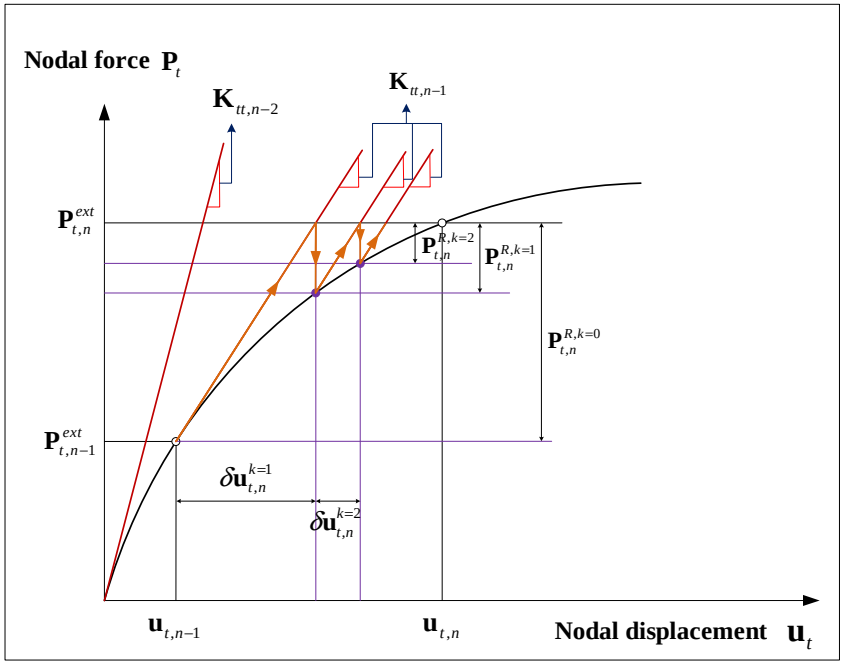


Fig. 4.3 Illustration of modified Newton-Raphson iterative method (Bathe 1996)

Force criteria This convergence criterion is obtained by determining the Euclidian norm of the residual nodal force vector $\mathbf{P}^{R,k}_{t,n}$ and:

$$\|\mathbf{P}^{R,k}_{t,n}\| < \epsilon_F \tag{4.17}$$

where ϵ_F is a force convergence tolerance.

Energy criteria A difficulty with the force criterion is that the displacement solution does not introduce the termination criterion. As an illustration, consider an elasto-plastic truss with a very small strain-hardening modulus entering the plastic region. In this case, the residual force vector may be very small while the displacements may still be much in error. Hence, the convergence criteria expressed by Eq. 4.16 and 4.17 may haver to be used with very small values of ϵ_D and ϵ_F. Also, the expressions must be modified appropriately when quantities of different units are measured.

In order to provide some indication of when both the displacements and the forces are near their equilibrium values, the energy criteria can be used. It is expressed as:

$$\left|\frac{1}{2}\cdot\mathbf{P}^{R,k}_{t,n}\cdot\delta\mathbf{u}^{k}_{n}\right| < \epsilon_E \tag{4.18}$$

where ϵ_E is an energy convergence tolerance.

Since Eq. 4.18 contains both the displacements and the forces, it is in practice an attractive measure.

An important point is that the convergence tolerance ϵ_D, ϵ_F and ϵ_E may need to be quite small in some solutions in order to reach a good solution accuracy.

4.4 The spherical arc-length method

Step 1 Residual nodal force vector

$$\begin{aligned}\Delta\mathbf{P}_{t,n} &= \mathbf{P}^{ext}_{t,n} - \mathbf{P}^{ext}_{t,n-1}\\ \mathbf{P}^{R,i=1}_{t} &= \Delta\mathbf{P}_{t,n}\\ \mathbf{P}^{R,i}_{t} &= \mathbf{P}^{int,i}_{t,n} - \lambda\cdot\mathbf{P}^{ext}_{t,n}\end{aligned}$$

where, $\Delta\mathbf{P}_t$, $\mathbf{P}^{ext}_t$ and $\mathbf{P}^{R}_t$ are incremental, external and residual nodal force vector, and λ is scalar force-level parameter

Step 2 Incremental tangent nodal displacement vector due to external nodal force vector

$$\delta\mathbf{u}^{ext,i}_{t,n} = \left[\mathbf{K}^{tan,i}_{tt,n}\right]^{-1}\cdot\mathbf{P}^{ext}_{t,n}$$

where $\delta\mathbf{u}^{ext}_t$ is incremental tangent nodal displacement vector, and $\mathbf{K}^{tan}_{tt}$ is tangent stiffness matrix.

Step 3 Incremental nodal displacement vector

$$\begin{aligned}\delta\bar{\mathbf{u}}^{i}_{t,n} &= \left[\mathbf{K}^{tan,i}_{tt,n}\right]^{-1}\cdot\mathbf{P}^{R,i}_{t,n}\\ \delta\mathbf{u}^{i}_{t,n} &= \delta\bar{\mathbf{u}}^{i}_{t,n} + \delta\mathbf{u}^{ext,i}_{t,n}\end{aligned}$$

where, $\delta\bar{\mathbf{u}}_t$ and $\delta\mathbf{u}_{t,n}$ are incremental nodal displacement vector due to residual nodal force vector and incremental nodal displacement vector.

Step 4 Incremental are-length

$$\left[\Delta l^i_n\right]^2 = \left[\Delta\mathbf{u}^{i-1}_{t,n}\right]^T\cdot\Delta\mathbf{u}^{i-1}_{t,n} + \Delta\lambda^{i-1}_n\cdot\psi^2\cdot\left[\mathbf{P}^{ext}_{t,n}\right]^T\cdot\mathbf{P}^{ext}_{t,n}$$

where, Δl is incremental arc-length, $\Delta\mathbf{u}_t$ and $\Delta\lambda$ incremental nodal displacement and incremental scalar force-lever parameter from last converged equilibrium state, and ψ scaling parameter for arc-length constraint.

Step 5 Incremental scalar force-level parameter

$$a_1\cdot\left[\delta\lambda^i_n\right]^2 + a_2\cdot\delta\lambda^i_n + a3 = 0 \tag{4.19}$$

where,

$$
\begin{aligned}
a_1 &= \left[\delta \mathbf{u}_{t,n}^{ext,i}\right]^T \cdot \delta \mathbf{u}_{t,n}^{ext,i} + \psi^2 \cdot \left[\mathbf{P}_{t,n}^{ext}\right]^T \cdot \mathbf{P}_{t,n}^{ext} \\
a_2 &= 2 \cdot \left[\delta \mathbf{u}_{t,n}^{ext,i}\right]^T \left(\Delta \mathbf{u}_{t,n}^{i-1} + \delta \bar{\mathbf{u}}_{t,n}^{i}\right) + 2 \cdot \Delta \lambda_n^{i-1} \cdot \psi^2 \cdot \left[\mathbf{P}_{t,n}^{ext}\right]^T \cdot \mathbf{P}_{t,n}^{ext} \\
a_3 &= \left(\Delta \mathbf{u}_{t,n}^{i-1} + \delta \bar{\mathbf{u}}_{t,n}^{i}\right)^T \cdot \left(\Delta \mathbf{u}_{t,n}^{i-1} + \delta \bar{\mathbf{u}}_{t,n}^{i}\right) - \left[\Delta l_n^i\right]^2 + \Delta \lambda_n^{i-1} \cdot \psi^2 \cdot \left[\mathbf{P}_{t,n}^{ext}\right]^T \cdot \mathbf{P}_{t,n}^{ext}
\end{aligned}
$$

where, $\delta\lambda$ is incremental scalar force-level parameter. Eq. 4.19 can be solved for $\delta\lambda$. If $\delta\lambda$ can not have real roots, numerical analysis for are-length method fails. Otherwise, if it has one real root

$$\delta \lambda_n^i = \delta \lambda_n^i$$

else if it has two real roots, R_1 and R_2

$$
\begin{aligned}
a_4 &= \left[\Delta \mathbf{u}_{t,n}^{i-1}\right]^T \cdot \delta \bar{\mathbf{u}}_{t,n}^{i} + \left[\Delta \mathbf{u}_{t,n}^{i-1}\right]^T \cdot \Delta \mathbf{u}_{t,n}^{i-1} \\
a_5 &= \left[\Delta \mathbf{u}_{t,n}^{i-1}\right]^T \cdot \delta \bar{\mathbf{u}}_{t,n}^{i} \\
\Delta l_n^i \cdot \cos\theta_1 &= a_4 + a_5 \cdot R_1 \\
\Delta l_n^i \cdot \cos\theta_2 &= a_4 + a_5 \cdot R_2
\end{aligned}
$$

If $\Delta l_n^i \cdot \cos\theta_1$ is greater than $\Delta l_n^i \cdot \cos\theta_2$, $\delta\lambda_n^i = R_1$, otherwise $\delta\lambda_n^i = R_2$.

Step 6 Update scalar force-lever parameter and nodal displacement vector

$$
\begin{aligned}
\lambda_n^i &= \lambda_n^{i-1} + \delta \lambda_n^i \\
\Delta \lambda_n^i &= \Delta \lambda_n^{i-1} + \delta \lambda_n^i \\
\delta \mathbf{u}_{t,n}^i &= \delta \bar{\mathbf{u}}_{t,n}^i + \delta \lambda_n^i \cdot \delta \mathbf{u}_{t,n}^{ext} \\
\mathbf{u}_{t,n}^i &= \mathbf{u}_{t,n}^{i-1} + \delta \mathbf{u}_{t,n}^i \\
\Delta \mathbf{u}_{t,n}^i &= \Delta \mathbf{u}_{t,n}^{i-1} + \delta \mathbf{u}_{t,n}^i
\end{aligned}
$$

Chapter 5

TIME INTEGRATION METHODS for TRANSIENT ANALYSIS

At the heart of the real time hybrid method in the numerical integration of the equation of motion, this chapter will review relevant numerical integration methods which will be later implemented in the newly developed computer program.

5.1 Introduction

The reduction of the structure from a continuum to a finite set of discrete equations may be achieved by application of the finite element method resulting in a second order ordinary differential equation.

$$\mathbf{M}_{tt} \cdot \ddot{\mathbf{u}}_t + \mathbf{C}_{tt} \cdot \dot{\mathbf{u}}_t + \mathbf{P}_t^{int} = \mathbf{P}_t^{ext} \tag{5.1}$$

where $\mathbf{M}_{tt}$ and $\mathbf{C}_{tt}$ are the mass and viscous damping matrices for the idealization of the structure; $\ddot{\mathbf{u}}_t$ is the nodal acceleration vector, $\dot{\mathbf{u}}_t$ is the nodal velocity vector, $\mathbf{P}_t^{int}$ is the static restoring or internal nodal force vector resulting from the nodal displacement vector $\mathbf{u}_t$, and $\mathbf{P}_t^{ext}$ is the vector of applied nodal forces due to a seismic loading. Numerical methods for solving Eq. 5.1 are divided into two major categories; explicit and implicit methods. This chapter will limit its coverage to implicit schemes and in particular: 1) Newmark β method, 2) the Hilber-Hughes-Taylor(HHT) method, and 3) modification of these method by P.B. Shing.

5.1.1 Mass Representation

The mass of the structure may be defined by both specified nodal lumped weights, distributed weights along the member or element material density. Nodal lumped weights will contribute only to the diagonal terms of the mass matrix where the terms associated with the rotational degrees of freedom are often taken as zero. However, in some cases the rotational inertias can be accounted for. The x, y and z inertia quantities may be different in a structure particularly in a two-dimensional analysis where the frame being analyzed is flanked by adjoining frames which carry vertical loads but have relatively insignificant lateral stiffness. In

this case the vertical inertia is associated only with that of the frame being analyzed while the horizontal inertia has to represent the sum of the inertia contributions of all frames being supported, in the lateral direction, by the frame being analyzed.

The mass matrix may take one of two forms.

1. The lumped mass is where contributions are made to the diagonal terms associated with the three translational degrees of freedom at the nodes of a member in 3D frame with no contribution to the rotational degrees of freedom. However, if there is the contribution to the rotational degrees of freedom, it may be considered with the diagonal term having the appropriate coefficient (Cook, Malkus, Plesha and Witt 2002).

2. The consistent mass using kinematically equivalent mass matrix where inertia forces are associated with all degrees of freedom. This will result in a mass matrix for the structures with the same skyline form as that of the stiffness matrix. The consistent mass model requires a greater computational cost int the multiplication by the nodal accelerations to get the inertia forces at each time-step in the analysis. It also includes all natural frequencies and consequently gives a slight bias to the frequency content of the structure.

Mercury only supports lumped mass for transient analysis.

5.1.1.1 Lumped mass

In prismatic 2D framed structure, the lumped mass matrix on global reference is presented as

$$\mathbf{M}_e = \rho \cdot A \cdot L \begin{bmatrix} 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \alpha_r \cdot L_e^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \alpha_r \cdot L_e^2 \end{bmatrix} \tag{5.2}$$

Here α_r is a nonnegative coefficient for rotation. α_r zero will result in a singular mass matrix which is undesirable if a mass-inverse appears. An *ad hoc* to prescribe α_r is to imagine that a uniform slender bar of length $L_e/2$ and mass $m/2$ is attached to each node and rotates with it. The associated mass moment of inertia is $I_z = (m/2)(L_e/2)^2/3$, for which $\alpha_r = 1/24$ (Cook et al. 2002).

5.1.1.2 Consistent mass

1. Local reference

   In prismatic 2D framed structure, consistent mass matrix in local reference is given by

$$\mathbf{m}_e = \int_0^{L_e} \rho \cdot A(x) \cdot \mathbf{N}_d(x)^T \cdot \mathbf{N}_d(x) \mathrm{d}x$$

   where,

$$\mathbf{N}_d(x) = \begin{bmatrix} 1 - \frac{x}{L_e} & 0 & 0 & \frac{x}{L_e} & 0 & 0 \\ 0 & 1 + 2(\frac{x}{L_e})^3 - 3(\frac{x}{L_e})^2 & x\left(1 - \frac{x}{L_e}\right)^2 & 0 & -2(\frac{x}{L_e})^3 + 3(\frac{x}{L_e})^2 & x\left((\frac{x}{L_e})^2 - \frac{x}{L_e}\right) \end{bmatrix}$$

Hence,

$$\mathbf{m}_e = \frac{\rho \cdot A \cdot L_e}{420} \begin{bmatrix} 140 & 0 & 0 & 70 & 0 & 0 \\ 0 & 156 & 22 \cdot L_e & 0 & 54 & -13 \cdot L_e \\ 0 & 22 \cdot L_e & 4 \cdot L_e^2 & 0 & 13 \cdot L_e & -3 \cdot L_e^2 \\ 70 & 0 & 0 & 140 & 0 & 0 \\ 0 & 54 & 13 \cdot L_e & 0 & 156 & -22 \cdot L_e \\ 0 & -13 \cdot L_e & -3 \cdot L_e^2 & 0 & -22 \cdot L_e & 4 \cdot L_e^2 \end{bmatrix} \tag{5.3}$$

2. Global reference

   In global reference, we rewrite Eq.5.3 using rotation matrix, $\mathbf{\Gamma}_e$.

$$\mathbf{M}_e = \mathbf{\Gamma}_e^T \cdot \mathbf{m}_e \cdot \mathbf{\Gamma}_e$$

5.1.2 Damping Representation

Transient response of multi-degree of freedom system is determined by the solution of simultaneous linear differential equations expressed in matrix form as

$$\mathbf{M}_{tt} \cdot \ddot{\mathbf{u}}_{t,n} + \mathbf{C}_{tt} \cdot \dot{\mathbf{u}}_{t,n} + \mathbf{P}_{t,n}^{int} = \mathbf{P}_{t,n}^{ext}$$

where, $\ddot{\mathbf{u}}_{t,n}$, $\dot{\mathbf{u}}_{t,n}$ and $\mathbf{u}_{t,n}$ are the nodal acceleration, velocity, and displacement vectors at the current time step, respectively; $\mathbf{P}_{t,n}^{int}$ is the static restoring or internal nodal force vector at the current time step.

In practice, there is not enough information to specify the coefficients of the damping matrix, $\mathbf{C}_{tt}$. Hence, we assume that

$$\mathbf{C}_{tt} = a_m \cdot \mathbf{M}_{tt} + b_k \cdot \mathbf{K}_{tt} \tag{5.4}$$

where, a_m and b_k are coefficients which pre-multiply the mass and stiffness terms respectively. Eq. 5.4 is known as proportional Rayleigh damping.

Rayleigh damping is the most widely used (but not only) model for damping. The coefficients a_m and b_k in Eq. 5.4 are calculated based upon two circular frequencies (ω_1 and ω_2, radians/sec.) to be damped at ξ_1 and ξ_2 respectively. Where ω_m and ξ_m are the circular frequency and the damping ratio of the m^{th} mode.

We recall that the damping ratio for a single degree of freedom (SDF) for mode m is given by

$$\zeta_m = \frac{C_{tt,m}}{2 \cdot M_{tt,m} \cdot \omega_m} \tag{5.5}$$

Thus for mass proportional damping of multi degree of freedom (MDF) system, with $\mathbf{C}_{tt,m} = a_m \cdot \mathbf{M}_{tt,m}$, this would lead to

$$\zeta_m = \frac{a_m}{2} \cdot \frac{1}{\omega_m}$$

The damping ratio is thus inversely proportional to the natural frequency and a_m can be selected

to obtain a specified damping ratio in any one mode i or

$$a_m = 2 \cdot \zeta_i \cdot \omega_i \tag{5.6}$$

Similarly, and recalling that $\mathbf{K}_{tt} \cdot \phi_m = \omega_m^2 \cdot \mathbf{M}_{tt} \cdot \phi_m$, a stiffness proportional damping $\mathbf{C}_{tt,m} = b_k \cdot \mathbf{K}_{tt,m}$ combined with Eq. 5.5 will lead to

$$\zeta_m = \frac{b_k}{2} \cdot \omega_m \tag{5.7}$$

In this case the damping ratio is proportional to the natural frequency and b_k can be selected to obtain a specified damping ratio in any one mode j or

$$b_k = \frac{2 \cdot \zeta_j}{\omega_j} \tag{5.8}$$

Combining Eq. 5.6 and 5.8 leads to the following linear equations

$$\frac{1}{2} \begin{bmatrix} \frac{1}{\omega_i} & \omega_i \\ \frac{1}{\omega_j} & \omega_j \end{bmatrix} \begin{Bmatrix} a_m \\ b_k \end{Bmatrix} = \begin{Bmatrix} \zeta_1 \\ \zeta_2 \end{Bmatrix} \tag{5.9}$$

If one assumes the same damping ratio ζ for both modes (reasonable practical assumption), then

$$a_m = \zeta \frac{2\omega_i \cdot \omega_j}{\omega_i + \omega_j}$$
$$b_k = \zeta \frac{2\omega_i \cdot \omega_j}{\omega_i + \omega_j}$$

Fig. 5.1 explains Rayleigh damping.

5.2 Time Integration Methods

Time integration methods by time step are used for solving the equations of motion at discrete time interval Δt. If the structural response at the previous time step $n-1$ is given and the equation of equilibrium is satisfied, then the equation of motion for a structure can be depicted as

$$\mathbf{M}_{tt} \cdot \ddot{\mathbf{u}}_{t,n-1} + \mathbf{C}_{tt} \cdot \dot{\mathbf{u}}_{t,n-1} + \mathbf{P}^{int}_{t,n-1} = \mathbf{P}^{ext}_{t,n-1}$$

At the current time step n, the equation of motion is

$$\mathbf{M}_{tt} \cdot \ddot{\mathbf{u}}_{t,n} + \mathbf{C}_{tt} \cdot \dot{\mathbf{u}}_{t,n} + \mathbf{P}^{int}_{t,n} = \mathbf{P}^{ext}_{t,n} \tag{5.10}$$

Time integration methods for solving above equation at each time step are divided into two major categories which are explicit and implicit methods.

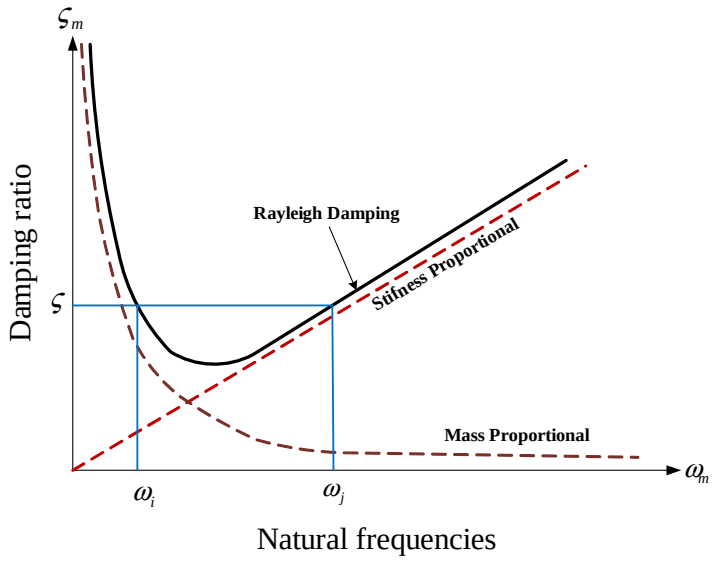


Fig. 5.1 Rayleigh damping

1. Explicit methods

   The nodal displacement vector $\mathbf{u}_{t,n}$ at the current time step can be calculated from a function of structural response solutions at the previous time step $n-1$. The effective stiffness matrix for solving the equation of motion consists of mass matrix and/or damping matrix. This indicates that it is not necessary to invert the structural stiffness matrix.

2. Implicit methods

   The nodal displacement vector $\mathbf{u}_{t,n}$ at the current time step is often represented as a function of structural response quantities with both the previous time step $n-1$ and the current time step n. For this reason, iteration procedure such as Newton-Raphson iterative method has to be applied so as to solve equations and to achieve convergence if the response is nonlinear.

Advantages of using explicit methods for transient analysis include the simplicity of the algorithm, easy implementation and their fast and efficient computation without the demand for iterations. In addition, there are no need for the knowledge of the tangent stiffness during the test. However, explicit methods are only conditionally stable. This means that the computed response may grow without bond when the product of the time step and the highest natural frequency of the structure $\Delta t \cdot \omega$ exceeds a limitation. In analyzing a structure with a very high natural frequency ω, a small time interval needs to be used and it can be too small to be practical. On the other hand, most implicit methods are unconditionally stable. This means that it is possible to analyze a multi-degree-of-freedom system with high frequency modes using a reasonably large time step without stability problems. Furthermore, implicit methods can be customized to provide favorable numerical energy dissipation properties. In addition to the prescribed viscous damping matrix $\mathbf{C}_{tt}$, implicit methods provide numerical damping which can be controlled by certain parameters. The value of these parameters can be selected in order to suppress the spurious higher-mode responses, excited by experimental errors during the test. However, an iterative solution procedure for a nonlinear system is computationally more demanding. It introduces the possibility of inducing undesirable loading and unloading

hysteresis in a structure whose behavior can be highly sensitive to prior inelastic deformation history. Last but not least implicit methods are ideal candidates fro real time hybrid simulation.

5.2.1 Newmark β method

5.2.1.1 General Formulation

The most widely used method of time integration methods for solving Eq.5.10 is the Newmark β method. We should solve the initial value problem for Eq.5.10 to find a nodal displacement vector, $\mathbf{u}_{t,n}$ from the given initial data ($\mathbf{u}_{t,0}$ and $\dot{\mathbf{u}}_{t,0}$). Let us consider the Taylor series expansions of the nodal displacement and velocity vector terms about the values at the previous time $n-1$.

$$\mathbf{u}_{t,n} \approx \mathbf{u}_{t,n-1} + \frac{\partial \mathbf{u}_{t,n-1}}{\partial t}\Delta t + \frac{\partial^2 \mathbf{u}_{t,n-1}}{\partial^2 t}\frac{\Delta t^2}{2!} + \frac{\partial^3 \mathbf{u}_{t,n}}{\partial^3 t}\frac{\Delta t^3}{3!} \tag{5.11}$$

$$\dot{\mathbf{u}}_{t,n} \approx \dot{\mathbf{u}}_{t,n-1} + \frac{\partial^2 \mathbf{u}_{t,n-1}}{\partial t^2}\Delta t + \frac{\partial^3 \mathbf{u}_{t,n}}{\partial^3 t}\frac{\Delta t^2}{2!} \tag{5.12}$$

The above two equations represent the approximate displacement and velocity vectors ($\mathbf{u}_{t,n}$ and $\dot{\mathbf{u}}_{t,n}$) except for high order terms of Taylor series. We represent the last terms of the above two equations as follow:

$$\begin{aligned}
\frac{\partial^3 \mathbf{u}_{t,n}}{\partial^3 t}\frac{\Delta t^3}{3!} &\approx \frac{\frac{\partial^2 \mathbf{u}_{t,n}}{\partial^2 t} - \frac{\partial^2 \mathbf{u}_{t,n-1}}{\partial^2 t}}{\Delta t}\frac{\Delta t^3}{3!} \\
&\approx \left(\ddot{\mathbf{u}}_{t,n} - \ddot{\mathbf{u}}_{t,n-1}\right)\frac{\Delta t^2}{3!} \\
&\approx \beta\left(\ddot{\mathbf{u}}_{t,n} - \ddot{\mathbf{u}}_{t,n-1}\right)\Delta t^2 \qquad (5.13) \\
\frac{\partial^3 \mathbf{u}_{t,n}}{\partial^3 t}\frac{\Delta t^2}{2!} &\approx \frac{\frac{\partial^2 \mathbf{u}_{t,n}}{\partial^2 t} - \frac{\partial^2 \mathbf{u}_{t,n-1}}{\partial^2 t}}{\Delta t}\frac{\Delta t^2}{2!} \\
&\approx \left(\ddot{\mathbf{u}}_{t,n} - \ddot{\mathbf{u}}_{t,n-1}\right)\frac{\Delta t}{2!} \\
&\approx \gamma\left(\ddot{\mathbf{u}}_{t,n} - \ddot{\mathbf{u}}_{t,n-1}\right)\Delta t \qquad (5.14)
\end{aligned}$$

where β and γ are parameters which depict numerical approximations. These parameters will account for Eq.5.13 and Eq.5.14 including additional terms which were dropped from Taylor series approximation. Substituting Eq.5.13 and Eq.5.14 into Eq.5.11 and Eq.5.12, respectively, we obtain the two equations as follow:

$$\begin{aligned}
\mathbf{u}_{t,n} &= \mathbf{u}_{t,n-1} + \Delta t \cdot \dot{\mathbf{u}}_{t,n-1} + \frac{\Delta t^2}{2} \cdot \ddot{\mathbf{u}}_{t,n-1} + \Delta t^2 \cdot \beta \cdot \left(\ddot{\mathbf{u}}_{t,n} - \ddot{\mathbf{u}}_{t,n-1}\right) \\
\dot{\mathbf{u}}_{t,n} &= \dot{\mathbf{u}}_{t,n-1} + \Delta t \cdot \ddot{\mathbf{u}}_{t,n-1} + \Delta t \cdot \gamma \cdot \left(\ddot{\mathbf{u}}_{t,n} - \ddot{\mathbf{u}}_{t,n-1}\right)
\end{aligned} \tag{5.15}$$

Hence, we obtain the Newmark β method, which consists of the following equations:

$$\mathbf{P}_{t,n}^{ext} = \mathbf{M}_{tt} \cdot \ddot{\mathbf{u}}_{t,n} + \mathbf{C}_{tt} \cdot \dot{\mathbf{u}}_{t,n} + \mathbf{P}_{t,n}^{int} \tag{5.16}$$

$$\mathbf{u}_{t,n} = \mathbf{u}_{t,n-1} + \Delta t \cdot \dot{\mathbf{u}}_{t,n-1} + \frac{\Delta t^2}{2}\left[(1-2\beta)\ddot{\mathbf{u}}_{t,n-1} + 2\beta \cdot \ddot{\mathbf{u}}_{t,n}\right] \tag{5.17}$$

$$\dot{\mathbf{u}}_{t,n} = \dot{\mathbf{u}}_{t,n-1} + \Delta t\left[(1-\gamma)\ddot{\mathbf{u}}_{t,n-1} + \gamma \cdot \ddot{\mathbf{u}}_{t,n}\right] \tag{5.18}$$

where, Eq.5.16 is the equation of equilibrium expressed at time $n+1$, and Eq.5.17 and Eq.5.18 are finite difference formulas describing the evolution of the approximation solution. β and γ are parameters that determine the stability and accuracy characteristics. Stability conditions for the Newmark β method follows:

1. unconditionally stable if

$$\begin{aligned} \gamma &\geq \frac{1}{2} \\ \beta &\geq \frac{\gamma}{2} \end{aligned}$$

2. conditionally stable if

$$\begin{aligned} \gamma &\geq \frac{1}{2} \\ \beta &< \frac{\gamma}{2} \end{aligned}$$

with the following stability limit:

$$\omega \cdot \Delta t \leq \Omega_{crit} = \frac{\xi(\gamma - 1/2) + [\gamma/2 - \beta + \xi^2(\gamma - 1/2)^2]^{1/2}}{\gamma/2 - \beta} \tag{5.19}$$

where ω is maximum natural frequency, Ω_{crit} is critical sampling frequency, and ξ is the damping ratio.

The constant acceleration (trapezoidal rule) method is implicit and unconditionally stable. The linear acceleration method is implicit and conditionally stable. The central difference method is conditionally stable. Note that if $\gamma = 1/2$ has no effect on stability. In practice it is more convenient to express Eq.5.19 in terms of the period of vibration, $T = 2\pi/\omega$, in which case Eq.5.19 becomes $\Delta t/T \leq \Omega_{crit}/(2\pi)$. In the case of the linear acceleration, $\Delta t/T$ is calculated as follows:

$$\frac{\Delta t}{T} \leq \frac{1}{2\pi}\frac{1}{\sqrt{\gamma - 2\beta}} = 0.551$$

A summary for the Newmark β method is shown in Table. 5.1.

Table 5.1 Properties of the Newmark β method

Method	Type	β	γ	Stability condition	Order of accuracy
Constant acceleration	Implicit	1/4	1/2	Unconditional	2
Linear acceleration	implicit	1/6	1/2	$\Delta t \leq 2\sqrt{3}/\omega$	2
Central difference	Explicit	0	1/2	$\Delta t \leq 2/\omega$	2

5.2.1.2 Newmark β implicit method

The Newmark β implicit method can be expressed with Eq. 5.16, 5.17, and 5.18. Rewritting Eq. 5.17 and 5.18:

$$\mathbf{u}_{t,n} = \tilde{\mathbf{u}}_{t,n} + \Delta t^2 \cdot \beta \cdot \ddot{\mathbf{u}}_{t,n} \tag{5.20}$$

$$\dot{\mathbf{u}}_{t,n} = \tilde{\dot{\mathbf{u}}}_{t,n} + \Delta t \cdot \gamma \cdot \ddot{\mathbf{u}}_{t,n} \tag{5.21}$$

where,

$$\tilde{\mathbf{u}}_{t,n} = \mathbf{u}_{t,n-1} + \Delta t \cdot \dot{\mathbf{u}}_{t,n-1} + \frac{\Delta t^2}{2}(1-2\beta)\ddot{\mathbf{u}}_{t,n-1}$$

$$\tilde{\dot{\mathbf{u}}}_{t,n} = \dot{\mathbf{u}}_{t,n-1} + \Delta t(1-\gamma)\ddot{\mathbf{u}}_{t,n-1}$$

Rewritting Eq. 5.20, we can solve for $\ddot{\mathbf{u}}_{t,n}$:

$$\ddot{\mathbf{u}}_{t,n} = \frac{\mathbf{u}_{t,n} - \tilde{\mathbf{u}}_{t,n}}{\Delta t^2 \cdot \beta} \tag{5.22}$$

Substituting Eq. 5.22 into Eq. 5.21, we can rewrite $\dot{\mathbf{u}}_{t,n}$:

$$\dot{\mathbf{u}}_{t,n} = \tilde{\dot{\mathbf{u}}}_{t,n} + \frac{\gamma}{\Delta t \cdot \beta}(\mathbf{u}_{t,n} - \tilde{\mathbf{u}}_{t,n}) \tag{5.23}$$

Substituting Eq. 5.22 and Eq. 5.23 into Eq. 5.16, we have:

$$\mathbf{M}_{tt}\left[\frac{\mathbf{u}_{t,n} - \tilde{\mathbf{u}}_{t,n}}{\Delta t^2 \cdot \beta}\right] + \mathbf{C}_{tt}\left[\tilde{\dot{\mathbf{u}}}_{t,n} + \frac{\gamma}{\Delta t \cdot \beta}(\mathbf{u}_{t,n} - \tilde{\mathbf{u}}_{t,n})\right] + \mathbf{P}^{int}_{t,n} = \mathbf{P}^{ext}_{t,n} \tag{5.24}$$

Rewriting Eq. 5.24,

$$\begin{aligned}&\frac{1}{\Delta t^2 \cdot \beta}\mathbf{M}_{tt} \cdot \mathbf{u}_{t,n} + \frac{\gamma}{\Delta t \cdot \beta}\mathbf{C}_{tt} \cdot \mathbf{u}_{t,n} + \mathbf{P}_{t,n}^{int} \\ &= \mathbf{P}_{t,n}^{ext} + \frac{1}{\Delta t^2 \cdot \beta}\mathbf{M}_{tt} \cdot \tilde{\mathbf{u}}_{t,n} + \frac{\gamma}{\Delta t \cdot \beta}\mathbf{C}_{tt} \cdot \tilde{\mathbf{u}}_{t,n} - \mathbf{C}_{tt} \cdot \tilde{\dot{\mathbf{u}}}_{t,n}\end{aligned} \tag{5.25}$$

If the trial solutions in given iteration step k are $\mathbf{u}_{t,n}^k$, and $\mathbf{P}_{t,n}^{int,k}$, then it does not satisfy the equations of motion. Hence, we can write for this particular step with residual force vector $\mathbf{P}_{t,n}^{R,k}$:

$$\mathbf{P}_{t,n}^{R,k} = \mathbf{P}_{t,n}^{ext} + \overline{\mathbf{M}}_{tt}\left(\tilde{\mathbf{u}}_{t,n} - \mathbf{u}_{t,n}^k\right) - \mathbf{C}_{tt} \cdot \tilde{\dot{\mathbf{u}}}_{t,n} - \mathbf{P}_{t,n}^{int,k}$$

where,

$$\overline{\mathbf{M}}_{tt} = \frac{\mathbf{M}_{tt} + \Delta t \cdot \gamma \cdot \mathbf{C}}{\Delta t^2 \cdot \beta}$$

Using initial stiffness iterative method, we can solve for $\Delta\mathbf{u}_{t,n}^k$:

$$\mathbf{P}_{t,n}^{R,k} = \mathbf{K}_{eff} \cdot \Delta\mathbf{u}_{t,n}^k \tag{5.26}$$

where, $\mathbf{K}_{eff}$ is the effective stiffness matrix, and $\Delta\mathbf{u}_{t,n}^k$ is:

$$\Delta\mathbf{u}_{t,n}^k = \mathbf{u}_{t,n} - \mathbf{u}_{t,n}^k$$

In elastic section, we can rewrite $\mathbf{P}_{t,n}^{int}$ to compute the effective stiffness matrix with initial stiffness matrix $\mathbf{K}_{tt}$:

$$\mathbf{P}_{t,n}^{int} = \mathbf{K}_{tt} \cdot \mathbf{u}_{t,n} \tag{5.27}$$

Substituting Eq. 5.27 into Eq. 5.25, we can solve for $\mathbf{u}_{t,n}$:

$$\mathbf{K}_{eff} \cdot \mathbf{u}_{t,n} = \mathbf{P}_{t,n}^{ext} + \overline{\mathbf{M}}_{tt} \cdot \tilde{\mathbf{u}}_{t,n} - \mathbf{C}_{tt} \cdot \tilde{\dot{\mathbf{u}}}_{t,n}$$

where,

$$\mathbf{K}_{eff} = \overline{\mathbf{M}}_{tt} + \mathbf{K}_{tt}$$

From Eq. 5.26, we can solve for $\delta\mathbf{u}_{t,n}^k$ and the updated displacement vector $\mathbf{u}_{t,n}^{k+1}$ at the next iteration step $k+1$:

$$\begin{aligned}\delta\mathbf{u}_{t,n}^k &= [\mathbf{K}_{eff}]^{-1} \cdot \mathbf{P}_{t,n}^{R,k} \\ \mathbf{u}_{t,n}^{k+1} &= \mathbf{u}_{t,n}^k + \delta\mathbf{u}_{t,n}^k\end{aligned}$$

Fig. 5.2 and 5.3 explain the implementation of transient analysis using the Newmark β implicit method with flexibility-based 2D beam-column elements.

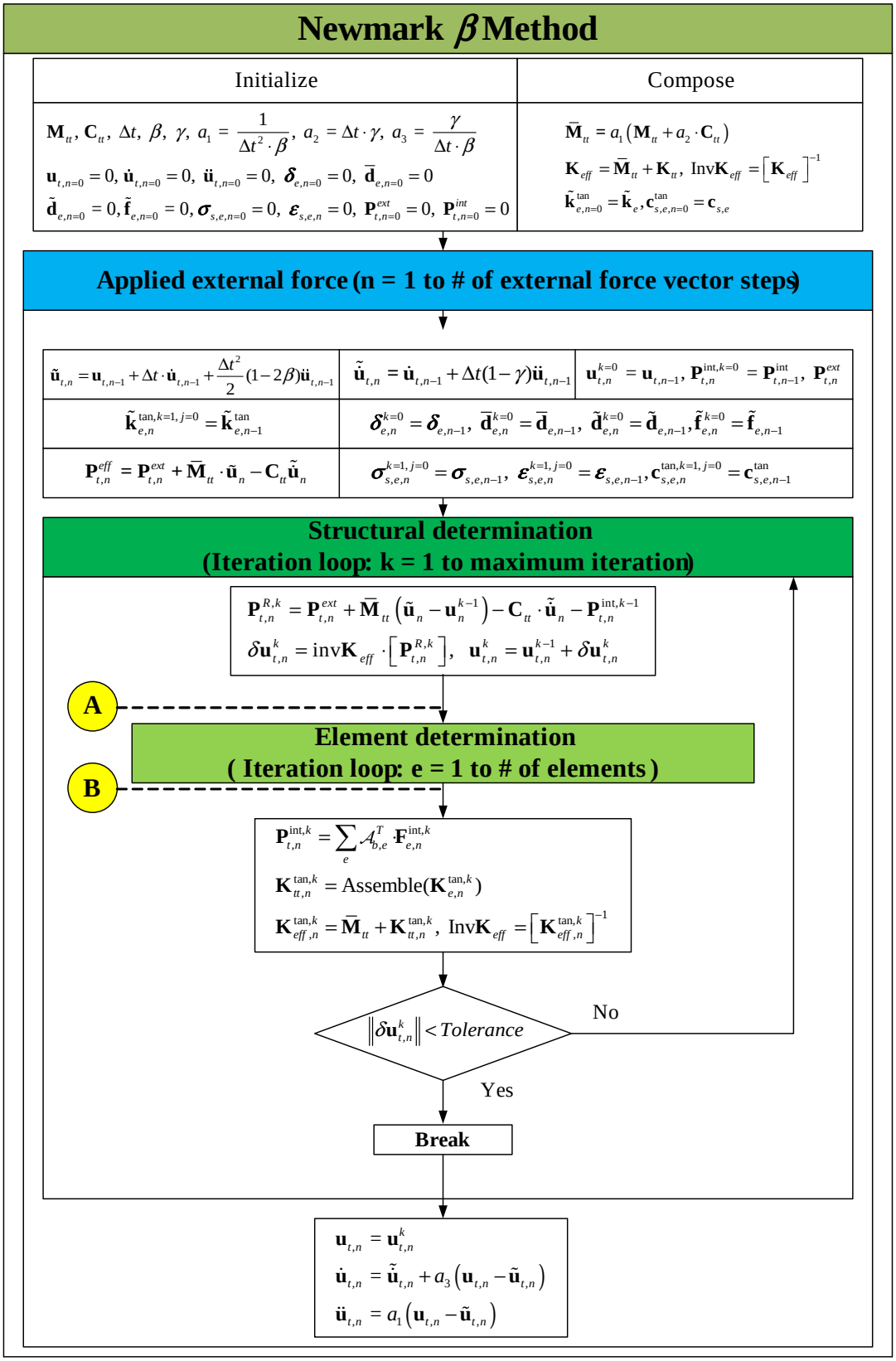


Fig. 5.2 Flow chart (1) of transient analysis using the Newmark β implicit method

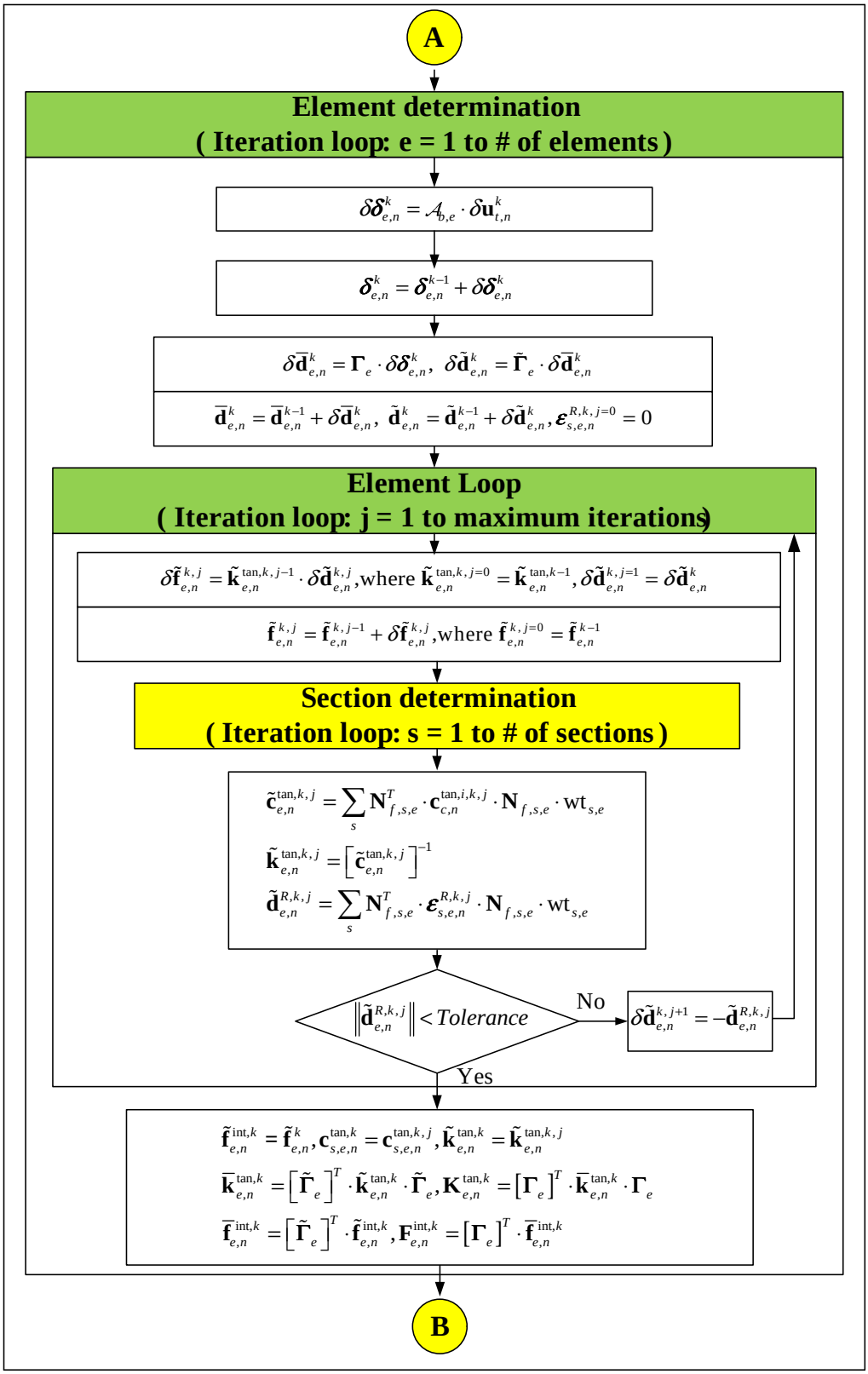


Fig. 5.3 Flow chart (2) of transient analysis using the Newmark β implicit method

5.2.2 Hilber-Hughes-Taylor method(HHT method)

5.2.2.1 General Formulation

A major drawback of Newmark β method is the tendency for high frequency noise to persist in the solution. On the other hand, when linear damping or artificial viscosity is added via the parameter γ, the accuracy is markedly degraded. The α method, (Hilber, Hughes and Taylor 1977) improves numerical dissipation for high frequency without degrading the accuracy as much.

Equation of motion in HHT method is written at current time step n (forward difference) as:

$$\mathbf{M}_{tt} \cdot \ddot{\mathbf{u}}_{t,n} + \mathbf{P}^{int}_{t,n} = \mathbf{P}^{ext}_{t,n}$$

Seeking an approximate solution of this equation by one-step difference, we write,

$$\mathbf{M}_{tt} \cdot \ddot{\mathbf{u}}_{t,n} + (1+\alpha)\mathbf{P}^{int}_{t,n} - \alpha \cdot \mathbf{P}^{int}_{t,n-1} = \mathbf{P}^{ext}_{t,n}$$

with Eq. 5.15.

We note that the HHT method introduces $\alpha(\mathbf{P}^{int}_{t,n} - \mathbf{P}^{int}_{t,n-1})$ which is akin of stiffness proportional damping. If the above equation is expanded, effect of damping introduced, and possible material nonlinearity introduced, we obtain:

$$(1+\alpha)\mathbf{P}^{ext}_{t,n} - \alpha\mathbf{P}^{ext}_{t,n-1} = \mathbf{M}_{tt}\cdot\ddot{\mathbf{u}}_{t,n} + (1+\alpha)\mathbf{C}_{tt}\cdot\dot{\mathbf{u}}_{t,n} - \alpha\cdot\mathbf{C}_{tt}\cdot\dot{\mathbf{u}}_{t,n-1} + (1+\alpha)\mathbf{P}^{int}_{t,n} - \alpha\cdot\mathbf{P}^{int}_{t,n-1} \quad (5.28)$$

If $-1/3 \leq \alpha \leq 0$, $\beta = (1-\alpha)^2/4$, and $\gamma = (1-2\alpha)/2$, then the α method is unconditionally stable and has a second-order accuracy. Hence, Eq. 5.15 and 5.28 must all be simultaneously satisfied through an iterative method.

Assuming that we have obtained the response at the previous time step $n-1$, i.e. $\mathbf{u}_{t,n-1}$, $\dot{\mathbf{u}}_{t,n-1}$ and $\ddot{\mathbf{u}}_{t,n-1}$ which satisfy the equation of motion, we now seek to determine the solution at the current time step n by iteration. First of all, we need to determine effective external force and effective stiffness. These are calculated from Eq.5.22, 5.23 and 5.28.

$$\begin{aligned}
&\frac{1}{\Delta t^2 \cdot \beta}\mathbf{M}_{tt}\cdot\mathbf{u}_{t,n} + \frac{\gamma}{\Delta t\cdot\beta}(1+\alpha)\mathbf{C}_{tt}\cdot\mathbf{u}_{t,n} + (1+\alpha)\mathbf{P}^{int}_{t,n} \\
&= (1+\alpha)\mathbf{P}^{ext}_{t,n} - \alpha\cdot\mathbf{P}^{ext}_{t,n-1} + \frac{1}{\Delta t^2\cdot\beta}\mathbf{M}_{tt}\cdot\tilde{\mathbf{u}}_{t,n} + \frac{\gamma}{\Delta t\cdot\beta}(1+\alpha)\mathbf{C}_{tt}\cdot\tilde{\mathbf{u}}_{t,n} \\
&- (1+\alpha)\mathbf{C}_{tt}\cdot\tilde{\dot{\mathbf{u}}}_{t,n} + \alpha\cdot\mathbf{C}_{tt}\cdot\dot{\mathbf{u}}_{n-1} + \alpha\cdot\mathbf{P}^{int}_{t,n}
\end{aligned} \quad (5.29)$$

The trial solutions in iteration step k are $\mathbf{u}^k_{t,n}$, and $\mathbf{P}^{int,k}_{t,n}$, does not necessarily satisfy the equations of motion. Hence, we can write for this particular step:

$$\begin{aligned}
\mathbf{P}^{R,k}_{t,n} &= (1+\alpha)\mathbf{P}^{ext}_{t,n} - \alpha\cdot\mathbf{P}^{ext}_{t,n-1} + \overline{\mathbf{M}}_{tt}\left(\tilde{\mathbf{u}}_{t,n} - \mathbf{u}^k_{t,n}\right) - (1+\alpha)\mathbf{C}_{tt}\cdot\tilde{\dot{\mathbf{u}}}_{t,n} + \alpha\cdot\mathbf{C}_{tt}\cdot\dot{\mathbf{u}}_{t,n-1} \\
&- (1+\alpha)\mathbf{P}^{int,k}_{t,n} + \alpha\cdot\mathbf{P}^{int}_{t,n-1}
\end{aligned}$$

where,

$$\overline{\mathbf{M}}_{tt} = \frac{\mathbf{M}_{tt} + \Delta t \cdot \gamma(1+\alpha)\mathbf{C}_{tt}}{\Delta t^2 \cdot \beta}$$

and $\mathbf{P}_{t,n}^{R,k}$ is the residual force vector.

Using the initial stiffness iterative method, we can solve for $\Delta\mathbf{u}_{t,n}^{k}$:

$$\mathbf{P}_{t,n}^{R,k} = \mathbf{K}_{eff} \cdot \Delta\mathbf{u}_{t,n}^{k} \tag{5.30}$$

where, $\mathbf{K}_{eff}$ is the effective stiffness matrix, and $\Delta\mathbf{u}_{t,n}^{k}$ is:

$$\Delta\mathbf{u}_{t,n}^{k} = \mathbf{u}_{t,n} - \mathbf{u}_{t,n}^{k}$$

In elastic section, we can express $\mathbf{P}_{t,n}^{int}$ to compute the effective stiffness matrix with initial stiffness matrix $\mathbf{K}_{tt}$ as:

$$\mathbf{P}_{t,n}^{int} = \mathbf{K}_{tt} \cdot \mathbf{u}_{t,n} \tag{5.31}$$

Substituting Eq. 5.31 into Eq. 5.29, we solve for $\mathbf{u}_{t,n}$:

$$\begin{aligned}\mathbf{K}_{eff} \cdot \mathbf{u}_{t,n} = (1+\alpha)\mathbf{P}_{t,n}^{ext} - \alpha \cdot \mathbf{P}_{t,n-1}^{ext} + \overline{\mathbf{M}}_{tt} \cdot \tilde{\mathbf{u}}_{t,n} \\ - (1+\alpha) \cdot \mathbf{C}_{tt} \cdot \tilde{\dot{\mathbf{u}}}_{t,n} + \alpha \cdot \mathbf{C}_{tt} \cdot \dot{\mathbf{u}}_{t,n-1} + \alpha \cdot \mathbf{P}_{t,n-1}^{int}\end{aligned}$$

where,

$$\mathbf{K}_{eff} = \overline{\mathbf{M}}_{tt} + (1+\alpha)\mathbf{K}_{tt}$$

From Eq. 5.30, we solve for $\delta\mathbf{u}_{t,n}^{k}$ and the updated displacement vector $\mathbf{u}_{t,n}^{k+1}$ at the next iteration step $k+1$:

$$\begin{aligned}\delta\mathbf{u}_{t,n}^{k} &= [\mathbf{K}_{eff}]^{-1} \cdot \mathbf{P}_{t,n}^{R,k} \\ \mathbf{u}_{t,n}^{k+1} &= \mathbf{u}_{t,n}^{k} + \delta\mathbf{u}_{t,n}^{k}\end{aligned}$$

Finally, we note that:

1. α introduces a damping that grows with the ratio of time increment to the period of vibration of a node.
2. Negative values of α provide damping
3. If $\alpha = 0$, we have no artificial damping (energy preseving) and is exactly the constant acceleration (trapezoidal rule) - Newmark's β method if $\beta = 1/4$ and $\gamma = 1/2$.
4. Maximum value is $\alpha = -1/3$ which provides the maximum artificial damping. This results in a damping ratio of about 6% when the time increment is 40% of the period of oscillation of the mode being studied and smaller if the oscillation period increases.
5. This artificial damping is not very substantial for realistic time increment and low frequencies, but is non-negligible for high frequencies.

6. A default value of -0.05 is recommended.

Fig. 5.4 explains the implementation of transient analysis using the HHT implicit method in structural level. Element state determination is identical to Fig. 5.3.

5.2.3 Modification of Newmark β and HHT method by P.B. Shing

Existing iterative method such as the Newton-Raphson or initial stiffness needs several iterations for $\mathbf{u}_{t,n}^{k}$ to converge to the exact solution and this can be computationally expensive. To avoid this problem, a special iterative method with fixed number of iterations and quadratic interpolation function during iteration (Jung 2005) in each time step was proposed. The quadratic interpolation function is based on the updated nodal displacement vector $\mathbf{u}_{t,n}^{k}$ and the converged nodal displacement vector $\mathbf{u}_{t,n-1}$ in the previous time step. In this procedure, the number of iterations within a time step is limited with m. It should be emphasized that this fixed number of iteration per time step is to address constraints imposed by real time hybrid simulation. Fig. 5.5 illustrates the procedure in detail.

Instead of the updated nodal displacement vector $\mathbf{u}_{t,n}^{k}$ during iteration, the desired nodal displacement vector $\mathbf{u}_{t,n}^{d,k}$ at current iteration step can be expressed through the following equation:

$$\mathbf{u}_{t,n}^{d,k} = \begin{bmatrix} C1 & C2 & C3 \end{bmatrix} \begin{Bmatrix} \mathbf{u}_{t,n-2}^{d,m} \\ \mathbf{u}_{t,n-1}^{d,m} \\ \mathbf{u}_{t,n}^{k} \end{Bmatrix} \tag{5.32}$$

where, k vary from 1 to m, m denotes the total number of specified iteration in each time step, and

$$C1 = \frac{1}{2}\left[\left(\frac{k}{m}\right)^2 - \left(\frac{k}{m}\right)\right]$$
$$C2 = 1 - \left(\frac{k}{m}\right)^2$$
$$C3 = \frac{1}{2}\left[\left(\frac{k}{m}\right)^2 + \left(\frac{k}{m}\right)\right]$$

m becomes $\Delta t/\delta t$, where δt may correspond to the smallest time interval of an actuator controller in real time hybrid simulation. For the first and second time steps, the quadratic interpolation is based on the initial displacement and velocity vectors as follow:

1. n=1

$$\mathbf{u}_{t,1}^{d,k} = C2 \cdot \mathbf{u}_{t,0} - 2 \cdot m \cdot C1 \cdot \dot{\mathbf{u}}_{t,0} + \left(\frac{k}{m}\right)^2 \mathbf{u}_{t,1}^{k}$$

2. n=2

$$\mathbf{u}_{t,2}^{d,k} = C1 \cdot \mathbf{u}_{t,0} + C2 \cdot \mathbf{u}_{t,n-1}^{d,m} + C3 \cdot \mathbf{u}_{t,n}^{k}$$

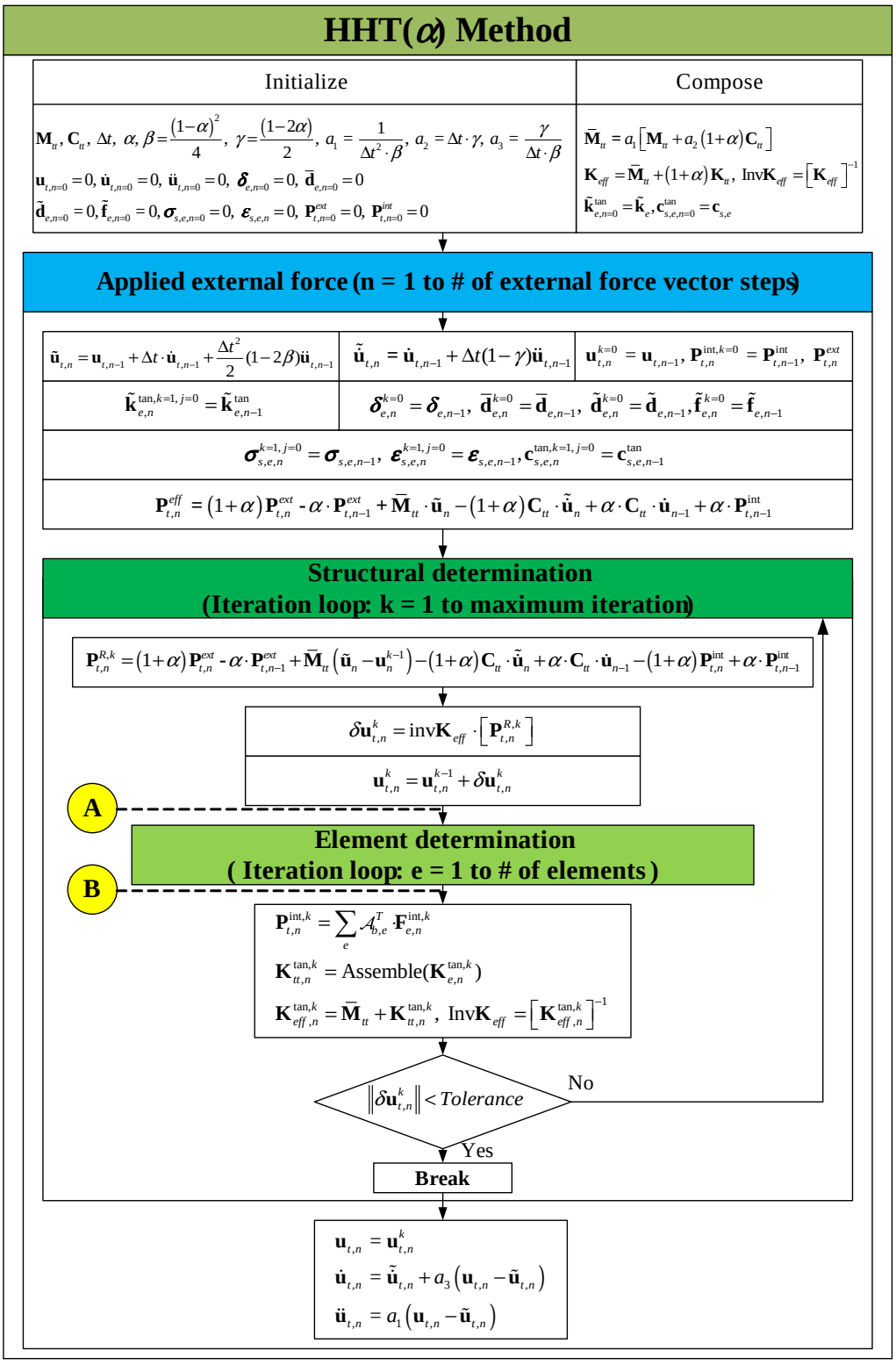


Fig. 5.4 Flow chart of transient analysis using the HHT implicit method

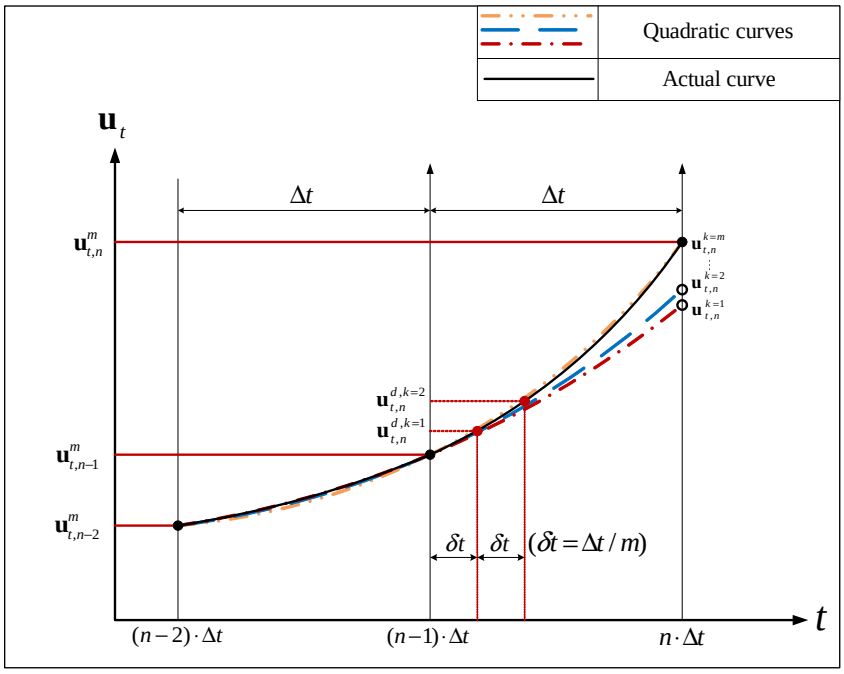


Fig. 5.5 Shing iteration scheme using quadratic interpolation

In this procedure, experience indicates that 10 iterations in a time step will provide satisfactory convergence even for strongly nonlinear structural responses provided that there is no severe strain softening (Wei 2005).

Fig. 5.6 explains the implementation of transient analysis using the Newmark β implicit method in structural level with initial stiffness method modified by P.B.Shing. Element state determination is identical to Fig. 5.3.

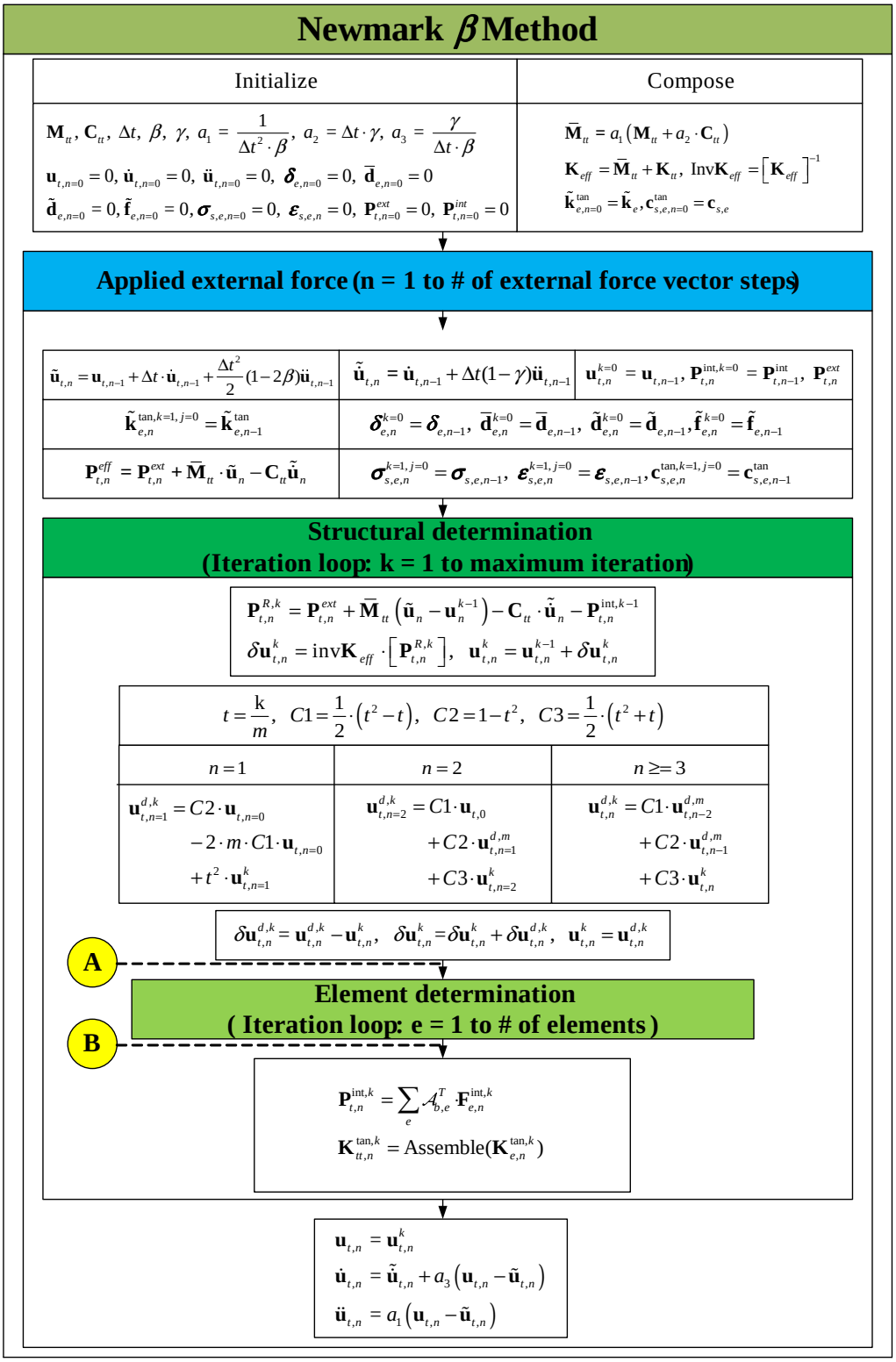


Fig. 5.6 Flow chart of transient analysis using the Newmark β implicit method in structural level with initial stiffness method modified by P.B.Shing

Chapter 6

MERCURY IMPLEMENTATION

Incomplete

The theory behind Mercury having been developed in the previous chapters, this one will address its computer implementation and strategy.

6.1 General Description

Mercury was developed in two stages. First a Matlab version was written in order to check accuracy and reliability of the various models within a user-friendly environment. Once the Matlab version was deemed satisfactory, then a c++ version was written from scratch by a professional programmer (Dr. G. Haussmann, Technical Director of CU-NEES) in such a way to be optimized for real time hybrid simulation.

This two tier approach proved most valuable as the Matlab version not only greatly facilitated subsequent implementation in c++ but by it self it constitutes an excellent pedagogical tool for those interested in learning the intricacies of nonlinear structural analysis with modern elements (flexibility based with fiber sections, and appropriate constitutive models). Such a code exist, (Fedeaslab), (Filippou 2004) however the source code is locked. Another code would be OpenSees, however few graduate students feel comfortable in programming in c++ and would rather modify a Matlab one, (Faniel 2009). Finally, there is FEAP, (Taylor 2008) which could be an intermediary solution, as modifying Fortran remains a rather simple task.

Furthermore, the Matlab version of Mercury is set up to perform hybrid simulation. Currently, a client version of Mercury could simulate the experiment as a prelude to an actual test. This platform is also ideal from a pedagogical point of view to understand the concept of hybrid simulation.

The integration of the two versions of Mercury, as well as supporting code is shown in Fig. 6.1. The Matlab and the c++ version are designed to accept similar (but not identical files), hence `m2lua.m` is a Matlab based translator which will convert `.m` files into `.lua` ones. Similarly, the mesh definition for a multistroy orthogonal frame with fiber sections, zero length elements and/or sections can be a daunting task. Hence this task is facilitated by `mmg.m` (Mercury Mesh Generator) which reads key information from a preformatted Excel file and generates a Mercury compatible file ready to run.

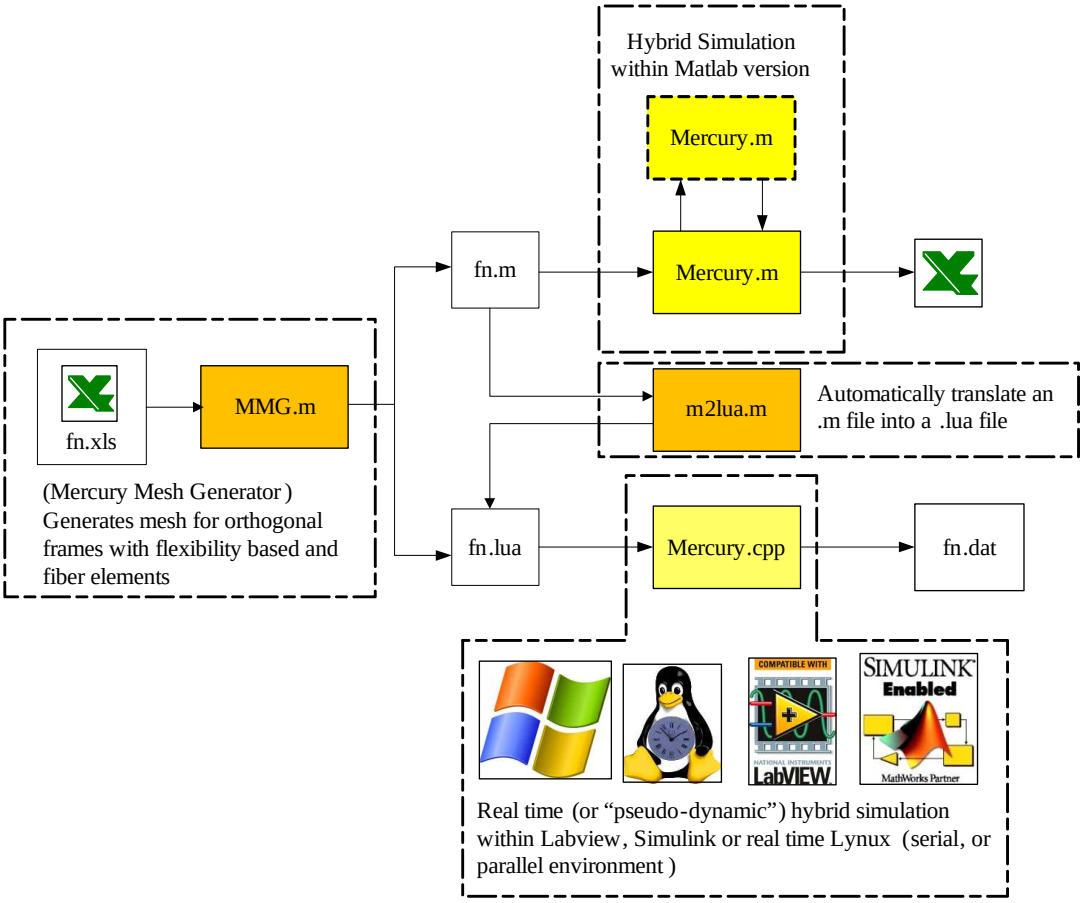


Fig. 6.1 Integration of Mercury versions and supporting codes

Finally, the c++ version of Mercury can be embedded in Simulink, LabView or operate within Windows, Linux or Real Time Linux kernels.

6.2 MatLab Implementation Strategy

The Matlab version of Mercury is structured as shown in Fig. 6.2

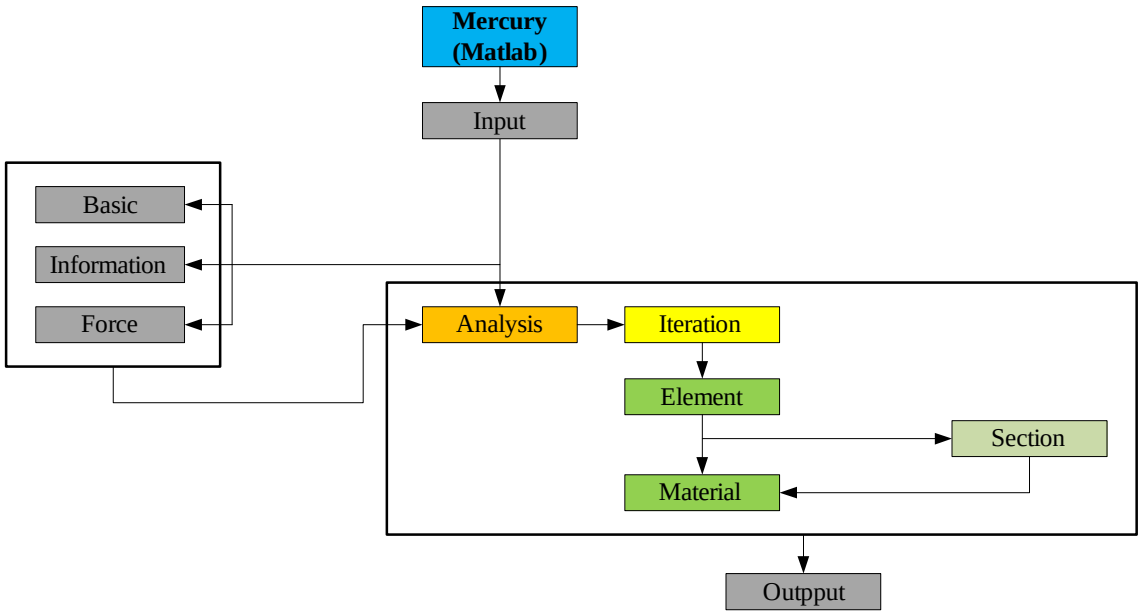


Fig. 6.2 The program architecture of `Mercury` on Matlab in Mercury

Each block in Fig. 6.2 refers to several `.m` files defined below:

Input groups all `.m` files associated with problem definition by the user.

Basic

- `DefaultValuse`: Initializes all variables used in analysis.
- `GlobalVariables`: Defines global variables.
- `MatrixVectorSize`: Assigns matrix and vector sizes based on from dimension and number of d.o.f per node.

Information

- `Assemble`: Extracts stiffness matrix ($\mathbf{K}_{tt}$) for free d.o.f and stiffness matrix ($\mathbf{K}_{ut}$)used in displacement control from augmented stiffness matrix.
- `DynMassDamping`: Computes mass and damping matrices for dynamic analysis.
- `ElementInfo`: Defines geometries of each element and defines section type or material type of each element.

- `GaussLobatto`: For frame element, Calculates locations of each section and weighting coefficients in each element.
- `InitialStiffnessMatrix`: Augments initial stiffness matrices of each element.
- `FlexibilityBasedElementStiffnessMatrix`: Computes initial stiffness matrix of flexibility-based element.
- `StiffnessBasedElementStiffnessMatrix`: Computes initial stiffness matrix of stiffness-based element.
- `TrussElementStiffnessMatrix`: Computes initial stiffness matrix of truss element.
- `InterpolationInfo`: Computes the displacement interpolation functions matrix for stiffness-based elements and force interpolation functions for flexibility-based elements.
- `NodeInfo`: Stores all nodal information of each element such as coordinates, `ID` matrix and `LM` matrix.
- `PropertyInfo`: Stores material properties of each element.
- `TransformationInfo`: Computes transformation matrix.

Force

- `AccForces`: Computes acceleration forces such as earthquake data depending on force steps.
- `ExternalForce`: Assembles all external forces for each force step.
- `IncrementalElementDistributedForces`: Computes incremental element distributed forces based on force steps.
- `IncrementalNodalDisplacements`: Computes incremental nodal displacements for displacement control based on force steps.
- `IncrementalNodalForces`: Computes incremental nodal forces based on force steps.
- `InitialElementDistributedForces`: Computes initial element distributed forces.
- `InitialNodalDisplacements`: Computes initial nodal displacements for displacement control.
- `InitialNodalForces`: Computes initial nodal forces.

Analysis

- `DynamicAnalysisHHT`: Master file for HHT method of dynamic analysis.
- `DynamicAnalysisNM`: Master file for Newmark β method of dynamic analysis.
- `StaticAnalysis`: Master file for nonlinear static analysis.

Iteration

- `InitialStiffnessMethod`: Master file for initial stiffness method.
- `ModifiedInitialStiffnessMethod`: Master file for modified initial stiffness method.
- `ModifiedNRMethod`: Master file for modified Newton-Raphson method.

- `NRMethod`: Master file for Newton-Raphson method.
- `TempModifiedInitialStiffnessMethod`: Master file for Shing' method.

Element

- `ElementDetermination`: In element level, computes element nodal forces and displacements etc.

Section

- `SectionDetermination`: In section level, computes section forces and deformations etc.
- `FiberSection`: Defines fiber section analysis.
- `GeneralSection`: Defines elastic section analysis.

Material

- `HardeningMaterial`: Determines uniaxial stress and strain for hardening material.
- `AnisotropicDamageMaterial`: Determines uniaxial stress and strain for anisotropic damage material.

Output

User modified.

6.3 c++ Implementation Strategy

The C++ software described here is intended to be a compact finite-element solver capable of numerical and real-time hybrid simulation of structures. The software is meant to be run stand-alone or embedded within other software applications such as LavView or MATLAB Real-time workshop.

6.3.1 General Desired Functionality

All components should have a certain level of basic functionality, including:

- An identifier so the component can be looked up via its name.
- A component should be able to write its complete state to and read from disk and the network.
- A component should expose its core functionality (whatever that may be) to the available scripting language(s).

6.3.2 Named Composite Hierarchy (NCH)

- Use Composite (Gamma, Helm, Johnson and Vlissides 1994, page 395) to allow for nested analysis, assembly, nodes, elements
- All named objects have an identifier, even if implicit.
- Named objects can be children nodes of other named objects.
- Named objects can be discovered via a full path name or their identifier.
- ComponentGroup collects a set of objects.
- Subtrees can be copied, deleted, written to a stream, etc.

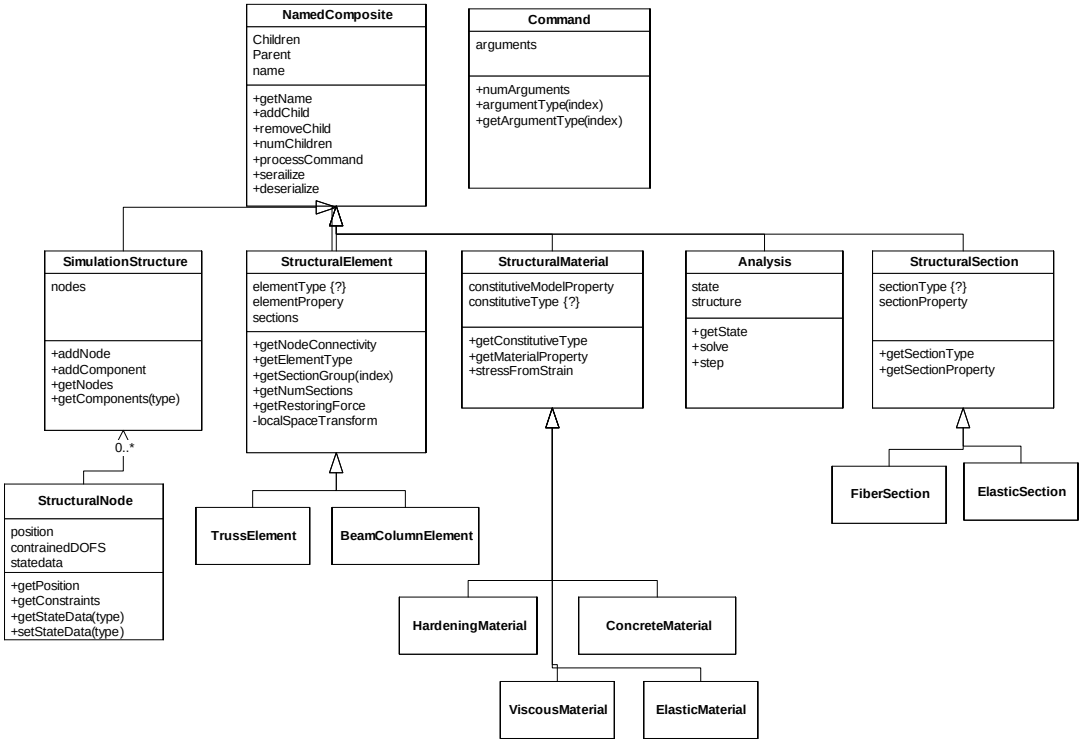


Fig. 6.3 Named Composite Hierarchy.

6.3.3 Data Streaming and Serialization

Specific functionality we wish to include is that the configuration and data can be saved to disk, or sent over the network to another machine. This can be done to save the simulation state and restart it later, or to run a simulation on one machine while visualizing it on another machine. Thus most, if not all, of the structural and simulation components should be able to serialize and unserialize themselves to a data stream. The data stream can represent disk access, network data traffic, or perhaps some other sort of data process. The specific format of the serialization is stream dependent; for instance, one type of data stream may record everything as XML records and ASCII strings, while another may use compact or even lossy representations.

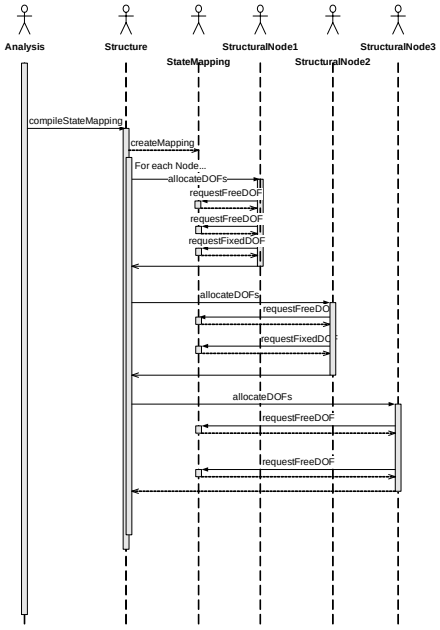


Fig. 6.4 State Construction.

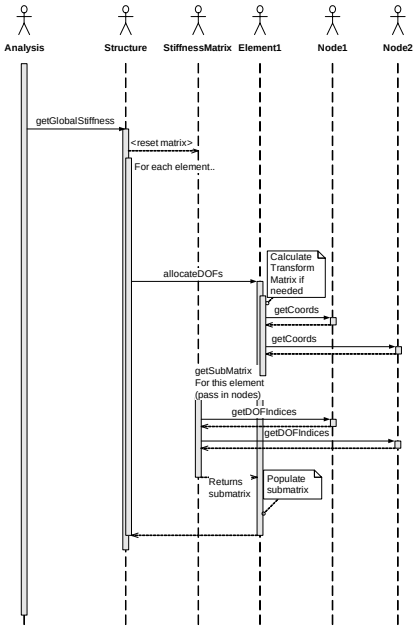


Fig. 6.5 Stiffness Matrix Assembly.

6.3.4 Configuration and Input

To configure the structure, integration, and data recording, some sort of simple scripting is required. The scripting should be:

- Compact, so that it does not require a lot of source code to add to the program and does not increase the program size noticeably.
- Simple, so that uninitiated users can quickly grasp basic usage and perform simple analysis without reading a manual on how to use the language.
- Portable, so that it can be used on multiple platforms such as Windows, RT-Linux, and others without a large amount of work to switch platforms.

A good fit for these requirements is to use *Lua* (Lua 2009) for the default scripting language. To expose component functionality to *Lua*all objects should be able to process Command objects, which are typically generated by script commands. For instance, the Lua script

```
1 analysis.setDeltat(0.02)      -- set dt to be 20mS
2 analysis.advanceStep(10)      -- do 10 timesteps
```

Will send two Command objects to the Component named "analysis," with the first Command specifying a new δt and the second command advancing the analysis by 10 timesteps.

6.3.5 Domain and Structural Assembly

- A structure consists of assembly components: nodes, elements, constitutive models, section models, and forcing functions (excitation).
- Components can be added in any order up until the structure is "compiled"—that is, the state variables are mapped and relevant matrices are constructed.
- Nodes contain position and constraint information. The total number of free and constrained degrees-of-freedom can be determined by calling appropriate methods of all nodes in the structure.
- Elements are aware of what nodes they connect and what materials/sections they are built from. Elements provide relevant matrices for simulation (stiffness, damping, mass) and, based on current displacements at nodes, can provide the current restoring force.
- Constitutive models provide the relevant relations used by element computations. These include stress/strain as well as damping and mass.
-

6.3.6 Analysis

- Analysis is specified by the IAnalysis interface, or the IDynamicAnalysis interface which also provides functionality for timestepping.

6.3.7 Timing and Synchronization

In order to control simulation timing for a real-time hybrid test, several semantics must be available:

- Timing sources can be specified either as an OS-inherent timer or an externally triggered timer.
- Synchronization is performed by waiting for time to pass as specified by a timing source, or by waiting for a signal from a specific thread.
- Timing requirements can be attached to an Analysis in order to determine if computation (or specific parts of the computation) violate timing requirements.

6.3.8 General Source Code Guidelines

The comment style is the JavaDoc (JavaDoc 2009) style, written to be compatible with Doxygen (Doxygen 2009) documentation generation program. A sample is included in the appendix.

Variable naming uses minimal Hungarian notation; specifically, static variables are prefixed with `s_`, global variables are prefixed with `g_`, and local variables have no prefix. Local and member variables are lowercase, while globals should be capitalized. Local variables and members are typically not distinguished; if distinguishing the two is important, a member can be explicitly pointed out using `this`. Capitalization is typically reserved for class names, not instance variables. Examples:

```
1 statevariables
2 this->membervariable
3 g_errorstream
4 s_singleton
```

Method naming is in Camel Case–that is, words run together with words after the first word capitalized. Method names are generally in the form "verbWordWordWord." Examples:

```
1 void eatLargeMeal();
2 int getWildebeestCount() const;
3 bool isChuckNorris() const;
4 virtual Stream watchMidgetWrestling();
```

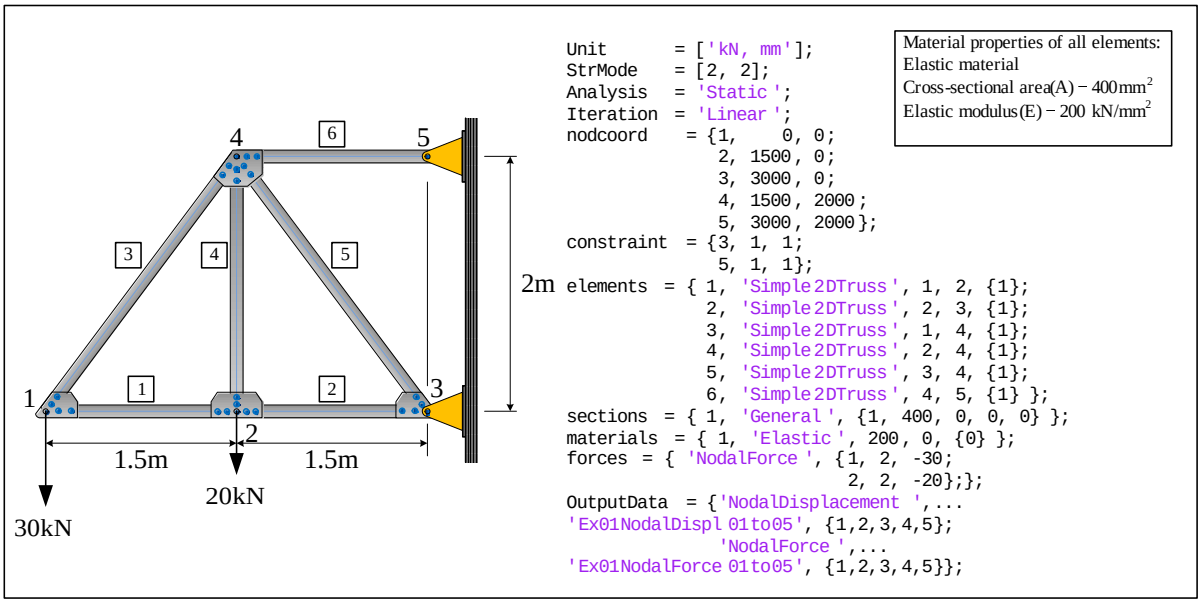
Chapter 7

MERCURY VALIDATION

7.1 Simple 2D Truss Validation

7.1.1 Example 01

Fig 7.1



Material properties of all elements:
Elastic material
Cross-sectional area(A) – 400mm^2
Elastic modulus(E) – 200 kN/mm^2

```
Unit      = ['kN, mm'];
StrMode   = [2, 2];
Analysis  = 'Static ';
Iteration = 'Linear ';
nodcoord    = {1,    0, 0;
               2, 1500, 0;
               3, 3000, 0;
               4, 1500, 2000;
               5, 3000, 2000};
constraint  = {3, 1, 1;
               5, 1, 1};
elements  = { 1, 'Simple2DTruss ', 1, 2, {1};
              2, 'Simple2DTruss ', 2, 3, {1};
              3, 'Simple2DTruss ', 1, 4, {1};
              4, 'Simple2DTruss ', 2, 4, {1};
              5, 'Simple2DTruss ', 3, 4, {1};
              6, 'Simple2DTruss ', 4, 5, {1} };
sections  = { 1, 'General ', {1, 400, 0, 0, 0} };
materials  = { 1, 'Elastic ', 200, 0, {0} };
forces = { 'NodalForce ', {1, 2, -30;
                           2, 2, -20};};
OutputData  = {'NodalDisplacement ',...
'Ex01NodalDispl 01to05', {1,2,3,4,5};
                 'NodalForce ',...
'Ex01NodalForce 01to05', {1,2,3,4,5}};
```

Fig. 7.1 Simple 2D Truss EX 01

7.1.2 Example 02

Fig 7.2

Example	Structural Mode	Iteration Mode	Integration Mode	Norm	Element 1	Element 2	Element 3	Section 1	Section 2	Section 3	Const. Model 1	Const. Model 2	Const. Model 3	Excitation 1	Excitation 2
Ex01	Static	Linear	-	-	Simple2DTruss	-	-	General	-	-	Elastic	-	-	NF	ND
Ex-02	Static	Linear	-	-	Simple2DTruss	-	-	General	-	-	Elastic	-	-	VNF	-
Ex03	Static	Linear	-	-	Simple3DTruss	-	-	General	-	-	Elastic	-	-	VNF	-
Ex04	Static	Linear	-	-	Simple4DTruss	-	-	General	-	-	Elastic	-	-	VND	-
Ex05	Static	NR	-	ForceNorm	Simple5DTruss	-	-	General	-	-	Elastic	Hardening	-	NF	-
Ex06	Static	NR	-	ForceNorm	Simple6DTruss	-	-	General	-	-	Elastic	Hardening	-	VNF	-
Ex07	Static	NR	-	ForceNorm	Simple7DTruss	-	-	General	-	-	Elastic	Hardening	-	VNF	-
Ex08	Static	NR	-	ForceNorm	Simple8DTruss	-	-	General	-	-	Elastic	Hardening	-	VND	-
Ex09	Static	NR	-	ForceNorm	Simple9DTruss	-	-	General	-	-	Elastic	Bilinear	-	VNF	-
Ex10	Static	NR	-	ForceNorm	Simple10DTruss	-	-	General	-	-	Elastic	ModifiedGMP	-	VNF	-
Ex11	Static	NR	-	ForceNorm	Simple11DTruss	-	-	General	-	-	AnisoDamage	-	-	VND	-
Ex12	Static	IS	-	ForceNorm	Simple12DTruss	-	-	General	-	-	ModifiedKP	-	-	VND	-
Ex13	Static	NR	-	ForceNorm	Simple2DTruss	-	-	General	-	-	Hardening	-	-	VNF	-
Ex14	Static	NR	-	ForceNorm	Simple2DTruss	-	-	General	-	-	Hardening	-	-	VNF	-
Ex15	Static	NR	-	ForceNorm	Simple2DTruss	-	-	General	-	-	Hardening	-	-	VNF	-
Ex16	Static	NR	-	ForceNorm	Simple2DTruss	-	-	General	-	-	Bilinear	-	-	VND	-
Ex17	Static	NR	-	ForceNorm	Simple2DTruss	-	-	General	-	-	Bilinear	-	-	VND	-
Ex18	Static	NR	-	ForceNorm	Simple2DTruss	-	-	General	-	-	ModifiedGMP	-	-	VND	-
Ex19	Static	NR	-	ForceNorm	Simple2DTruss	-	-	General	-	-	ModifiedGMP	-	-	VND	-
Ex20	Static	NR	-	ForceNorm	Simple2DTruss	-	-	General	-	-	ModifiedKP	-	-	VND	-
Ex21	Static	NR	-	ForceNorm	SB2DBC	-	-	General	-	-	Elastic	-	-	VNF	VND
Ex22	Static	NR	-	ForceNorm	SB2DBC	-	-	General	-	-	Elastic	-	-	VND	VEDF
Ex23	Static	NR	-	ForceNorm	FB2DBC	-	-	Fiber	-	-	ModifiedKP	ModifiedGMP	-	VNF	VND
Ex24	Static	NR	-	ForceNorm	FB2DBC	-	-	Fiber	-	-	ModifiedKP	ModifiedGMP	-	VNF	VND
Ex25	Static	NR	-	ForceNorm	FB2DBCNI	-	-	Fiber	-	-	ModifiedKP	ModifiedGMP	-	VNF	VND
Ex26	Static	NR	-	ForceNorm	FB2DBCNI	-	-	Fiber	-	-	ModifiedKP	ModifiedGMP	-	VNF	VND
Ex27	Static	IS	-	ForceNorm	FB2DBC	-	-	Layer	-	-	Hardening	-	-	NF	VND
Ex28	Static	IS	-	ForceNorm	FB2DBC	-	-	Layer	-	-	Hardening	Bilinear	-	NF	VND
Ex29	Static	IS	-	ForceNorm	FB2DBC	-	-	Layer	-	-	ModifiedGMP	Bilinear	-	NF	VND

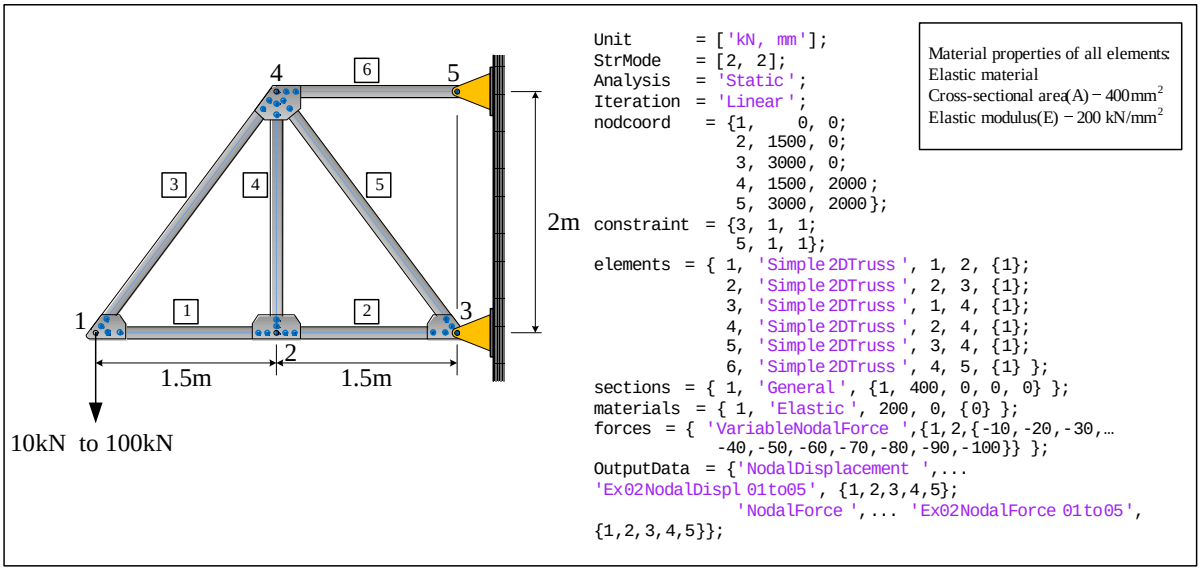


Fig. 7.2 Simple 2D Truss EX 02

7.1.3 Example 03

Fig 7.3

7.1.4 Example 04

Fig 7.4

7.1.5 Example 05

Fig 7.5

7.1.6 Example 06

Fig 7.6

7.1.7 Example 07

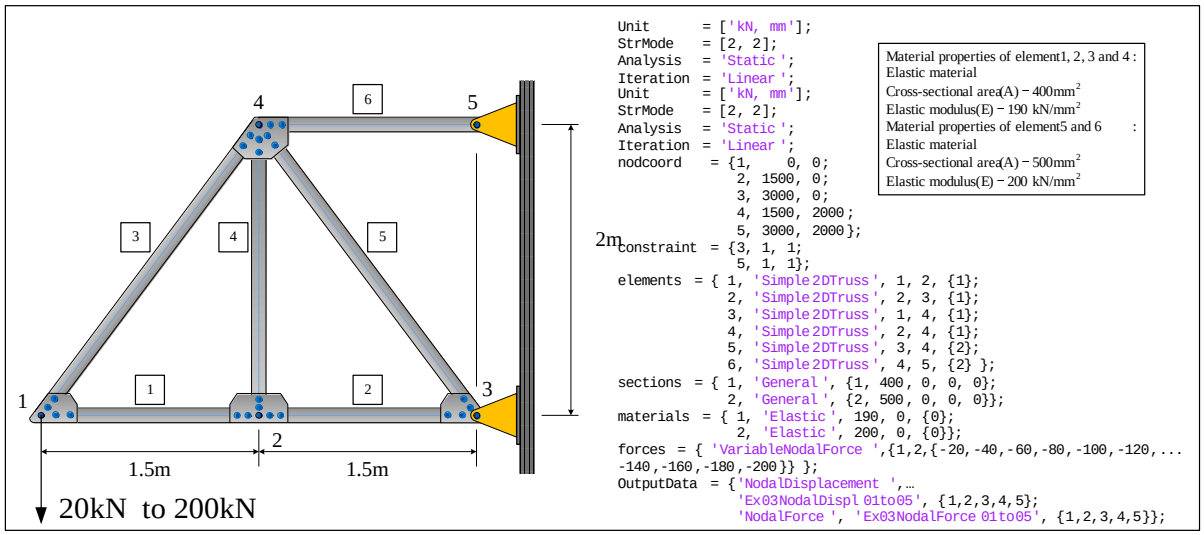
Fig 7.7

7.1.8 Example 08

Fig 7.8

7.1.9 Example 09

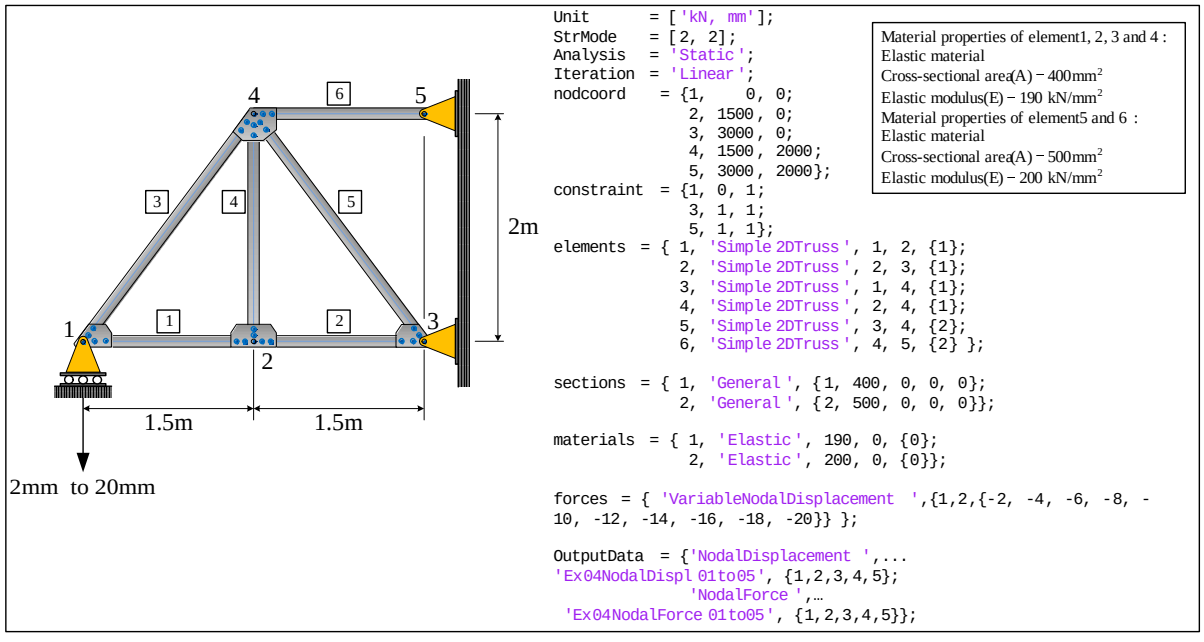
Fig 7.9




```
Unit      = ['kN, mm'];
StrMode   = [2, 2];
Analysis  = 'Static ';
Iteration = 'Linear ';
Unit      = ['kN, mm'];
StrMode   = [2, 2];
Analysis  = 'Static ';
Iteration = 'Linear ';
nodcoord   = {1,    0, 0;
              2, 1500, 0;
              3, 3000, 0;
              4, 1500, 2000;
              5, 3000, 2000};
constraint = {3, 1, 1;
              5, 1, 1};
elements = { 1, 'Simple 2DTruss ', 1, 2, {1};
             2, 'Simple 2DTruss ', 2, 3, {1};
             3, 'Simple 2DTruss ', 1, 4, {1};
             4, 'Simple 2DTruss ', 2, 4, {1};
             5, 'Simple 2DTruss ', 3, 4, {2};
             6, 'Simple 2DTruss ', 4, 5, {2} };
sections = { 1, 'General ', {1, 400, 0, 0, 0};
             2, 'General ', {2, 500, 0, 0, 0}};
materials = { 1, 'Elastic ', 190, 0, {0};
              2, 'Elastic ', 200, 0, {0}};
forces = { 'VariableNodalForce ',{1,2,{-20,-40,-60,-80,-100,-120,...
-140,-160,-180,-200}} };
OutputData = {'NodalDisplacement ',…
              'Ex03NodalDispl 01to05', {1,2,3,4,5};
              'NodalForce ', 'Ex03NodalForce 01to05', {1,2,3,4,5}};
```

Material properties of element1, 2, 3 and 4 :
Elastic material
Cross-sectional area(A) – 400mm^2
Elastic modulus(E) – 190 kN/mm^2
Material properties of element5 and 6 :
Elastic material
Cross-sectional area(A) – 500mm^2
Elastic modulus(E) – 200 kN/mm^2

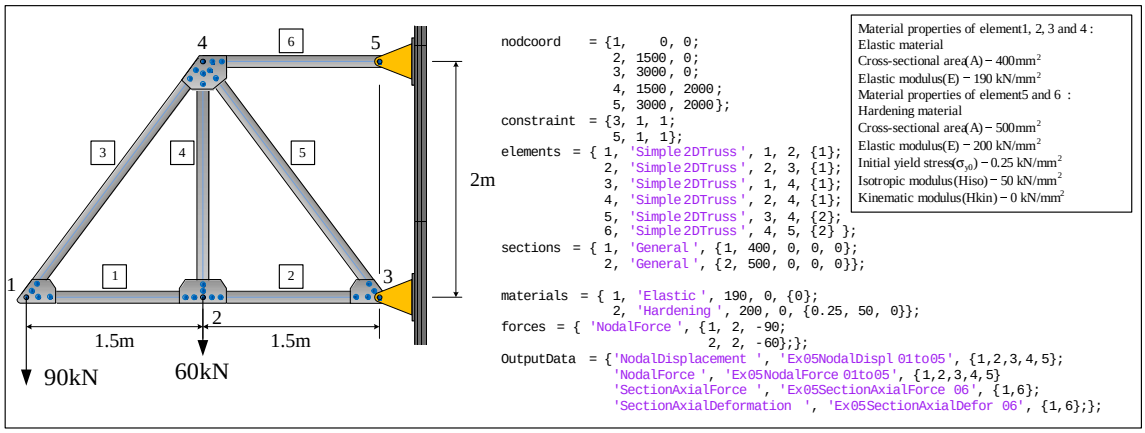
Fig. 7.3 Simple 2D Truss EX 03




```
Unit      = ['kN, mm'];
StrMode   = [2, 2];
Analysis  = 'Static';
Iteration = 'Linear';
nodcoord   = {1,    0, 0;
              2, 1500, 0;
              3, 3000, 0;
              4, 1500, 2000;
              5, 3000, 2000};
constraint = {1, 0, 1;
              3, 1, 1;
              5, 1, 1};
elements = { 1, 'Simple 2DTruss ', 1, 2, {1};
             2, 'Simple 2DTruss ', 2, 3, {1};
             3, 'Simple 2DTruss ', 1, 4, {1};
             4, 'Simple 2DTruss ', 2, 4, {1};
             5, 'Simple 2DTruss ', 3, 4, {2};
             6, 'Simple 2DTruss ', 4, 5, {2} };

sections = { 1, 'General ', {1, 400, 0, 0, 0};
             2, 'General ', {2, 500, 0, 0, 0}};

materials = { 1, 'Elastic', 190, 0, {0};
              2, 'Elastic', 200, 0, {0}};

forces = { 'VariableNodalDisplacement  ',{1,2,{-2, -4, -6, -8, -
10, -12, -14, -16, -18, -20}} };

OutputData  = {'NodalDisplacement ',...
'Ex04NodalDispl 01to05', {1,2,3,4,5};
                'NodalForce ',…
 'Ex04NodalForce 01to05', {1,2,3,4,5}};
```

Material properties of element1, 2, 3 and 4 :
Elastic material
Cross-sectional area(A) – 400mm^2
Elastic modulus(E) – 190 kN/mm^2
Material properties of element5 and 6 :
Elastic material
Cross-sectional area(A) – 500mm^2
Elastic modulus(E) – 200 kN/mm^2

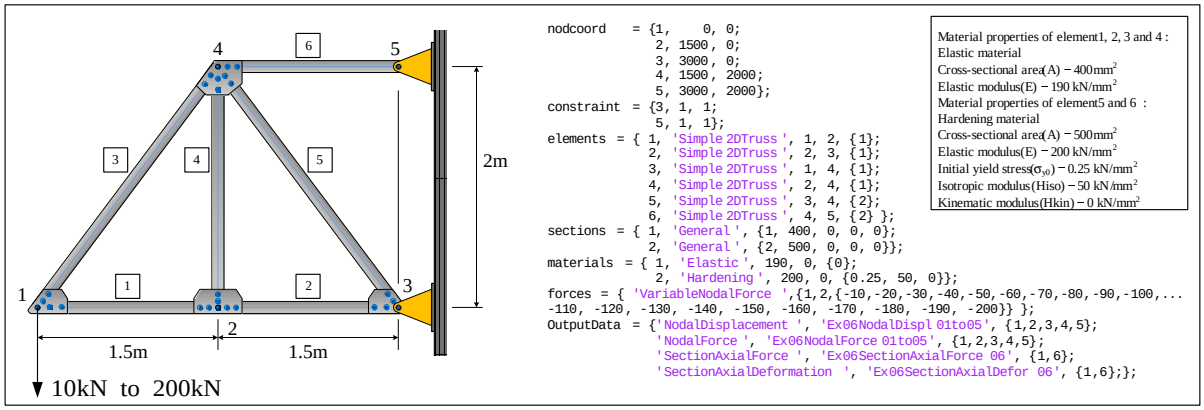
Fig. 7.4 Simple 2D Truss EX 04




```
nodcoord    = {1,    0, 0;
               2, 1500, 0;
               3, 3000, 0;
               4, 1500, 2000;
               5, 3000, 2000};
constraint  = {3, 1, 1;
               5, 1, 1};
elements  = { 1, 'Simple 2DTruss ', 1, 2, {1};
              2, 'Simple 2DTruss ', 2, 3, {1};
              3, 'Simple 2DTruss ', 1, 4, {1};
              4, 'Simple 2DTruss ', 2, 4, {1};
              5, 'Simple 2DTruss ', 3, 4, {2};
              6, 'Simple 2DTruss ', 4, 5, {2} };
sections  = { 1, 'General ', {1, 400, 0, 0, 0};
              2, 'General ', {2, 500, 0, 0, 0}};

materials  = { 1, 'Elastic ', 190, 0, {0};
               2, 'Hardening ', 200, 0, {0.25, 50, 0}};
forces = { 'NodalForce ', {1, 2, -90;
                           2, 2, -60};};
OutputData  = {'NodalDisplacement ', 'Ex05NodalDispl 01to05', {1,2,3,4,5};
               'NodalForce ', 'Ex05NodalForce 01to05', {1,2,3,4,5}
               'SectionAxialForce ', 'Ex05SectionAxialForce 06', {1,6};
               'SectionAxialDeformation  ', 'Ex05SectionAxialDefor 06', {1,6};};
```

Material properties of element1, 2, 3 and 4 :
Elastic material
Cross-sectional area(A) – 400mm^2
Elastic modulus(E) – 190 kN/mm^2
Material properties of element5 and 6 :
Hardening material
Cross-sectional area(A) – 500mm^2
Elastic modulus(E) – 200 kN/mm^2
Initial yield stress(σ_{y0}) – 0.25 kN/mm^2
Isotropic modulus(Hiso) – 50 kN/mm^2
Kinematic modulus(Hkin) – 0 kN/mm^2

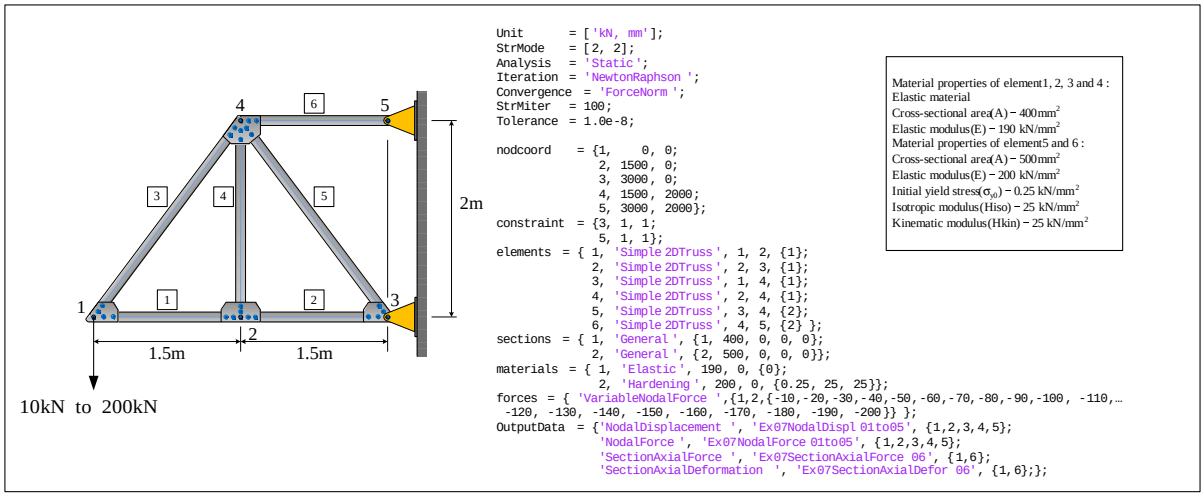
Fig. 7.5 Simple 2D Truss EX 05




```
nodcoord    = {1,    0, 0;
               2, 1500, 0;
               3, 3000, 0;
               4, 1500, 2000;
               5, 3000, 2000};
constraint  = {3, 1, 1;
               5, 1, 1};
elements  = { 1, 'Simple 2DTruss ', 1, 2, {1};
              2, 'Simple 2DTruss ', 2, 3, {1};
              3, 'Simple 2DTruss ', 1, 4, {1};
              4, 'Simple 2DTruss ', 2, 4, {1};
              5, 'Simple 2DTruss ', 3, 4, {2};
              6, 'Simple 2DTruss ', 4, 5, {2} };
sections  = { 1, 'General ', {1, 400, 0, 0, 0};
              2, 'General ', {2, 500, 0, 0, 0}};
materials  = { 1, 'Elastic ', 190, 0, {0};
               2, 'Hardening ', 200, 0, {0.25, 50, 0}};
forces = { 'VariableNodalForce ',{1,2,{-10,-20,-30,-40,-50,-60,-70,-80,-90,-100,...
-110, -120, -130, -140, -150, -160, -170, -180, -190, -200}} };
OutputData  = {'NodalDisplacement ', 'Ex06NodalDispl 01to05', {1,2,3,4,5};
               'NodalForce ', 'Ex06NodalForce 01to05', {1,2,3,4,5};
               'SectionAxialForce ', 'Ex06SectionAxialForce 06', {1,6};
               'SectionAxialDeformation  ', 'Ex06SectionAxialDefor 06', {1,6};};
```

Material properties of element1, 2, 3 and 4 :
Elastic material
Cross-sectional area(A) – 400mm^2
Elastic modulus(E) – 190 kN/mm^2
Material properties of element5 and 6 :
Hardening material
Cross-sectional area(A) – 500mm^2
Elastic modulus(E) – 200 kN/mm^2
Initial yield stress(σ_{y0}) – 0.25 kN/mm^2
Isotropic modulus(Hiso) – 50 kN/mm^2
Kinematic modulus(Hkin) – 0 kN/mm^2

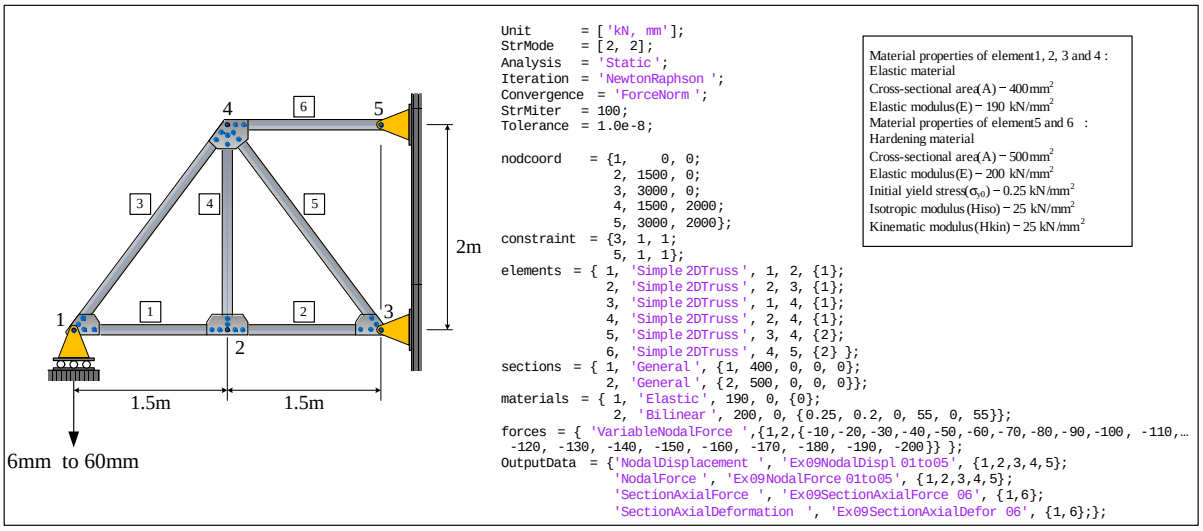
Fig. 7.6 Simple 2D Truss EX 06




```
Unit      = ['kN, mm'];
StrMode   = [2, 2];
Analysis  = 'Static';
Iteration = 'NewtonRaphson ';
Convergence = 'ForceNorm ';
StrMiter  = 100;
Tolerance = 1.0e-8;

nodcoord    = {1,    0, 0;
               2, 1500, 0;
               3, 3000, 0;
               4, 1500, 2000;
               5, 3000, 2000};
constraint  = {3, 1, 1;
               5, 1, 1};
elements  = { 1, 'Simple 2DTruss ', 1, 2, {1};
              2, 'Simple 2DTruss ', 2, 3, {1};
              3, 'Simple 2DTruss ', 1, 4, {1};
              4, 'Simple 2DTruss ', 2, 4, {1};
              5, 'Simple 2DTruss ', 3, 4, {2};
              6, 'Simple 2DTruss ', 4, 5, {2} };
sections  = { 1, 'General ', {1, 400, 0, 0, 0};
              2, 'General ', {2, 500, 0, 0, 0}};
materials  = { 1, 'Elastic ', 190, 0, {0};
               2, 'Hardening ', 200, 0, {0.25, 25, 25}};
forces = { 'VariableNodalForce ',{1,2,{-10,-20,-30,-40,-50,-60,-70,-80,-90,-100, -110,…
 -120, -130, -140, -150, -160, -170, -180, -190, -200}} };
OutputData  = {'NodalDisplacement ', 'Ex07NodalDispl 01to05', {1,2,3,4,5};
               'NodalForce ', 'Ex07NodalForce 01to05', {1,2,3,4,5};
               'SectionAxialForce ', 'Ex07SectionAxialForce 06', {1,6};
               'SectionAxialDeformation  ', 'Ex07SectionAxialDefor 06', {1,6};};
```

Material properties of element1, 2, 3 and 4 :
Elastic material
Cross-sectional area(A) – 400mm^2
Elastic modulus(E) – 190 kN/mm^2
Material properties of element5 and 6 :
Cross-sectional area(A) – 500mm^2
Elastic modulus(E) – 200 kN/mm^2
Initial yield stress(σ_{y0}) – 0.25 kN/mm^2
Isotropic modulus(Hiso) – 25 kN/mm^2
Kinematic modulus(Hkin) – 25 kN/mm^2

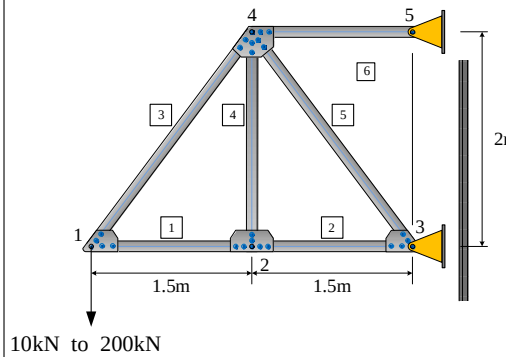
Fig. 7.7 Simple 2D Truss EX 07




```
Unit      = ['kN, mm'];
StrMode   = [2, 2];
Analysis  = 'Static';
Iteration = 'NewtonRaphson ';
Convergence = 'ForceNorm ';
StrMiter  = 100;
Tolerance = 1.0e-8;

nodcoord    = {1,    0, 0;
               2, 1500, 0;
               3, 3000, 0;
               4, 1500, 2000;
               5, 3000, 2000};
constraint  = {3, 1, 1;
               5, 1, 1};
elements = { 1, 'Simple 2DTruss ', 1, 2, {1};
             2, 'Simple 2DTruss ', 2, 3, {1};
             3, 'Simple 2DTruss ', 1, 4, {1};
             4, 'Simple 2DTruss ', 2, 4, {1};
             5, 'Simple 2DTruss ', 3, 4, {2};
             6, 'Simple 2DTruss ', 4, 5, {2} };
sections = { 1, 'General ', {1, 400, 0, 0, 0};
             2, 'General ', {2, 500, 0, 0, 0}};
materials = { 1, 'Elastic ', 190, 0, {0};
              2, 'Bilinear ', 200, 0, {0.25, 0.2, 0, 55, 0, 55}};
forces = { 'VariableNodalForce ',{1,2,{-10,-20,-30,-40,-50,-60,-70,-80,-90,-100, -110,…
 -120, -130, -140, -150, -160, -170, -180, -190, -200}} };
OutputData  = {'NodalDisplacement ', 'Ex09NodalDispl 01to05', {1,2,3,4,5};
               'NodalForce ', 'Ex09NodalForce 01to05', {1,2,3,4,5};
               'SectionAxialForce ', 'Ex09SectionAxialForce 06', {1,6};
               'SectionAxialDeformation ', 'Ex09SectionAxialDefor 06', {1,6};};
```

Material properties of element1, 2, 3 and 4 :
Elastic material
Cross-sectional area(A) – 400mm^2
Elastic modulus(E) – 190 kN/mm^2
Material properties of element5 and 6 :
Hardening material
Cross-sectional area(A) – 500mm^2
Elastic modulus(E) – 200 kN/mm^2
Initial yield stress(σ_{y0}) – 0.25 kN/mm^2
Isotropic modulus(Hiso) – 25 kN/mm^2
Kinematic modulus(Hkin) – 25 kN/mm^2

Fig. 7.8 Simple 2D Truss EX 08




```
Unit      = ['kN, mm'];
StrMode   = [2, 2];
Analysis  = 'Static';
Iteration = 'NewtonRaphson ';
Convergence = 'ForceNorm ';
StrMiter  = 100;
Tolerance = 1.0e-8;

nodcoord    = {1,    0, 0;
               2, 1500, 0;
               3, 3000, 0;
               4, 1500, 2000;
               5, 3000, 2000};
constraint  = {3, 1, 1;
               5, 1, 1};
elements = { 1, 'Simple 2DTruss ', 1, 2, {1};
             2, 'Simple 2DTruss ', 2, 3, {1};
             3, 'Simple 2DTruss ', 1, 4, {1};
             4, 'Simple 2DTruss ', 2, 4, {1};
             5, 'Simple 2DTruss ', 3, 4, {2};
             6, 'Simple 2DTruss ', 4, 5, {2} };
sections = { 1, 'General ', {1, 400, 0, 0, 0};
             2, 'General ', {2, 500, 0, 0, 0}};
materials = { 1, 'Elastic ', 190, 0, {0};
              2, 'Bilinear ', 200, 0, {0.25, 0.2, 0, 55, 0, 55}};
forces = { 'VariableNodalForce ',{1,2,{-10,-20,-30,-40,-50,-60,-70,-80,-90,-100, -110,...
-120, -130, -140, -150, -160, -170, -180, -190, -200}} };
OutputData = {'NodalDisplacement ', 'Ex09NodalDispl 01to05', {1,2,3,4,5};
              'NodalForce ', 'Ex09NodalForce 01to05', {1,2,3,4,5}
              'SectionAxialForce ', 'Ex09SectionAxialForce 06', {1,6};
              'SectionAxialDeformation ', 'Ex09SectionAxialDefor 06', {1,6};};
```

Material properties of element1, 2, 3 and 4 :
Elastic material
Cross-sectional area(A) – 400mm^2
Elastic modulus(E) – 190 kN/mm^2
Material properties of element5 and 6 :
Bilinear material
Cross-sectional area(A) – 500mm^2
Elastic modulus(E) – 200 kN/mm^2
Initial yield stress(σ_{y0}) – 0.25 kN/mm^2
a1=0, a2=55, a3=0, a4=55

Fig. 7.9 Simple 2D Truss EX 09

7.1.10 Example 10

Fig 7.10

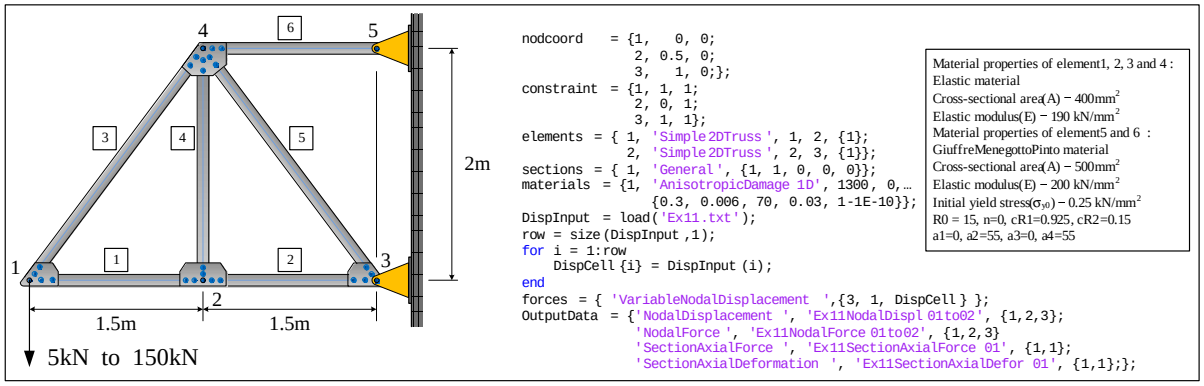


Fig. 7.10 Simple 2D Truss EX 10

7.1.11 Example 11

Fig 7.11

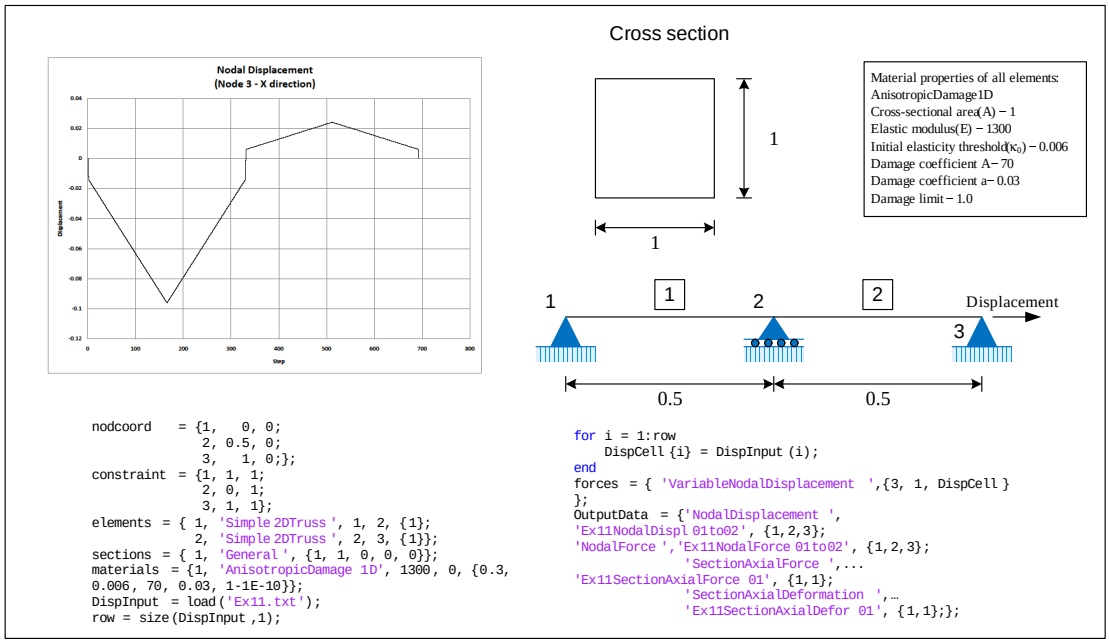
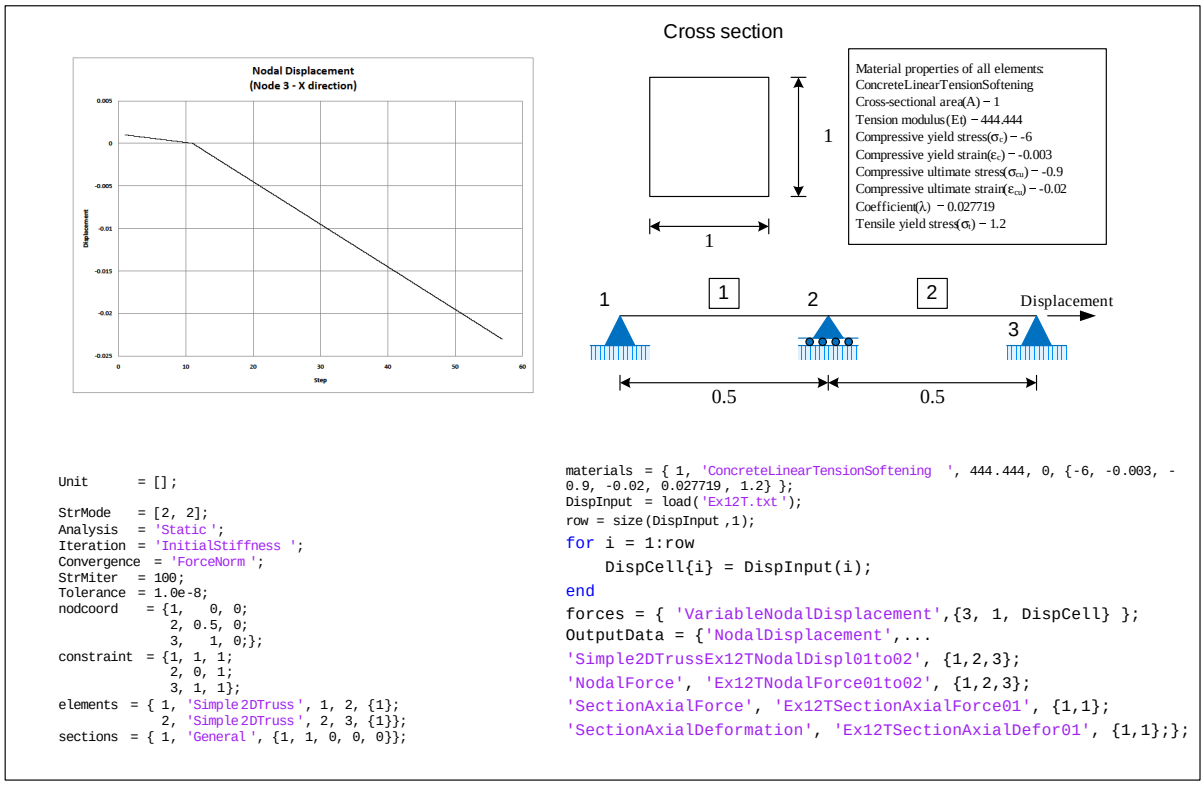


Fig. 7.11 Simple 2D Truss EX 11

7.1.12 Example 12

Fig 7.12




```
Unit      = [];

StrMode    = [2, 2];
Analysis   = 'Static ';
Iteration  = 'InitialStiffness ';
Convergence  = 'ForceNorm ';
StrMiter   = 100;
Tolerance  = 1.0e-8;
nodcoord    = {1,   0, 0;
               2, 0.5, 0;
               3,   1, 0;};
constraint  = {1, 1, 1;
               2, 0, 1;
               3, 1, 1};
elements  = { 1, 'Simple2DTruss ', 1, 2, {1};
              2, 'Simple2DTruss ', 2, 3, {1}};
sections  = { 1, 'General ', {1, 1, 0, 0, 0}};
```

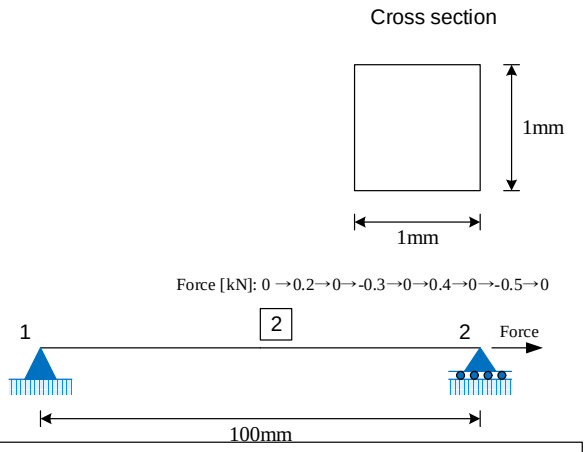
```
materials  = { 1, 'ConcreteLinearTensionSoftening  ', 444.444, 0, {-6, -0.003, -
0.9, -0.02, 0.027719, 1.2} };
DispInput  = load('Ex12T.txt');
row = size(DispInput,1);
for i = 1:row
    DispCell{i} = DispInput(i);
end
forces = { 'VariableNodalDisplacement',{3, 1, DispCell} };
OutputData = {'NodalDisplacement',...
'Simple2DTrussEx12TNodalDispl01to02', {1,2,3};
'NodalForce', 'Ex12TNodalForce01to02', {1,2,3};
'SectionAxialForce', 'Ex12TSectionAxialForce01', {1,1};
'SectionAxialDeformation', 'Ex12TSectionAxialDefor01', {1,1};};
```

Fig. 7.12 Simple 2D Truss EX 12

7.1.13 Example 13 to 15

Fig 7.13

```
Unit      = ['kN, mm'];
StrMode   = [2, 2];
Analysis  = 'Static';
Iteration = 'NewtonRaphson ';
Convergence = 'ForceNorm ';
StrMiter  = 10;
Tolerance = 1.0e-8;
nodcoord   = {1,    0, 0;
              2,  100, 0};
constraint = {1, 1, 1;
              2, 0, 1};
elements = { 1, 'Simple 2DTruss', 1, 2, {1}};
sections = { 1, 'General', {1, 1, 0, 0, 0}};
materials = { 1, 'Hardening', 200, 0, {0.25, 50, 0} };
ForceInput = load('Ex13.txt');
row = size(ForceInput,1);
for i = 1:row
    ForceCell{i} = ForceInput(i);
end
forces = { 'VariableNodalForce ',{2,1,ForceCell} };
OutputData = {'NodalDisplacement ', 'Ex13NodalDispl 02', {2};
              'NodalForce ', 'Ex13NodalForce 02', {2};
              'SectionAxialForce ', 'Ex13SectionAxialForce 01', {1,1};
              'SectionAxialDeformation ', 'Ex13SectionAxialDefor 01',
{1,1};};
```



Ex 13 : Isotropic Material ; Ex 14 : Kinematic Material ; Ex 15 : Hardeing Material ;
Material properties of all elements : Hardening material
Cross-sectional area (A) − 1
Elastic modulus (E) − 200 kN/mm^2
Initial yield stress (σ_{y0}) − 0.25 kN/mm^2
Isotropic modulus (Hiso) − 50 kN/mm^2, 0kN/mm^2, 25 kN/mm^2
Kinematic modulus (Hkin) − 0 kN/mm^2, 50kN/mm^2, 25 kN/mm^2

Fig. 7.13 Simple 2D Truss EX 13 to 15

Fig 7.14 Results for Isotropic Material (Ex 13) Fig **??** Results for Kinematic Material (Ex 14) Fig **??** Results for Hardening Material (Ex 15)

Fig 7.15 Mercury Matlab results for examples 13 to 15. Fig **??** Full results for examples 13 to 15.

7.1.14 Example 16 and 17

Fig **??**

Fig 7.16 Results for Bilinear Material OpenSees/Mercury Matlab comparison Fig **??** Results for Bilinear Isotropic Material OpenSees/Mercury Matlab comparison.

Fig 7.17 Mercury Matlab Billinear/Bilinear Isotropic comprasion.

Fig 7.18 Full results for examples 16 and 17

7.1.15 Examples 18 and 19

Fig 7.19 Fig **??**

Fig 7.20

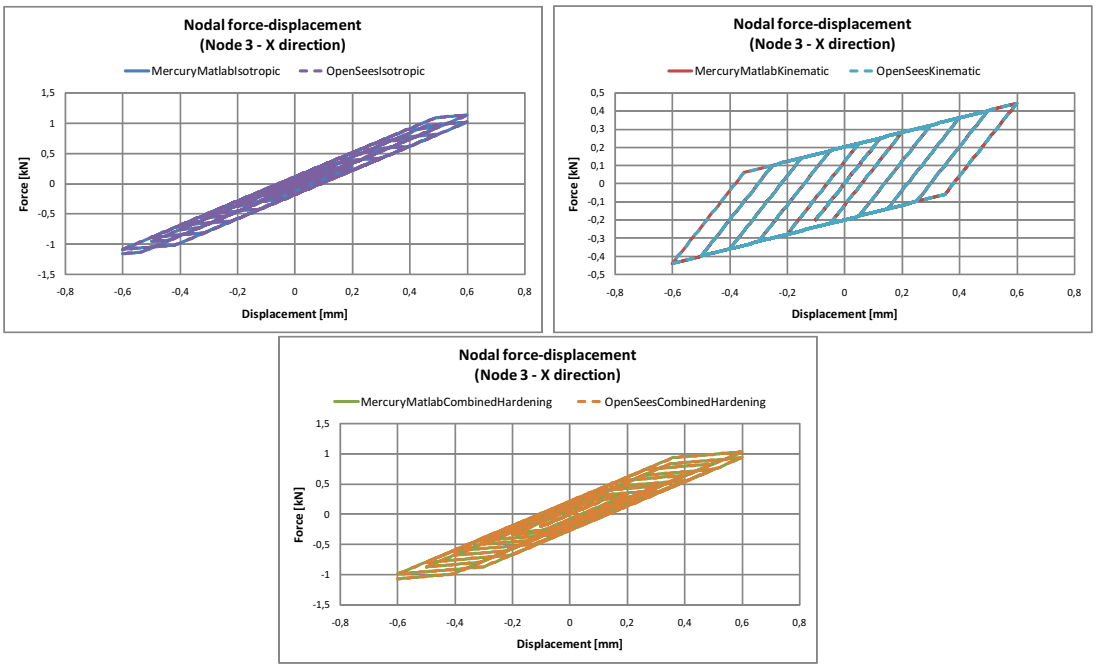


Fig. 7.14 Ex13 Isotropic Model, Ex 14 Kinematic Model, Ex 15 Hardening Model

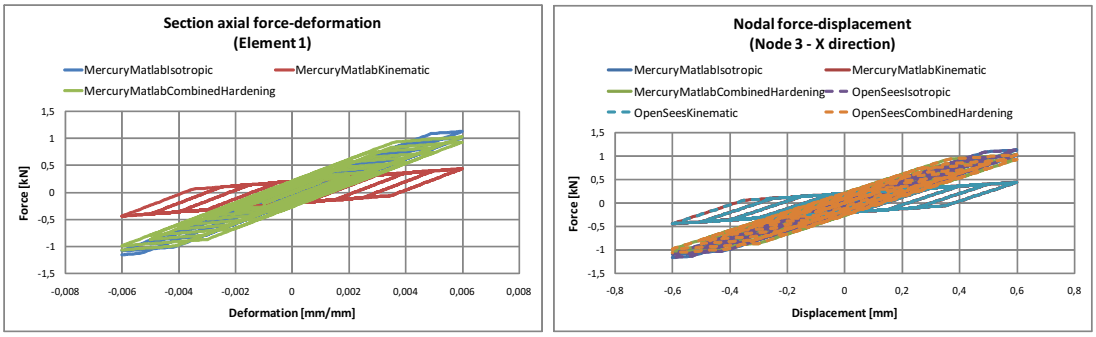


Fig. 7.15 Example 13 to 15 Mercury Results, Complete Results for Example 13 to 15

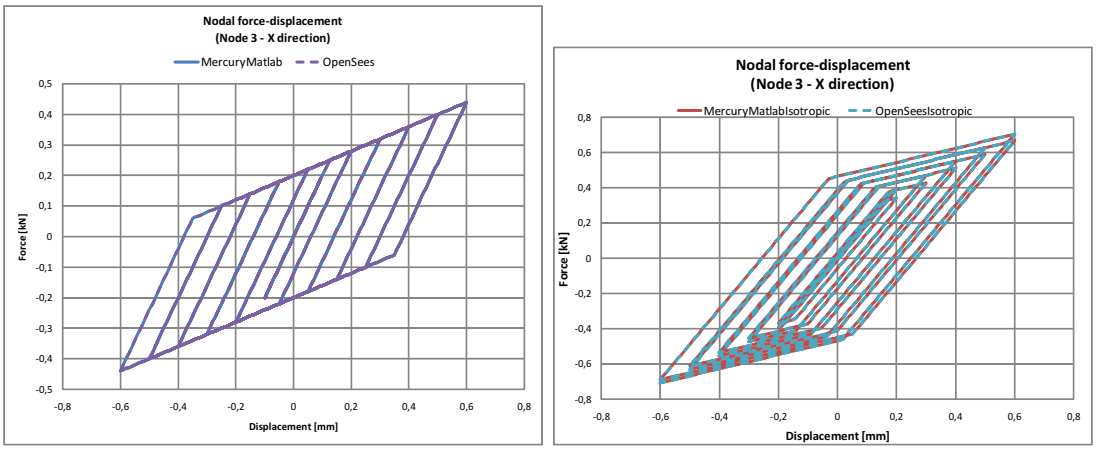


Fig. 7.16 Ex 16 Bilinear material for OpenSees and Mercury, Ex 17 Bilinear Isotropic Material for Opensees and Mercury

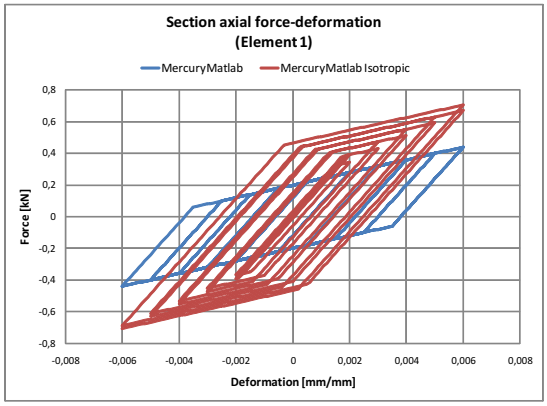


Fig. 7.17 Ex16 and Ex17 Billinear/Bilinear Isotropic comprasion.

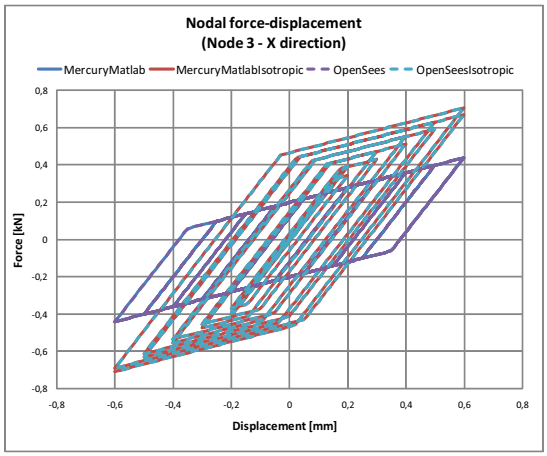


Fig. 7.18 Complete Results for Examples 16 and 17

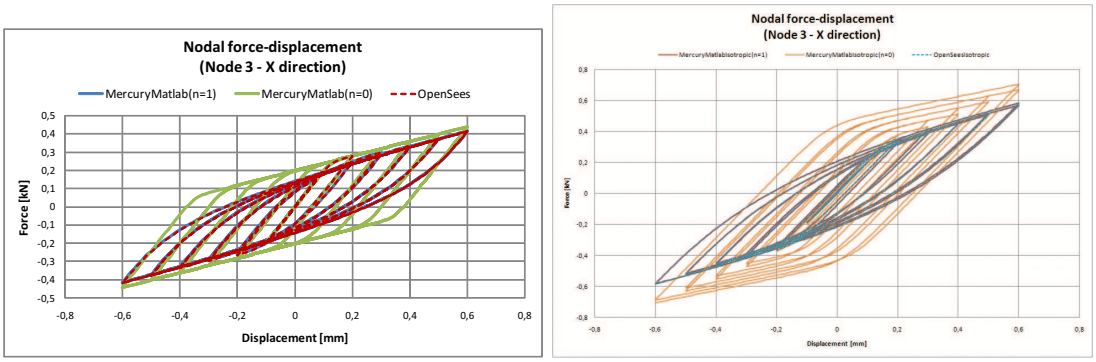


Fig. 7.19 Example 18, Example 19 Isotropic Material

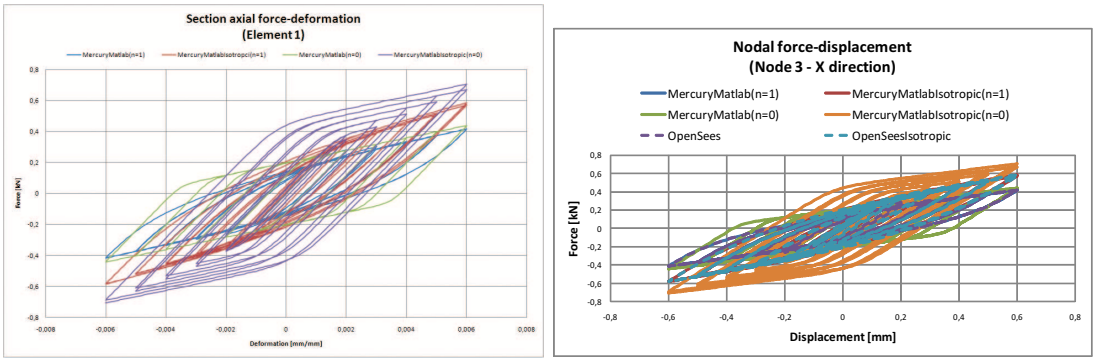
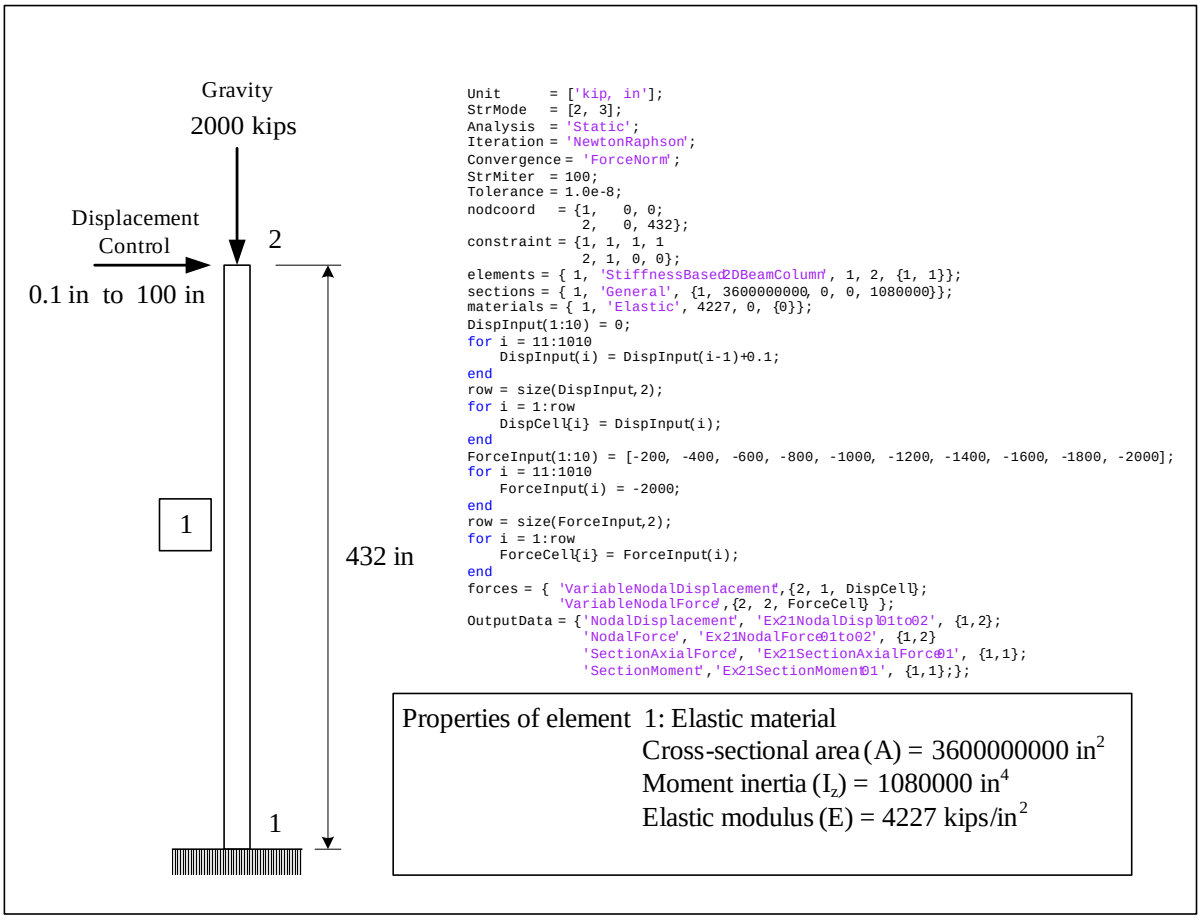


Fig. 7.20 Example 18 and 19 Section Axial force/Deformation,Full Results for examples 18 and 19

7.2 Beam Column Validation

7.2.1 Example 21

Fig 7.21




```
Unit      = ['kip, in'];
StrMode   = [2, 3];
Analysis  = 'Static';
Iteration = 'NewtonRaphson';
Convergence = 'ForceNorm';
StrMiter  = 100;
Tolerance = 1.0e-8;
nodcoord   = {1,   0, 0;
              2,   0, 432};
constraint = {1, 1, 1, 1
              2, 1, 0, 0};
elements = { 1, 'StiffnessBased2DBeamColumn', 1, 2, {1, 1}};
sections = { 1, 'General', {1, 3600000000, 0, 0, 1080000}};
materials = { 1, 'Elastic', 4227, 0, {0}};
DispInput(1:10) = 0;
for i = 11:1010
    DispInput(i) = DispInput(i-1)+0.1;
end
row = size(DispInput,2);
for i = 1:row
    DispCell{i} = DispInput(i);
end
ForceInput(1:10) = [-200, -400, -600, -800, -1000, -1200, -1400, -1600, -1800, -2000];
for i = 11:1010
    ForceInput(i) = -2000;
end
row = size(ForceInput,2);
for i = 1:row
    ForceCell{i} = ForceInput(i);
end
forces = { 'VariableNodalDisplacement',{2, 1, DispCell};
           'VariableNodalForce',{2, 2, ForceCell} };
OutputData = {'NodalDisplacement', 'Ex21NodalDisplo1to02', {1,2};
              'NodalForce', 'Ex21NodalForce01to02', {1,2}
              'SectionAxialForce', 'Ex21SectionAxialForce01', {1,1};
              'SectionMoment','Ex21SectionMoment01', {1,1};};
```

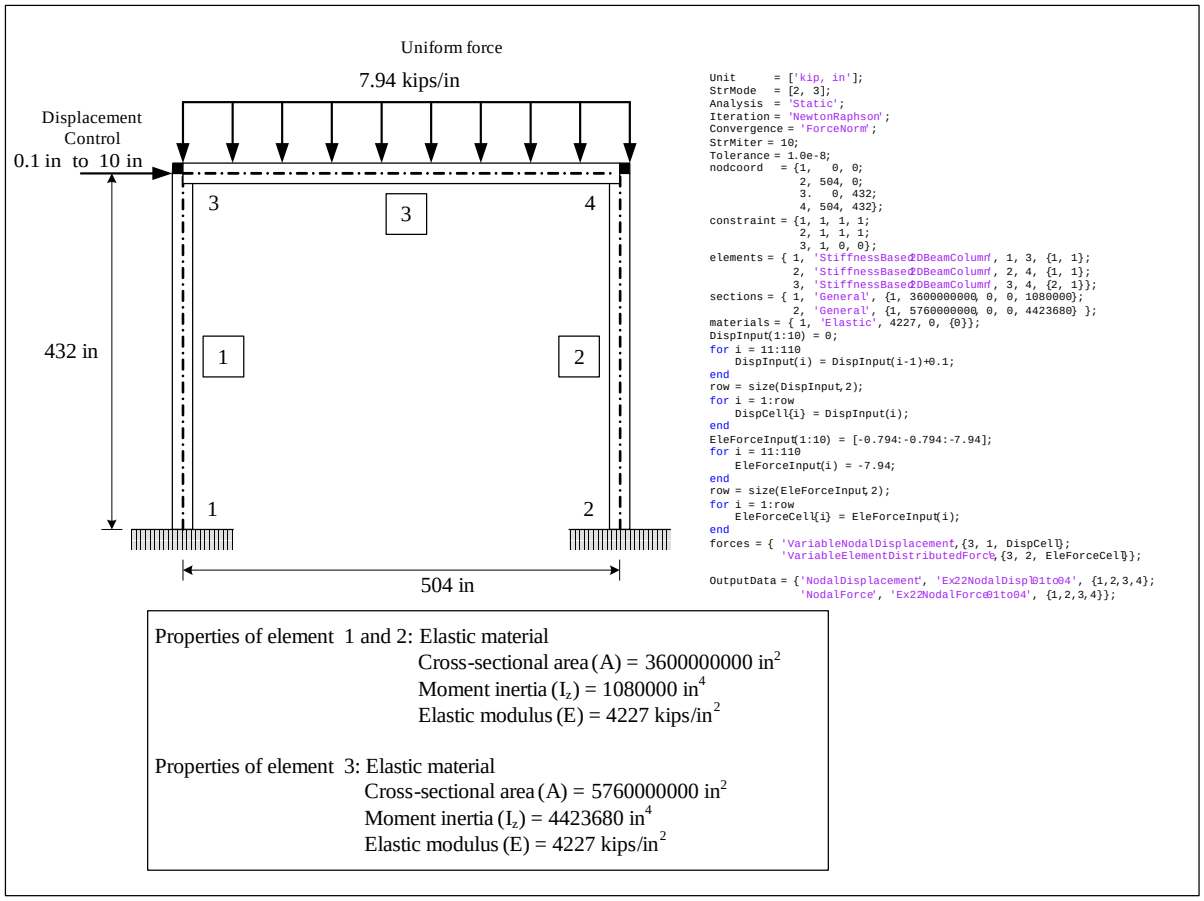
Fig. 7.21 Example 21

7.2.2 Example 22

Fig 7.22

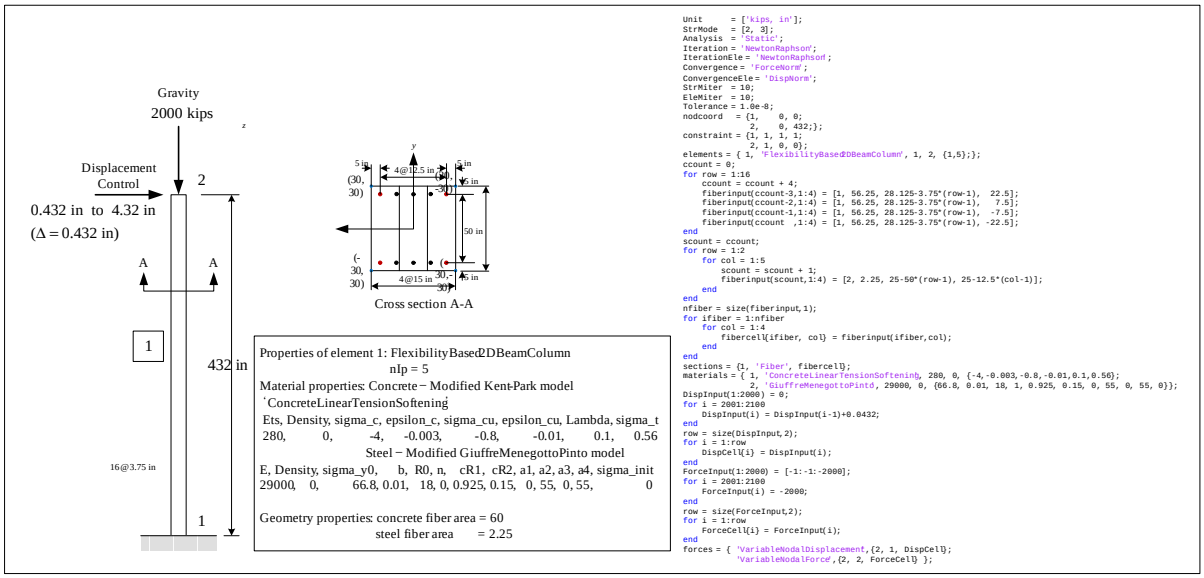
7.2.3 Example 23 to 26

Fig 7.23
Fig 7.24
Fig 7.25
Fig 7.26
Fig 7.27



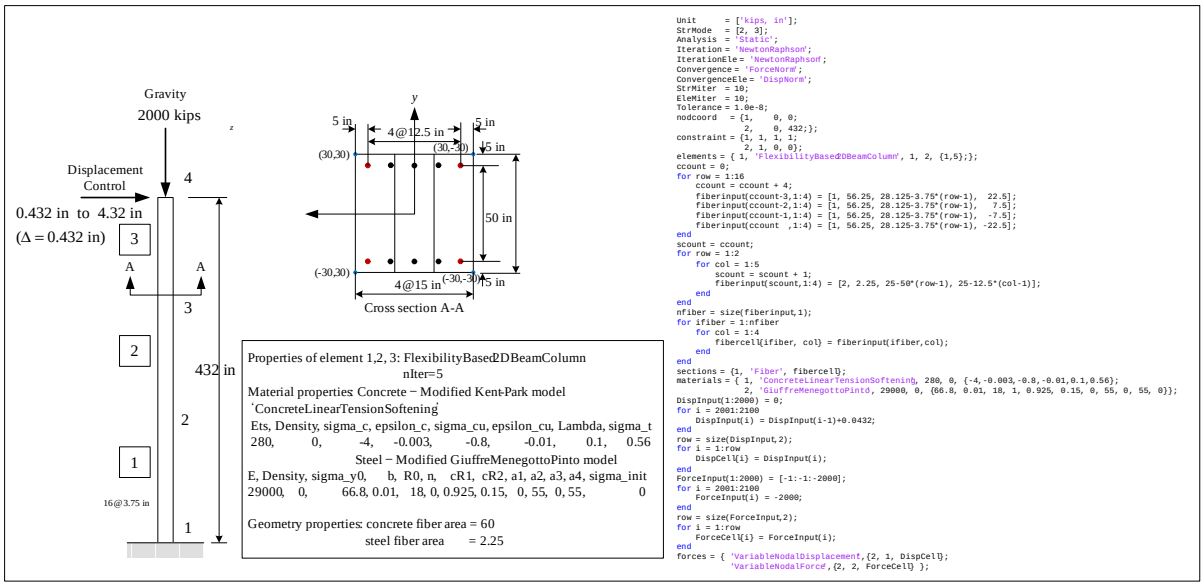

```
Unit      = ['kip, in'];
StrMode   = [2, 3];
Analysis  = 'Static';
Iteration = 'NewtonRaphson';
Convergence = 'ForceNorm';
StrMiter = 10;
Tolerance = 1.0e-8;
nodcoord   = {1,   0, 0;
              2, 504, 0;
              3,   0, 432;
              4, 504, 432};
constraint = {1, 1, 1, 1;
              2, 1, 1, 1;
              3, 1, 0, 0};
elements = { 1, 'StiffnessBased2DBeamColumn', 1, 3, {1, 1};
             2, 'StiffnessBased2DBeamColumn', 2, 4, {1, 1};
             3, 'StiffnessBased2DBeamColumn', 3, 4, {2, 1}};
sections = { 1, 'General', {1, 3600000000, 0, 0, 1080000};
             2, 'General', {1, 5760000000, 0, 0, 4423680} };
materials = { 1, 'Elastic', 4227, 0, {0}};
DispInput(1:10) = 0;
for i = 11:110
    DispInput(i) = DispInput(i-1)+0.1;
end
row = size(DispInput,2);
for i = 1:row
    DispCell{i} = DispInput(i);
end
EleForceInput(1:10) = [-0.794:-0.794:-7.94];
for i = 11:110
    EleForceInput(i) = -7.94;
end
row = size(EleForceInput,2);
for i = 1:row
    EleForceCell{i} = EleForceInput(i);
end
forces = { 'VariableNodalDisplacement',{3, 1, DispCell};
           'VariableElementDistributedForce',{3, 2, EleForceCell}};

OutputData = {'NodalDisplacement', 'Ex22NodalDisp01to04', {1,2,3,4};
              'NodalForce', 'Ex22NodalForce01to04', {1,2,3,4}};
```

Fig. 7.22 Example 22



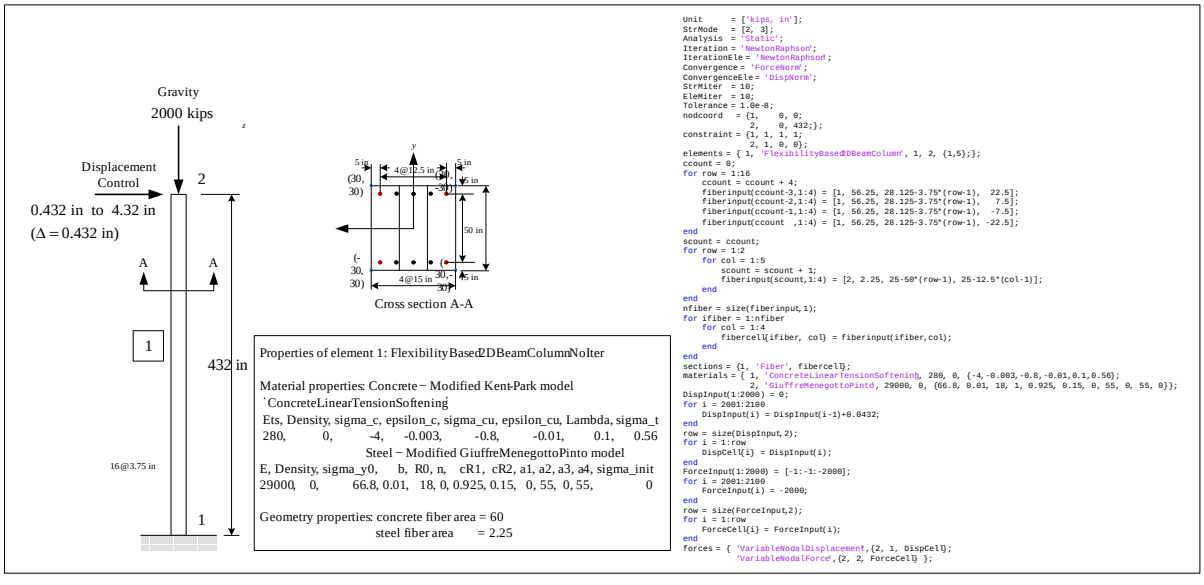

```
Unit      = ['kips, in'];
StrMode   = [2, 3];
Analysis  = 'Static';
Iteration = 'NewtonRaphson';
IterationEle = 'NewtonRaphson';
Convergence = 'ForceNorm';
ConvergenceEle = 'DispNorm';
StrMiter  = 10;
EleMiter  = 10;
Tolerance = 1.0e-8;
nodcoord   = {1,    0, 0;
              2,    0, 432;};
constraint = {1, 1, 1, 1;
              2, 1, 0, 0};
elements = { 1, 'FlexibilityBased2DBeamColumn', 1, 2, {1,5};};
ccount = 0;
for row = 1:16
    ccount = ccount + 4;
    fiberinput(ccount-3,1:4) = [1, 56.25, 28.125-3.75*(row-1),  22.5];
    fiberinput(ccount-2,1:4) = [1, 56.25, 28.125-3.75*(row-1),   7.5];
    fiberinput(ccount-1,1:4) = [1, 56.25, 28.125-3.75*(row-1),  -7.5];
    fiberinput(ccount  ,1:4) = [1, 56.25, 28.125-3.75*(row-1), -22.5];
end
scount = ccount;
for row = 1:2
    for col = 1:5
        scount = scount + 1;
        fiberinput(scount,1:4) = [2, 2.25, 25-50*(row-1), 25-12.5*(col-1)];
    end
end
nfiber = size(fiberinput,1);
for ifiber = 1:nfiber
    for col = 1:4
        fibercell{ifiber, col} = fiberinput(ifiber,col);
    end
end
sections = {1, 'Fiber', fibercell};
materials = { 1, 'ConcreteLinearTensionSoftening', 280, 0, {-4,-0.003,-0.8,-0.01,0.1,0.56};
              2, 'GiuffreMenegottoPinto', 29000, 0, {66.8, 0.01, 18, 1, 0.925, 0.15, 0, 55, 0, 55, 0};};
DispInput(1:2000) = 0;
for i = 2001:2100
    DispInput(i) = DispInput(i-1)+0.0432;
end
row = size(DispInput,2);
for i = 1:row
    DispCell{i} = DispInput(i);
end
ForceInput(1:2000) = [-1:-1:-2000];
for i = 2001:2100
    ForceInput(i) = -2000;
end
row = size(ForceInput,2);
for i = 1:row
    ForceCell{i} = ForceInput(i);
end
forces = { 'VariableNodalDisplacement',{2, 1, DispCell};
           'VariableNodalForce',{2, 2, ForceCell} };
```

Fig. 7.23 Example 23



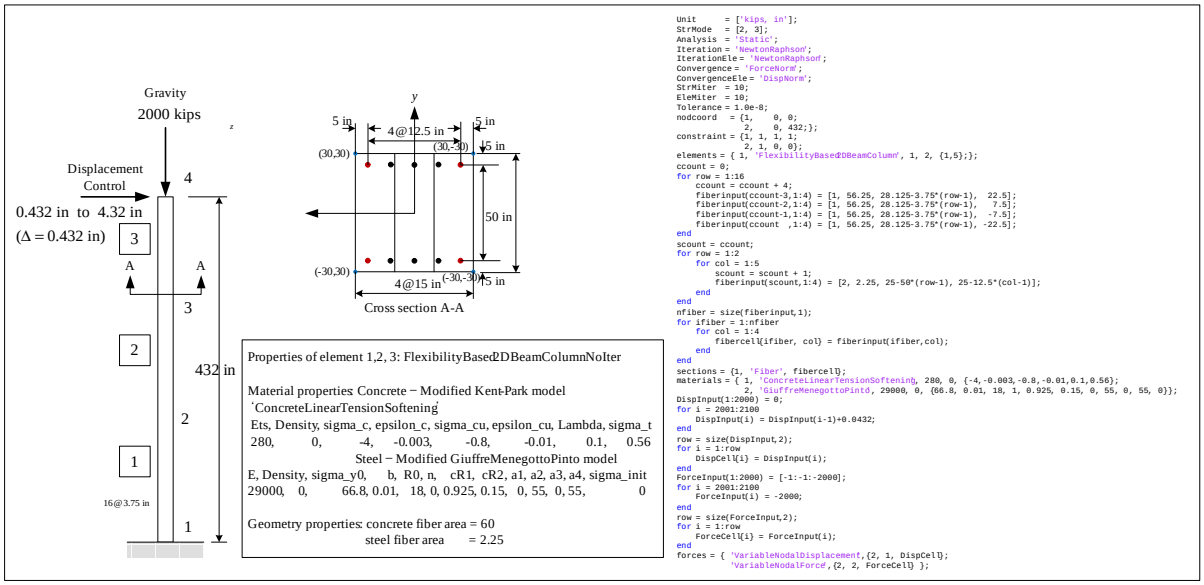

```
Unit      = ['kips, in'];
StrMode   = [2, 3];
Analysis  = 'Static';
Iteration = 'NewtonRaphson';
IterationEle = 'NewtonRaphson';
Convergence= 'ForceNorm';
ConvergenceEle= 'DispNorm';
StrMiter  = 10;
EleMiter  = 10;
Tolerance = 1.0e-8;
nodcoord   = {1,    0, 0;
              2,    0, 432;};
constraint = {1, 1, 1, 1;
              2, 1, 0, 0};
elements = { 1, 'FlexibilityBased2DBeamColumn', 1, 2, {1,5};};
ccount = 0;
for row = 1:16
    ccount = ccount + 4;
    fiberinput(ccount-3,1:4) = [1, 56.25, 28.125-3.75*(row-1),  22.5];
    fiberinput(ccount-2,1:4) = [1, 56.25, 28.125-3.75*(row-1),   7.5];
    fiberinput(ccount-1,1:4) = [1, 56.25, 28.125-3.75*(row-1),  -7.5];
    fiberinput(ccount  ,1:4) = [1, 56.25, 28.125-3.75*(row-1), -22.5];
end
scount = ccount;
for row = 1:2
    for col = 1:5
        scount = scount + 1;
        fiberinput(scount,1:4) = [2, 2.25, 25-50*(row-1), 25-12.5*(col-1)];
    end
end
nfiber = size(fiberinput,1);
for ifiber = 1:nfiber
    for col = 1:4
        fibercell{ifiber, col} = fiberinput(ifiber,col);
    end
end
sections = {1, 'Fiber', fibercell};
materials = { 1, 'ConcreteLinearTensionSoftening', 280, 0, {-4,-0.003,-0.8,-0.01,0.1,0.56};
              2, 'GiuffreMenegottoPinto', 29000, 0, {66.8, 0.01, 18, 1, 0.925, 0.15, 0, 55, 0, 55, 0}};
DispInput(1:2000) = 0;
for i = 2001:2100
    DispInput(i) = DispInput(i-1)+0.0432;
end
row = size(DispInput,2);
for i = 1:row
    DispCell{i} = DispInput(i);
end
ForceInput(1:2000) = [-1:-1:-2000];
for i = 2001:2100
    ForceInput(i) = -2000;
end
row = size(ForceInput,2);
for i = 1:row
    ForceCell{i} = ForceInput(i);
end
forces = { 'VariableNodalDisplacement',{2, 1, DispCell};
           'VariableNodalForce',{2, 2, ForceCell} };
```

Fig. 7.24 Example 24




```
Unit      = ['kips, in'];
StrMode   = [2, 3];
Analysis  = 'Static';
Iteration = 'NewtonRaphson';
IterationEle = 'NewtonRaphson';
Convergence= 'ForceNorm';
ConvergenceEle= 'DispNorm';
StrMiter  = 10;
EleMiter  = 10;
Tolerance = 1.0e-8;
nodcoord   = {1,    0, 0;
              2,    0, 432;};
constraint = {1, 1, 1, 1;
              2, 1, 0, 0};
elements = { 1, 'FlexibilityBased2DBeamColumn', 1, 2, {1,5};};
ccount = 0;
for row = 1:16
    ccount = ccount + 4;
    fiberinput(ccount-3,1:4) = [1, 56.25, 28.125-3.75*(row-1),  22.5];
    fiberinput(ccount-2,1:4) = [1, 56.25, 28.125-3.75*(row-1),   7.5];
    fiberinput(ccount-1,1:4) = [1, 56.25, 28.125-3.75*(row-1),  -7.5];
    fiberinput(ccount  ,1:4) = [1, 56.25, 28.125-3.75*(row-1), -22.5];
end
scount = ccount;
for row = 1:2
    for col = 1:5
        scount = scount + 1;
        fiberinput(scount,1:4) = [2, 2.25, 25-50*(row-1), 25-12.5*(col-1)];
    end
end
nfiber = size(fiberinput,1);
for ifiber = 1:nfiber
    for col = 1:4
        fibercell{ifiber, col} = fiberinput(ifiber,col);
    end
end
sections = {1, 'Fiber', fibercell};
materials = { 1, 'ConcreteLinearTensionSoftening', 280, 0, {-4,-0.003,-0.8,-0.01,0.1,0.56};
              2, 'GiuffreMenegottoPinto', 29000, 0, {66.8, 0.01, 18, 1, 0.925, 0.15, 0, 55, 0, 55, 0}};
DispInput(1:2000) = 0;
for i = 2001:2100
    DispInput(i) = DispInput(i-1)+0.0432;
end
row = size(DispInput,2);
for i = 1:row
    DispCell{i} = DispInput(i);
end
ForceInput(1:2000) = [-1:-1:-2000];
for i = 2001:2100
    ForceInput(i) = -2000;
end
row = size(ForceInput,2);
for i = 1:row
    ForceCell{i} = ForceInput(i);
end
forces = { 'VariableNodalDisplacement',{2, 1, DispCell};
           'VariableNodalForce',{2, 2, ForceCell} };
```

Fig. 7.25 Example 25




```
Unit      = ['kips, in'];
StrMode   = [2, 3];
Analysis  = 'Static';
Iteration = 'NewtonRaphson';
IterationEle= 'NewtonRaphson';
Convergence= 'ForceNorm';
ConvergenceEle= 'DispNorm';
StrMiter  = 10;
EleMiter  = 10;
Tolerance = 1.0e-8;
nodcoord   = {1,    0, 0;
              2,    0, 432;};
constraint = {1, 1, 1, 1;
              2, 1, 0, 0};
elements = { 1, 'FlexibilityBased2DBeamColumn', 1, 2, {1,5};};
ccount = 0;
for row = 1:16
    ccount = ccount + 4;
    fiberinput(ccount-3,1:4) = [1, 56.25, 28.125-3.75*(row-1),  22.5];
    fiberinput(ccount-2,1:4) = [1, 56.25, 28.125-3.75*(row-1),   7.5];
    fiberinput(ccount-1,1:4) = [1, 56.25, 28.125-3.75*(row-1),  -7.5];
    fiberinput(ccount  ,1:4) = [1, 56.25, 28.125-3.75*(row-1), -22.5];
end
scount = ccount;
for row = 1:2
    for col = 1:5
        scount = scount + 1;
        fiberinput(scount,1:4) = [2, 2.25, 25-50*(row-1), 25-12.5*(col-1)];
    end
end
nfiber = size(fiberinput,1);
for ifiber = 1:nfiber
    for col = 1:4
        fibercell{ifiber, col} = fiberinput(ifiber,col);
    end
end
sections = {1, 'Fiber', fibercell};
materials = { 1, 'ConcreteLinearTensionSoftening', 280, 0, {-4,-0.003,-0.8,-0.01,0.1,0.56};
              2, 'GiuffreMenegottoPintd', 29000, 0, {66.8, 0.01, 18, 1, 0.925, 0.15, 0, 55, 0, 55, 0}};
DispInput(1:2000) = 0;
for i = 2001:2100
    DispInput(i) = DispInput(i-1)+0.0432;
end
row = size(DispInput,2);
for i = 1:row
    DispCell{i} = DispInput(i);
end
ForceInput(1:2000) = [-1:-1:-2000];
for i = 2001:2100
    ForceInput(i) = -2000;
end
row = size(ForceInput,2);
for i = 1:row
    ForceCell{i} = ForceInput(i);
end
forces = { 'VariableNodalDisplacement',{2, 1, DispCell};
           'VariableNodalForce',{2, 2, ForceCell} };
```

Fig. 7.26 Example 26

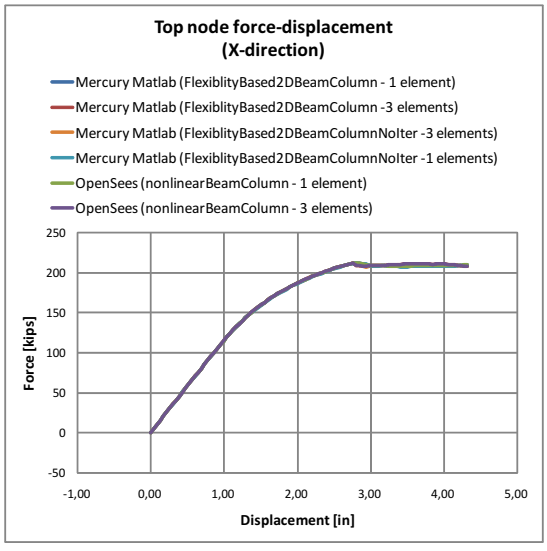


Fig. 7.27 Example 23 to 26

7.2.4 Example 27

Fig 7.28

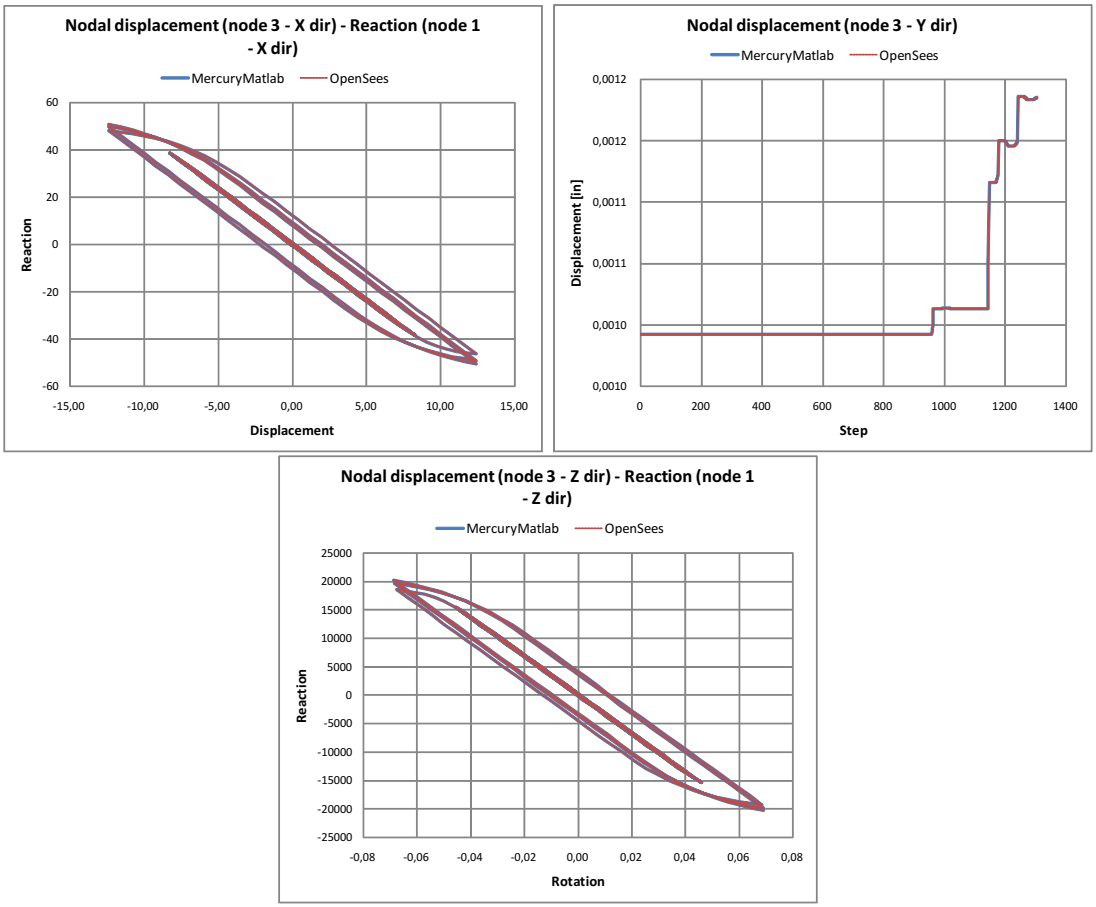


Fig. 7.28 Example 27, x, y, z direction

7.2.5 Example 28

Fig 7.29

7.2.6 Example 29

Fig 7.30

7.2.7 Example 30

Fig 7.31

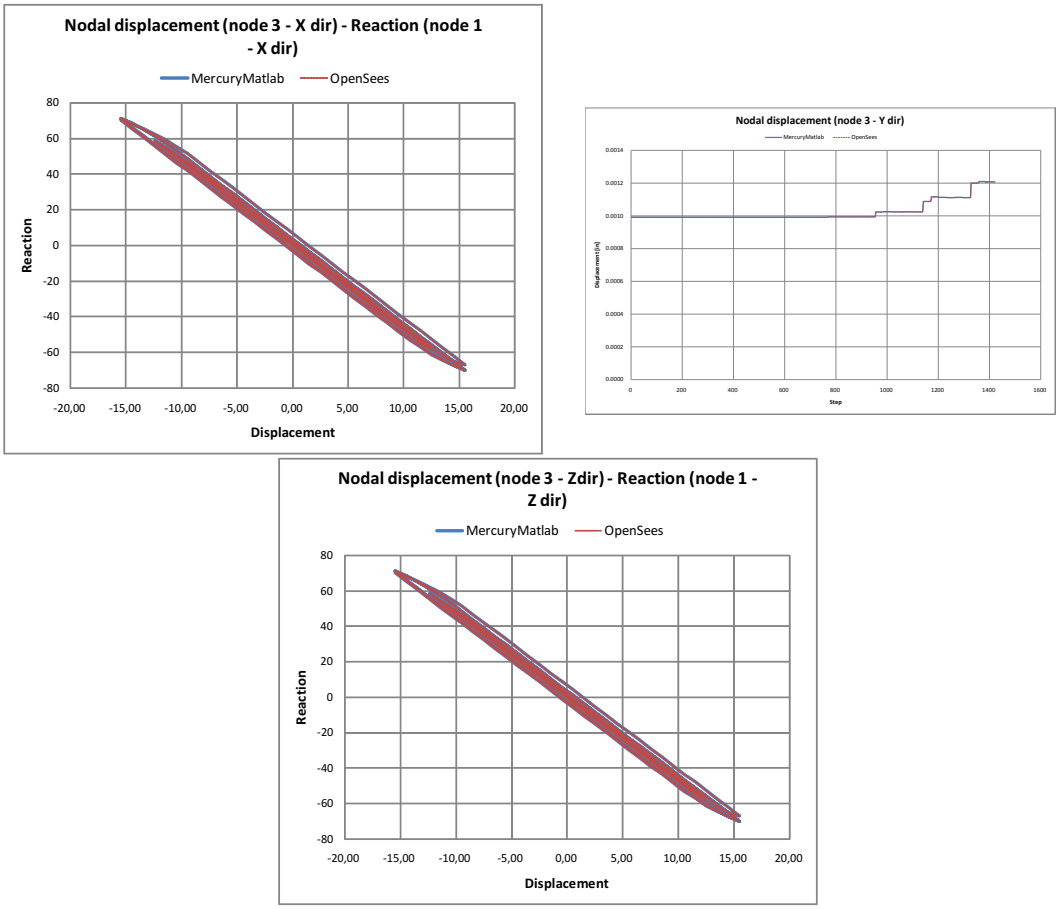


Fig. 7.29 Example 28, x, y, z direction

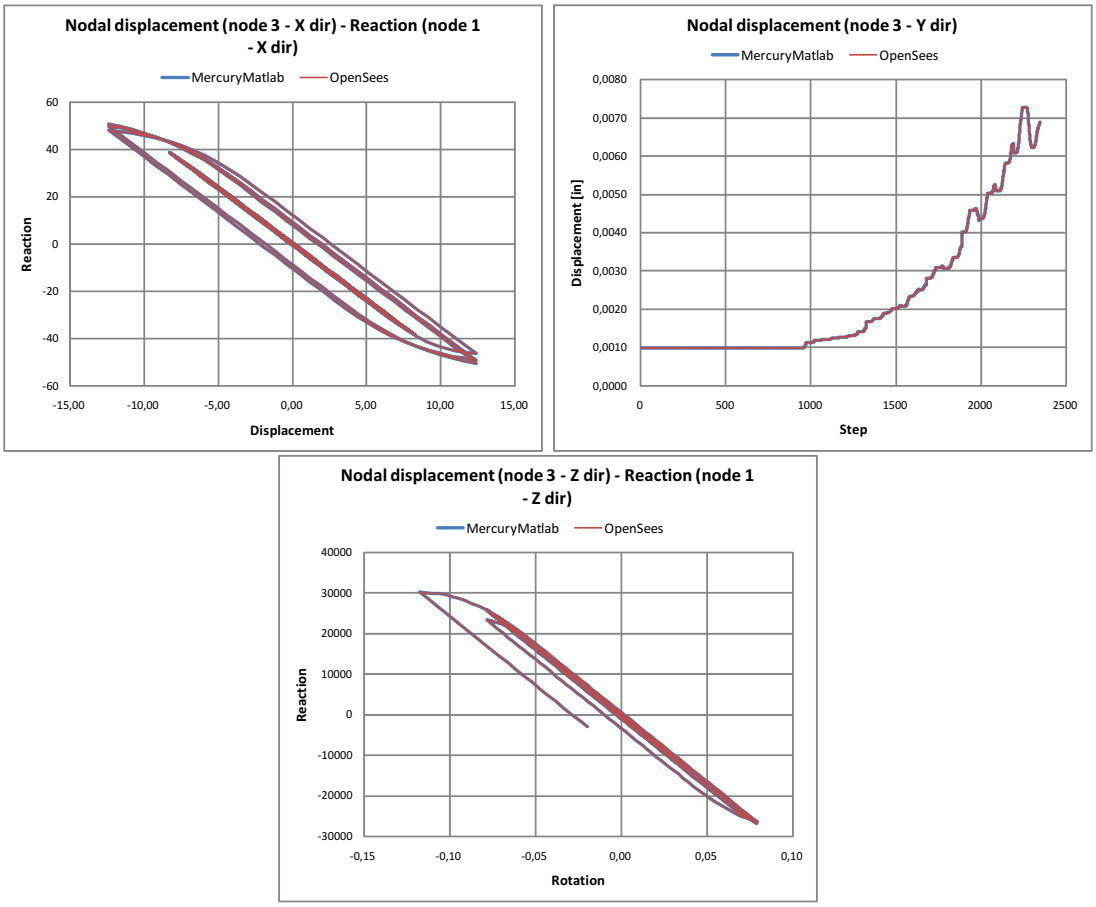


Fig. 7.30 Example 29, x direction

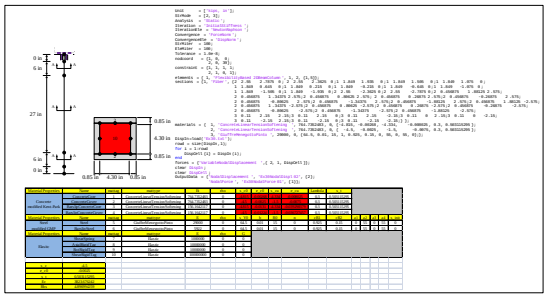
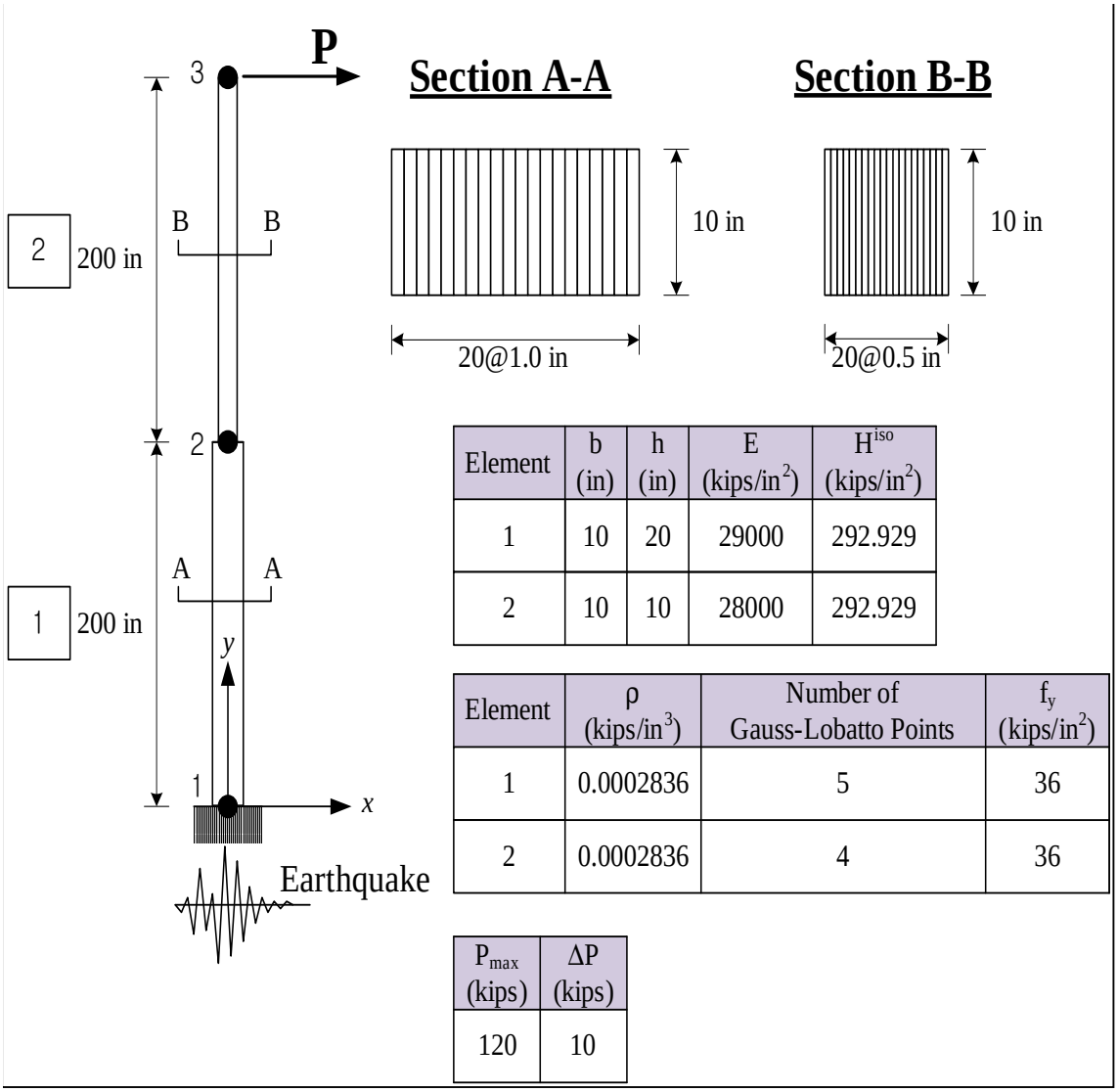


Fig. 7.31 Example 30

7.2.8 Ex110 to 113

Fig 7.32 Setup for example 110 to 113. Fig 7.33 Acceleration, Velocity and Displacement for Example 112 and 113. Fig 7.34 Acceleration, Velocity and Displacement for Example 110 and 111.



Element	b (in)	h (in)	E (kips/in^2)	H^{iso} (kips/in^2)
1	10	20	29000	292.929
2	10	10	28000	292.929

Element	ρ (kips/in^3)	Number of Gauss-Lobatto Points	f_y (kips/in^2)
1	0.0002836	5	36
2	0.0002836	4	36

P_{max} (kips)	ΔP (kips)
120	10

Fig. 7.32 Example 110 to 113 setup

7.2.9 Ex116 to 119

Fig 7.35 Setup for example 116 to 119. Fig 7.36 Acceleration, Velocity and Displacement for Example 116 and 117. Fig 7.37 Acceleration, Velocity and Displacement for Example 118 and 119.

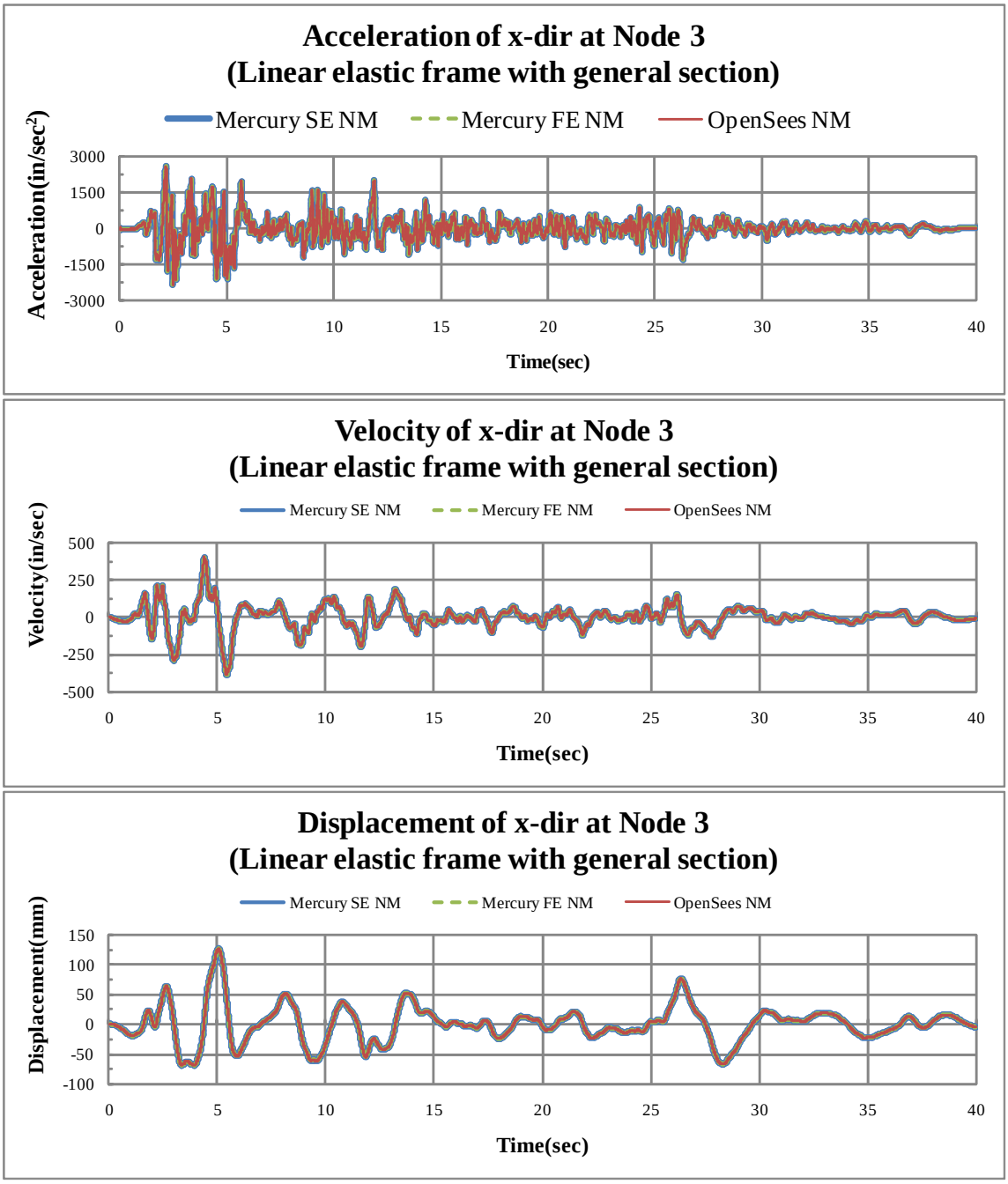


Fig. 7.33 Example 110 & 111 acceleration, velocity and displacement

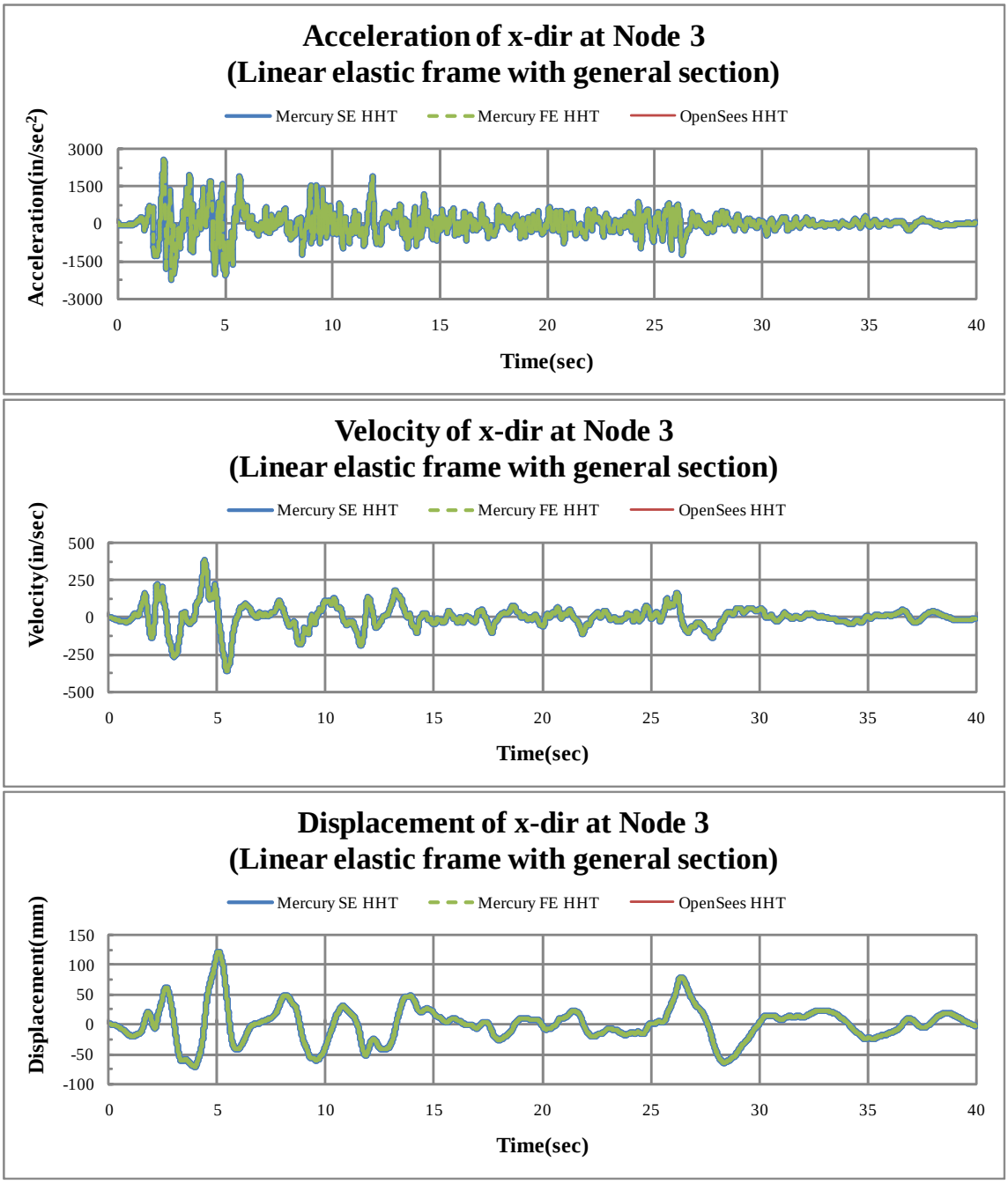
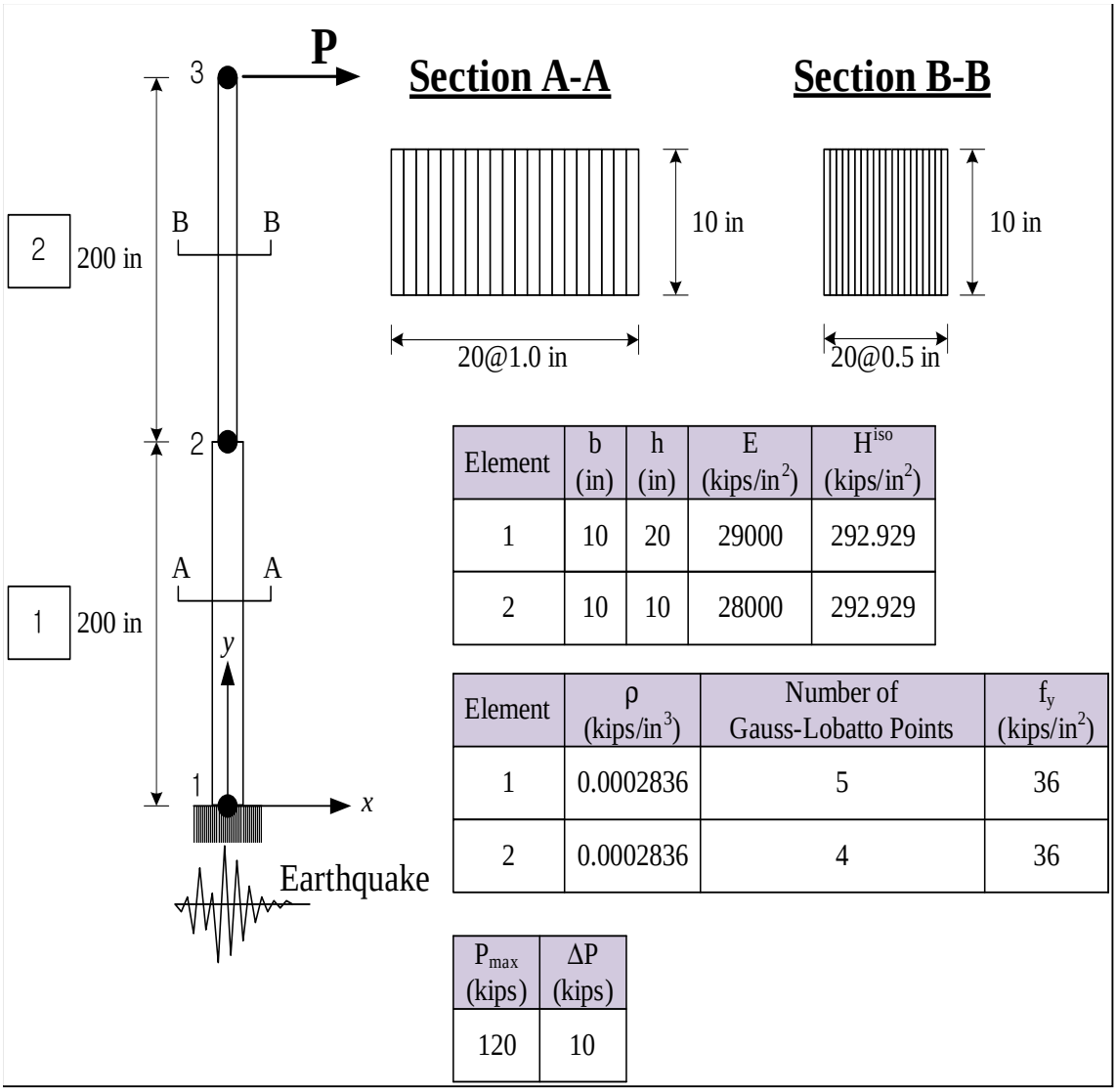


Fig. 7.34 Example 112 & 113 acceleration, velocity and displacement



Element	b (in)	h (in)	E (kips/in^2)	H^{iso} (kips/in^2)
1	10	20	29000	292.929
2	10	10	28000	292.929

Element	ρ (kips/in^3)	Number of Gauss-Lobatto Points	f_y (kips/in^2)
1	0.0002836	5	36
2	0.0002836	4	36

P_{max} (kips)	ΔP (kips)
120	10

Fig. 7.35 Example 116 to 119 setup

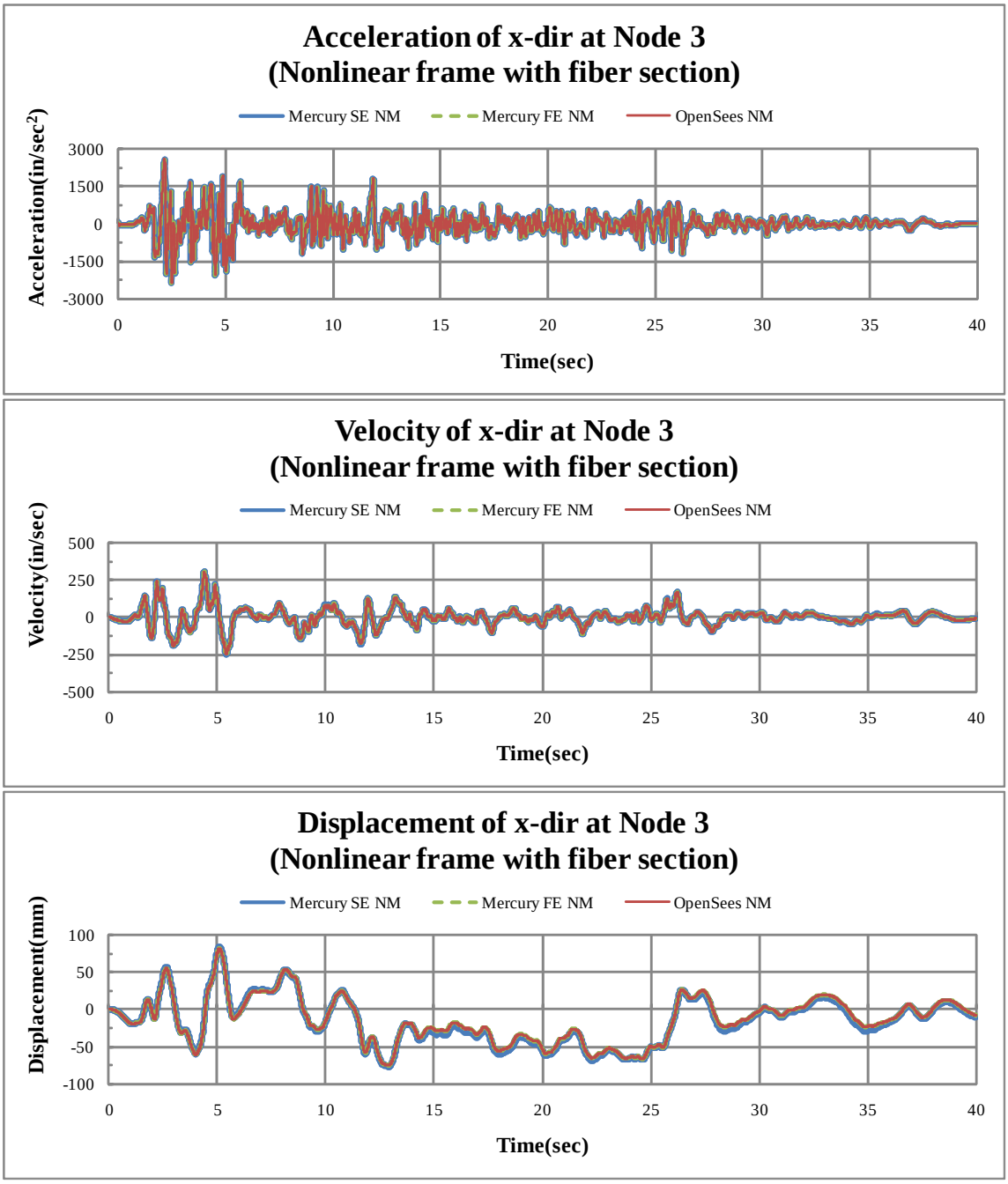


Fig. 7.36 Example 116 & 117 acceleration, velocity and displacement

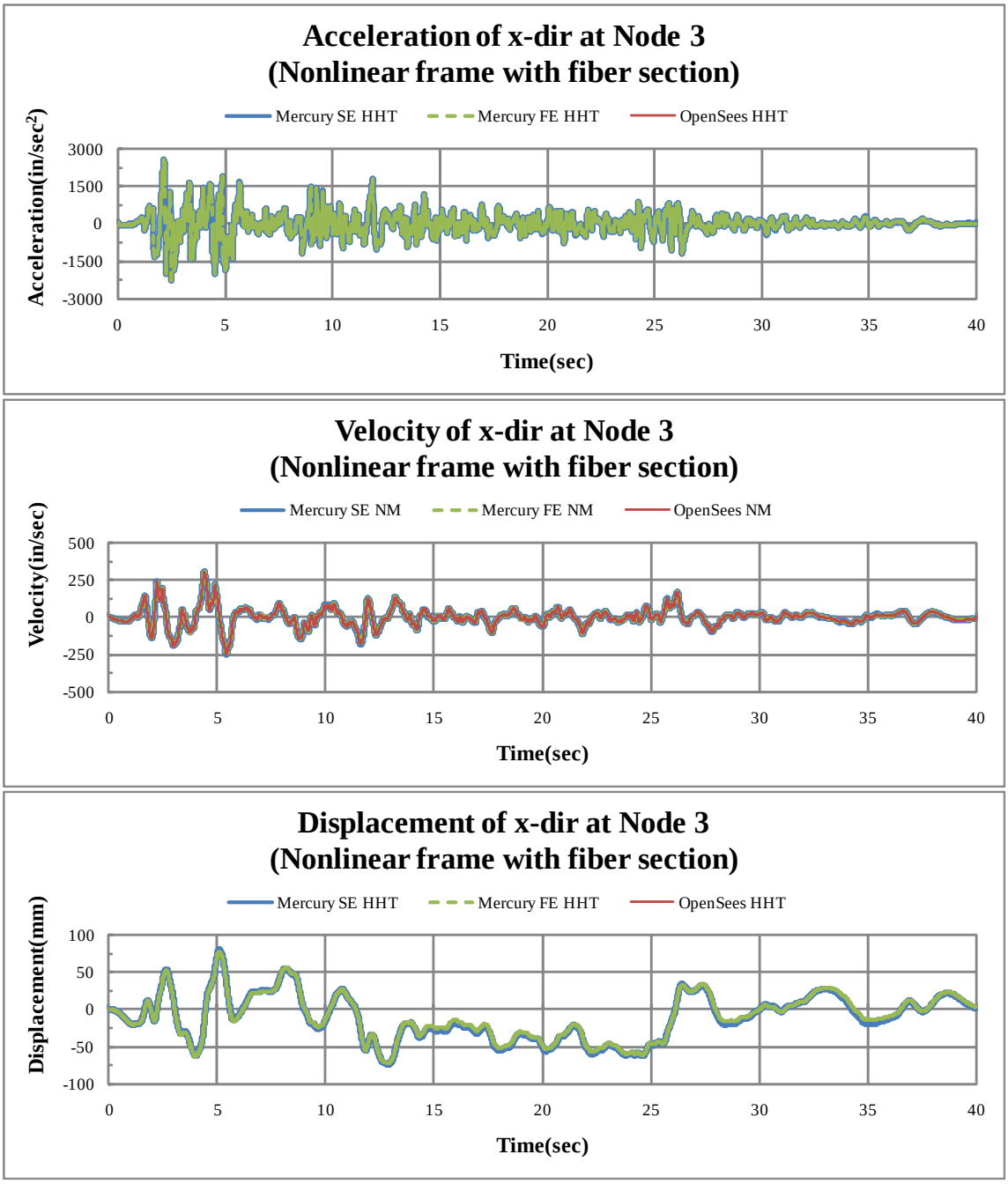
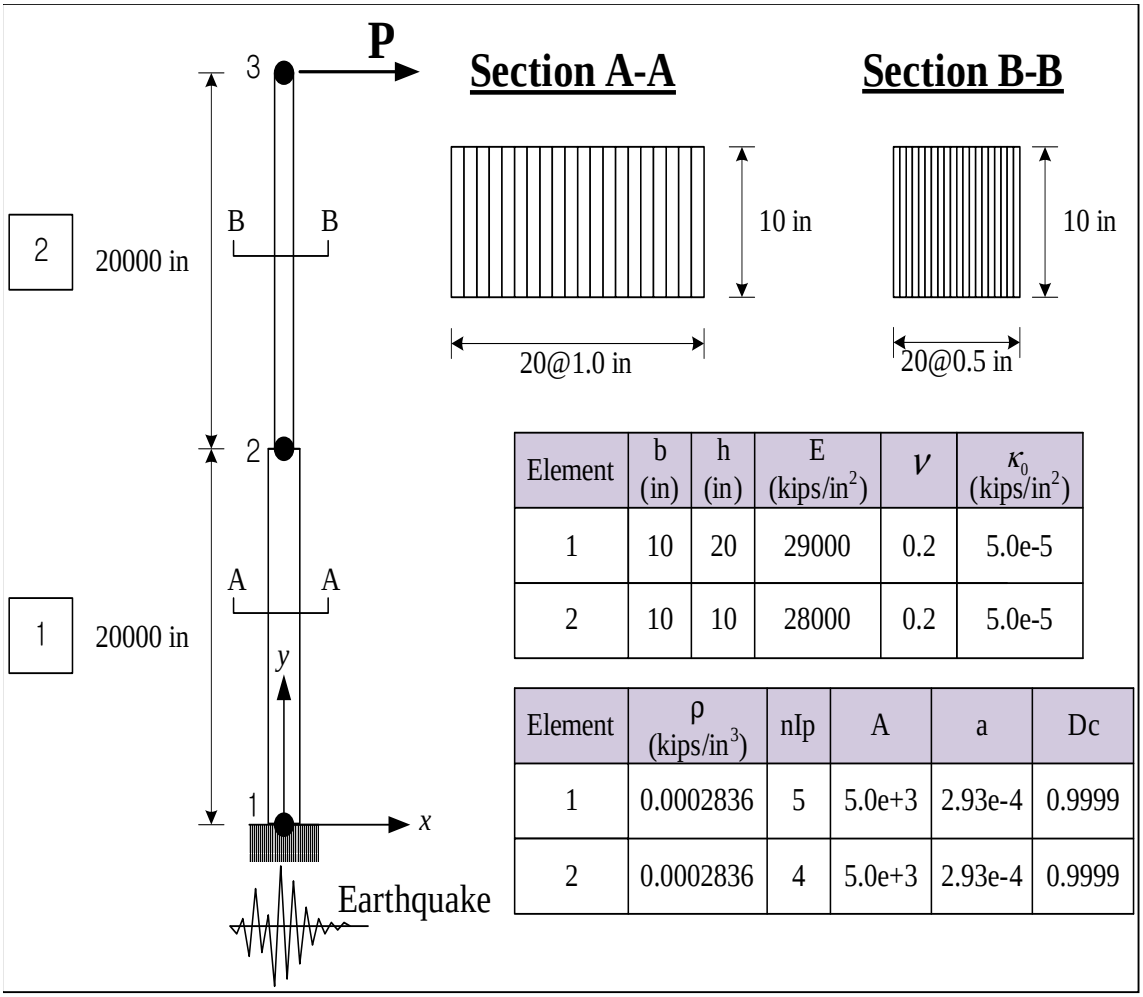


Fig. 7.37 Example 118 & 119 acceleration, velocity and displacement

7.2.10 Example 140

Fig 7.38 Setup for example 140. Fig 7.39 Acceleration, Velocity and Displacement.



Element	b (in)	h (in)	E (kips/in^2)	ν	κ_0 (kips/in^2)
1	10	20	29000	0.2	5.0e-5
2	10	10	28000	0.2	5.0e-5

Element	ρ (kips/in^3)	nIp	A	a	Dc
1	0.0002836	5	5.0e+3	2.93e-4	0.9999
2	0.0002836	4	5.0e+3	2.93e-4	0.9999

Fig. 7.38 Example 140 setup

7.2.11 Example 141

Fig 7.40 Setup for example 141. Fig 7.41 Acceleration, Velocity and Displacement.

7.3 Comparison of Modified Kent & Park with the Anisotropic Model

Two concrete models were implemented in Mercury, the well established Modified Kent & Park model, and the more recently developed anisotropic damage model of

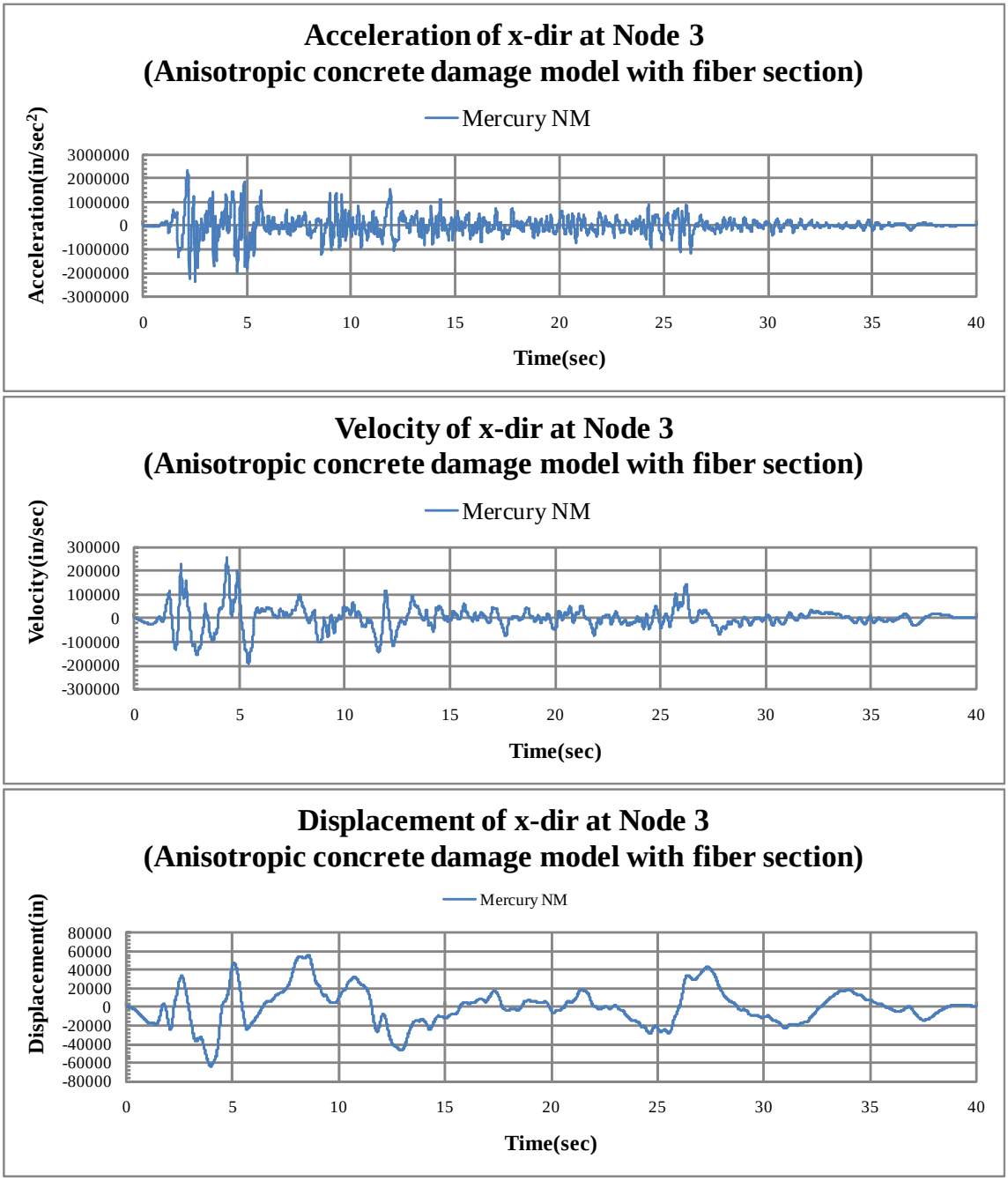
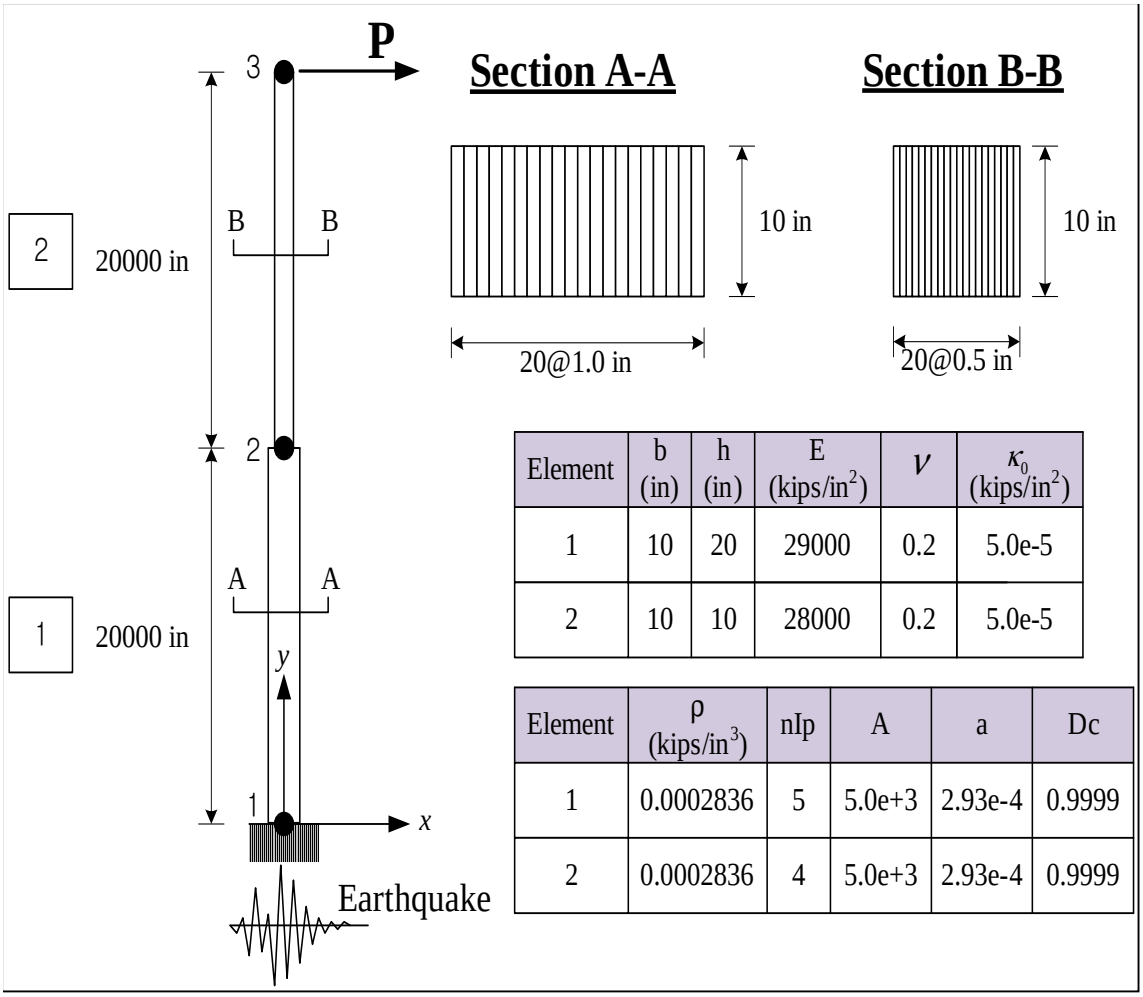


Fig. 7.39 Example 140 acceleration, velocity and displacement



Element	b (in)	h (in)	E (kips/in^2)	ν	κ_0 (kips/in^2)
1	10	20	29000	0.2	5.0e-5
2	10	10	28000	0.2	5.0e-5

Element	ρ (kips/in^3)	nIp	A	a	Dc
1	0.0002836	5	5.0e+3	2.93e-4	0.9999
2	0.0002836	4	5.0e+3	2.93e-4	0.9999

Fig. 7.40 Example 141 setup

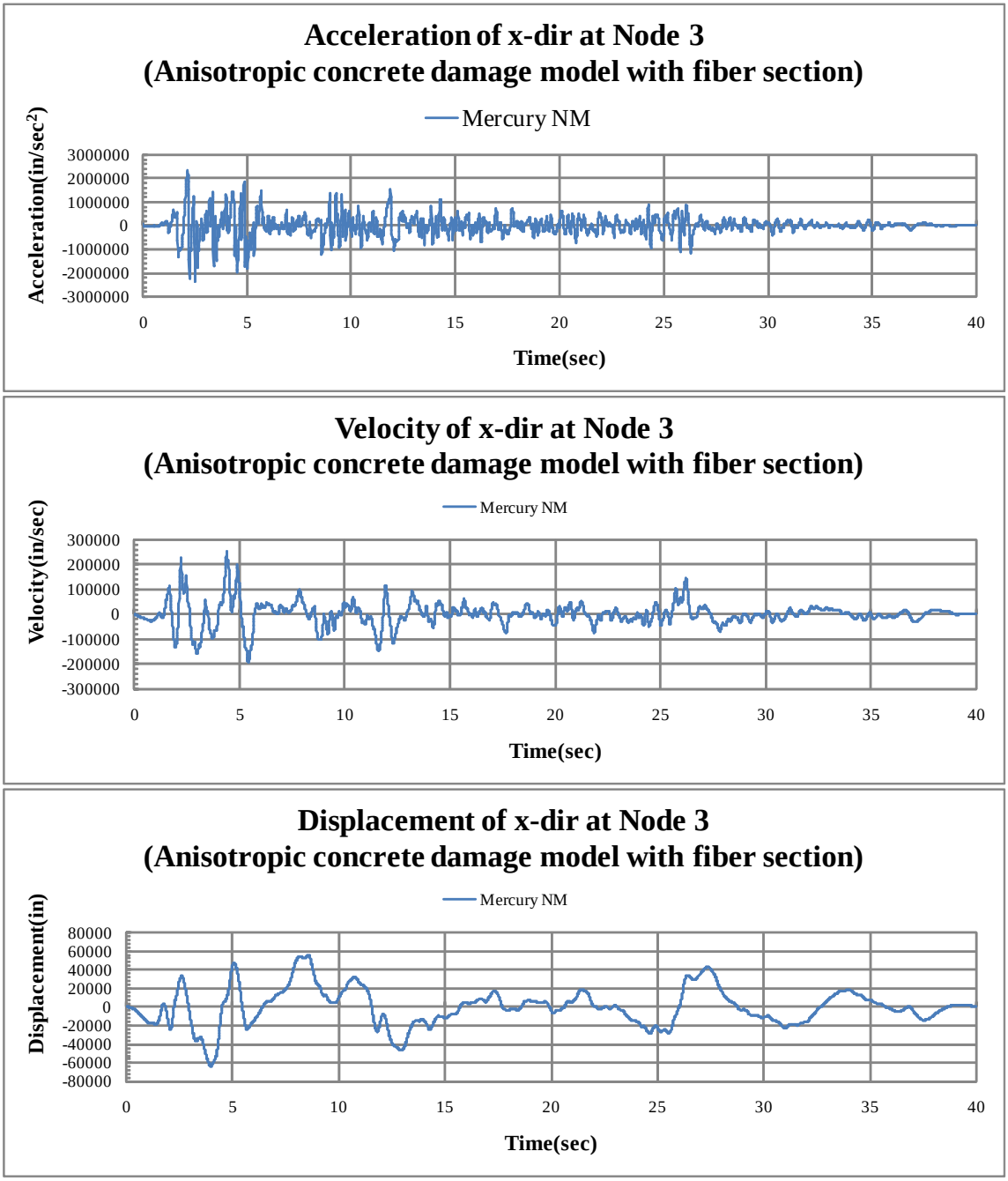


Fig. 7.41 Example 141 acceleration, velocity and displacement

7.3.1 Description of Test Problems

Two representative beam-columns were identified:

Beam-Column 1: is a single flexibility based beam-column element with confined and unconfined concrete and internally composed of 30 concrete fibers, and 8 steel ones, Fig. 7.42.

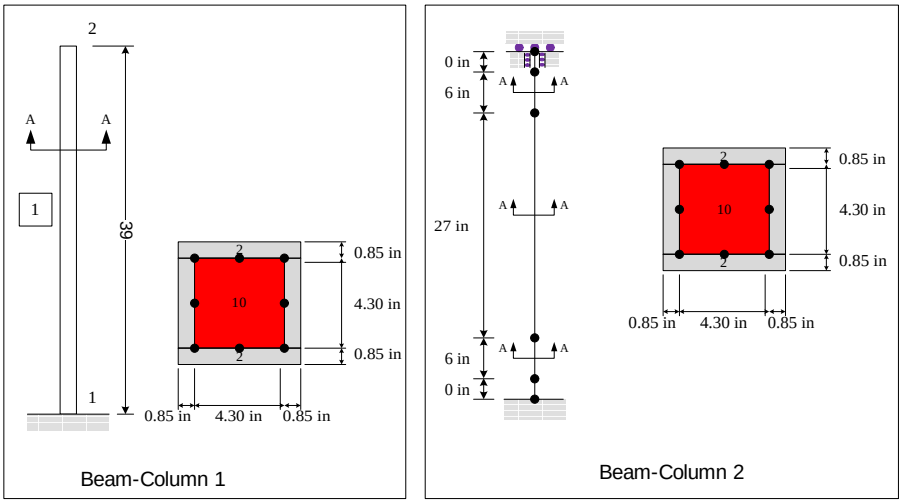


Fig. 7.42 Test Beam Columns

Beam-Column 2: has the same cross section (68 concrete fibers and 8 steel fibers) as the previous column, however it is composed of 3 flexibility based elements, and one zero length element and one zero length section element at each end, Fig. 7.42.

Material properties are shown in Table 7.1.

We consider three type of loads:

Loading 1: Simple pre-peak push-over Where the imposed displacement do not induce failure. A total of 1 inch displacement is imposed by increments of 0.01 in. Fig. 7.43.

Loading 2: Pushover with vertical compression Second loading is a pushover (in displacement control as well) of 2.57 inches with a vertical loading of 16.7 kips and steps of 0.0097 inches.

Loading 3: Cyclic load Third loading is cyclic load,three cycle are done for each the peak : 0.1 inch., 0.21 inch., 0.43 inch., 0.86 inch., 1.29 inch., 1.72 inch., and 2.57 inch.. Vertical compression is also 16.7 kips.

7.3.2 Results

Modified Kent & Park and anisotropic damage models yield practically same results for Column-1 Load1 and 2, and column-2 Load-2, Fig. **??** and 7.45.

Fig. 7.46 compares Modified Kent & Park (from an OpenSees analysis) anisotropic damage model in Mercury for Load-3 and Column-2. We note that both analysis yield practically same results.

The CPU times reported in Table 7.2 indicate that:

Table 7.1 Material Properties

Anistropic Damage													
	E	ν	κ_0	a	A	D_c							
Concrete Core	3960	0.18	4.95E-05	3.36E-04	4,200	0.999							
Concrete Cover	3823	0.18	4.65E-05	3.47E-04	4,150	0.999							
Bar slip Concrete Core	3960	0.18	4.85E-05	3.47E-04	4,150	0.999							
Bar slip Concrete Cover	3823	0.18	5.00E-05	3.17E-04	4,500	0.999							
Modified Kent-Park													
	E_t	ρ	σ_{c0}	e_{c0}	σ_{cu}	ε_{cu}	λ	σ_t					
Concrete Core	764.7	0	-4.815	-0.002675	-4.3335	-0.008025	0.3	0.50311					
Concrete Cover	764.7	0	-4.5	-0.0025	-1.5	-0.0075	0.3	0.50311					
Bar slip Concrete Core	156.16	0	-4.815	-0.01309	-4.3335	-0.0393	0.3	0.50311					
Bar slip Concrete Cover	156.16	0	-4.5	-0.0122	-1.5	-0.03672	0.3	0.50311					
Modified Giuffre Menegotto and Pinto													
	E	ρ	σ_{Yo}	b	R_0	n	c_{R1}	c_{R2}	a_1	a_2	a_3	a_4	σ_{init}
Steel	29000	0	64.5	0.01	15	0	0.925	0.15	0	55	0	55	0
Bar slip Steel	5922	0	64.5	0.01	15	0	0.925	0.15	0	55	0	55	0
Zero Length Section													
	E	ρ	G										
Shear Stiffness	10^6	0	0										
Axial Stiffness	10^7	0	0										
Rotational Stiffness	110^7	0	0										
ShearRigidTag	10^8	0	0										

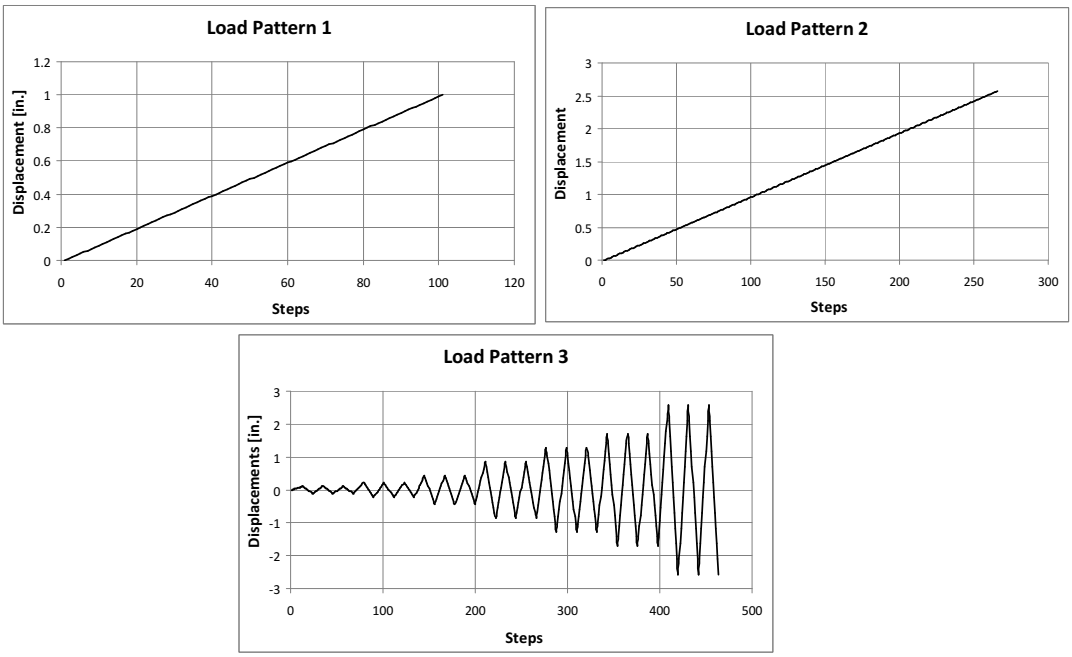


Fig. 7.43 Load Diagrams

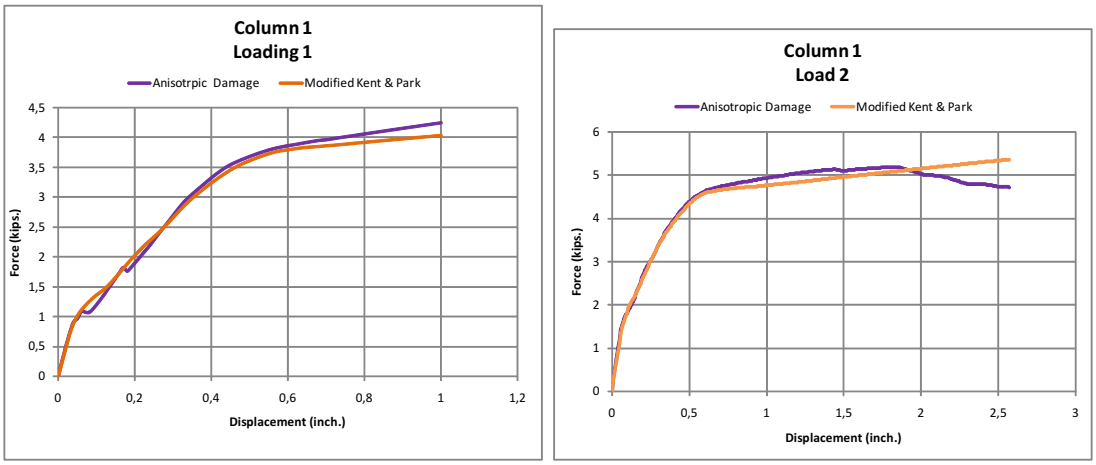


Fig. 7.44 Force Displacement for Column 1, Load Pattern 1 and 2

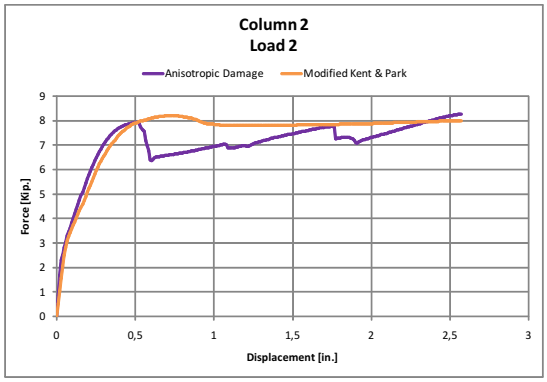


Fig. 7.45 Force Displacement for Column 2, Load Pattern 2

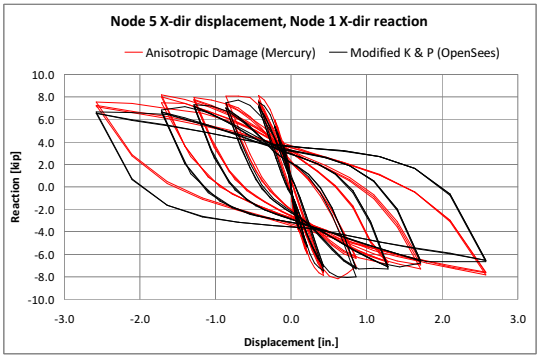


Fig. 7.46 Modified Kent & Park (OpenSees) vs Anisotropic Damage (Mercury), Tol. 10^{-8}

1. For Column-1 Modified Kent & Park appear to be faster than anisotropic damage model for pushover analysis.

2. For Column-1 Modified Kent & Park appear to be ??? than anisotropic damage model for cyclic load.

3. Anisotropic damage model is much faster than the Modified Kent & Park for Column-2 Load-2 (which is a pushover up to failure).

	Column 1		Column 2	
	Load 1	Load 2	Load 2	Load 3
Anisotropic Damage	133.1	636.5	1,601	10,8574
Modified Kent & Park	24.8	60.5	11,562	*

* No convergence after 20,000 iterations at incr. 399

Table 7.2 Comparison of CPU time for various loads for a convergence criteria of 10^{-8}

In Table 7.3 we determine that for Column 2, Load-2, we determine that increasing the convergence tolerance up to 10^{-5}, the anisotropic damage model is faster than the modified Kent & Park; however the reverse is true for convergence tolerance up to 10^{-3}.

Tolerance	10^{-8}	10^{-5}	10^{-4}	10^{-3}
Total CPU Time				
Modified Kent & Park	11,500	15,000	118	101
Anisotropic Damage	1,600	714	536	427
Inverse of Normalized CPU Time wrt Damage (10^{-8})				
Modified Kent & Park	0.1	0.1	13.6	15.8
Anisotropic Damage	1.0	2.2	3.0	3.7

Table 7.3 Comparison of CPU time between Modified Kent & Park and Anisotropic Damage for different convergence criteria for Column 2 and Load 2

However, for both models we can increase the convergence criteria up to 10^{-3} without significant loss of accuracy, Fig. 7.47.

7.3.3 Effect of Fixed Number of Iterations

The RTHS integration schem at CU-NEES fixes the number of iterations per increment to 10. As such, we examined the errors induced by fixing the number of iterations in the Anisotropic Damage model on Column-2 Load-3, Fig. 7.48. We note that by dropping the maximum number of iterations from 100 to 10, there is practically little differences in the results.

Fig. 7.49 shows that for Column-2 Load-3 with a tolerance of 10^{-3} there are few increments with more than 10 iterations.

Comparing CPU times for Column-2 and Load-3 with the anisotropic damage model, we increased the speed by a factor of 300 from a tolerance of 10^{-8} and maximum iterations equal 100, to a tolerance of 10^{-3} and maximum iterations equal 10, Table 7.4.

The effect of tolerance and number of iterations on the accuracy of results in the anisotropic model for column 2 load 3 are shown in Fig. **??**. We observe that we can reasonably use a

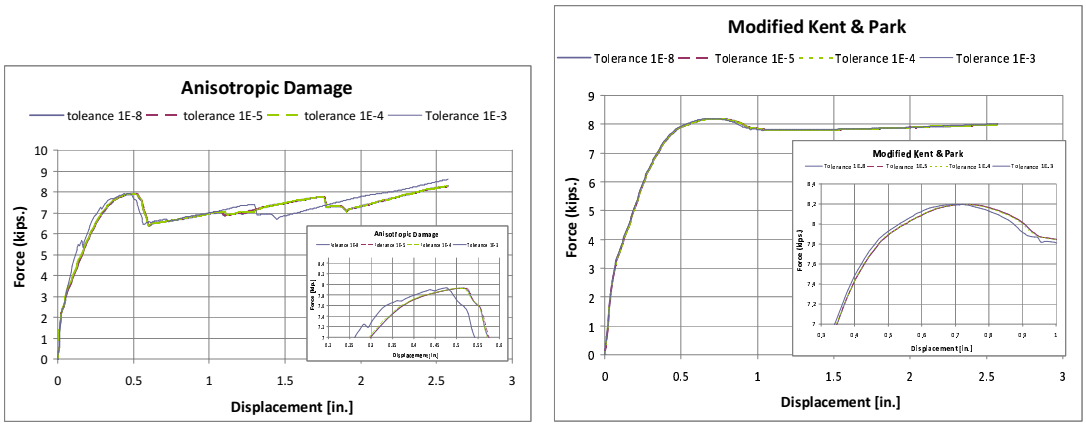


Fig. 7.47 Force Displacement for Column 2, Load 2, Anisotropic Damage and Modified Kent & Park Model. Tolerance form 1E-8 to 1E-3

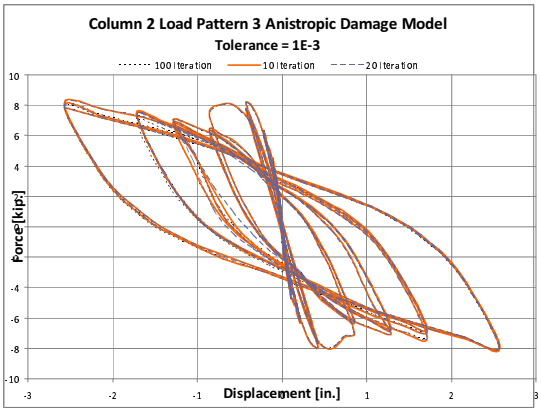


Fig. 7.48 Effect of fixed iteration number in the anisotropic damage model

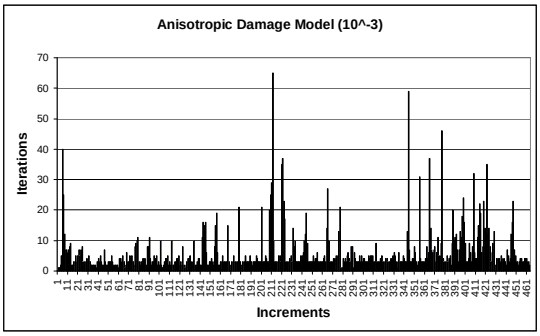


Fig. 7.49 Number of iterations per increment in anisotropic damage model with 10^{-3}

Tolerance	$\varepsilon = 10^{-8}$	$\varepsilon = 10^{-3}$		
Max. Iteration	100	100	20	10
	1.08E+05	7.20E+03	4.07E+02	3.50E+02
Speed-up	1.0	15	265	309

Table 7.4 CPU Comparison for anisotropic damage model with various convergence criteria

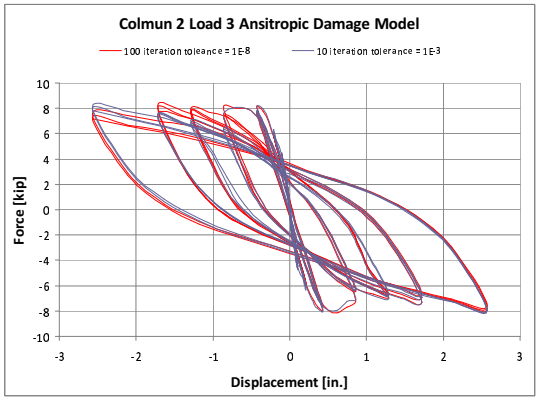


Fig. 7.50 Effect of fixed iteration number in the anisotropic damage model

tolerance of 10^{-3} and a fixed maximum number of iterations equal to 10 in the anisotropic damage model for Column-2 and Load-3 which is representative of seismic load.

Chapter 8

HYBRID TEST

8.1 Introduction

8.1.1 Methods To Evaluate Seismic Performance

The numerical and experimental methods for evaluating the seismic performance of a variety of structures have been developed during the past decades. Nevertheless, it is too hard to predict confidently the structural response under severe seismic loadings when a structure has inelastic behavior and includes elements with the rate dependent behavior. Actually, the accuracy of the predictions of structural response in the existed nonlinear analytical methods are limited by assumptions in the mathematical models. Therefore, experimental methods have been regarded as methods that are worth trusting most to evaluate the nonlinear behavior of structures subjected to severe earthquake loadings. Such experimental methods can provide the effective, professional information for understanding and evaluating nonlinear behavior of structures as well as for the new structural designs to improve seismic performance.

To simulate the dynamic response and evaluate the seismic performance of structures and their components, all numerical methods can fall generally within either explicit or implicit methods, and experimental methods can be subdivided into three main methods: quasi-static tests, shaking table tests, and hybrid tests.

8.1.2 Numerical Methods

The idealized structural model is described by the equations of motion in spatially discrete form by a set of second-order ordinary differential equations, which is approximate representation of the real structure. The equations of motion 8.1 can be solved using numerical step-by-step integration methods.

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{C}\dot{\mathbf{u}} + \mathbf{r}_S = \mathbf{f} \tag{8.1}$$

where, $\mathbf{M}$ and $\mathbf{C}$ are the mass and viscous damping matrix, and $\mathbf{f}$ is the external excitation vector;$\ddot{\mathbf{u}}$, $\dot{\mathbf{u}}$, and $\mathbf{r}_S$, are the acceleration vector, the velocity vector, and the static restoring force vector of the structure.

Both explicit and implicit methods have been applied for simulating the dynamic response of a structural model. If the displacement at each time is determined entirely in terms of the solutions at the previous time step without the need to invert the stiffness of structure, the numerical integration methods are said to be explicit methods. The advantage of explicit methods is that the required displacement increments can be determined directly from the results of the previous time-step, while the drawback of these is that they are conditionally stable with respect to the time interval for integration, so may require a short-time step for integration and require that systematic error be controlled within given parameters in order to remain stable. Explicit methods may not be used for multi-degree-of-freedom structures due to systematic undershoot errors exciting higher mode responses. Otherwise, implicit methods require knowledge of the structural response in the current time step. For structural response with nonlinear behavior, these require a Newton-type iterative procedure in order to satisfy equilibrium at the end of the current time step. However, most implicit methods are unconditionally stable and can better handle structural nonlinearity and experimental error. Hence, implicit methods allow for the possibility of analyzing a multi-degree-of- freedom structure with high frequency modes using a reasonably large time step. However, an iterative solution procedure for a nonlinear system may require a lot of computational time, which may have the possibility of inducing undesiable loading and unloading hystereses in a structure.

In summary, while explicit methods make it easier to implement in hybrid simulation, implicit methods can improve stability characteristics and use larger integration time steps. Therefore, the use of implicit methods can get better accuracy than the use of explicit methods. However, implicit methods may be undesirable for test of structure because of the risk of overshooting, which may have a significant effect on the response of the structure (Williams and Blakeborough 2001).

8.1.3 Experimental Methods

The most widely used experimental methods to obtain the structural response under seismic loadings are quasi-static test, shaking table test, and hybrid test. Shaking table tests, hybrid tests, and real-time hybrid tests similar to hybrid tests are introduced.

8.2 Literature Survey

8.2.1 Deveolpment Of Hybrid Test

8.2.2 Time Integration Methods For Hybrid Test

Early hybrid tests have required the test of the entire structural model and a large-scale testing facilities. For this reason, substructure technique has been developed and applied to hybrid tests by Dermitzakis and Mahin (1985), Nakashima and Takai (1985), and Shing and Vanna (1991). This is explaned in section 8.2.2.1.

To represent the dynamic behavior of structure for hybrid test, we can use a set of spatially discretized equations of motion depicted by matrix form. In step-by-step time integration with discrete time interval Δt, the equations of motion at current time step t_n can be written:

$$\mathbf{M}\ddot{\mathbf{u}}_n + \mathbf{C}\dot{\mathbf{u}}_n + \mathbf{r}_{S,n} = \mathbf{f}_n \tag{8.2}$$

where, $\mathbf{M}$ and $\mathbf{C}$ are the mass and viscous damping matrix, and $\mathbf{f}_n$ is the external excitation vector at the current time step t_n; $\ddot{\mathbf{u}}_n$, $\dot{\mathbf{u}}_n$, and $\mathbf{r}_{S,n}$, are the acceleration vector, the velocity vector, and the static restoring force vector of the structure at the current time step t_n. Eq. 8.2 at each time step can be solved by approximate solutions within each time interval. Time integration methods generally can be classified into two catagories: either explicit or implict methods. They are introduced in section 8.2.2.2 and section 8.2.2.3.

8.2.2.1 Substructure technique

In substructure technique, a structure is partitioned into analytical and physical substructures. Physical substructure in which structural nonlinearity or severe damage due to seismic loading is expected will be actually tested in laboratory, and anlytical substructure is the remainder of the structure to be modelled in a computer. The equilibrium and displacement compatiblity conditions between the analytical and physical substructures can be enforced with a standard substructure technique. Hence, we can rewrite the equaions of motion for the entire structure with both analytical and physical substructures:

$$(\mathbf{M}^A + \mathbf{M}^P)\ddot{\mathbf{u}} + (\mathbf{C}^A + \mathbf{C}^P)\ddot{\mathbf{u}} + \mathbf{r}_S^A + \mathbf{r}_S^P = \mathbf{f} \tag{8.3}$$

where, the terms with subscripts A represent the properties and response variables of the analytical substructure and the terms with subscripts P are those of the physical substructure. Since the quantities of the inertia and damping forces of physical substructure are measured in test, Eq. 8.3 can be expressed approximately:

$$\mathbf{M}^A\ddot{\mathbf{u}} + \mathbf{C}^A\ddot{\mathbf{u}} + \mathbf{r}_S^A + \mathbf{r}^P = \mathbf{f} \tag{8.4}$$

with

$$\begin{aligned} \mathbf{r}^P &= \mathbf{r}_I^P + \mathbf{r}_D^P + \mathbf{r}_S^P \\ &= \mathbf{M}^P\ddot{\mathbf{u}} + \mathbf{C}^P\ddot{\mathbf{u}} + \mathbf{r}_S^P \end{aligned} \tag{8.5}$$

Depending on the properties of the analytical and physical substructures and how the two are coupled in the analysis, either explict or implicit methods can be used to solve Eq. 8.4 numberically. Fig. 8.1 explains the total approach (Shing 2008). In this approach, it is assumed that the dynamics of the entire system can be accurately described by the degrees of freedom

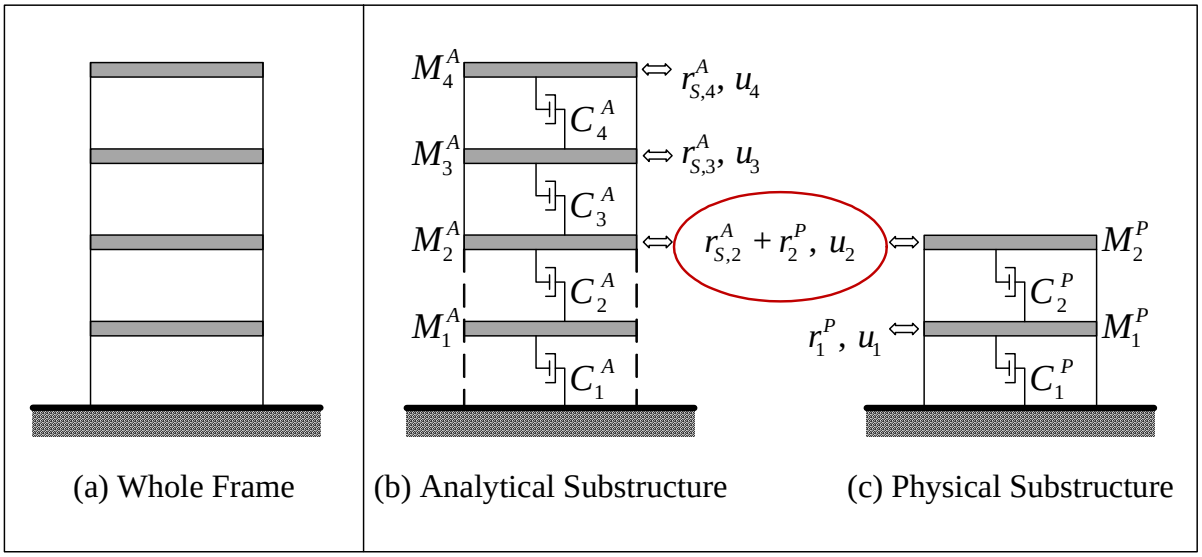


Fig. 8.1 Partitioning of a structure into analytical and physical substructures with a total approach (Shing, 2008)

indentified in Eq. 8.4. Eq. 8.4 based on the total approach can be expressed as follows:

$$\begin{bmatrix} M_1^A & 0 & 0 & 0 \\ 0 & M_2^A & 0 & 0 \\ 0 & 0 & M_3^A & 0 \\ 0 & 0 & 0 & M_4^A \end{bmatrix} \begin{Bmatrix} \ddot{u}_1 \\ \ddot{u}_2 \\ \ddot{u}_3 \\ \ddot{u}_4 \end{Bmatrix}$$
$$+ \begin{bmatrix} C_1^A + C_2^A & -C_2^A & 0 & 0 \\ -C_2^A & C_2^A + C_3^A & -C_3^A & 0 \\ 0 & -C_3^A & C_3^A + C_4^A & -C_4^A \\ 0 & 0 & -C_4^A & C_4^A \end{bmatrix} \begin{Bmatrix} \dot{u}_1 \\ \dot{u}_2 \\ \dot{u}_3 \\ \dot{u}_4 \end{Bmatrix}$$
$$+ \begin{Bmatrix} 0 \\ r_{S,2}^A \\ r_{S,3}^A \\ r_{S,4}^A \end{Bmatrix} + \begin{Bmatrix} r_1^P \\ r_2^P \\ 0 \\ 0 \end{Bmatrix} = \begin{Bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{Bmatrix}$$

In this approach, the restoring force vector can be rewritten:

$$\begin{Bmatrix} \mathbf{r}^P \\ \mathbf{r}_B \\ \mathbf{r}_S^A \end{Bmatrix} = \begin{Bmatrix} \mathbf{r}^P \\ \mathbf{r}_B^A + \mathbf{r}_B^P \\ \mathbf{r}_S^A \end{Bmatrix}$$

where, subscript B denotes the degrees of freedom at the boundary of the analytical and physical substructures. Hence, the restoring force vector ($\mathbf{r}_B^P$) developed by the physical substructure is measured in the test and assembled with the restoring force vector ($\mathbf{r}_B^A$) from the analytical substructure to obtain the system restoring force vector.

8.2.2.2 Explicit methods

1. Central difference method

   The central diffenence method is one of the most widely used explicit methods because of its simplicity and computational efficiency. In this method, the velocity and acceleration vectors at the current time t_n are approximated by Eq. 8.6 and Eq. 8.7, respectively:

$$\dot{\mathbf{u}}_n = \frac{\mathbf{u}_{n+1} - \mathbf{u}_{n-1}}{2\Delta t} \tag{8.6}$$

$$\ddot{\mathbf{u}}_n = \frac{\mathbf{u}_{n+1} - 2\mathbf{u}_n + \mathbf{u}_{n-1}}{\Delta t^2} \tag{8.7}$$

   Substituting Eq. 8.6 and Eq. 8.7 into Eq. 8.4, we have Eq. 8.8:

$$\begin{aligned} \mathbf{M}^A \left[\frac{\mathbf{u}_{n+1} - 2\mathbf{u}_n + \mathbf{u}_{n-1}}{\Delta t^2}\right] + \mathbf{C}^A \left[\frac{\mathbf{u}_{n+1} - \mathbf{u}_{n-1}}{2\Delta t}\right] + \mathbf{r}^A_{S,n} + \mathbf{r}^P_n &= \mathbf{f}_n \\ \mathbf{K}_{eff}\mathbf{u}_{n+1} &= \mathbf{F}_{n,eff} \end{aligned} \tag{8.8}$$

   where,

$$\begin{aligned} \mathbf{K}_{eff} &= \left[\mathbf{M}^A + \frac{\Delta t}{2}\mathbf{C}^A\right] \\ \mathbf{F}_{n,eff} &= \left[\Delta t^2\left(\mathbf{f}_n - \mathbf{r}^A_{S,n} - \mathbf{r}^P_n\right) + 2\mathbf{M}^A\mathbf{u}_n - \left(\mathbf{M}^A - \frac{\Delta t}{2}\mathbf{C}^A\right)\mathbf{u}_{n-1}\right] \end{aligned}$$

   where, $\mathbf{K}_{eff}$ is the effective stiffness matrix and $\mathbf{F}_{n,eff}$ is the effective external force vector; $\mathbf{r}^P_n$ is measured from the physical substructure in each time step. Hence, we can solve for $\mathbf{u}_{n+1}$:

$$\mathbf{u}_{n+1} = \left[\mathbf{K}_{eff}\right]^{-1}\mathbf{F}_{n,eff}$$

   The effective stiffness $\mathbf{K}_{eff}$ only have mass and damping matrix with discrete time interval Δt. For this reason, the structural stiffness matrix doesn't need to be inverted.

   This method is a two-step method because the displacement response at t_{n+1} is expressed in terms of the solution at the previous time steps t_{n-1} and t_n. To determine the displacement vector $\mathbf{u}_1$ in the first time step t_1, we need a fictitious displacement quantitiy $\mathbf{u}_{-1}$ that can be obtained from Eq. 8.6 and Eq. 8.7. Hence, we have:

$$\mathbf{u}_{-1} = \mathbf{u}_0 - \Delta t\dot{\mathbf{u}}_0 + \frac{\Delta t^2}{2}\ddot{\mathbf{u}}_0$$

   where, the initial acceleration vector $\ddot{\mathbf{u}}_0$ can be solved with the initial conditions $\mathbf{u}_0$ and $\dot{\mathbf{u}}_0$.

$$\ddot{\mathbf{u}}_0 = \left[\mathbf{M}^A\right]^{-1}\left(\mathbf{f}_0 - \mathbf{C}^A\dot{\mathbf{u}}_0 - \mathbf{r}^A_{S,0} - \mathbf{r}^P_0\right)$$

   The central difference method is conditionally stable. Thus, a stable solution can be obtatined only by selecting a time step $\Delta t \leq \Delta t_{cr}$ (Tedesco, McDougal and Ross 1999)

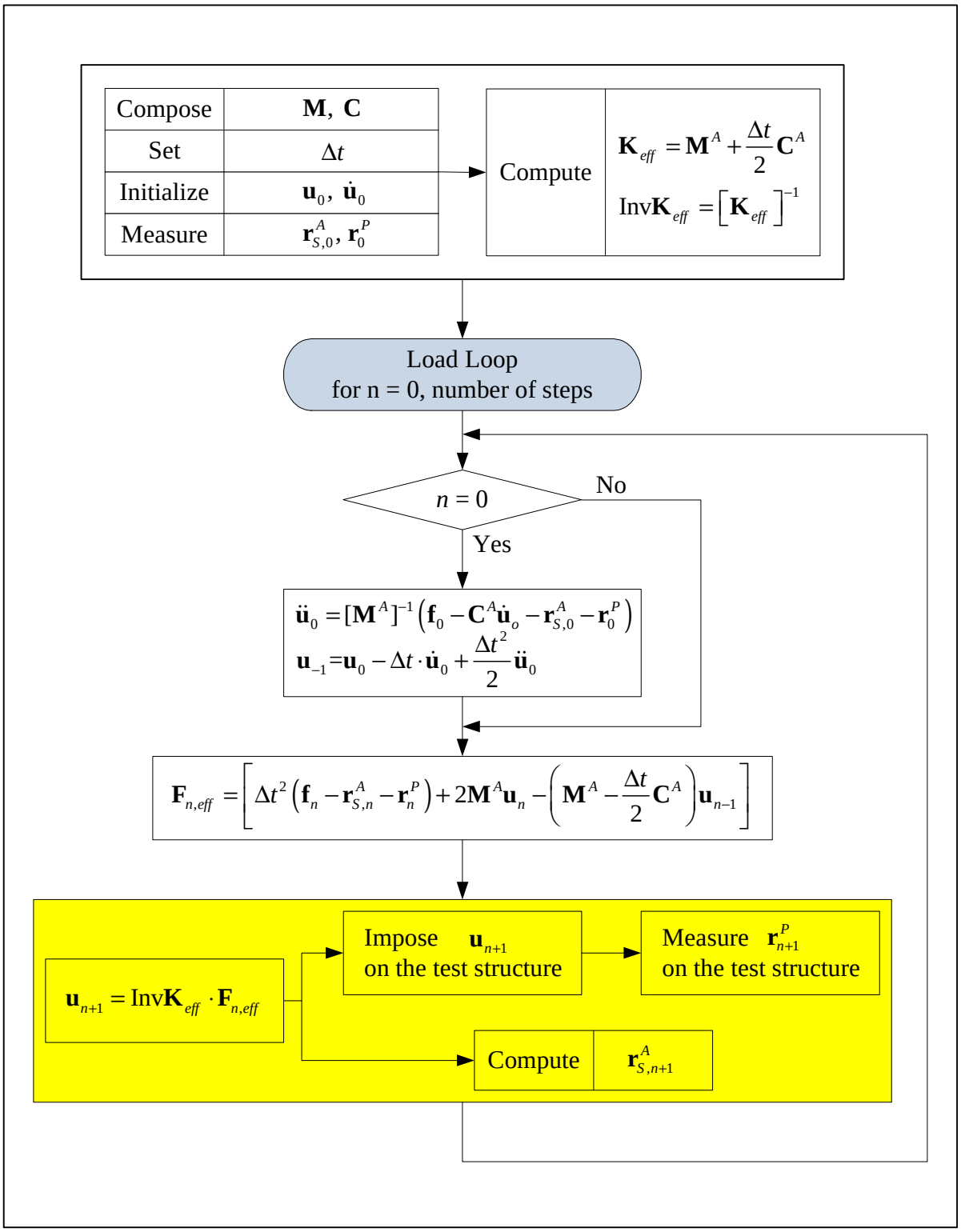


Fig. 8.2 Flow chart of central difference method for hybrid test

given by

$$\Delta t_{cr} = \frac{T_n}{\pi}$$

where, T_n is the smallest natural period corresponding to highest natural frequency ω. Fig. 8.2 explains the implementation of the central difference method for hybrid test.

2. Newmark β explicit method

   Mahin and Williams (1981) proposed the application of Newmark β explicit method for hybrid test. This mehtod consists of the following equations in each time step:

$$\begin{aligned} \mathbf{f}_{n+1} &= \mathbf{M}^A \ddot{\mathbf{u}}_{n+1} + \mathbf{C}^A \dot{\mathbf{u}}_{n+1} + \mathbf{r}^A_{S,n+1} + \mathbf{r}^P_{n+1} && (8.9) \\ \mathbf{u}_{n+1} &= \mathbf{u}_n + \Delta t \dot{\mathbf{u}}_n + \frac{\Delta t^2}{2}\left[(1-2\beta)\ddot{\mathbf{u}}_n + 2\beta \ddot{\mathbf{u}}_{n+1}\right] && (8.10) \\ \dot{\mathbf{u}}_{n+1} &= \dot{\mathbf{u}}_n + \Delta t\left[(1-\gamma)\ddot{\mathbf{u}}_n + \gamma \ddot{\mathbf{u}}_{n+1}\right] && (8.11) \end{aligned}$$

   where, β and γ as integration parameters determine the stability and accuracy characteristic of the algorithm under consideration. If $\beta = 0$, this method is explicit method, and Eq. 8.10 becomes:

$$\mathbf{u}_{n+1} = \mathbf{u}_n + \Delta t \dot{\mathbf{u}}_n + \frac{\Delta t^2}{2}\ddot{\mathbf{u}}_n \qquad (8.12)$$

   Substituting Eq. 8.12 and Eq. 8.11 into Eq. 8.9, we can solve for $\ddot{\mathbf{u}}_{n+1}$.

$$\begin{aligned} \mathbf{M}^A \ddot{\mathbf{u}}_{n+1} + \mathbf{C}^A\left\{\dot{\mathbf{u}}_n + \Delta t\left[(1-\gamma)\ddot{\mathbf{u}}_n + \gamma\ddot{\mathbf{u}}_{n+1}\right]\right\} + \mathbf{r}^A_{S,n+1} + \mathbf{r}^P_{n+1} &= \mathbf{f}_{n+1} \\ \mathbf{K}_{eff}\ddot{\mathbf{u}}_{n+1} &= \mathbf{F}_{n+1,eff} \quad (8.13) \end{aligned}$$

   where,

$$\begin{aligned} \mathbf{K}_{eff} &= \mathbf{M}^A + \Delta t \gamma \mathbf{C}^A \\ \mathbf{F}_{n+1,eff} &= \mathbf{f}_{n+1} - \mathbf{C}^A\left[\dot{\mathbf{u}}_n + \Delta t(1-\gamma)\ddot{\mathbf{u}}_n\right] - \mathbf{r}^A_{S,n+1} - \mathbf{r}^P_{n+1} \end{aligned}$$

   Hence,

$$\ddot{\mathbf{u}}_{n+1} = [\mathbf{K}_{eff}]^{-1}\mathbf{F}_{n+1,eff}$$

   Fig. 8.3 explains the implementation of the Newmark β explicit method for hybrid test.

   If $\beta = 0$ and $\gamma = 1/2$, this method has the same stability and numerical properties as the central difference method, but has a more favorable error-propagation characteristic for displacement feedback errors in an experiment (Shing and Mahin 1983).

3. Modified Newmark method

   Shing and Mahin (1983) developed the modified Newmark method that provides a favorable numerical energy dissipation effect which is not available in the central difference method or Newmark β explicit method. This numerical energy dissipation damps out the spurious higher-mode reponse introduced by experimental errors. This method consists

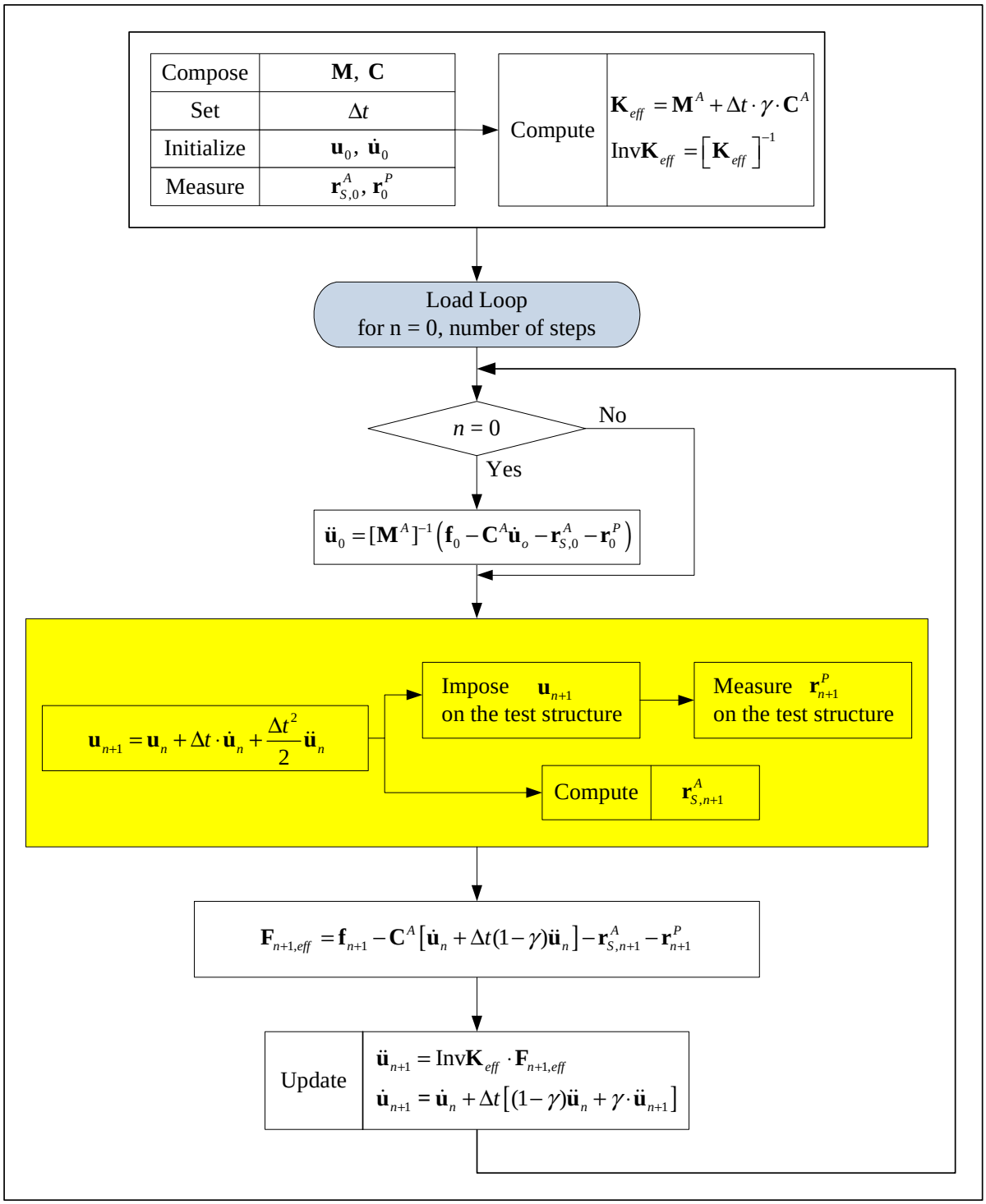


Fig. 8.3 Flow chart of Newmark β explicit method for hybrid test

of the following equations:

$$\mathbf{M}^A\ddot{\mathbf{u}}_{n+1}+(1+\alpha)(\mathbf{r}^A_{S,n+1}+\mathbf{r}^P_{n+1})+\frac{\rho}{\Delta t^2}\mathbf{M}^A\mathbf{u}_{n+1}=\mathbf{f}_{n+1}+\alpha(\mathbf{r}^A_{S,n}+\mathbf{r}^P_n)+\frac{\rho}{\Delta t^2}\mathbf{M}^A\mathbf{u}_n \quad (8.14)$$

$$\mathbf{u}_{n+1} = \mathbf{u}_n+\Delta t\dot{\mathbf{u}}_n+\frac{\Delta t^2}{2}\ddot{\mathbf{u}}_n \quad (8.15)$$

$$\dot{\mathbf{u}}_{n+1} = \dot{\mathbf{u}}_n+\frac{\Delta t}{2}(\ddot{\mathbf{u}}_n+\ddot{\mathbf{u}}_{n+1}) \quad (8.16)$$

Rewritting Eq. 8.14, we can solve for $\ddot{\mathbf{u}}_{n+1}$:

$$\begin{aligned}\mathbf{M}^A\ddot{\mathbf{u}}_{n+1} &= \mathbf{f}_{n+1}-(1+\alpha)(\mathbf{r}^A_{S,n+1}+\mathbf{r}^P_{n+1})+\alpha(\mathbf{r}^A_{S,n}+\mathbf{r}^P_n)+\frac{\rho}{\Delta t^2}\mathbf{M}^A(\mathbf{u}_n-\mathbf{u}_{n+1})\\ \ddot{\mathbf{u}}_{n+1} &= [\mathbf{M}^A]^{-1}\left[\mathbf{f}_{n+1}-(1+\alpha)(\mathbf{r}^A_{S,n+1}+\mathbf{r}^P_{n+1})+\alpha(\mathbf{r}^A_{S,n}+\mathbf{r}^P_n)\right]+\frac{\rho}{\Delta t^2}(\mathbf{u}_n-\mathbf{u}_{n+1})\end{aligned}$$

As shown in Eq. 8.14, with a careful selection of parameters α and ρ, damping is to be determined numerically in the algorithm instead of Damping matrix $\mathbf{C}^A$. Hence, this method can provide small damping for lower frequency modes and larger damping for the higher frequency modes. This is because this method provides numerical damping that increases with frequency. This method is recommanded for a structure with the multi-degree-of freedom, where experimental error propagation can be significant in the high frequency modes.

Shing and Mahin (1984) showed the stability limits as the following equation:

$$\sqrt{-\frac{\rho}{\alpha}}\le\omega\Delta t\le\frac{1+\sqrt{1-(1+\alpha)\rho}}{1+\alpha}$$

where, ω is the hightest natural frequency.

Fig. 8.4 explains the implementation of the Newmark β explicit method for hybrid test.

8.2.2.3 Implicit Methods

Since the explicit methods are conditionally stable, these can be limited in structures with mulit-degree-of freedom or very higher frequency mode. Hence, the implicit methods with unconditional stability can be useful method.

1. Newmark β implicit method

   The Newmark β implicit method can be expressed with Eq. 8.9, Eq. 8.10, and Eq. 8.11. Rewritting Eq. 8.10 and Eq. 8.11:

$$\mathbf{u}_{n+1} = \tilde{\mathbf{u}}_{n+1}+\Delta t^2\beta\ddot{\mathbf{u}}_{n+1} \quad (8.17)$$

$$\dot{\mathbf{u}}_{n+1} = \tilde{\dot{\mathbf{u}}}_{n+1}+\Delta t\gamma\ddot{\mathbf{u}}_{n+1} \quad (8.18)$$

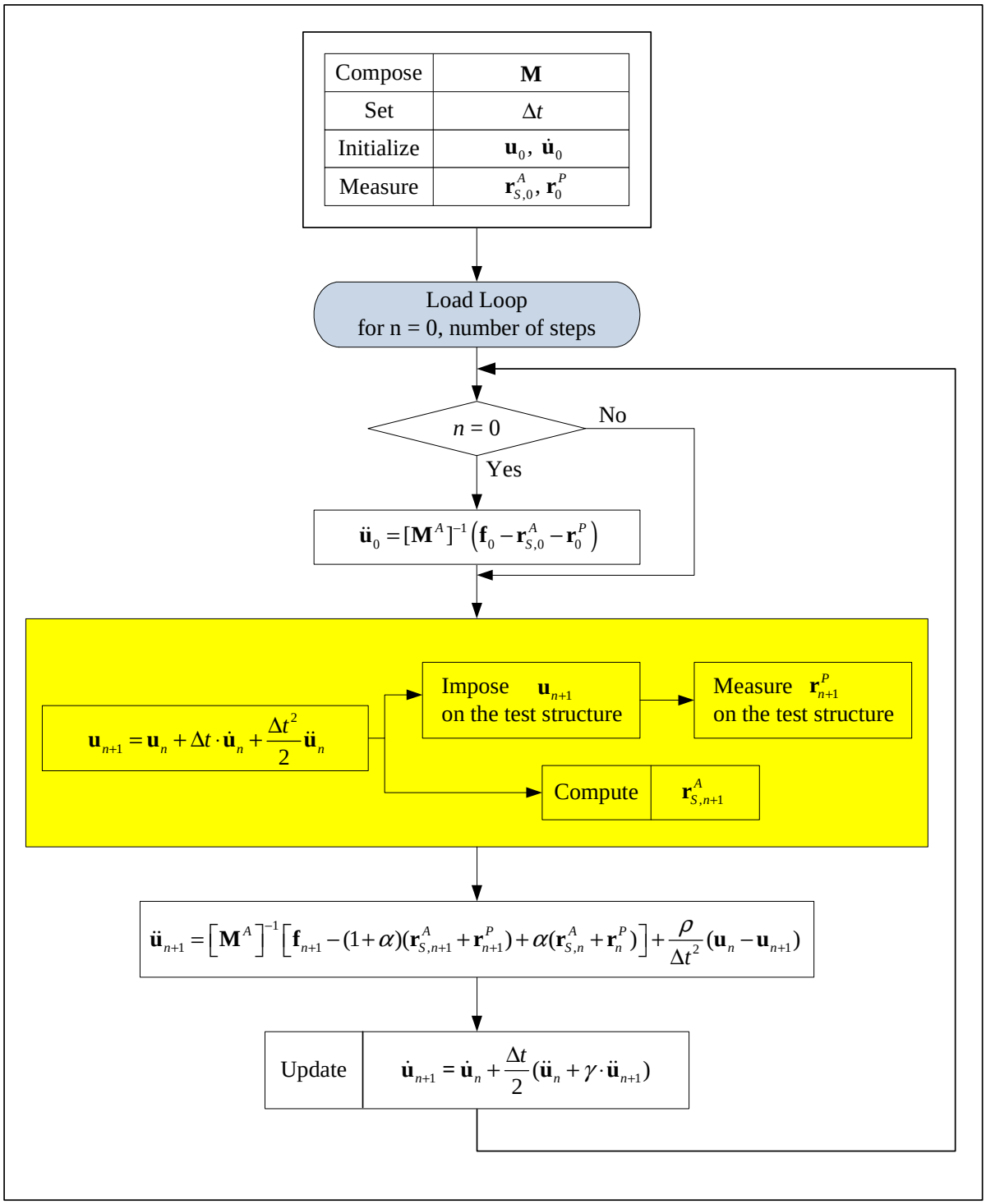


Fig. 8.4 Flow chart of the modified Newmark β explicit method for hybrid test

where,

$$\begin{aligned}\tilde{\mathbf{u}}_{n+1} &= \mathbf{u}_n + \Delta t\dot{\mathbf{u}}_n + \frac{\Delta t^2}{2}(1-2\beta)\ddot{\mathbf{u}}_n \\ \tilde{\dot{\mathbf{u}}}_{n+1} &= \dot{\mathbf{u}}_n + \Delta t(1-\gamma)\ddot{\mathbf{u}}_n\end{aligned}$$

Rewritting Eq. 8.17, we can solve for $\ddot{\mathbf{u}}_{n+1}$:

$$\ddot{\mathbf{u}}_{n+1} = \frac{\mathbf{u}_{n+1} - \tilde{\mathbf{u}}_{n+1}}{\Delta t^2\beta} \tag{8.19}$$

Substituting Eq. 8.19 into Eq. 8.18, we can rewrite $\dot{\mathbf{u}}_{n+1}$:

$$\dot{\mathbf{u}}_{n+1} = \tilde{\dot{\mathbf{u}}}_{n+1} + \frac{\gamma}{\Delta t\beta}(\mathbf{u}_{n+1} - \tilde{\mathbf{u}}_{n+1}) \tag{8.20}$$

Substituting Eq. 8.19 and Eq. 8.20 into Eq. 8.9, we have:

$$\mathbf{M}^A\left[\frac{\mathbf{u}_{n+1} - \tilde{\mathbf{u}}_{n+1}}{\Delta t^2\beta}\right] + \mathbf{C}^A\left[\tilde{\dot{\mathbf{u}}}_{n+1} + \frac{\gamma}{\Delta t\beta}(\mathbf{u}_{n+1} - \tilde{\mathbf{u}}_{n+1})\right] + \mathbf{r}^A_{S,n+1} + \mathbf{r}^P_{n+1} = \mathbf{f}_{n+1}$$

$$\begin{aligned}&\frac{1}{\Delta t^2\beta}\mathbf{M}^A\mathbf{u}_{n+1} + \frac{\gamma}{\Delta t\beta}\mathbf{C}^A\mathbf{u}_{n+1} + \mathbf{r}^A_{S,n+1} + \mathbf{r}^P_{n+1} \\ = \;& \mathbf{f}_{n+1} + \frac{1}{\Delta t^2\beta}\mathbf{M}^A\tilde{\mathbf{u}}_{n+1} + \frac{\gamma}{\Delta t\beta}\mathbf{C}^A\tilde{\mathbf{u}}_{n+1} - \mathbf{C}^A\tilde{\dot{\mathbf{u}}}_{n+1}\end{aligned} \tag{8.21}$$

If the trial solutions in given iteration step k are $\mathbf{u}^k_{n+1}$, $\mathbf{r}^{A,k}_{S,n+1}$, and $\mathbf{r}^{P,k}_{n+1}$, it does not satisfy the equations of motion. Hence, we can write for this particular step with residual force vector $\mathbf{R}^k_{n+1}$:

$$\mathbf{R}^k_{n+1} = \mathbf{f}_{n+1} + \overline{\mathbf{M}}\left(\tilde{\mathbf{u}}_{n+1} - \mathbf{u}^k_{n+1}\right) - \mathbf{C}^A\tilde{\dot{\mathbf{u}}}_{n+1} - \mathbf{r}^{A,k}_{S,n+1} - \mathbf{r}^{P,k}_{n+1} \tag{8.22}$$

where,

$$\overline{\mathbf{M}} = \frac{\mathbf{M}^A + \Delta t\gamma\mathbf{C}^A}{\Delta t^2\beta}$$

In this method, we need to estimate $\mathbf{M}^P$, $\mathbf{C}^P$, and the stiffness matrix of physical substructure $\mathbf{K}^P$ exactly. However, since measuring $\mathbf{K}^P$ during hybrid test is so complicated work and it is only used to estimate the updated displacements in the iteration process, the initial stiffness matrix $\mathbf{K}^P_0$ of physical substructure is used even though this may affect the convergence rate. For structures with softening behavior under seismic loading, $\mathbf{K}^P_0$ may be a little higher than the tangent or secant stiffness matrix to prevent the overshoot and undesiable loading and unloading cycles and to preserve the stability of the numberical solution (Shing 2008).

Assuming that the physical substructure has viscous damping and using the modified Newton Raphson iteration with the initial stiffness of the structure, we can solve for $\Delta\mathbf{u}_{n+1}^{k}$:

$$\mathbf{R}_{n+1}^{k} = \mathbf{K}_{eff}\Delta\mathbf{u}_{n+1}^{k} \tag{8.23}$$

where, $\mathbf{K}_{eff}$ is the effective stiffness matrix, and $\Delta\mathbf{u}_{n+1}^{k}$ is:

$$\Delta\mathbf{u}_{n+1}^{k} = \mathbf{u}_{n+1} - \mathbf{u}_{n+1}^{k}$$

To compute the effective stiffness matrix, we can express $\mathbf{r}_{n+1}^{P}$ as the following equation from Eq. 8.5:

$$\mathbf{r}_{n+1}^{P} = \mathbf{M}^{P}\ddot{\mathbf{u}}_{n+1} + \mathbf{C}^{P}\dot{\mathbf{u}}_{n+1} + \mathbf{r}_{S,n+1}^{P} \tag{8.24}$$

Also, we can rewrite $\mathbf{r}_{S,n+1}^{A}$ and $\mathbf{r}_{n+1}^{P}$:

$$\begin{aligned} \mathbf{r}_{S,n+1}^{A} &= \mathbf{K}_{0}^{A}\mathbf{u}_{n+1} && (8.25)\\ \mathbf{r}_{n+1}^{P} &= \mathbf{M}^{P}\ddot{\mathbf{u}}_{n+1} + \mathbf{C}^{P}\dot{\mathbf{u}}_{n+1} + \mathbf{K}_{0}^{P}\mathbf{u}_{n+1} && (8.26) \end{aligned}$$

Substituting Eq. 8.25 and Eq. 8.26 into Eq. 8.21, we can solve for $\mathbf{u}_{n+1}$:

$$\mathbf{K}_{eff}\mathbf{u}_{n+1} = \mathbf{f}_{n+1} + \left[\frac{(\mathbf{M}^{A}+\mathbf{M}^{P}) + \Delta t\gamma(\mathbf{C}^{A}+\mathbf{C}^{P})}{\Delta t^{2}\beta}\right]\tilde{\mathbf{u}}_{n+1} - (\mathbf{C}^{A}+\mathbf{C}^{P})\tilde{\dot{\mathbf{u}}}_{n+1}$$

where,

$$\mathbf{K}_{eff} = \frac{(\mathbf{M}^{A}+\mathbf{M}^{P}) + \Delta t\gamma(\mathbf{C}^{A}+\mathbf{C}^{P})}{\Delta t^{2}\beta} + (\mathbf{K}_{0}^{A}+\mathbf{K}_{0}^{P}) \tag{8.27}$$

From Eq. 8.23, we can solve for $\Delta\mathbf{u}_{n+1}^{k}$ and the updated displacement vector $\mathbf{u}_{n+1}^{k+1}$ at the next iteration step $k+1$:

$$\begin{aligned} \Delta\mathbf{u}_{n+1}^{k} &= [\mathbf{K}_{eff}]^{-1}\mathbf{R}_{n+1}^{k} \\ \mathbf{u}_{n+1}^{k+1} &= \mathbf{u}_{n+1}^{k} + \Delta\mathbf{u}_{n+1}^{k} && (8.28) \end{aligned}$$

The updated displacement vector $\mathbf{u}_{n+1}^{k+1}$ to be imposed on the test structure would be sent to the actuator controller for the current iteration. However, the measured displacement vector $\mathbf{u}_{n+1}^{m,k+1}$ may not be the same as the updated displacement vector $\mathbf{u}_{n+1}^{k+1}$ due to the limitation of a loading equipment. The measured displacement vector $\mathbf{u}_{n+1}^{m,k+1}$ for computing the next iteration will be used as the new updated displacement vector.

$$\mathbf{u}_{n+1}^{k} = \mathbf{u}_{n+1}^{m,k+1}$$

This method needs iterations using modified Newton-Raphson method as shown in Fig. 8.5 until $\mathbf{u}_{n+1}^{k}$ converges to the exact solution. For this reason, this may results in following problems for real-time hybrid test: (a) $\mathbf{u}_{n+1}^{k}$ cannot be directly imposed on the structural

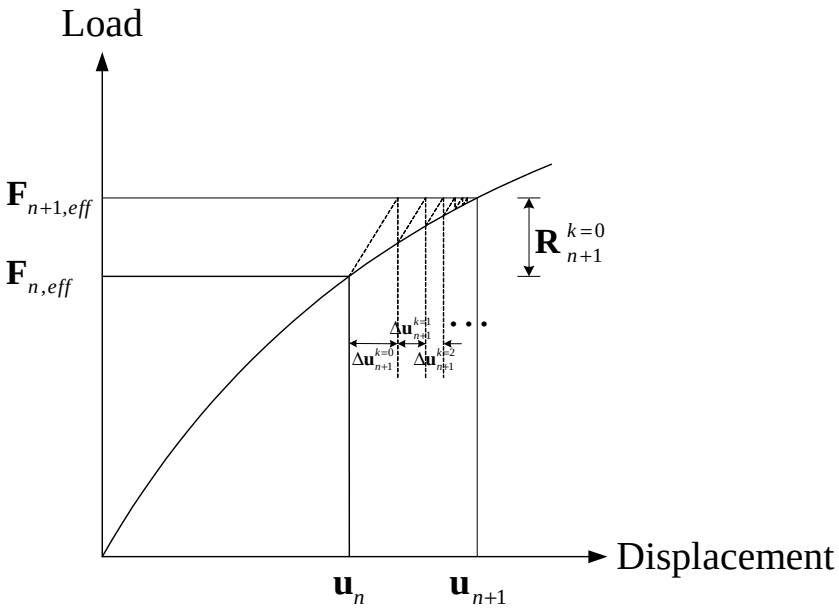


Fig. 8.5 Modified Newton-Raphson iteration in hybrid test

specimen because this may lead to undesired velocity fluctuations during iteration, and (b) this may lead to uncertainties in actuator speed such as the slowing of actuator speed because the number of iteration for correction of displacement vector can vary from one time step to the next time step depending on the degree of nonlinearity developed by the structure, and correction of displacement vector follows a 2^{nd}-order convergence.

To avoid these problems, a special iterative method that has a fixed number of iterations in each time step has been proposed, and relies on quadratic interpolation function to assure a smooth motion of the actuators during iteration (Jung 2005). The quadratic interpolation function is based on the updated displacement vector $\mathbf{u}^{k+1}_{n+1}$ and the converged displacement vector in the previous time steps. In this procedure, the number of the actuator movement within a time step is limited with i iteration steps. Fig. 8.6 illustrates the procedure in detail. Instead of the updated displacement vector $\mathbf{u}^{k+1}_{n+1}$, the desired diplacement vector $\mathbf{u}^{d,k+1}_{n+1}$ at current iteration step which is sent to actuator controller can be expressed by the following equation:

$$\mathbf{u}^{d,k+1}_{n+1} = \begin{bmatrix} C1 & C2 & C3 \end{bmatrix} \begin{Bmatrix} \mathbf{u}^{d,i}_{n-1} \\ \mathbf{u}^{d,i}_{n} \\ \mathbf{u}^{k+1}_{n+1} \end{Bmatrix} \tag{8.29}$$

where, k vary from 0 to $i-1$, i denotes the total number of specified iteration in each

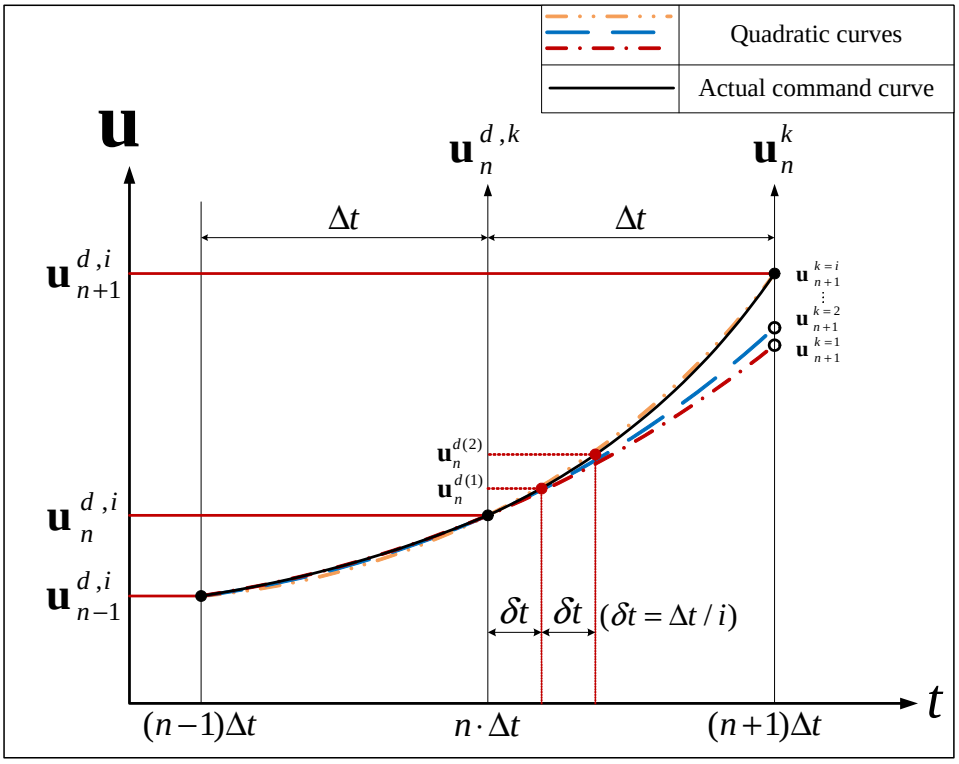


Fig. 8.6 Iteration scheme using quadratic interpolation

time step, and

$$
\begin{aligned}
C1 &= \frac{1}{2}\left[\left(\frac{k+1}{i}\right)^2 - \left(\frac{k+1}{i}\right)\right] \\
C2 &= 1 - \left(\frac{k+1}{i}\right)^2 \\
C3 &= \frac{1}{2}\left[\left(\frac{k+1}{i}\right)^2 + \left(\frac{k+1}{i}\right)\right]
\end{aligned}
$$

i becomes $\Delta t/\delta t$, where δt means the time interval of the actuator controller in a time step. In the first time step, the quadratic interpolation is based on the initial displacement and velocity vectors.

$$
\mathbf{u}_1^{d,k+1} = C2 \cdot \mathbf{u}_0 - 2 \cdot i \cdot C1 \cdot \dot{\mathbf{u}}_0 + \left(\frac{k+1}{i}\right)^2 \mathbf{u}_1^{k+1}
$$

In this procedure, experience indicates that 10 iterations in a time step will provide satisfactory convergence even for strongly nonlinear structural responses provided that there is no severe strain softening (Wei 2005).

Since the number of iterations in each time step is fixed, convergence errors are expected by

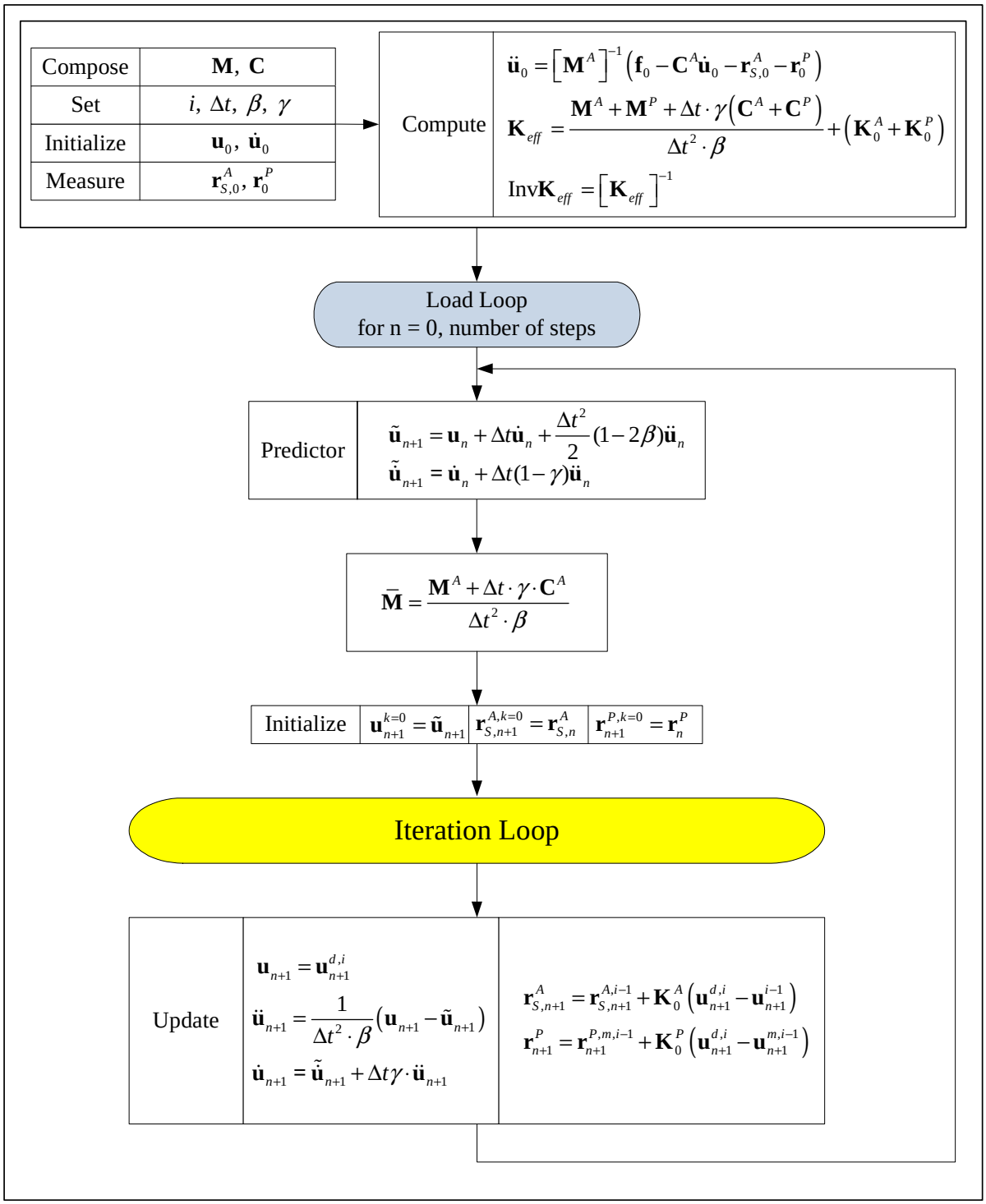


Fig. 8.7 Flow chart (1) of the Newmark β implicit method for hybrid test

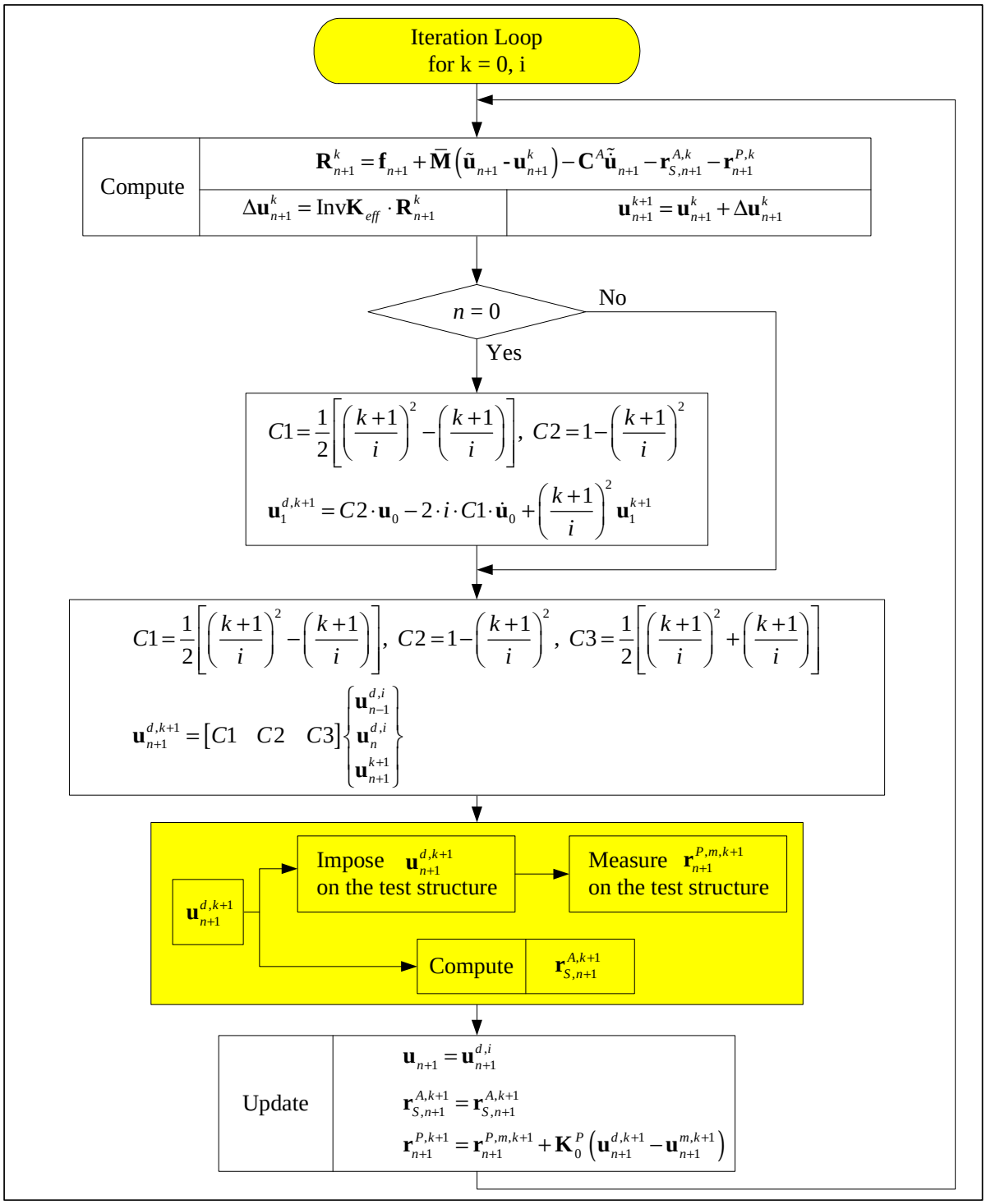


Fig. 8.8 Flow chart (2) of the Newmark β implicit method for hybrid test

the time delays in actuator response and in the data exchange between the controller/data acquisition processor and the target computer. To minimize the effect of these errors and enforce equilibrium at the end of each time step, the following approximate correction can be introduced for the updated displacement vector and force vector in the last iteration (Shing 2008).

$$\begin{aligned} \mathbf{u}_{n+1} &= \mathbf{u}_{n+1}^{d,i} \\ \mathbf{r}_{S,n+1}^{A} &= \mathbf{r}_{S,n+1}^{A,i-1} + \mathbf{K}_0^A \left[\mathbf{u}_{n+1}^{d,i} - \mathbf{u}_{n+1}^{i-1}\right] \qquad (8.30) \\ \mathbf{r}_{n+1}^{P} &= \mathbf{r}_{n+1}^{P,m,i-1} + \mathbf{K}^{P*} \left[\mathbf{u}_{n+1}^{d,i} - \mathbf{u}_{n+1}^{m,i-1}\right] \qquad (8.31) \end{aligned}$$

where, $\mathbf{u}_{n+1}$, $\mathbf{r}_{S,n+1}^{A}$, and $\mathbf{r}_{n+1}^{P}$ are the updated displacement vector, force vectors, $\mathbf{r}_{n+1}^{P,m,i-1}$ is restoring force vector measured from the test structure at the beginning of the last iteration, and $\mathbf{K}^{P*}$ is the portion of Eq. 8.27 that contaions only the properties of the physical substructure.

In addition, to avoid displacement incompatibility between the analytical and physical substructures at the boundary degrees of freedom, it is desired that the displacements and forces measured from the physical substructure be corrected in each iteration as follows (Wei 2005):

$$\begin{aligned} \mathbf{u}_{n+1}^{k+1} &= \mathbf{u}_{n+1}^{d,k+1} \\ \mathbf{r}_{n+1}^{P,k+1} &= \mathbf{r}_{n+1}^{P,m,k+1} + \mathbf{K}^{P*} \left[\mathbf{u}_{n+1}^{d,k+1} - \mathbf{u}_{n+1}^{m,k+1}\right] \end{aligned}$$

Fig. 8.7 and Fig. 8.8 explain the implementation of the Newmark β implicit method for hybrid test.

2. HHT implicit method

HHT implicit method basically has the sams procedure as Newmark β implicit method. However, this presents a technical difficulty when the inertia force vector developed by the physical substructure become significant and cannot be ignored (Shing 2008). If $-1/3 \leq \alpha \leq 0$, $(1-\alpha)^2/4$, and $\gamma = (1-2\alpha)/2$, then this method is unconditionally stable and has a second-order accuracy. HHT implicit method can be expressed with Eq. 8.17, Eq. 8.18, and the following eqauation:

$$\begin{aligned} (1+\alpha)\mathbf{f}_{n+1} - \alpha\mathbf{f}_n &= \mathbf{M}^A\ddot{\mathbf{u}}_{n+1} + (1+\alpha)\mathbf{C}^A\dot{\mathbf{u}}_{n+1} - \alpha\mathbf{C}^A\dot{\mathbf{u}}_n \\ &\quad +(1+\alpha)\mathbf{r}_{S,n+1}^A - \alpha\mathbf{r}_{S,n}^A + (1+\alpha)\mathbf{r}_{n+1}^P - \alpha\mathbf{r}_n^P \qquad (8.32) \end{aligned}$$

Substituting Eq. 8.19 and Eq. 8.20 into Eq. 8.32, we have:

$$\begin{aligned} (1+\alpha)\mathbf{f}_{n+1} - \alpha\mathbf{f}_n &= \mathbf{M}^A\left[\frac{\mathbf{u}_{n+1} - \tilde{\mathbf{u}}_{n+1}}{\Delta t^2\beta}\right] + (1+\alpha)\mathbf{C}^A\left[\tilde{\dot{\mathbf{u}}}_{n+1} + \frac{\gamma}{\Delta t\beta}(\mathbf{u}_{n+1} - \tilde{\mathbf{u}}_{n+1})\right] \\ &\quad -\alpha\mathbf{C}^A\dot{\mathbf{u}}_n + (1+\alpha)\mathbf{r}_{S,n+1}^A - \alpha\mathbf{r}_{S,n}^A + (1+\alpha)\mathbf{r}_{n+1}^P - \alpha\mathbf{r}_n^P \end{aligned}$$

$$\begin{aligned} & \frac{1}{\Delta t^2\beta}\mathbf{M}^A\mathbf{u}_{n+1} + \frac{\gamma}{\Delta t\beta}(1+\alpha)\mathbf{C}^A\mathbf{u}_{n+1} + (1+\alpha)\left(\mathbf{r}^A_{S,n+1} + \mathbf{r}^P_{n+1}\right) \\ = \;& (1+\alpha)\mathbf{f}_{n+1} - \alpha\mathbf{f}_n + \frac{1}{\Delta t^2\beta}\mathbf{M}^A\tilde{\mathbf{u}}_{n+1} + \frac{\gamma}{\Delta t\beta}(1+\alpha)\mathbf{C}^A\tilde{\mathbf{u}}_{n+1} \\ & -(1+\alpha)\mathbf{C}^A\tilde{\dot{\mathbf{u}}}_{n+1} + \alpha\mathbf{C}^A\dot{\mathbf{u}}_n + \alpha\left(\mathbf{r}^A_{S,n} + \mathbf{r}^P_n\right) \end{aligned} \tag{8.33}$$

If the trial solutions in given iteration step k are $\mathbf{u}^k_{n+1}$, $\mathbf{r}^{A,k}_{S,n+1}$, and $\mathbf{r}^{P,k}_{n+1}$, it does not satisfy the equations of motion. Hence, we can write for this particular step with residual force vector $\mathbf{R}^k_{n+1}$:

$$\begin{aligned} \mathbf{R}^k_{n+1} = \;& (1+\alpha)\mathbf{f}_{n+1} - \alpha\mathbf{f}_n + \overline{\mathbf{M}}\left(\tilde{\mathbf{u}}_{n+1} - \mathbf{u}^k_{n+1}\right) - (1+\alpha)\mathbf{C}^A\tilde{\dot{\mathbf{u}}}_{n+1} + \alpha\mathbf{C}^A\dot{\mathbf{u}}_n \\ & -(1+\alpha)\left(\mathbf{r}^{A,k}_{S,n+1} + \mathbf{r}^{P,k}_{n+1}\right) + \alpha\left(\mathbf{r}^A_{S,n} + \mathbf{r}^P_n\right) \end{aligned} \tag{8.34}$$

where,

$$\overline{\mathbf{M}} = \frac{\mathbf{M}^A + \Delta t\gamma(1+\alpha)\mathbf{C}^A}{\Delta t^2\beta}$$

Assuming that the physical substructure has viscous damping and using the modified Newton Raphson iteration with the initial stiffness of the structure, we can solve for $\Delta\mathbf{u}^k_{n+1}$:

$$\mathbf{R}^k_{n+1} = \mathbf{K}_{eff}\Delta\mathbf{u}^k_{n+1} \tag{8.35}$$

where, $\mathbf{K}_{eff}$ is the effective stiffness matrix, and $\Delta\mathbf{u}^k_{n+1}$ is:

$$\Delta\mathbf{u}^k_{n+1} = \mathbf{u}_{n+1} - \mathbf{u}^k_{n+1}$$

To compute the effective stiffness matrix, we can express $\mathbf{r}^P_{n+1}$ as the following equation:

$$\mathbf{r}^P_{n+1} = \mathbf{M}^P\ddot{\mathbf{u}}_{n+1} + (1+\alpha)\mathbf{C}^P\ddot{\mathbf{u}}_{n+1} - \alpha\mathbf{C}^P\ddot{\mathbf{u}}_n + (1+\alpha)\mathbf{r}^P_{S,n+1} - \alpha\mathbf{r}^P_{S,n} \tag{8.36}$$

Like Newmark β implicit method, we can rewrite $\mathbf{r}^A_{S,n+1}$ and $\mathbf{r}^P_{n+1}$:

$$\mathbf{r}^A_{S,n+1} = \mathbf{K}^A_0\mathbf{u}_{n+1} \tag{8.37}$$

$$\begin{aligned} (1+\alpha)\mathbf{r}^P_{n+1} - \alpha\mathbf{r}^P_n \approx \;& \mathbf{M}^P\ddot{\mathbf{u}}_{n+1} + (1+\alpha)\mathbf{C}^P\dot{\mathbf{u}}_{n+1} - \alpha\mathbf{C}^P\dot{\mathbf{u}}_n \\ & +(1+\alpha)\mathbf{K}^P_0\mathbf{u}_{n+1} - \alpha\mathbf{r}^P_{S,n} \end{aligned} \tag{8.38}$$

Substituting Eq. 8.37 and Eq. 8.38 into Eq. 8.33, we can solve for $\mathbf{u}_{n+1}$:

$$\begin{aligned}\mathbf{K}_{eff}\mathbf{u}_{n+1} &= (1+\alpha)\mathbf{f}_{n+1} - \alpha\mathbf{f}_n + \left[\frac{(\mathbf{M}^A+\mathbf{M}^P)+\Delta t\gamma(1+\alpha)(\mathbf{C}^A+\mathbf{C}^P)}{\Delta t^2\beta}\right]\tilde{\mathbf{u}}_{n+1} \\ &\quad -(1+\alpha)(\mathbf{C}^A+\mathbf{C}^P)\tilde{\dot{\mathbf{u}}}_{n+1} + \alpha(\mathbf{C}^A+\mathbf{C}^P)\dot{\mathbf{u}}_n + \alpha(\mathbf{r}^A_{S,n}+\mathbf{r}^P_{S,n})\end{aligned}$$

where,

$$\mathbf{K}_{eff} = \frac{(\mathbf{M}^A+\mathbf{M}^P)+\Delta t\gamma(1+\alpha)(\mathbf{C}^A+\mathbf{C}^P)}{\Delta t^2\beta} + (1+\alpha)(\mathbf{K}^A_0+\mathbf{K}^P_0) \tag{8.39}$$

From Eq. 8.35, we can solve for $\Delta\mathbf{u}^k_{n+1}$ and the updated displacement vector $\mathbf{u}^{k+1}_{n+1}$ at the next iteration step $k+1$:

$$\begin{aligned}\Delta\mathbf{u}^k_{n+1} &= [\mathbf{K}_{eff}]^{-1}\mathbf{R}^k_{n+1} \\ \mathbf{u}^{k+1}_{n+1} &= \mathbf{u}^k_{n+1} + \Delta\mathbf{u}^k_{n+1}\end{aligned} \tag{8.40}$$

Fig. 8.9 and Fig. 8.10 explain the implementation of the HHT implicit method for hybrid test.

3. Operator-Splitting method

Since Operator-Splitting (OS) method does not require iteration but is unconditiionally stable method, it is so ambiguous for Operator-Splitting method to be classified as either explicit or implicit methods. However, the OS method is based on implicit methods and uses a predictor-corrector approach not to require iterations for nonliner response of structure. For this reason, we can classify the OS method as implicit method. This method was developed by Nakashima, Kaminosomo and Ishida (1990) for hybrid test, and applied by Wu, Xu, Wang and Williams (2005) for real-time hybrid test to test a damper system where the actual mass of the specimen is negligible. Wu et al. (2005) assumed the predictor velocity $\tilde{\dot{\mathbf{u}}}_{n+1}$ is constant in each time step and have concluded that the unconditional stability of the integration scheme can no longer be preserved by having the predictor velocity that is not consistent with the OS method. For real-time hybrid tests with the velocity-dependent physical substructure, the equations of motion are expressed as follows:

$$\begin{aligned}\mathbf{f}_{n+1} &= \mathbf{M}^A\ddot{\mathbf{u}}_{n+1} + \mathbf{C}^A\dot{\mathbf{u}}_{n+1} + \mathbf{K}^A_0(\mathbf{u}_{n+1}-\tilde{\mathbf{u}}_{n+1}) + \tilde{\mathbf{r}}^A_{S,n+1}(\tilde{\mathbf{u}}_{n+1}) \\ &\quad +\mathbf{M}^P\ddot{\mathbf{u}}_{n+1} + \mathbf{C}^P(\dot{\mathbf{u}}_{n+1}-\tilde{\dot{\mathbf{u}}}_{n+1}) + \mathbf{K}^P_0(\mathbf{u}_{n+1}-\tilde{\mathbf{u}}_{n+1}) \\ &\quad +\tilde{\mathbf{r}}^P_{n+1}(\tilde{\dot{\mathbf{u}}}_{n+1}, \tilde{\mathbf{u}}_{n+1})\end{aligned} \tag{8.41}$$

where, $\tilde{\dot{\mathbf{u}}}_{n+1}$ and $\tilde{\mathbf{u}}_{n+1}$ are the predictor velocity and displacement vectors, and $\ddot{\mathbf{u}}_{n+1}$ is the acceleration vector, and $\dot{\mathbf{u}}_{n+1}$ and $\mathbf{u}_{n+1}$ are the corrected velocity and displacement

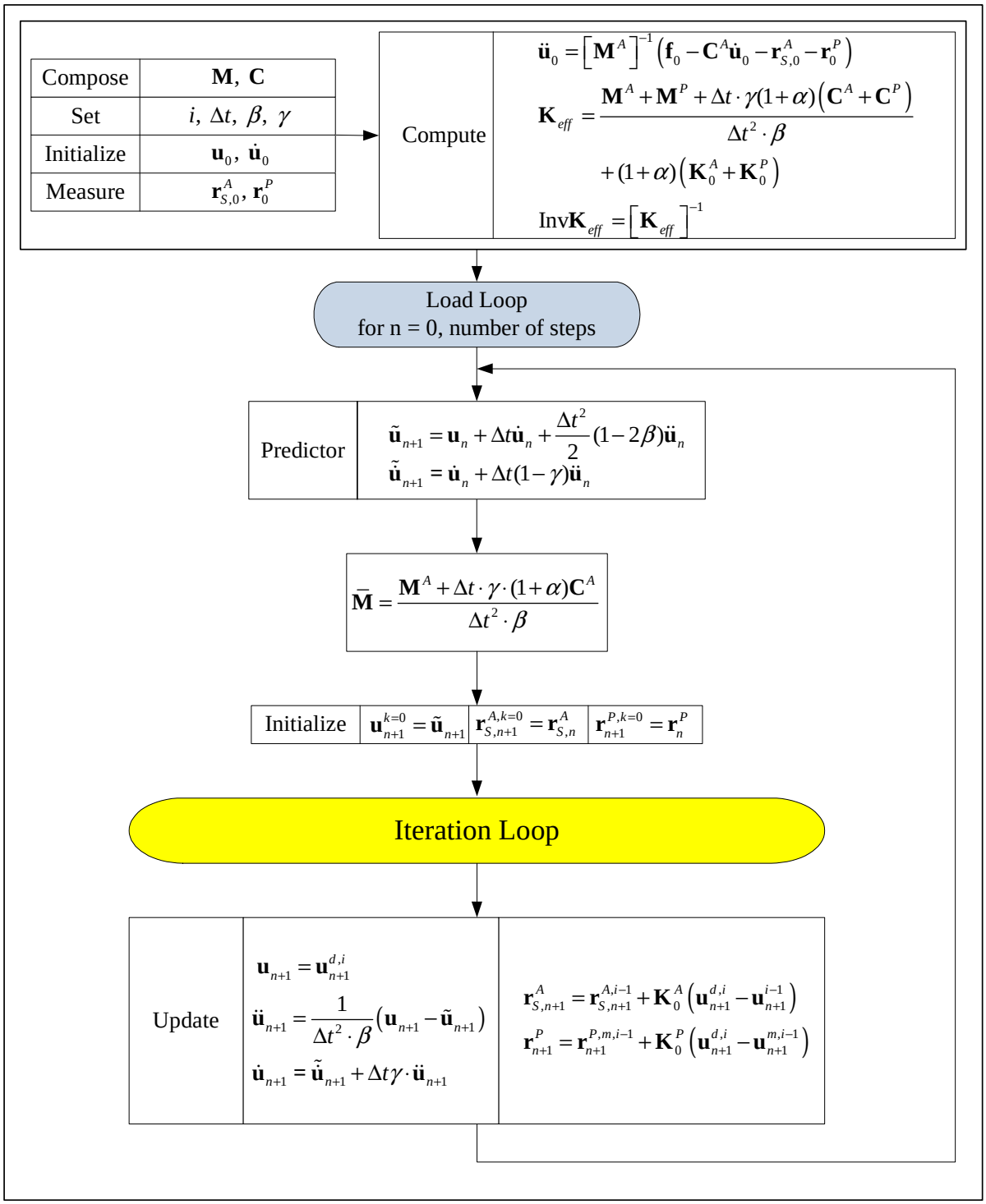


Fig. 8.9 Flow chart (1) of the HHT implicit method for hybrid test

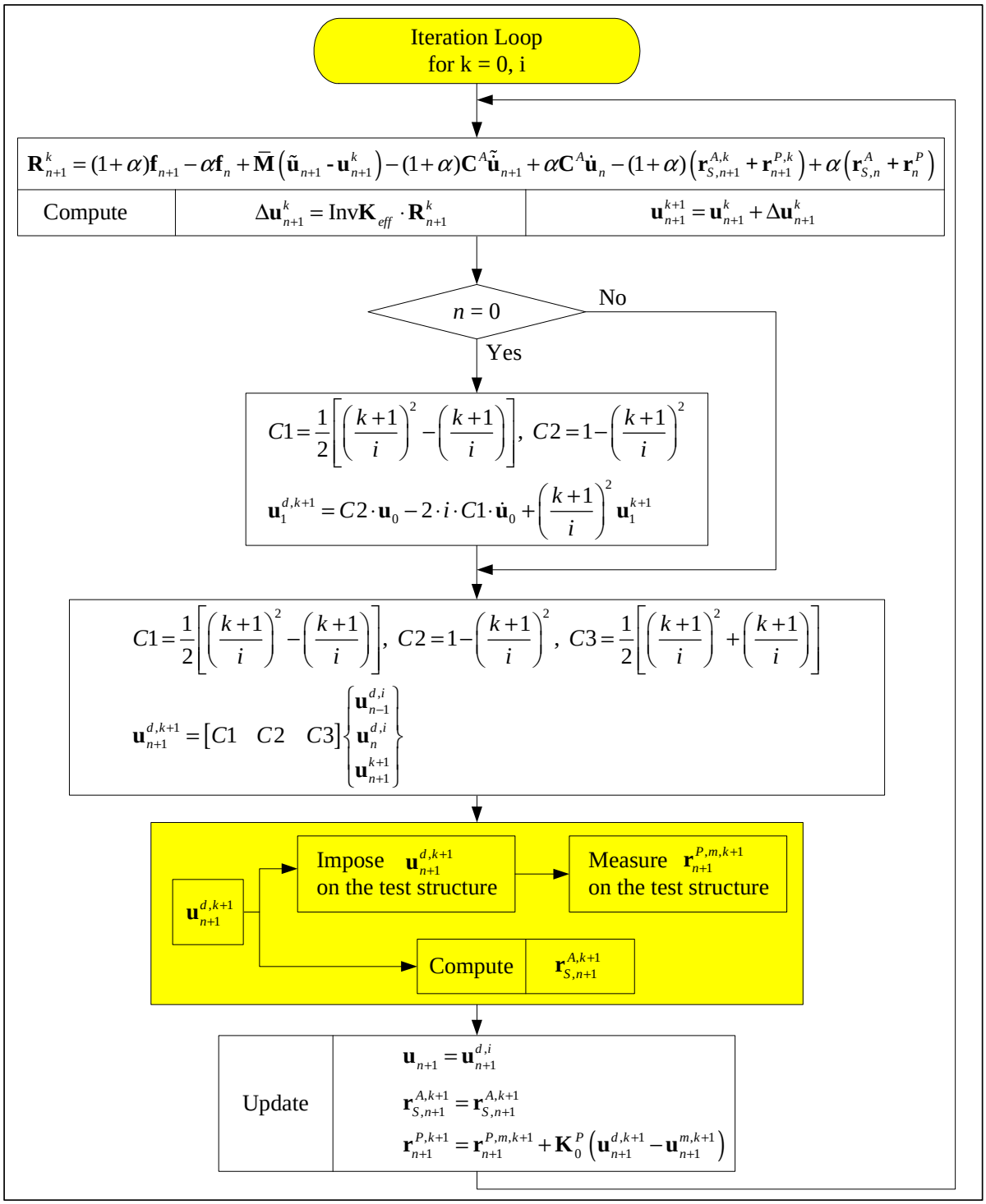


Fig. 8.10 Flow chart (2) of the HHT implicit method for hybrid test

vectors;

$$\tilde{\mathbf{u}}_{n+1} = \mathbf{u}_n + \Delta t \dot{\mathbf{u}}_n + \frac{\Delta t^2}{4}\ddot{\mathbf{u}}_n \tag{8.42}$$

$$\tilde{\dot{\mathbf{u}}}_{n+1} = \dot{\mathbf{u}}_n + \frac{\Delta t}{2}\ddot{\mathbf{u}}_n \tag{8.43}$$

$$\mathbf{u}_{n+1} = \tilde{\mathbf{u}}_{n+1} + \frac{\Delta t^2}{4}\ddot{\mathbf{u}}_{n+1} \tag{8.44}$$

$$\dot{\mathbf{u}}_{n+1} = \tilde{\dot{\mathbf{u}}}_{n+1} + \frac{\Delta t}{2}\ddot{\mathbf{u}}_{n+1} \tag{8.45}$$

and $\tilde{\mathbf{r}}^A_{S,n+1}(\tilde{\mathbf{u}}_{n+1})$ and $\tilde{\mathbf{r}}^P_{n+1}(\tilde{\dot{\mathbf{u}}}_{n+1}, \tilde{\mathbf{u}}_{n+1})$ are the restoring force vectors of analytical and physical substructure to be computed and measured with the predictor vectors, respectively. In this method, the predictor displacement vector $\tilde{\mathbf{u}}_{n+1}$ is imposed on the test structure with $\tilde{\dot{\mathbf{u}}}_{n+1}$, and then the measured restoring force vector $\tilde{\mathbf{r}}^{P,m}_{n+1}$ is measured and $\tilde{\mathbf{r}}^A_{S,n+1}$ is computed. Substituting Eq. 8.42 and Eq. 8.43 into Eq. 8.41, we have Eq. 8.46:

$$\mathbf{K}_e f f \ddot{\mathbf{u}}_{n+1} = \mathbf{F}_{n+1,eff} \tag{8.46}$$

where,

$$\begin{aligned}
\mathbf{K}_{eff} &= \left(\mathbf{M}^A + \mathbf{M}^P\right) + \frac{\Delta t}{2}\left(\mathbf{C}^A + \mathbf{C}^P\right) + \frac{\Delta t^2}{4}\left(\mathbf{K}^A_0 + \mathbf{K}^P_0\right) \\
\mathbf{F}_{n+1,eff} &= \mathbf{f}_{n+1} - \mathbf{C}^P\tilde{\dot{\mathbf{u}}}_{n+1} - \tilde{\mathbf{r}}^A_{S,n+1}(\tilde{\mathbf{u}}_{n+1}) - \tilde{\mathbf{r}}^P_{n+1}(\tilde{\dot{\mathbf{u}}}_{n+1}, \tilde{\mathbf{u}}_{n+1})
\end{aligned}$$

In this method, we can slove for $\ddot{\mathbf{u}}_{n+1}$ without iterations. We needs a correction of the measured force vector based on the initial stiffness and the error in displacement feedback. For this reason, the restoring force vector of physical substructure will be corrected as follows:

$$\tilde{\mathbf{r}}^P_{n+1} = \tilde{\mathbf{r}}^{P,m}_{n+1} + \mathbf{K}^P_0\left(\tilde{\mathbf{u}}_{n+1} - \tilde{\mathbf{u}}^m_{n+1}\right)$$

where, $\tilde{\mathbf{u}}^m_{n+1}$ is the displacement vector measured from the test structure.

Fig. 8.11 explains the implementation of the Operator-Splitting method for hybrid test.

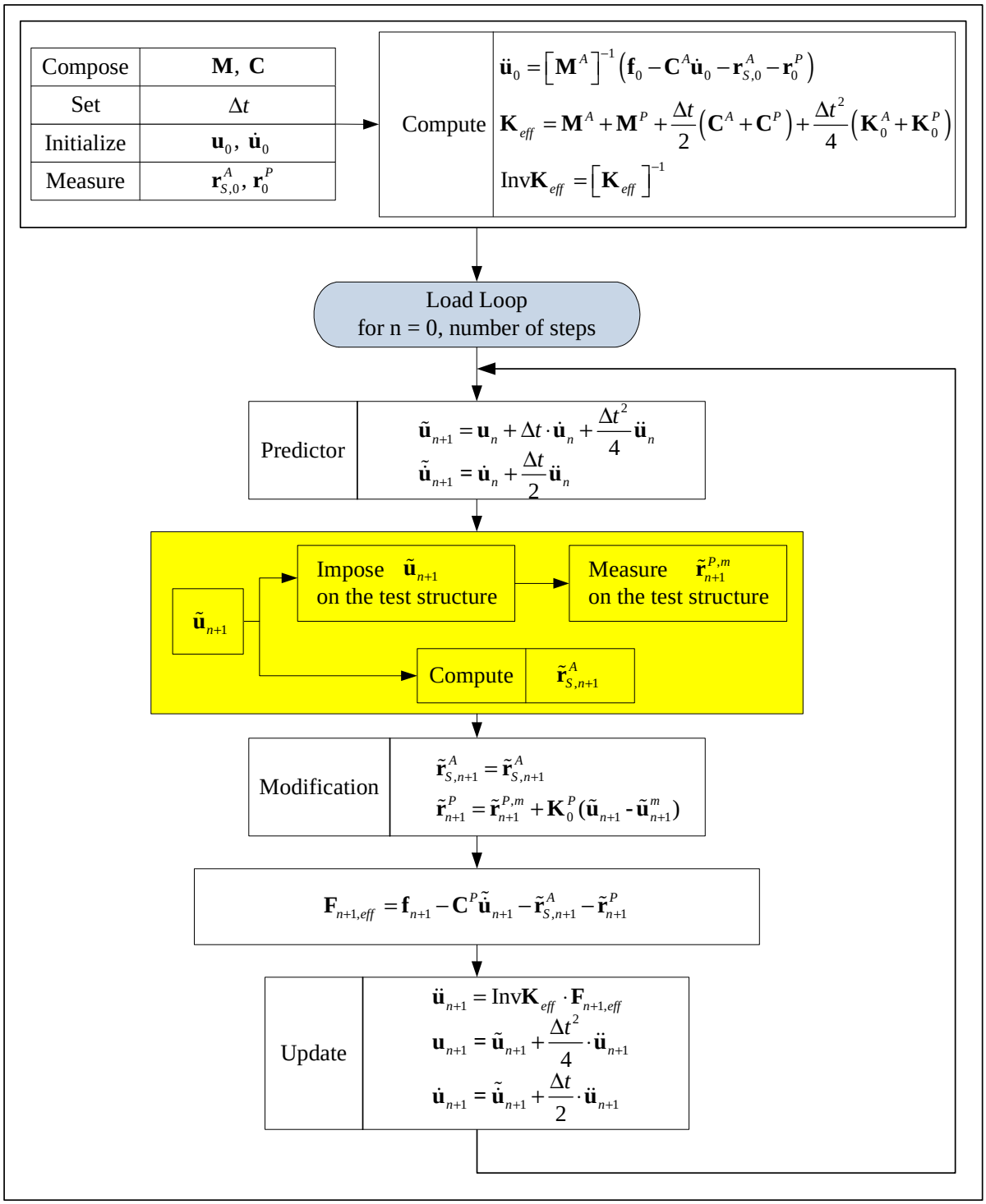


Fig. 8.11 Flow chart of the Operator-Splitting method for hybrid test

Chapter 9

PARALLEL COMPUTATION

9.1 Using Parallel Computation in Finite-Element Analysis

For Finite-Element analysis of structures, there are several opportunities for parallelization; here we examine the parallelization of two major computation processes in a typical analysis: the matrix solver and per-element computation.

The matrix solver is a key component in finite-element computation because it is used every iteration to compute the displacement delta from the residual using the stiffness matrix: $\delta u = K^{-1}r$. For large systems the matrix solver takes up the bulk of analysis time. Speeding up the matrix solution time with a parallel matrix solver can produce a dramatic speedup of the overall analysis time.

Per-element computation can consume a large amount of overall analysis time, particularly if the computation in each element is non-trivial. Many of the structures considered a CU-Boulder contain many flexibility-based beam-column elements, which rely on internal iteration to solve a complex relation involving multiple sections and materials. The computation required for any single element is small, but when the structure contains several hundred or thousand elements the total computation is considerable.

Speeding up both the matrix solver and per-element computation can be accomplished with paralleization. The computation is divided up amongst several processors, each of which work on part of the problem. The results from each processor are then re-assembled into a final product for use in further computation.

We will consider two forms of parallelization:

Shared-memory parallelism or multithreading, where there exist multiple processors on a single computer. This would typically be a high performance desktop computer with two or more processors, each possibly quad-core, and a total cost under $8k. When the program starts up, several "threads" or streams of computation are started up and are used to distribute computation over several processors inside the computer. Multithreaded programs are often easier to write and test than distributed-memory parallelism, but are limited by the maximum number of processors that can be embedded within a single computer.

Distributed-memory parallelism where there are many computers each with their own memory space which send data to each other using a computer network. Distributed-memory

parallel programs are usually harder to write and suffer more overhead due to the communication between computers. MPI is the standard programming environment for such application. However, they are often capable of running much larger problems than an equivalent multithreaded program. Typically, this would have at least 65 processors, and the number can be up to thousands. For reference, the University of Colorado is acquiring a 100+ teraflops peak performance computer (an NSF award to Prof. Tuffo), with half a petabyte of storage, 12 gigabytes of RAM. It will have a tightly coupled mesh interconnect via a quaddata-rate InfiniBand and available through a 10 Gbps network on campus.

9.2 Parallel Computation in Real-Time Hybrid Testing

When targetting parallelism for hybrid testing, specifically real-time hybrid testing, the upper bounds in per-iteration computation time become a dominant factor. For instance, in the real-time hybrid tests performed at CU-Boulder, each iteraton of the Newton-Raphson solver must complete in 0.97mS. A message sent using MPI over Infiniband (a high-speed interconnect used in cluster computing) typically has 1.3uS to 3uS end-to-end latency, or about 1-3% of the total time allowed for each iteration. If multiple messages are sent back and forth, this number can quickly grow. For interconnects with lower performance this can be much larger; for example, MPICH over TCP can exhibit delays of hundreds of milliseconds (*Euro PVM/MPI 2003* 2003).

CU-Boulder is pursuing a dual track approach to the parallelizatio of Mercury:

1. Shared-memory multithreading approach which avoids the message passing overhead, and which can be implemented on a 2 Quad-Core CP's on our Dell Workstation. In multithreading the overhead of message passing is lower or nonexistant, and limits on problem size are less of a factor since the models are relatively small (less than 1,000 DOFs). Issues such as network reliability and bandwidth are also not a factor in multithreaded computation. We intend to use this model for the real time hybrid simulation of a reinforced concrete frame previously tested at Berkeley (Ghannoum and Moehle), and which has about 300 highly nonlinear elements.

2. Distributed memory approach which is currently being tested in our laboratory, and ultimately deployed on the supercomputer described above. Ultimately, the experience gained in parallelizing MERCURY in this environment, will help us pioneed RTHSof far more complex structures such as cars and rockets.

9.3 Parallelism Using Task-Based Multithreading

In order to speed up computation for Mercury, multithreading was introduced to two key sections of the program: solving the linear system, and per-element computation.

Solving the linear system is performed each and every iteration and consumes a large portion of the computational time. In Mercury the direct-sparse solver *CSparse* is replaced with a multithreaded direct sparse solver *PARDISO* available in the Intel MKL matrix library.

The per-element computation is in essence a large loop where each iteration of the loop processes a single element. Since the computations within each element are independent of other elements, the loop iterations can be sub-divided in smaller chunks which are processed by separate processors.

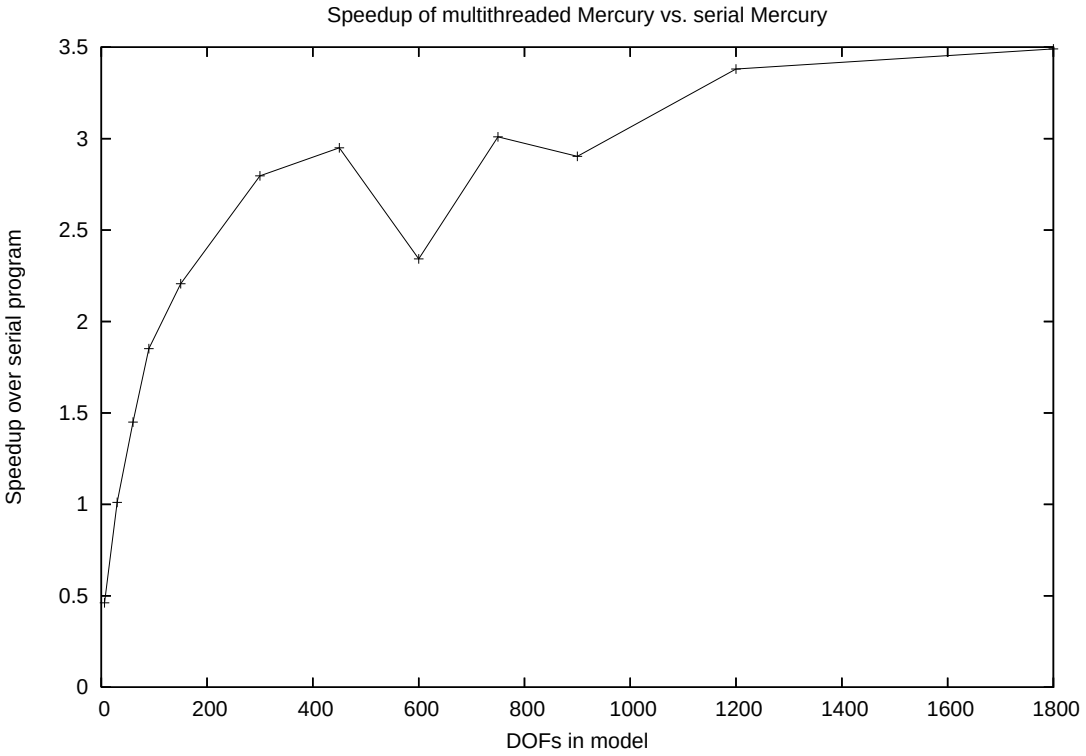


Fig. 9.1 Speedup of multithreaded Mercury compared to non-parallel Mercury

9.3.1 Parallel Speedup Using a Test Model

To quantify the speedup obtained with a shared-memory approach, a model was constructed for Mercury representing a multi-story building composed of flexibility-based non-linear beam-column elements. The number of stories in the building can be varied to produce a model with the desired number of degrees-of-freedom (DOFs). The point is to examine the difference that model size has on the effective speedup gained by multithreading. The beam-column elements contain various nonlinear materials, including both hardening and concrete damage material models, which combined with the flexibility-based formulation can require a large of computation per element during the analysis.

As a result of the multithreading changes, Mercury computation speed can be increased by up to a factor of 3.5 running on a local shared-memory processor (SMP) computer containing four processors, as shown in figure 9.1. It can be observed that larger problems produce more speedup; with smaller problem sizes the overhead of thread synchronization becomes a factor, but with larger problems this overhead becomes less significant. Future work involves tuning the threading granularity to reach a theoretical 4× speedup for the four processors in the SMP computer.

9.4 Parallel computation with MPI

Mercury generally uses an implicit integration scheme based on a variation of the Newton-Raphson method. From Fig 9.2, we note that there are two major CPU intensive operations:

1. Determination of inverse matrix, ① and ③ in Fig. 9.2

2. Element state determination for force recovery, ② in Fig. 9.2

In our current implementation, the stiffness matrix (always symmetric and positive definite) is only decomposed once in case of the Shing method in Fig. 9.3 though each step may require the updated stiffness matrix for accuracy. To improve the accuracy for our current implementation, we need to update a stiffness matrix which is made during structural determination as ⑤ in Fig. 9.3. It may effect the computation cost. Hence, we use MPI to determine the inverse matrix of a updated stiffness matrix for reducing computational time and improving accuracy.

9.4.1 Message Passing Interface(MPI)

Parallel computation is accomplished by dividing a computation into parts and making use of multiple processors, each executing on a separate processor, to carry out these parts. Sometimes an ordinary or sequential program can be used by all the processor, but with distinct input files or parameters. In such a situation, no communication occurs among the separate tasks. When the power of a parallel computation is needed to attack a large problem with a more complex structure, however, such communication is necessary.

One of the most straightforward approaches to communication is to have the processors coordinate their activities by sending and receiving messages, much as a group of people might cooperate to perform a complex task. This approach to achieving parallelism is called message passing. MPI is a message-passing library specification.

9.4.2 Determination of inverse matrix with MPI

For the transient analysis, Mercury can uses initial stiffness method, modified Newton-Raphson method, Newton-Raphson method, and Shing method which modified initial stiffness method for transient analysis as implicit integration scheme. However, implementation methods except for Shing method and initial stiffness method require the inverse matrix of a updated stiffness matrix for next iteration or step. It increases computational time. While, Shing method and initial stiffness method have a disadvantage. As we only consider Shing method, the disadvantage of our approach is that as the analysis proceeds, the errors introduced by the initial stiffness matrix, and the fixed number of iterations (10) increase (due to nonlinearity), Fig. 9.3. Great improvements could be achieved if we could update the stiffness matrix, however this would interfere with the determinism imposed by our environment (real time operation).

Hence, modified Shing method includes the implementation of the algorithm shown in Fig. 9.4 (c) where we always assign to one processors the task of performing "background" decomposition of the updated stiffness matrix, Fig. 9.6. When completed, this matrix is swapped with the active one, Fig. 9.5 and 9.7. It should be noted that there always is a "time lagg"' between the active inverse stiffness matrix (3 in the figure), and the current time increment.

9.4.3 Element state determination with MPI

Here, we modified Mercury to operate in a distributed memory computational environment, and initially focus on the parallelization of element state determination for the force recovery portion of the code. Again there are two possible implementations:

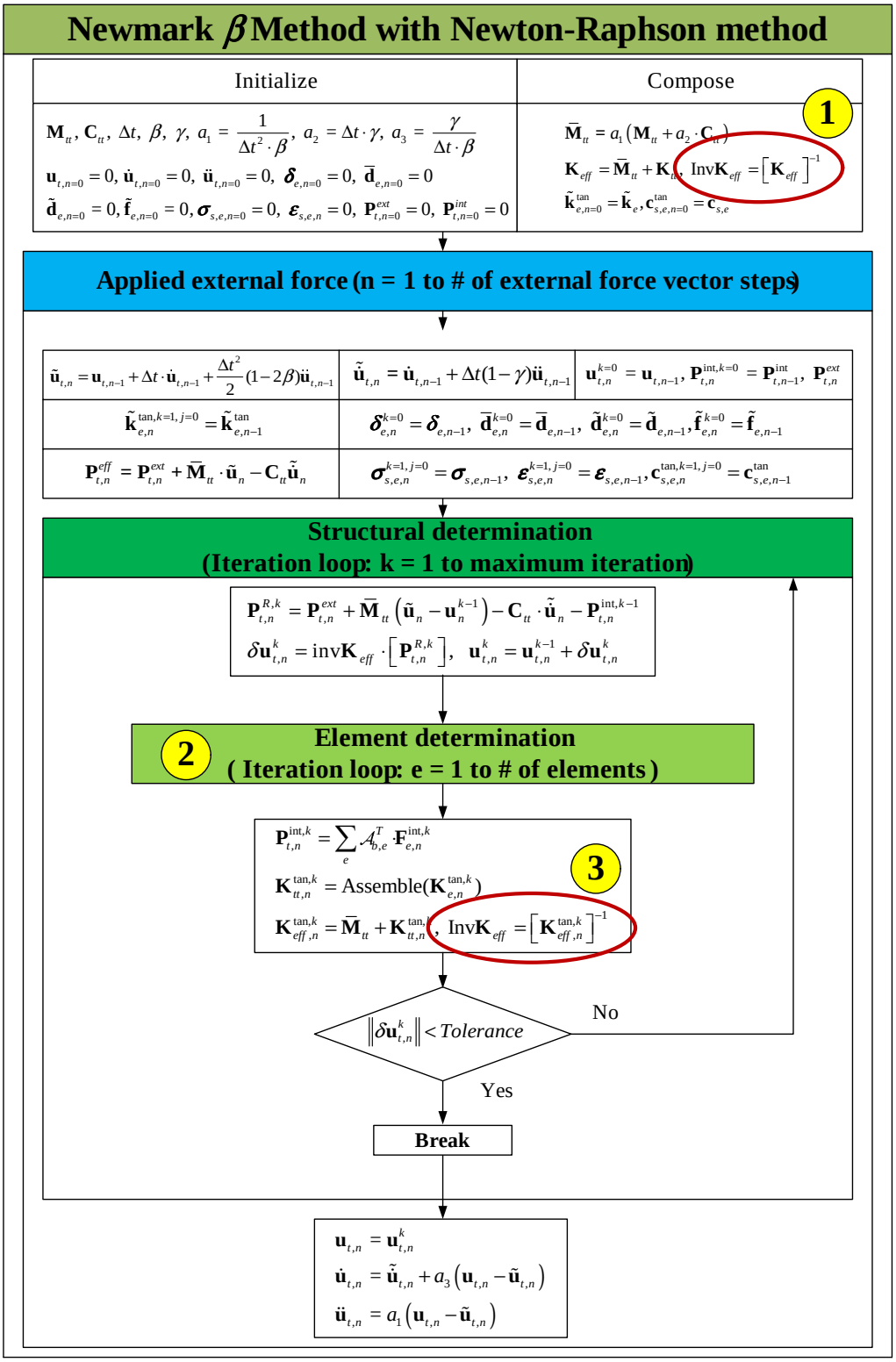


Fig. 9.2 Flow chart of nonlinear analysis with truss element using the Newton-Raphson method

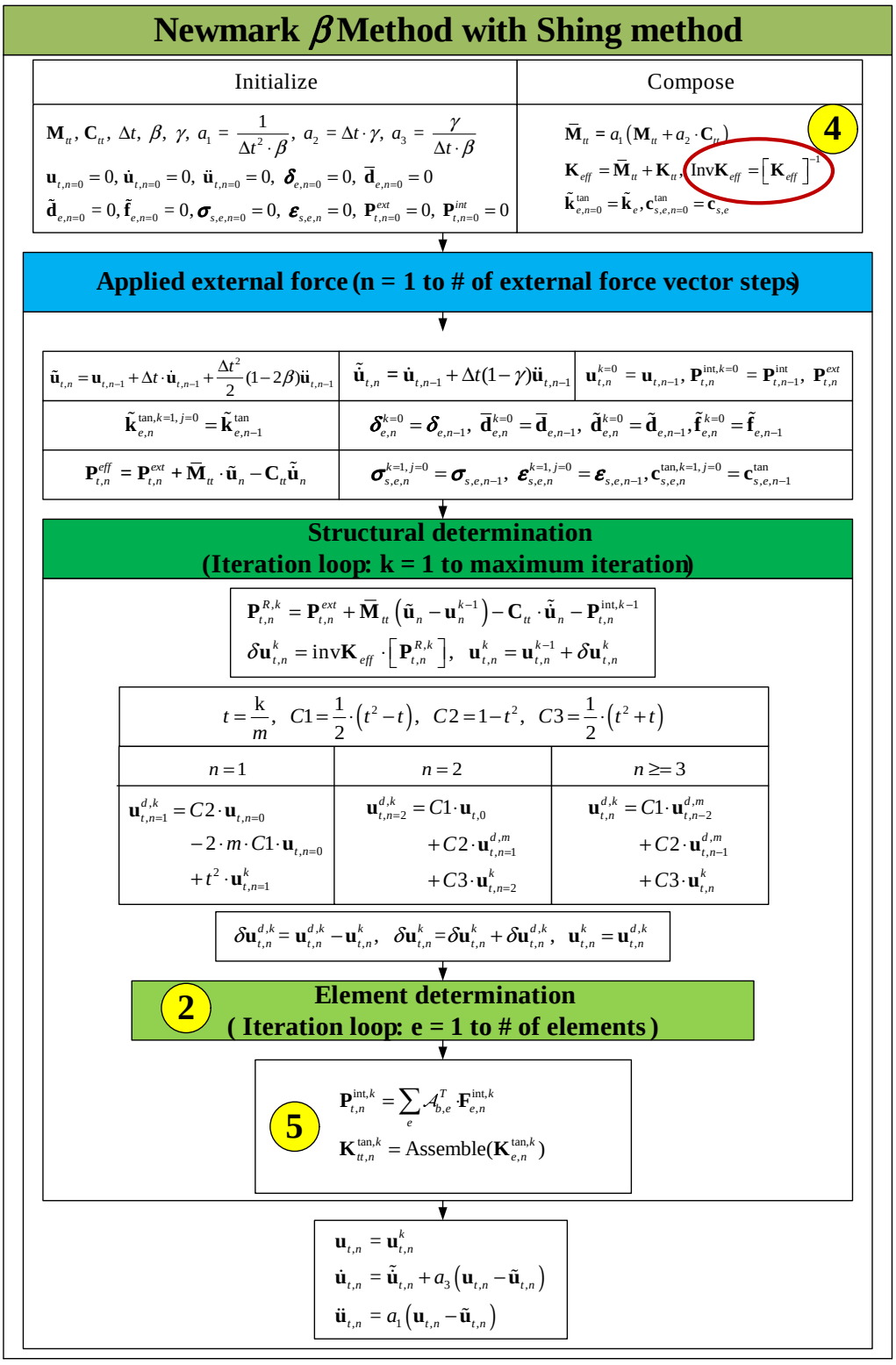


Fig. 9.3 Flow chart of nonlinear analysis with truss element using the Shing method

n	1				2				3			
$\mathbf{K}_0^{-1}$	$\mathbf{u}_1^1$	$\mathbf{u}_1^2$	$\cdots$	$\mathbf{u}_1^m$	$\mathbf{u}_2^1$	$\mathbf{u}_2^2$	$\cdots$	$\mathbf{u}_2^m$	$\mathbf{u}_3^1$	$\mathbf{u}_3^2$	$\cdots$	$\mathbf{u}_3^m$

(a) Initial Stiffness; One processor

n	1				2				3			
$\mathbf{K}_1^{-1}$	$\mathbf{u}_1^1$	$\mathbf{u}_1^2$	$\cdots$	$\mathbf{u}_1^m$	$\mathbf{K}_2^{-1}$	$\mathbf{u}_2^1$	$\mathbf{u}_2^2$	$\cdots$	$\mathbf{u}_2^m$	$\mathbf{K}_3^{-1}$	$\mathbf{u}_3^1$	$\mathbf{u}_3^2$	$\cdots$	$\mathbf{u}_3^m$

(b) Modified Newton-Raphson Stiffness; One processor

n	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
CPU 1	$\mathbf{K}_0^{-1}$				$\mathbf{K}_0^{-1}$				$\mathbf{K}_4^{-1}$			$\mathbf{K}_7^{-1}$			$\mathbf{K}_{10}^{-1}$	
SCRAMNet Card																
CPU 2					$\mathbf{K}_4^{-1}$				$\mathbf{K}_7^{-1}$			$\mathbf{K}_{10}^{-1}$			$\mathbf{K}_{13}^{-1}$	

n	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
CPU 1	$\mathbf{K}_{13}^{-1}$			$\mathbf{K}_{16}^{-1}$			$\mathbf{K}_{19}^{-1}$			$\mathbf{K}_{22}^{-1}$			$\mathbf{K}_{25}^{-1}$		
SCRAMNet Card															
CPU 2	$\mathbf{K}_{16}^{-1}$			$\mathbf{K}_{19}^{-1}$			$\mathbf{K}_{22}^{-1}$			$\mathbf{K}_{25}^{-1}$			$\mathbf{K}_{28}^{-1}$		

(c) Modified Shing; Two processors

Fig. 9.4 Modified Shing method using MPI

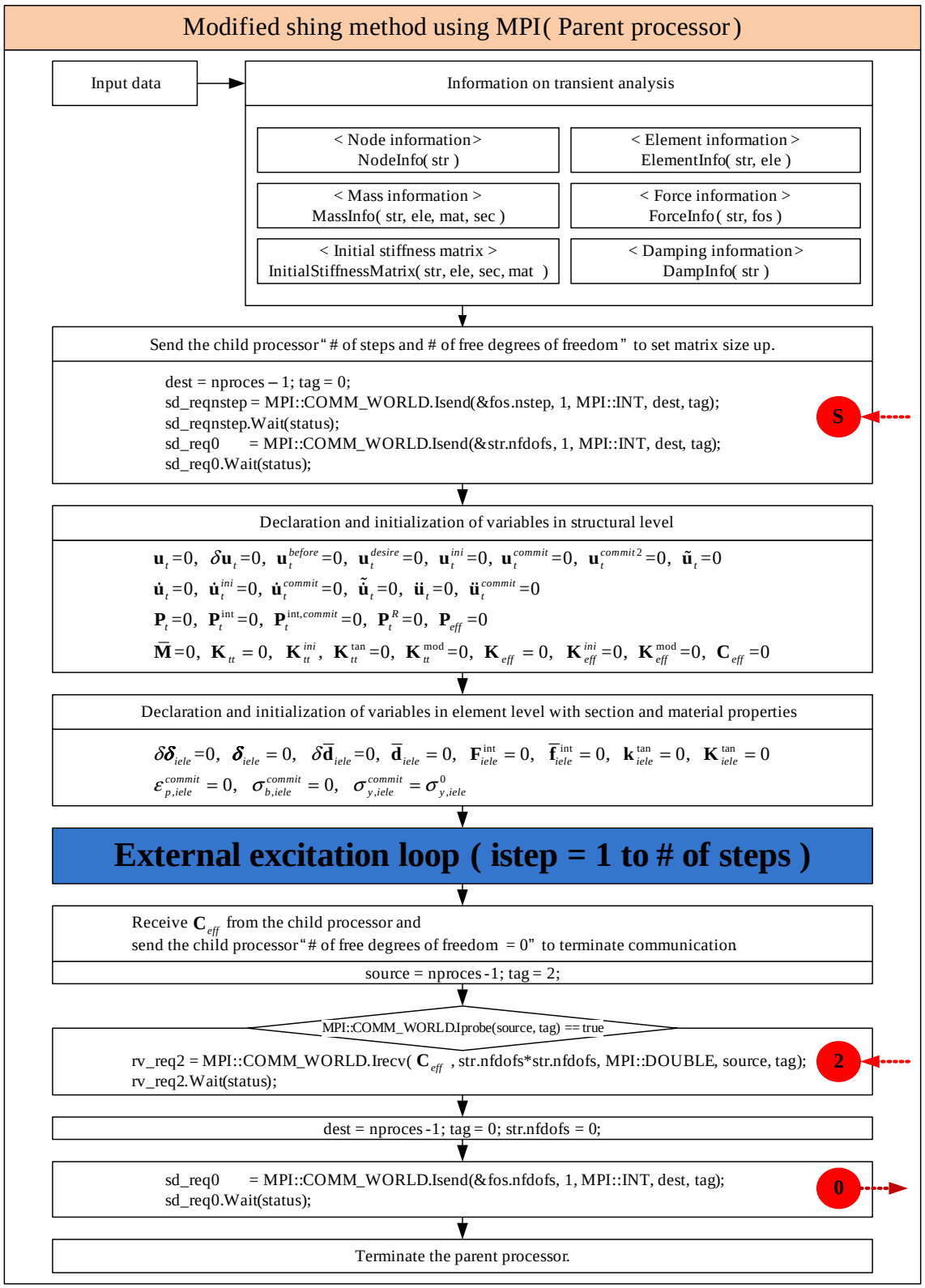


Fig. 9.5 Master processor for implementation

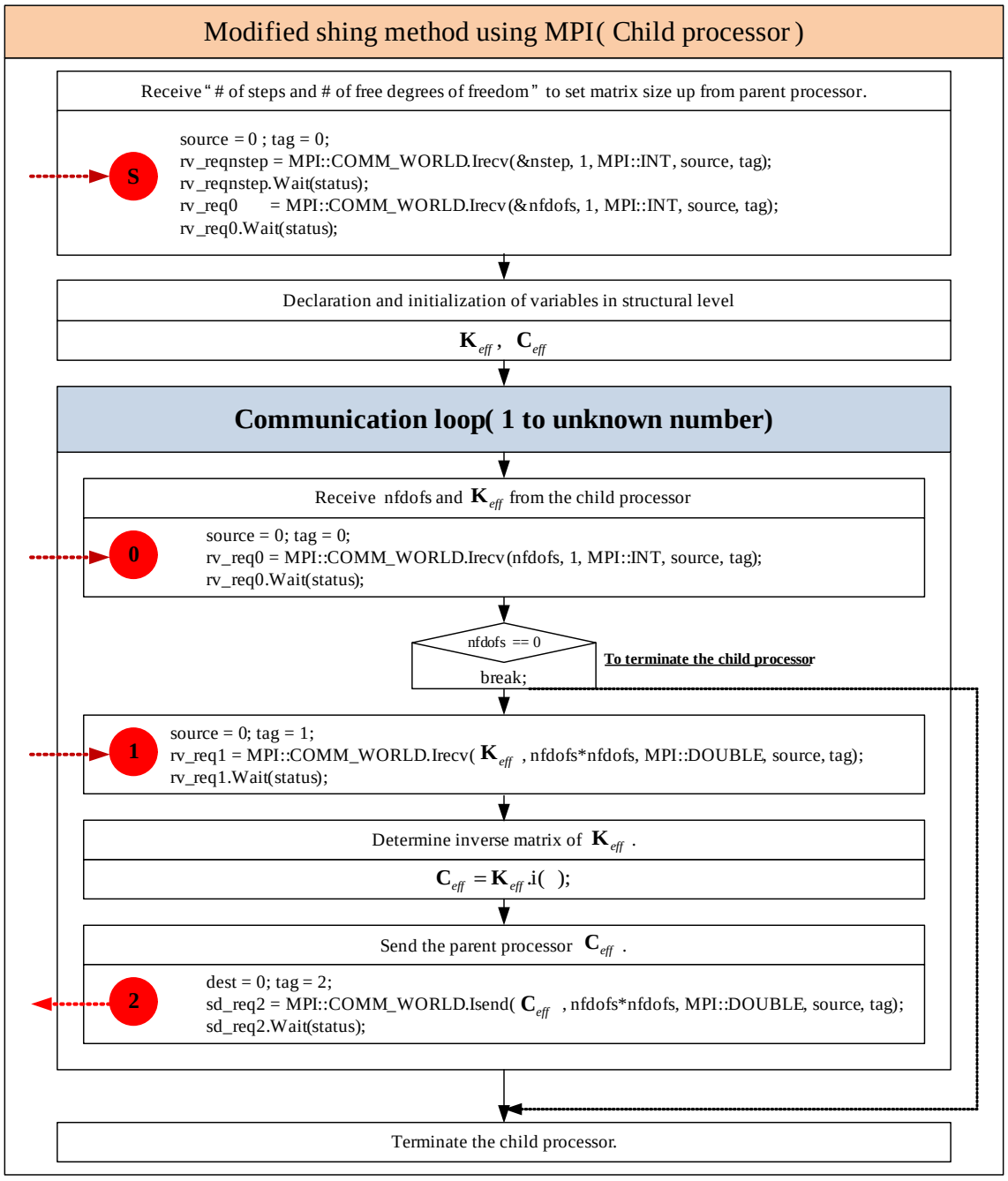


Fig. 9.6 Slave processor for determination of the inverse matrix of a updated stiffness matrix

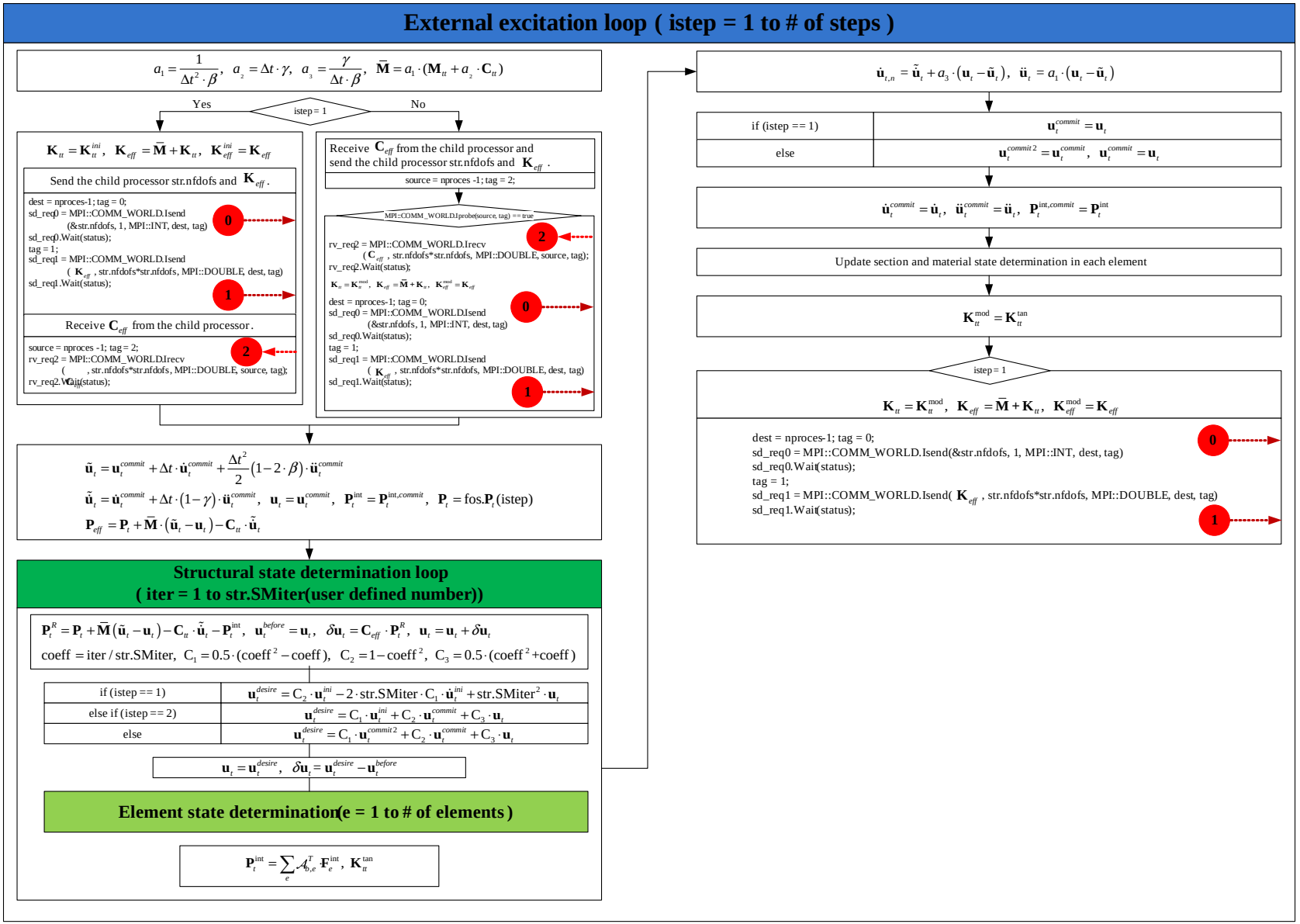


Fig. 9.7 External excitation loop in master processor

1. Perform a domain decomposition of the finite element mesh, and assign to each CPU the task of running that substructure.

2. Designate one master node which will run Mercury, and then assign a number of slave nodes to perform the force recovery for the elements.

Second approach should be implemented with MPI. Modified Mercury for element state determination has number of elements from 53 to 551 by 56, and uses number of processors from 1 to 7. To compare speed-up depending on number of elements and processors, speed-up is determined (Lazou 1988):

$$S = \frac{\text{Execution time for uniprocessor}}{\text{Execution time for } P \text{ processors}} \tag{9.1}$$

where, S is speed-up and P is number of processors.

Fig. 9.8 and 9.9 describe average elapsed time for element state determination and speed-up depending on number of processors.

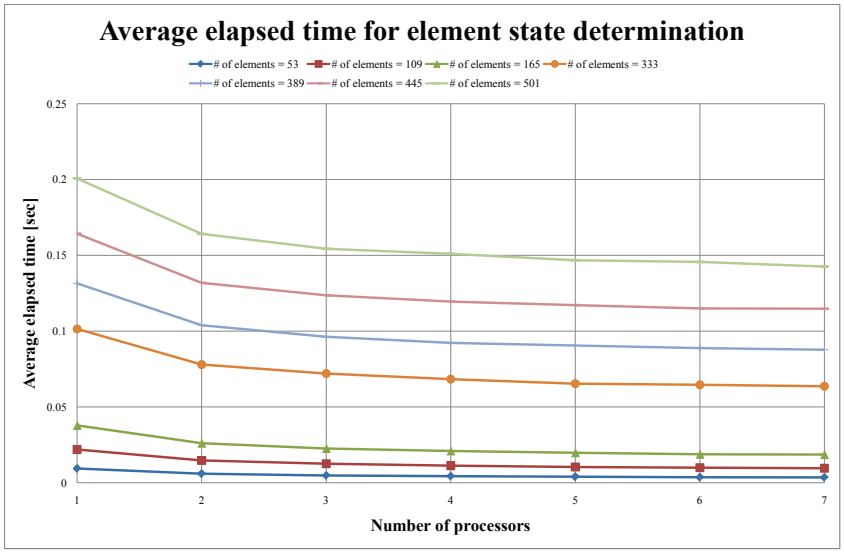


Fig. 9.8 Average elapsed time for element state determination depending on number of processors

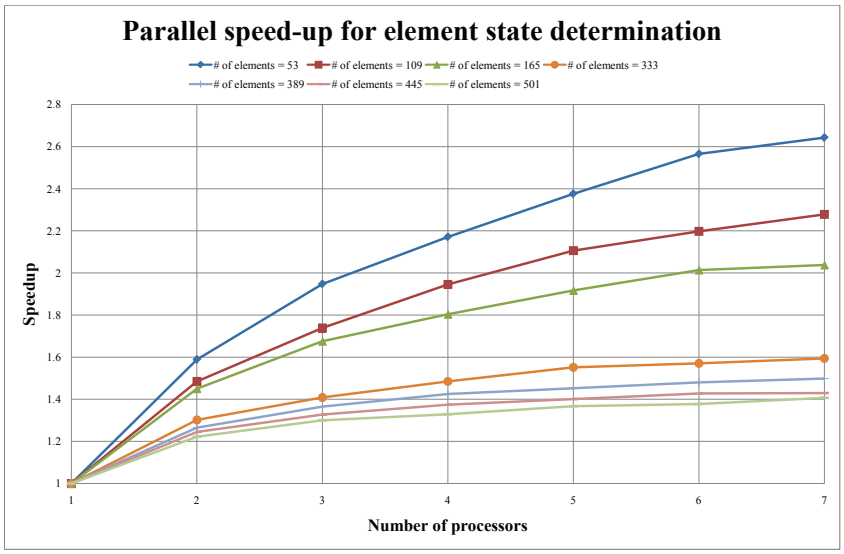


Fig. 9.9 Speed-up depending on number of processors

Appendix A

MATLAB USER's MANUAL

Mercury's input data for the Matlab and the c++ version are nearly identical. This document describes the input for Matlab. Since the c++ version uses the Lua scripting language (analogous to TCL in OpenSees), a simple program (to be embedded in the c++ version) will translate the Matlab format into Lua scripts.

Note: In this preliminary version of Mercury, no attempt has been made to simplify (generate/automate) data entry, and there is not (yet) a mesh generator for the program. Those are simple future developments.

A.1 Preface Block

This initial block declares units, spatial dimension of the structure, and the number of degrees of freedom per node.

A.1.1 Unit

The `Unit` declares selected units for analysis.

`Unit` = ['`F`, `L`']

`F`: force, and `L`: length units. For example **Unit = ['kN, mm']**

A.1.2 Structural mode

The `StrMode` declares dimension of structure and number of degrees of freedom per node.

`StrMode` = [`ndim`, `ndofpn`]

---

where `ndim` refers to the spatial dimension of the structure [2|3] and `ndofpn` to the number of degrees of freedom per node in global reference [2|3|6]

A.2 Control Block

The control block defines basic information about the structural analysis.

A.2.1 Analysis

The `Analysis` defines the analysis mode.

`Analysis` = '`AnalysisMode`'

---

where `AnalysisMode`: [Static—Transient].

A.2.2 Iteration

The `Iteration` defines the iterative method adopted in the solution of the nonlinear system of equations.

`Iteration` = '`IterationMode`'

---

where `IterationMode` can be

- `Linear`: Use only one single iteration to solve the nonlinear system of equations in a step.
- `NewtonRaphson`: Use the Newton-Raphson method in which the tangent stiffness matrix is updated at each iteration.
- `ModifiedNewtonRaphson`: Use the modified Newton-Raphson method in which the tangent stiffness is updated at each step.
- `InitialStiffness`: Use the initial stiffness matrix and the the tangent stiffness matrix is never updated.
- `ModifiedInitialStiffness`: Use the initial stiffness matrix as modified by Shing for hybrid simulation. This is limited to transient analysis.

A.2.3 Iteration for element

The `IterationEle` defines the iterative method adopted in the solution of the nonlinear system of equations in element level. In `Mercury`, this command is only used in `FlexibilityBased2DBeamColumn`.

`IterationEle` = '`IterationEleMode`'

---

where `IterationEleMode` can be

- `NewtonRaphson`: Use the Newton-Raphson method in which the tangent stiffness matrix is updated at each iteration.
- `InitialStiffness`: Use the initial stiffness matrix and the the tangent stiffness matrix is never updated.

A.2.4 Integration

The `Integration` defines the types of numerical integration used in the transient analysis.

- Newmark β method
`Integration` = {'`Newmark`', {α_m, `a`, `b`, β, γ}}
- Hilber-Hughes-Taylor method(HHT method)
`Integration` = {'`HHT`', {α_m, `a`, `b`, α}}

- `IntegrationMode`
  - Newmark: Use the Newmark β method for transient analysis with ground acceleration.
  - HHT: Use the Hilber-Hughes-Taylor method for transient analysis with ground acceleration. This method is often referred to as 'α' method.
- α_m: Coefficient premultiplying the rotational mass in a beam column. Recall that $[\mathbf{m}] = \frac{m}{2}\lfloor\ 1 \quad \alpha_m L^2/210 \quad 1 \quad \alpha_m L^2/210\ \rfloor$, (Cook et al. 2002).
- `a`: Rayleigh damping coefficient of mass matrix
- `b`: Rayleigh damping coefficient of stiffness matrix
- β: Coefficient β in the Newmark β method
- γ: Coefficient γ in the Newmark β method
- α: Coefficient α in the HHT method

For Example: **Integration = {'NewtonRaphson', {1/78, 0.02, 0.08, 1/6, 1/2 }}** for linear acceleration in the Newmark β method,

A.2.5 MassInput

Lumped masses are automatically determined by Mercury. If additional masses are to be assigned, then the `MassInput` command must be used and a mass specified for each node.

`MassInput`	= {	`nodtag`^{1}, m_1^1, m_2^1 [m_3^1, m_4^1, m_5^1, m_6^1]	;
		...	;
		`nodtag`^{i}, m_1^i, m_2^i [m_3^i, m_4^i, m_5^i, m_6^i]	;
		...	;
		`nodtag`^{n}, m_1^n, m_2^n [m_3^n, m_4^n, m_5^n, m_6^n]	}

MassInput is based on lumped mass. Where n is smaller than or eqaul to the total number of nodes. At each component m_j^i, j means the j^{th} degree of freedom of the i^{th} nodtag

A.2.6 Convergence Criteria

`StrMiter`	= StrMax
`EleMiter`	= EleMax
`Convergence`	= Norm
`ConvergenceEle`	= NormEle
`Tolerance`	= Tol

Where

- `StrMax`: Maximum number of iterations
- `EleMax`: Maximum number of iterations within an element when using flexibility-based 2D beam-column element with internal iteration
- `Norm`: User can select norm criterion for structural level. Mercury support three types, 'DispNorm', 'ForceNorm' and 'EnergyNorm' criterion.
- `NormEle`: User can select norm criterion for element level when using flexibility-based 2D beam-column element. Mercury support three types, 'DispNorm', 'ForceNorm' and 'EnergyNorm' criterion.

- `Tol`: Convergence criteria on the residuals

Example:
`FlexibilityBased2DBeamColumn` in Sec. A.4, then we could have: `StrMiter = 20`; `EleMiter = 50`; `Convergence = 'ForceNorm'`; `ConvergenceEle = 'EnergyNorm'`; `Tolerance = 1.0e-8`

A.3 Geometry Block

The geometry block defines nodal coordinates and their constraints assuming a right handed coordinate system.

A.3.1 Nodal coordinates

The `nodcoord` assigns coordinates of nodes.

`nodcoord`	= {	$\texttt{nodtag}_1,\ x_1,\ y_1\ [z_1]$	;
		...	;
		$\texttt{nodtag}_i,\ x_i,\ y_i\ [z_i]$	;
		...	;
		$\texttt{nodtag}_n,\ x_n,\ y_n\ [z_n]$	}

for example:

```
Node  =  {  1,   0.0,   0.0   ;
            2,   1.0,   3.0   ;
            3,   2.0,   0.0   }
```

A.3.2 Boundary condition

The `constraint` command assigns boundary conditions to the nodes. Each node has to have as many constraint as d.o.f's per node.

`constraint`	= {	$\texttt{nodtag}^1,\ id_1^1,\ id_2^1\ [\ id_3^1,\ id_4^1,\ id_5^1,\ id_6^1]$	;
		...	;
		$\texttt{nodtag}^i,\ id_1^i,\ id_2^i\ [\ id_3^i,\ id_4^i,\ id_5^i,\ id_6^i]$	;
		...	;
		$\texttt{nodtag}^n,\ id_1^n,\ id_2^n\ [\ id_3^n,\ id_4^n,\ id_5^n,\ id_6^n]$	}

Where 0 corresponds to a free dof, and 1 to a fixed one. For example:

```
constraint  =  {  3,   1,   1   ;
                  5,   1,   0   }
```

A.4 Element Block

The `elements` command defines element type, nodal connectivity, and basic sectional information. These may vary with the element type.

elements	= {	$\texttt{eletag}_1$, $\texttt{eletype}_1$, $\texttt{in}_1$, $\texttt{jn}_1$, { $\texttt{SecInfo}_1$ }	;
		...	;
		$\texttt{eletag}_i$, $\texttt{eletype}_i$, $\texttt{in}_i$, $\texttt{jn}_i$, { $\texttt{SecInfo}_i$ }	;
		...	;
		$\texttt{eletag}_n$, $\texttt{eletype}_n$, $\texttt{in}_n$, $\texttt{jn}_n$, { $\texttt{SecInfo}_n$ }	}

Where

- $\texttt{eletag}_i$: Sequential integer identifying the i^{th} element
- $\texttt{eletype}_i$: i^{th} element type (see below)
- $\texttt{in}_i$: First node
- $\texttt{jn}_i$: Second node
- $\texttt{SecInfo}_i$: Basic section information for the element (see below)

A.4.1 Truss element (Sec. 2.1)

This is the classical two noded axial element, however its cross section can be characterized by either a constant properties; general, layered or fiber.

```
eletag, 'Simple2DTruss', in, jn, { sectag }
```

Where:

- eletag: Element tag
- in: First node of element eletag
- jn: Second node of element eletag
- sectag: Integer number identifying section of the truss element eletag

A.4.2 Stiffness-based 2D beam-column element(Sec. 2.2.1)

Stiffness-based 2D beam-column element can have a constant, layered, or fiber section. Its numerical integration is based on Gauss-Legendre quadrature rule.

```
eletag, 'StiffnessBased2DBeamColumn', in, jn, { sectag, nIp }
```

Where:

- eletag: Element tag
- in: First node of element eletag
- jn: Second node of element eletag
- sectag: Integer number identifying section of element eletag
- nIp: Order of integration of element eletag

A.4.3 Flexibility-based 2D beam-column element (Sec. 2.2.2)

Flexibility-based 2D beam-column element can have a constant, layered or fiber section. Its numerical integration is based on Gauss-Lobatto quadrature rule.

- Flexibility-based 2D beam-column with element iteration loop
`eletag, 'FlexibilityBased2DBeamColumn', in, jn, { sectag, nIp }`
- Flexibility-based 2D beam-column without element iteration loop
`eletag, 'FlexibilityBased2DBeamColumnNoIter', in, jn, { sectag, nIp }`

Where

- `eletag`: Element tag
- `in`: First node of element `eletag`
- `jn`: Second node of element `eletag`
- `sectag`: Integer number identifying section of element `eletag`
- `nIp`: Order of integration of element `eletag`

A.4.4 Zero-length 2D element (Sec. 2.4)

Zero length element is used to model lumped plasticity. It can account for stiffness degradation in flexure and shear. It neglects axial-flexural coupling effect and depends on force and deformation history as well as on the section characteristics.

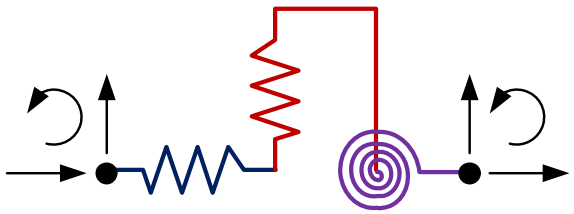


Fig. A.1 Zero-length 2D element

`eletag`, 'ZeroLength2D', `in`, `jn`, { `ndofpnlocal`, $mattag_1$, $mattag_2$, $mattag_3$, *angle* }

Where:

- `eletag`: Element tag
- `in`: First node of element `eletag`
- `jn`: Second node of element `eletag`
- `ndofpnlocal`: Integer number of degrees of freedom in local reference of element `eletag`
- $mattag_i$: Integer number identifying material of *ith* d.o.f in local reference for element `eletag`
- *angle*: Angle (in radians) between local reference and global reference of element `eletag`

A.4.5 Zero-length 2D section element (Sec. 2.5)

This is the counterpart of the zero length element for layered/fiber sections. It is particularly recommended if the center of rotation in zero-length element changes with axial force and moment.

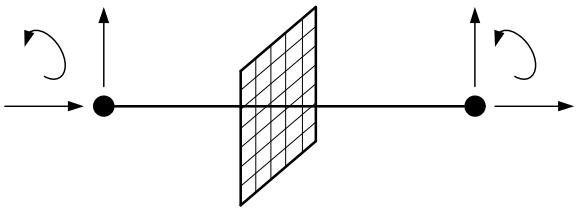


Fig. A.2 Zero-length 2D section element

`eletag`, 'ZeroLength2DSection', `in`, `jn`, { `sectag`, *angle* }

Where:

- `eletag`: Element tag
- `in`: First node of element `eletag`
- `jn`: Second node of element `eletag`
- `sectag`: Integer number identifying section of element `eletag`
- *angle*: Angle (in radians) between local reference and global reference of element `eletag`
  - Unit is radian.

For example, if the structure has `StiffnessBased2DBeamColumn`, `FlexibilityBased2DBeamColumn`, and `ZeroLength2DSection` elements,

```
elements  = {  1, 'ZeroLength2DSection', 1, 2, {1, 0}              ;
           = {  2, 'StiffnessBased2DBeamColumn', 2, 3, {1, 3}     ;
           = {  3, 'FlexibilityBased2DBeamColumn', 3, 4, {2, 5}   ;
           = {  4, 'ZeroLength2DSection', 4, 5, {2, 0}            }
```

A.5 Section Block

Section block declares section properties. `sections` defines section types, section properties, and basic material information of section. Description on sections may be different depending on types of section.

```
sections  = {  sectag_1,  sectype_1,  { SecProp_1 }   ;
               ...                                    ;
               sectag_i,  sectype_i,  { SecProp_i }   ;
               ...                                    ;
               sectag_n,  sectype_n,  { SecProp_n }   }
```

Where:

- `sectag`$_i$: Sequential integer number identifying section at i^{th} section
- `sectype`$_i$: Section type at i^{th} section
- `SecProp`$_i$: Section properties and basic material information on i^{th} section

A.5.1 General section

General section has only one layer or fiber, and it in `Mercury` only supports elastic material currently. Usually, if section has nonlinear material, user may use multi-layer or multi-fiber section.

```
sectag, 'General', { mattag, A, Ix, Iy, Iz }
```

Where:

- `sectag`: Section tag
- `mattag`: Integer number identifying material with `sectag`
- `A`: Section area with `sectag`
- `Ixx`: Moment inertia on x-axis with `sectag`
- `Iyy`: Moment inertia on y-axis with `sectag`
- `Izz`: Moment inertia on z-axis with `sectag`

A.5.2 Layer section (Sec. 2.3)

All elements in Mercury can be layered.

```
sectag, 'Layer',  {  mattag_1,  A_1,  y-distance_1   ;
                                 ...                 ;
                     mattag_i,  A_i,  y-distance_i   ;
                                 ...                 ;
                     mattag_n,  A_n,  y-distance_n   }
```

Where:

- `sectag`: Section tag
- `mattag`$_i$: Integer number identifying material of i^{th} layer in section `sectag`
- `A`$_i$: i^{th} layer section area in section `sectag`

- `y-distance`$_i$: i^{th} layer distance from neutral axis to centroid of i^{th} layer along y-axis in section with `sectag`

A.5.3 Fiber section (Sec. 2.3)

All elements in Mercury can have fiber sections.

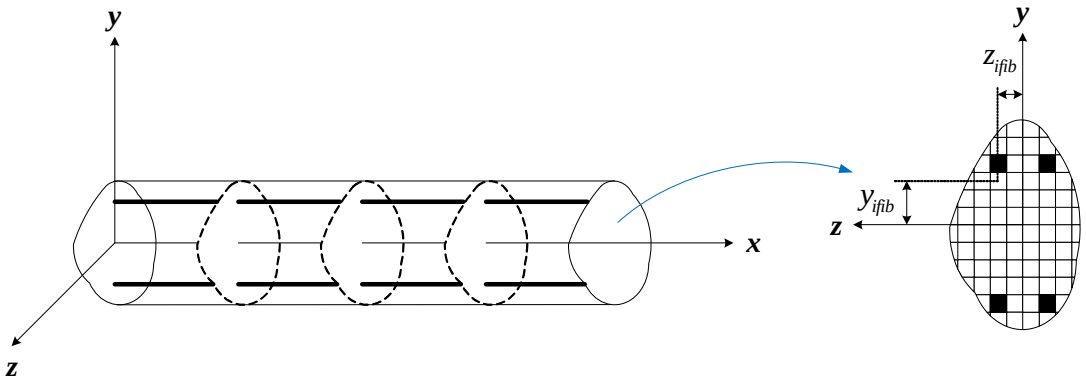


Fig. A.3 Fiber sections

`sectag, 'Fiber',`	{	`mattag`$_1$,	`A`$_1$,	`y-distance`$_1$,	`z-distance`$_1$;
			...		;
		`mattag`$_i$,	`A`$_i$,	`y-distance`$_i$,	`z-distance`$_i$;
			...		;
		`mattag`$_n$,	`A`$_n$,	`y-distance`$_n$,	`z-distance`$_n$ }

Where:

- `sectag`: Section tag
- `mattag`$_i$: Integer number identifying material of i^{th} fiber in section `sectag`
- `A`$_i$: i^{th} fiber section area in section `sectag`
- `y-distance`$_i$: i^{th} fiber distance from section centroid to centroid of i^{th} fiber along y-axis in section `sectag`
- `z-distance`$_i$: i^{th} fiber distance from neutral axis to centroid of i^{th} fiber along z-axis in section `sectag`

For example, if a structure has `General` and two `Layer` sections,

```
section= {1,'General', {1, 100, 0, 0, 833.33\}
         2, 'Layer',  2, 1, 0.35 ;
                      3, 1, 0.25 ;
                      2, 1, 0.15 ;
                      2, 1, 0.05 ;
                      2, 1, -0.05 ;
                      2, 1, -0.15 ;
                      3, 1, -0.25 ;
                      2, 1, -0.25 ;
```

```
3, 'Layer',   3, 1, 0.35 ;
              2, 1, 0.25 ;
              3, 1, 0.15 ;
              3, 1, 0.05 ;
              3, 1, -0.05 ;
              3, 1, -0.15 ;
              2, 1, -0.25 ;
              3, 1, -0.25}
```

A.6 Material Block

Material block declares material properties.

materials	= {	$\texttt{mattag}_1$, $\texttt{mattype}_1$, $\texttt{modulus}_1$, $\texttt{density}_1$, { $\texttt{MatProp}_1$ }	;
		...	;
		$\texttt{mattag}_i$, $\texttt{mattype}_i$, $\texttt{modulus}_i$, $\texttt{density}_i$, { $\texttt{MatProp}_i$ }	;
		...	;
		$\texttt{mattag}_n$, $\texttt{mattype}_n$, $\texttt{modulus}_n$, $\texttt{density}_n$, { $\texttt{MatProp}_n$ }	}

where:

- $\texttt{mattag}_i$: Consecutive integer number identifying material at i^{th} material
- $\texttt{mattype}_i$: Material type at i^{th} material
- $\texttt{modulus}_i$: Material modulus at i^{th} material
- $\texttt{density}_i$: Density at i^{th} material
- $\texttt{MatProp}_i$: Material properties at i^{th} material

A.6.1 Elastic material

mattag, 'Elastic', modulus, density, { G }

Where:

- mattag: Material tag
- modulus: Young's modulus of a material with mattag
- density: Density of a material with mattag
- G: Shear modulus of a material with mattag

A.6.2 Hardening material (Sec. 3.1.1)

mattag, 'Hardening', modulus, density, { sigmaY0, Hiso, Hkin }

Where:

- mattag: Material tag
- modulus: Young's modulus of a material with mattag
- density: Density of a material with mattag

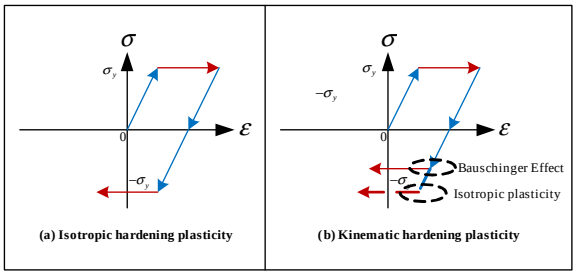


Fig. A.4 Zero-length 2D element

- `sigmaY0`: Initial yield stress of a material with `mattag`
- `Hiso`: Isotropic hardening modulus of a material with `mattag`
- `Hkin`: Kinematic hardening modulus of a material with `mattag`

A.6.3 Bilinear material with isotropic hardening (Sec. 3.1.2)

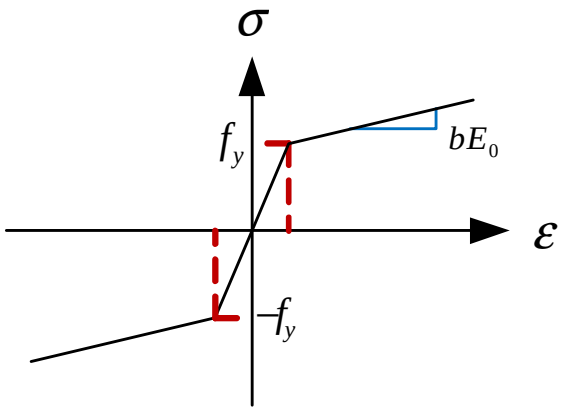


Fig. A.5 Bilinear material with isotropic hardening

`mattag`, '`Bilinear`', `modulus`, `density`, { `SigmaY0`, `b`, `a1`, `a2`, `a3`, `a4` }

Where

- `mattag`: Material tag
- `modulus`: Young's modulus of a material with `mattag`
- `density`: Density of a material with `mattag`
- `sigmaY0`: Initial yield stress of a material with `mattag`
- `b`: Strain-hardening ratio between post-yield tangent and Young's modulus of a material with `mattag`
- `a1`: Isotropic hardening coefficient 1 of a material with `mattag` - increase of compression yield envelope as proportion of initial yield stress after a plastic strain of `a2` × (`SigmaY0`/`modulus`)
- `a2`: Isotropic hardening coefficient 2 of a material with `mattag`

- a3: Isotropic hardening coefficient 3 of a material with mattag - increase of tension yield envelope as proportion of initial yield stress after a plastic strain of a4 × (SigmaY0/modulus)
- a4: Isotropic hardening coefficient 4 of a material with mattag

A.6.4 Modified Giuffre-Menegotto-Pinto material with isotropic hardening (Sec. 3.1.3)

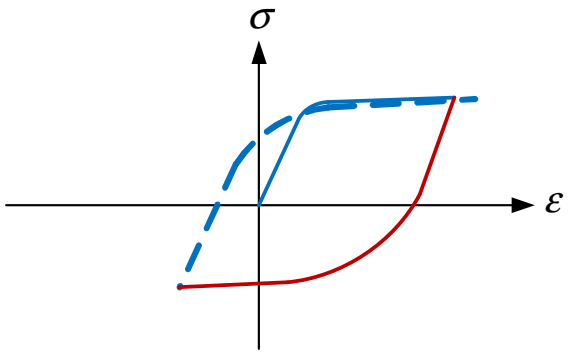


Fig. A.6 Modified Giuffre-Menegotto-Pinto material with isotropic hardening

mattag, 'GiuffreMenegottoPinto', modulus, density, { SigmaY0, b, R0, n, cR1, cR2, a1, a2, a3, a4, σ_{init} }

Where:

- mattag: Material tag
- modulus: Young's modulus of a material with mattag
- density: Density of a material with mattag
- sigmaY0: Initial yield stress of a material with mattag
- b: Strain-hardening ratio between post-yield tangent and Young's modulus of a material with mattag
- R0: Coefficient 0 of a material with mattag to control the transition from elastic to plastic branches - value between 10 and 20 is recommended
- n: R = R0 − ($R0^n$) ∗ cR1 * ξ / (cR2 + ξ) - n can have 0 and 1.
- cR1: Coefficient 1 of a material with mattag to control the transition from elastic to plastic branches - 0.925 is recommended
- cR2: Coefficient 1 of a material with mattag to control the transition from elastic to plastic branches - 0.15 is recommended
- a1: Isotropic hardening coefficient 1 of a material with mattag - increase of compression yield envelope as proportion of initial yield stress after a plastic strain of a2 × (SigmaY0/modulus)
- a2: Isotropic hardening coefficient 2 of a material with mattag
- a3: Isotropic hardening coefficient 3 of a material with mattag - increase of tension yield envelope as proportion of initial yield stress after a plastic strain of a4 × (SigmaY0/modulus)
- a4: Isotropic hardening coefficient 4 of a material with mattag
- σ_{init}: Initial stress of a material with mattag

A.6.5 Anisotropic damage 1D material (Sec. 3.2.2)

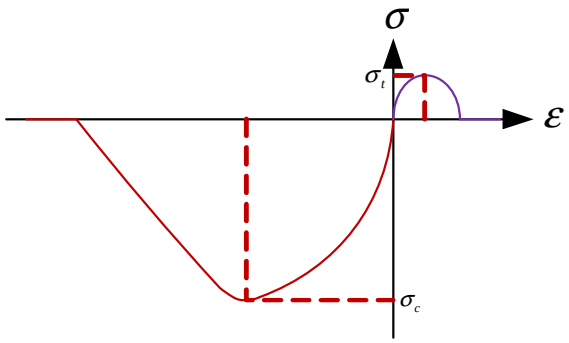


Fig. A.7 Concrete 1D anisotropic damage model

`mattag, 'AnisotropicDamage1D', modulus, density, { ν, κ0, Adamage, adamage, Dc }`

Where:

- `mattag`: Material tag
- `modulus`: Young's modulus of a material with `mattag`
- `density`: Density of a material with `mattag`
- ν: Poisson's ratio of a material with `mattag`
- κ_0: Initial elasticity threshold of a material with `mattag`
- `Adamage`: Damage coefficient A of a material with `mattag`
- `adamage`: Damage coefficient a of a material with `mattag`
- `Dc`: Damage limit of a material with `mattag`

A.6.6 Modified Kent and Park model (Sec. 3.2.1)

`mattag, 'ConcreteLinearTensionSoftening', modulus, density, { σc, εc, σcu, εcu, λ, σt }`

Where:

- `mattag`: Material tag
- `modulus`: Tension softening stiffness(absolute value) - slope of the linear tension softening branch of a material with `mattag`
- `density`: Density of a material with `mattag`
- σ_c: Compressive yield stress of a material with `mattag` - **Negative value**
- ε_c: Compressive yield strain of a material with `mattag` - **Negative value**
- σ_{cu}: Compressive crushing stress of a material with `mattag` - **Negative value**
- ε_{cu}: Compressive crushing strain of a material with `mattag` - **Negative value**
- λ: Ratio between unloading slope at ε_c and slope Young's modulus of a material with `mattag`
- σ_t: Tensile yield stress of a material with `mattag`

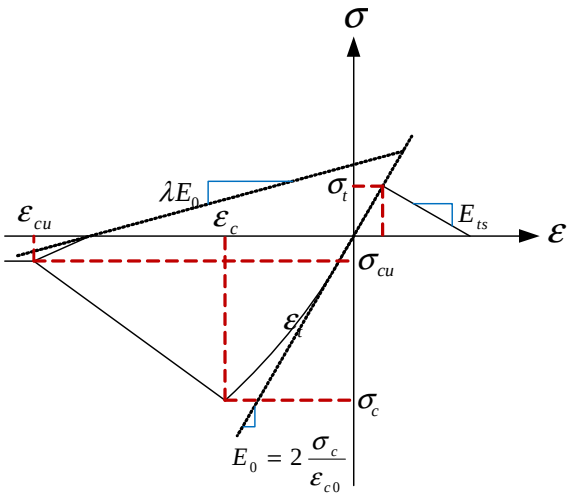


Fig. A.8 Concrete tension linear softening model

For example, if the structure for static analysis has `ConcreteLinearTensionSoftening` and `Bilinear` materials,

```
materials = { 1, 'ConcreteLinearTensionSoftening', 10, 0,  {-29, -0.00221, -5.8,
                                                            -0.02, 0.027719, 2.9}  ;
              2, 'Bilinear', 210, 0, {290, 0.01, 0.0, 1, 0, 1}                     }
```

A.7 Force Block

Force block declares all the external forces on the structure.

```
forces = { forcetype1, { ForceProp1 }  ;
           ...                         ;
           forcetypei, { ForcePropi }  ;
           ...                         ;
           forcetypen, { ForcePropn }  }
```

Where:

- `forcetype`$_i$: is type of force [`NodalForce| NodalDisplacement| ElementDistributedForce| VariableNodalForce|VariableNodalDisplacement| VariableElementDistributedForce| GreoundAcceleration| Variable|GroundAcceleration`] Force type of i^{th}
- `ForceProp`$_i$: Force properties of i^{th} `forcetype`

A.7.1 Nodal force

```
'NodalForce', { nodtag1, gx1, magnitude1  ;
                         ...              ;
                nodtagi, gxi, magnitudei  ;
                         ...              ;
                nodtagn, gxn, magnituden  }
```

Where

- $\texttt{nodtag}_i$: Node tag of i^{th} nodal force
- $\texttt{gx}_i$: Direction on global external force at node, [1—2—3] for X, or Y or Z global axis.
- $\texttt{magnitude}_i$: Magnitude of external force with node tag at i^{th} nodal force

A.7.2 Nodal displacement

'NodalDisplacement',	{	$\texttt{nodtag}_1,\ \texttt{gx}_1,\ \texttt{magnitude}_1$	;
		...	;
		$\texttt{nodtag}_i,\ \texttt{gx}_i,\ \texttt{magnitude}_i$	;
		...	;
		$\texttt{nodtag}_n,\ \texttt{gx}_n,\ \texttt{magnitude}_n$	}

Where:

- $\texttt{nodtag}_i$: Node tag of i^{th} nodal displacement
- $\texttt{gx}_i$: Direction on global reference of external displacement with node tag at i^{th} nodal displacement; [1—2—3] for X, or Y or Z global axis.
- $\texttt{magnitude}_i$: Magnitude of external displacement with node tag at i^{th} nodal displacement

A.7.3 Element distributed force

'ElementDistributedForce',	{	$\texttt{eletag}_1,\ \texttt{lx}_1,\ \texttt{magnitude}_1$	;
		...	;
		$\texttt{eletag}_i,\ \texttt{lx}_i,\ \texttt{magnitude}_i$	;
		...	;
		$\texttt{eletag}_n,\ \texttt{lx}_n,\ \texttt{magnitude}_n$	}

Where:

- $\texttt{eletag}_i$: Element tag at i^{th} element distributed force
- $\texttt{lx}_i$: Direction on local reference of external force with element tag at i^{th} element distributed force; [1—2—3] for X, or Y or Z global axis.
- $\texttt{magnitude}_i$: Magnitude of external force with element tag at i^{th} element distributed force

A.7.4 Variabel nodal force

'VariableNodalForce',	{	$\texttt{nodtag}_1,\ \texttt{gx}_1,\ \{\ \texttt{m}_1^1,\ \cdots,\ \texttt{m}_1^j,\ \cdots,\ \texttt{m}_1^m\ \}$	;
		...	;
		$\texttt{nodtag}_i,\ \texttt{gx}_i,\ \{\ \texttt{m}_i^1,\ \cdots,\ \texttt{m}_i^j,\ \cdots,\ \texttt{m}_i^m\ \}$	;
		...	;
		$\texttt{nodtag}_n,\ \texttt{gx}_n,\ \{\ \texttt{m}_n^1,\ \cdots,\ \texttt{m}_n^j,\ \cdots,\ \texttt{m}_n^m\ \}$	}

Where:

- $\texttt{nodtag}_i$: Node tag for i^{th} nodal force
- $\texttt{gx}_i$: Direction on global reference of external force with node tag at i^{th} nodal force; [1—2—3] for X, or Y or Z global axis.
- $\texttt{m}_i^j$: Magnitude of external force associated with node tag at i^{th} nodal force and j^{th} external force step

A.7.5 Variable nodal displacement

$$\begin{array}{llrl}
\texttt{'VariableNodalDisplacement'}, & \{ & \texttt{nodtag}_1,\ \texttt{gx}_1,\ \{\ \texttt{m}_1^1,\ \cdots,\ \texttt{m}_1^j,\ \cdots,\ \texttt{m}_1^m\ \} & ; \\
 & & \text{...} & ; \\
 & & \texttt{nodtag}_i,\ \texttt{gx}_i,\ \{\ \texttt{m}_i^1,\ \cdots,\ \texttt{m}_i^j,\ \cdots,\ \texttt{m}_i^m\ \} & ; \\
 & & \text{...} & ; \\
 & & \texttt{nodtag}_n,\ \texttt{gx}_n,\ \{\ \texttt{m}_n^1,\ \cdots,\ \texttt{m}_n^j,\ \cdots,\ \texttt{m}_n^m\ \} & \}
\end{array}$$

Where:

- $\texttt{nodtag}_i$: Node tag at i^{th} nodal displacement
- $\texttt{gx}_i$: Direction on global reference of external displacement with node tag at i^{th} nodal displacement; [1—2—3] for X, or Y or Z global axis.
- $\texttt{m}_i^j$: Magnitude of external displacement with node tag at i^{th} nodal displacement and j^{th} external displacement step

A.7.6 Variable element distributed force

$$\begin{array}{llrl}
\texttt{'VariableElementDistributedForce'}, & \{ & \texttt{eletag}_1,\ \texttt{lx}_1,\ \{\ \texttt{m}_1^1,\ \cdots,\ \texttt{m}_1^j,\ \cdots,\ \texttt{m}_1^m\ \} & ; \\
 & & \text{...} & ; \\
 & & \texttt{eletag}_i,\ \texttt{lx}_i,\ \{\ \texttt{m}_i^1,\ \cdots,\ \texttt{m}_i^j,\ \cdots,\ \texttt{m}_i^m\ \} & ; \\
 & & \text{...} & ; \\
 & & \texttt{eletag}_n,\ \texttt{lx}_n,\ \{\ \texttt{m}_n^1,\ \cdots,\ \texttt{m}_n^j,\ \cdots,\ \texttt{m}_n^m\ \} & \}
\end{array}$$

Where

- $\texttt{eletag}_i$: Element tag at i^{th} element distributed force
- $\texttt{lx}_i$: Direction on local reference of external force with element tag at i^{th} element distributed force; [1—2—3] for X, or Y or Z global axis.
- $\texttt{m}_i^j$: Magnitude of external force with element tag at i^{th} element distributed force and j^{th} external element distributed force step

A.7.7 Ground acceleration

$$\begin{array}{llrl}
\texttt{'GroundAcceleration'}, & \{\ \ \texttt{factor},\ \texttt{timestep}, & \{\ \texttt{m}_1^x,\ \texttt{m}_1^y\ [\ \texttt{m}_1^z]\ \} & ; \\
 & & \text{...} & ; \\
 & & \{\ \texttt{m}_i^x,\ \texttt{m}_i^y\ [\ \texttt{m}_i^z]\ \} & ; \\
 & & \text{...} & ; \\
 & & \{\ \texttt{m}_n^x,\ \texttt{m}_n^y\ [\ \texttt{m}_n^z]\ \} & \}
\end{array}$$

Where:

- `factor`: Factor on gravity acceleration
- `timestep`: Consistent time step
- $\texttt{m}_i^X$: Magnitude of ground acceleration along X-axis at i^{th} step
- $\texttt{m}_i^Y$: Magnitude of ground acceleration along Y-axis at i^{th} step
- $\texttt{m}_i^Z$: Magnitude of ground acceleration along Z-axis at i^{th} step

A.7.8 Variable ground acceleration

'VariableGroundAcceleration',	{	factor,	{ $\mathtt{time}_1$, $\mathtt{m}_1^X$, $\mathtt{m}_1^Y$ [$\mathtt{m}_1^Z$] }	;
			...	;
			{ $\mathtt{time}_i$, $\mathtt{m}_i^X$, $\mathtt{m}_i^Y$ [$\mathtt{m}_i^Z$] }	;
			...	;
			{ $\mathtt{time}_n$, $\mathtt{m}_n^X$, $\mathtt{m}_n^Y$ [$\mathtt{m}_n^Z$] }	}

Where:

- `factor`: Factor on gravity acceleration
- $\mathtt{time}_i$: Time at i^{th} step
- $\mathtt{m}_i^X$: Magnitude on ground acceleration along x-axis at i^{th} step
- $\mathtt{m}_i^Y$: Magnitude on ground acceleration along y-axis at i^{th} step
- $\mathtt{m}_i^Z$: Magnitude on ground acceleration along z-axis at i^{th} step

For example if the structure has `NodalForce` and `VariableNodalForce` forces to node 3 along x-axis,

```
forces  = {  'NodalForce', {3, 1, 50}                                          ;
             'VariableNodalForce', {3, 1, {0, 50, 100, 150, 200, 250, 300, 350}}  }
```

A.8 Output Block

The `OutputData` command is file recorder for output data.

OutputData	= {	$\mathtt{OutputType}_1$, $\mathtt{Filename}_1$, { $\mathtt{Info}_1$ }	;	
		...	;	
		$\mathtt{OutputType}_i$, $\mathtt{Filename}_i$, { $\mathtt{Info}_i$ }	;	
		...	;	
		$\mathtt{OutputType}_n$, $\mathtt{Filename}_n$, { $\mathtt{Info}_n$ }	}	

Where

- $\mathtt{Outputtype}_i$: is [NodalDisplacement| NodalVelocity| NodalAcceleration| NodalForce| SectionAxialForce| SectionAxialDeformation| SectionMoment| SectionCurvature| UniaxialStressStrain|].
- $\mathtt{Filename}_i$: User define

A.8.1 Nodal displacement

'NodalDisplacement', 'Filename', { $\mathtt{nodtag}_1$, $\cdots$, $\mathtt{nodtag}_i$, $\cdots$, $\mathtt{nodtag}_n$ }

Where:

- $\mathtt{nodtag}_i$: Node tag at i^{th} node

A.8.2 Nodal velocity

'NodalVelocity', 'Filename', { $\texttt{nodtag}_1$, $\cdots$, $\texttt{nodtag}_i$, $\cdots$, $\texttt{nodtag}_n$ }

Where:

- $\texttt{nodtag}_i$: Node tag at i^{th} node

A.8.3 Nodal acceleration

'NodalAcceleration', 'Filename', { $\texttt{nodtag}_1$, $\cdots$, $\texttt{nodtag}_i$, $\cdots$, $\texttt{nodtag}_n$ }

Where:

- $\texttt{nodtag}_i$: Node tag at i^{th} node

A.8.4 Nodal force

'NodalForce', 'Filename', { $\texttt{nodtag}_1$, $\cdots$, $\texttt{nodtag}_i$, $\cdots$, $\texttt{nodtag}_n$ }

Where:

- $\texttt{nodtag}_i$: Node tag at i^{th} node

A.8.5 Section axial force

'SectionAxialForce',	'Filename',	{ $\texttt{secnum}_1$, $\texttt{eletag}_1$;
		... ;
		$\texttt{secnum}_i$, $\texttt{eletag}_i$;
		... ;
		$\texttt{secnum}_n$, $\texttt{eletag}_n$; }

Where:

- $\texttt{secnum}_i$: Section of $\texttt{secnum}^{th}$ integration point in an element with eletag
- $\texttt{eletag}_i$: Element tag at i^{th} element

A.8.6 Section axial deformation

'SectionAxialDeformation',	'Filename',	{ $\texttt{secnum}_1$, $\texttt{eletag}_1$;
		... ;
		$\texttt{secnum}_i$, $\texttt{eletag}_i$;
		... ;
		$\texttt{secnum}_n$, $\texttt{eletag}_n$; }

Where:

- `secnum`$_i$: Section of `secnum`th integration point in an element with eletag
- `eletag`$_i$: Element tag at i^{th} element

A.8.7 Section moment

'SectionMoment',	'Filename',	{ `secnum`$_1$, `eletag`$_1$	;
		...	;
		`secnum`$_i$, `eletag`$_i$	;
		...	;
		`secnum`$_n$, `eletag`$_n$	; }

Where:

- `secnum`$_i$: Section of `secnum`th integration point in an element with eletag
- `eletag`$_i$: Element tag at i^{th} element

A.8.8 Section curvature

'SectionCurvature',	'Filename',	{ `secnum`$_1$, `eletag`$_1$	;
		...	;
		`secnum`$_i$, `eletag`$_i$	;
		...	;
		`secnum`$_n$, `eletag`$_n$	; }

Where:

- `secnum`$_i$: Section of `secnum`th integration point in an element with eletag
- `eletag`$_i$: Element tag at i^{th} element

A.8.9 Uniaxial stress and strain

'UniaxialStressStrain',	'Filename',	{ fibernum$_1$, `secnum`$_1$, `eletag`$_1$	;
		...	;
		fibernum$_i$, `secnum`$_i$, `eletag`$_i$	;
		...	;
		fibernum$_n$, `secnum`$_n$, `eletag`$_n$	; }

Where:

- `fibernum`$_i$: $fibernum^{th}$ layer/fiber with secnum and eletag
- `secnum`$_i$: Section of `secnum`th integration point in an element with eletag
- `eletag`$_i$: Element tag at i^{th} element

Appendix B

C++ USER's MANUAL

Appendix C

Notation

$\mathbf{P}_S^{int}$	:	Internal nodal force vector
$\mathbf{P}_t^{ext}$	:	External nodal force vector at free degrees of freedom at structural level
$\mathbf{P}_t^{int}$	:	Internal nodal force vector at free degrees of freedom at structural level
$\mathbf{P}_u^{int}$	:	Internal nodal force vector at constraint degrees of freedom at structural level
$\mathbf{P}_t^R$	:	Residual nodal force vector at free degrees of freedom at structural level
$\mathbf{u}_t$	:	Nodal displacement vector at free degrees of freedom at structural level
$\mathbf{F}_e$	:	Element nodal force vector in global reference
$\mathbf{F}_e$	=	$\lfloor N_{X1},\ V_{Y1},\ M_{Z1},\ N_{X2},\ V_{Y2},\ M_{Z2} \rfloor^T$
$\mathbf{F}_e^{int}$	:	Internal element nodal force vector in global reference
$\boldsymbol{\delta}_e$	:	Element nodal displacement vector in global reference
$\boldsymbol{\delta}_e$	=	$\lfloor u_{X1},\ v_{Y1},\ \theta_{Z1},\ u_{X2},\ v_{Y2},\ \theta_{Z2} \rfloor^T$
$\overline{\mathbf{f}}_e$	:	Element nodal force vector in local reference with rigid body modes
$\overline{\mathbf{f}}_e$	=	$\lfloor \overline{N}_{x1},\ \overline{V}_{y1},\ \overline{M}_{z1},\ \overline{N}_{x2},\ \overline{V}_{y2},\ \overline{M}_{z2} \rfloor^T$
$\overline{\mathbf{f}}_e^{int}$	:	Internal element nodal force vector in local reference with rigid body modes
$\overline{\mathbf{d}}_e$	:	Element nodal displacement vector in local reference with rigid body modes
$\overline{\mathbf{d}}_e$	=	$\lfloor \overline{u}_{x1},\ \overline{v}_{y1},\ \overline{\theta}_{z1},\ \overline{u}_{X2},\ \overline{v}_{Y2},\ \overline{\theta}_{Z2} \rfloor^T$
$\tilde{\mathbf{f}}_e$	:	Element nodal force vector in local reference without rigid body modes
$\tilde{\mathbf{f}}_e$	=	$\lfloor \tilde{M}_{z1},\ \tilde{M}_{z2},\ \tilde{N}_{x2} \rfloor^T$
$\tilde{\mathbf{f}}_e^{int}$	:	Internal element nodal force vector in local reference without rigid body modes
$\tilde{\mathbf{f}}_e^R$	:	Residual element nodal force vector in local reference without rigid body modes
$\tilde{\mathbf{d}}_e$	:	Element nodal displacement vector in local reference without rigid body modes
$\tilde{\mathbf{d}}_e$	=	$\lfloor \tilde{\theta}_{z1},\ \tilde{\theta}_{z2},\ \tilde{u}_{x2} \rfloor^T$
$\tilde{\mathbf{d}}_e^R$	:	Residual element nodal displacement vector in local reference without rigid body modes
$\mathbf{d}_s(x)$	:	Section displacement vector
$\mathbf{d}_s(x)$	=	$\lfloor u(x),\ v(x) \rfloor^T$
$\boldsymbol{\sigma}_s(x)$	:	Section force vector
κ	:	Plastic stress

$\boldsymbol{\sigma}_s(x)$	=	$\lfloor N(x),\ M(x) \rfloor^T$
$\boldsymbol{\sigma}_s^{int}(x)$	:	Internal section force vector
$\boldsymbol{\sigma}_s^{R}(x)$	:	Residual section force vector
$\boldsymbol{\varepsilon}_s(x)$	:	Section deformation vector
$\boldsymbol{\varepsilon}_s(x)$	=	$\lfloor \varepsilon_x(x),\ \phi_z(x) \rfloor^T$
$\boldsymbol{\varepsilon}_s^{int}(x)$	:	Residual section deformation vector
σ	:	Uniaxial stress
ε	:	Uniaxial strain
σ_r	:	Uniaxial stress of layer/fiber
ε_r	:	Uniaxial strain of layer/fiber
$\mathbf{N}_d(x)$	:	Shape function on displacement field
$\mathbf{B}_d(x)$	:	The matrix derived from the derivatives of $\mathbf{N}_d(x)$
$\mathbf{N}_f(x)$	:	Shape function on force field
$\mathbf{K}_S$	:	Augmented stiffness matrix at structural level
$\mathbf{K}_{tt}$	:	
$\mathbf{K}_{tu}$	:	
$\mathbf{K}_{ut}$	:	
$\mathbf{K}_{uu}$	:	
$\mathbf{K}_e$	:	Element stiffness matrix in global reference
$\overline{\mathbf{k}}_e$	:	Element stiffness matrix in local reference with rigid body modes
$\overline{\mathbf{k}}_e^{tan}$	:	Element tangent stiffness matrix in local reference with rigid body modes
$\tilde{\mathbf{k}}_e$	:	Element stiffness matrix in local reference without rigid body modes
$\tilde{\mathbf{c}}_e$	:	Element flexibility matrix in local reference without rigid body modes
$\mathbf{k}_s(x)$	:	Section stiffness matrix
$\mathbf{k}_s^{tan}(x)$	:	Section tangent stiffness matrix
$\mathbf{c}_s(x)$	:	Section flexibility matrix
$E(x)$	:	Elastic modulus
$A(x)$	:	Section area
$I_z(x)$	:	Moment of inertia on section area
$\delta\overline{\mathbf{d}}_e$	:	Virtual element nodal displacement vector in local reference
$\delta\boldsymbol{\varepsilon}_s(x)$	:	Virtual section deformation vector
L_e	:	Element length
$\boldsymbol{\Gamma}_e$	:	Transformation matrix between local and global coordinate system
$\tilde{\boldsymbol{\Gamma}}_e$	:	Transformation matrix between rigid body modes and no rigid body modes

`Subscript`

t	:	Known traction
u	:	Known displacement
S	:	Structural level
e	:	Element level or e^{th} element at element state determination
r	:	Layer/fiber level or r^{th} layer/fiber at layer/fiber state determination
s	:	Section level or s^{th} section at section state determination
d	:	Displacement field
f	:	Force field
n	:	Current step of External force/displacement vector

`Superscript`

int	:	Internal
ext	:	External
R	:	Residual
k	:	k^{th} iteration at structural level
j	:	j^{th} iteration at element level

Bibliography

Anagnostopoulos, S.: 1981, Inelastic beams for seismic analysis of structures., *Journal of Structural Engineering, ASCE* **107(ST 7)**, 1297–1311.

Banon, H., Biggs, J. and Irvine, M.: 1981, Seismic damage in reinforced concrete frames., *Journal of Structural Engineering, ASCE* **107(ST 9)**, 1713–1729.

Bathe, K.-J.: 1996, *Finite Element Procddures*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632.

Bertero, V., Aktan, A., Charney, F. and Sause, R.: 1984, Earthquake simulator tests and associated experimental, analytical and correlation studies of one-fifth scale model, *American Concrete Institute* **SP-84-13**, 375–424. in Earthquake Effects on Reinforced Concrete Structures, Detroit.

Brancaleoni, R., Ciampi, V. and Antonio, R. D.: 1983, Rate-type models for non linear hysteretic structural behavior, EUROMECH Colloquium, Palermo, Italy.

Carol, I. and Murcia, J.: 1989, Nonlinear time-dependent analysis of planar frames using an 'exact' formulation - **i**. theory, *Computers and Structures* **33**, 77–87.

Charney, F. and Bertero, V.: 1982, An evaluation of the design and analytical seismic response of a seven story reinforced concrete frame-wall structure, *Technical Report UCB/EERC-82/08*, Earthquake Engineering Research Center College of Engineering, University of California, Berkeley.

Ciampi, V. and Nicoletti, M.: 1986, Parameter identification for cyclic constitutive models with stiffness and strength degradation, 8th European Conference on Earthquake Engineering, Lisbon, Portugal.

Clough, R. and Benuska, L.: 1967, Nonlinear earthquake behavior of tall buildings., *Journal of Mechanical Engineering, ASCE* **93(EM 3)**, 129–146.

Clough, R. and Johnston, S.: 1966, Effect of stiffness degradation on earthquake ductility requirements, Transactions of Japan Earthquake Engineering Symposium, Tokyo, Japan, pp. 195–198.

Cook, R. D., Malkus, D. S., Plesha, M. E. and Witt, R. J.: 2002, *CONCEPTS AND APPLICATIONS OF FINITE ELEMENT ANALYSIS*, John Wiley and Sons, University of Wisconsin-Madison.

Darvall, L. and Mendis, P.: 1985, Elastic-plastic-softening anlysis of plane frames., *Journal of Structural Engineering, ASCE* **11(ST 4)**, 871–888.

Dermitzakis, S. and Mahin, S.: 1985, Development of substructuring techniques for on-line computer controlled seismic performance testing, *Technical Report UCB/EERC-85/04*, Earthquake Engineering Research Institute, University of California, Berkeley.

Desmorat, R., Ragueneau, F., Souid, A. and Delaplace, A.: 2008, Anisotropic damage model for multifibre dynamics and pseudo-dynamics analyses.

Doxygen: 2009, Doxygen source code documentation generator tool. http://www.stack.nl/ dimitri/doxygen/index.html.

Euro PVM/MPI 2003: 2003, *in* J. Dongarra, D. Laforenza and S. Orlando (eds), *LNCS 2840*, Springer-Verlag Berlin Heidelberg, pp. 112–116.

Faniel, I.: 2009, Unrealized potential: The socio-technical challenges of a large scale cyberinfrastructure initiative, *Technical report*, University of Michigan, School of Information.

Filippou, F.: 2004, Fedeaslab getting started guide and simulation examples, *Technical report*, Technical Report NEESgrid-2004-22.

Filippou, F. and Issa, A.: 1988, Nonlinear analysis of reinforced concrete frames under cyclic load reversals., *Technical Report UCB/EERC-88/12*, Earthquake Engineering Research Center College of Engineering, University of California, Berkeley.

Filippou, F., Popov, E. and Bertero, V.: 1983, Modeling of reinfored concrete joints under cyclic excitations, *ASCE, Journal of Structrual Engineering* **109**(11).

Gamma, E., Helm, R., Johnson, R. and Vlissides, J.: 1994, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley.

Giberson, M.: 1967, The response of nonlinear multi-story structures subjected to earthquake excitations., *Technical report*, Earthquake Engineering Research Laboratory, Pasadena.

Hilber, H., Hughes, T. and Taylor, R.: 1977, Improved numerical dissipation for time integration algorithms in structural dynamics, *Earthquake Engineering and Structural Dynamics* **5**, 283–292.

Iwan, W.: 1978, Application of nonlinear analysis techniques., *Applied Mechanics in Earthquake Engineering, ASME, AMD* **8**, 135–161. New York.

JavaDoc: 2009, How to write doc comments for the javadoc tool. http://java.sun.com/j2se/javadoc/writingdoccomments/index.html.

Jung, R.: 2005, *Development of Real-Time Hybrid Test System*, Ph. d. thesis, Department of Civil Engineering, University of Colorado, Boulder.

Kent, D. and Park, R.: 1971, Flexural members with confined concrete, *Journal of the Structural Division, ASCE* **97**(ST7).

Ladeveze, P.: 1983, Sur une théorie de l'endommagement anisotrope, *Rapport Interne LMT* **34**, 1–34.

Lai, S., Will, G. and Otani, S.: 1984, Model for inelastic biaxial bending of concrete members., *Journal of Structural Engineering, ASCE* **110(ST 11)**, 2563–2584.

Lazou, C.: 1988, *Supercomputers and their use*, Oxford Science Publications.

Lee, T.-H. and Mosalam, K. M.: 2004, Probabilistic fiber element modeling of reinforced concrete structures, *Computers and Structures* **82**, 2285–2299.

Liel, A. B.: 2008, *Assessing the Collapse Risk of California's Existing Reinforced Concrete Frame Structures*, PhD thesis, Metrics for Seismic Safety Decisions, Stanford University.

Lua: 2009, The programming language lua. http://www.lua.org/.

Mahin, S. and Williams, M.: 1981, Computer controlled seismic performance testing, In 2nd ASCE-EMD Specialty Conference on Dynamic Response of Structure.

Mazzoni, S., McKenna, F., Scott, M. H., Fenves, G. L. and et al.: 2006, Opensees command language manual, *Technical report*, The Open System for Earthquake Engineering Simulation, University of California, Berkeley.

Menegotto, M. and Pinto, P. E.: 1973, Method of analysis for cyclically loaded reinforced concrete plane frames including changes in geometry and non-elastic behavior, *La Resistance des structures et leur deformabilite ultime sous l'action de charges definies, repetees ou periodiquement variables = Resistance* pp. 15–22.

Meyer, C., Roufaiel, M. and Arzoumanidis, S.: 1983, Analysis of damaged concrete frames for cyclic loads, *Earthquake Engineering and Structural Dynamics* **11**, 207–228.

Nakashima, M., Kaminosomo, T. and Ishida, M.: 1990, Integration techniques for substructure pseudodynamic test, Proc. of the 4th U.S. National Conference on Earthquake Engineering, Palm Springs, California, pp. 515–524.

Nakashima, M. and Takai, H.: 1985, Use of substructuring techniques in pseudodynamic testing, *Technical Report Technical Report Research paper No.111*, Building Research Institute, Ministry of Construction, Tsukuba, Japan.

Otani, S.: 1974, Inelastic analysis r/c frame structures., *Journal of Structural Division, ASCE* **100(ST 7)**.

Ozdemir, H.: 1981, *Nonlinear Transient Dynamic Analysis of Yielding Structures*, Ph. d. thesis, Department of Civil Engineering, University of California Berkeley.

Prager, W. and Hodge, P.: 1951, *Theory of Perfectly Plastic Solids*, John Wiley and Sons, New York.

Ragueneau, F., Souid, A., Delaplace, A. and Desmorat, R.: 2006, Anisotropic damage for concrete within multifibre beam finite element theory, *First European Conference on Earthquake Engineering and Seismology*.

Regueiro, R. A.: 2007, Computational mechanics for geomaterials, lecture note, *Technical report*, University of Colorado at Boulder, Boulder, Colorado.

Roufaiel, M. and Meyer, C.: 1987, Analytical modeling of hysteretic behavior of r/c frames., *Journal of Structural Engineering, ASCE* **113(ST 3)**, 429–444.

Scott, B., Park, R. and Priestley, M.: 1982, Stress-strain behavior of concrete confined by overlapping hoops at low and high strain rages, *ACI Journal* **79**(1), 13–27.

Shing, P.: 2008, Integration schemes for real-time hybrid testing, *in* V. Saouma and M. Sivaselvan (eds), *Hybrid Simulation; Theory, Implementation and Applications*, Balkema.

Shing, P. and Mahin, S.: 1983, Experimental error propagation in pseudodynamic testing, *Technical Report UCB/EERC-83/12*, Earthquake Engineering Research Institute, University of California, Berkeley.

Shing, P. and Mahin, S.: 1984, Pseudodynamic test method for seismic performance evaluation: Theory and implementation, *Technical Report UCB/EERC-84/01*, Earthquake Engineering Research Institute, University of California, Berkeley.

Shing, P. and Vanna, M.: 1991, Implicit algorithm for pseudodynamic tests: Convergence and energy dissipation, *Earthquake Engineering and Structural Dynamics* **20**, 809–819.

Soleimani, D., Popov, E. and Bertero, V.: 1979, Nonlinear beam model for r/c frame analysis., 7th ASCE Conference on Electronic Computation, St. Louis.

Spacone, E., Ciampi, V. and Filippou, F. C.: 1992, A beam element for seismic damage analysis, *Technical Report UCB/EERC-92/07*, Earthquake Engineering Research Center College of Engineering, University of California, Berkeley.

Takayanagi, T. and Schnobrich, W.: 1979, Non-linear analysis of coupled wall systems, *Earthquake Engineering and Structural Dynamics* **7**, 1–22.

Takeda, T., Sozen, M. and Nielsen, N.: 1970, Reinforced concrete response to simulated earthquakes., *Journal of Structural Engineering, ASCE* **96(ST 12)**, 2557–2573.

Takizawa, H.: 1976, Notes on some basic problems in inelastic analysis of planar rc structures, *Trans. of Arch. Inst. of Japan* **240**. Part I in Feb., pp.51–62, Part II in March, pp.65–77.

Taucer, F. F., Spacone, E. and Filippou, F. C.: 1991, A fiber beam-column element for seismic response analysis of reinforced concrete structure, *Technical Report UCB/EERC-91/17*, Earthquake Engineering Research Center College of Engineering, University of California, Berkeley.

Taylor, R.: 2008, Feap; a finite element analysis program; version 8.2, *Technical report*, University of California, Berkeley.

Tedesco, J. W., McDougal, W. G. and Ross, C. A.: 1999, *Structural dynamics*, Addison Wesley Longman, Inc.

Wei, Z.: 2005, *Fast Hybrid Test System For Substructure Evaluation*, Ph. d. thesis, Department of Civil Engineering, University of Colorado, Boulder.

Williams, M. and Blakeborough, A.: 2001, Laboratory testing of structures under dynamic loads: an introductory review, *Phil. Trans. R. Soc. Lond.* **A**(359), 1651–1669.

Wu, B., Xu, G., Wang, Q. and Williams, M. S.: 2005, Operator-splitting method for real-time substructure testing, *Earthquake Engineering and Structural Dynamics* **35**(3), 293–314.

Yassin, M. H. M.: 1994, *Nonlinear analysis of prestressed concrete structures under monotonic and cyclic loads*, PhD thesis, University of California, University of California, Berkeley.

Zeris, C. and Mahin, S.: 1988, Analysis of reinforced concrete beam-columns under uniaxial excitation., *Journal of Structural Engineering, ASCE* **114(ST 4)**, 804–820.