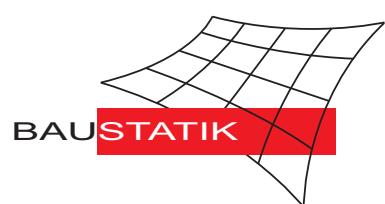
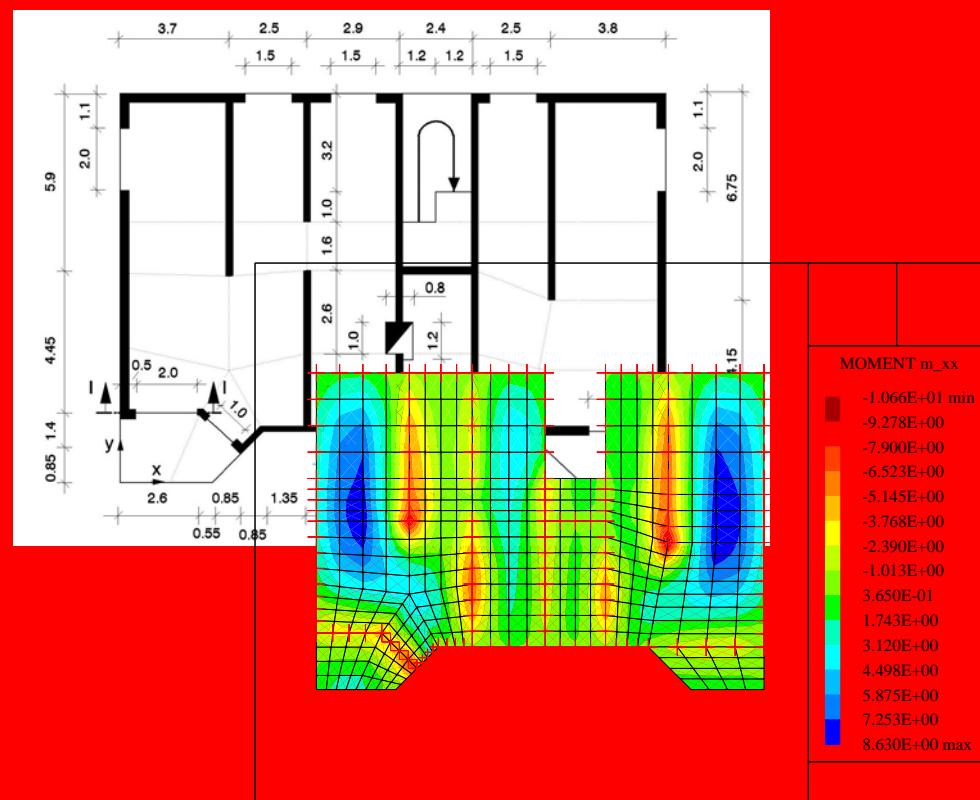


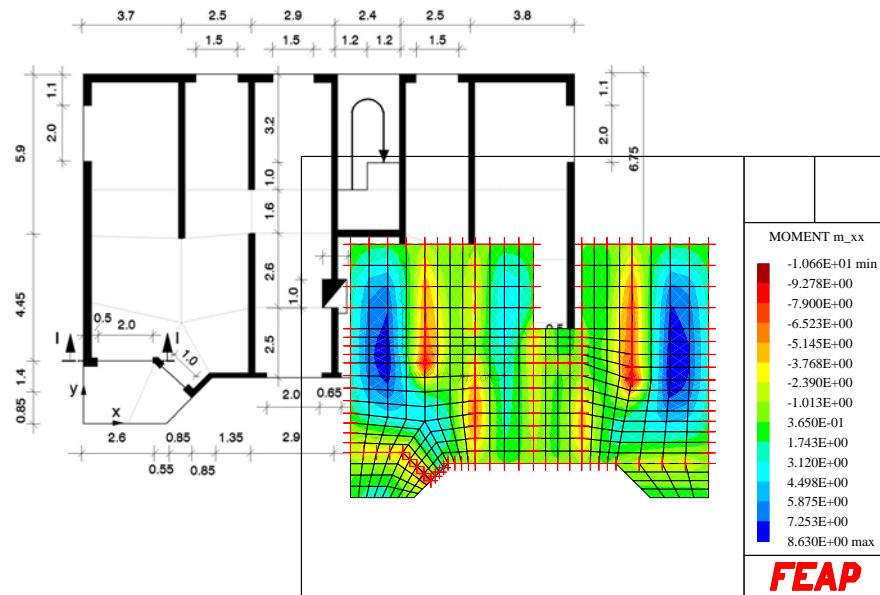
**F E A P**  
**A FINITE ELEMENT ANALYSIS  
PROGRAM**  
**Version 02/2015**



# F E A P

## A FINITE ELEMENT ANALYSIS PROGRAM

### Version 02/2015



#### Description and Users-Manual

Robert L. Taylor

University of California, Berkeley, U.S.A.

- Teil 1 : Allgemeine Beschreibung
- Teil 2 : Eingabekommandos
- Teil 3 : Makrokommandos
- Teil 4 : Graphische Ausgabe
- Teil 5 : 3D-Material-Bibliothek
- Teil 6 : Elemente
- Teil 7 : Prinzip-Beispiel
- Teil 8 : FEAP - Editor
- Teil 9 : FEAP - Netzgenerierung NEGE
- Teil 10 : FEAP - Netzgenerierung GMESH
- Teil 11 : FEAP - ISOGEO
- Teil 12 : FEAP - Fliesslinien Platten CYLT
- Teil 13 : Beispieldatensammlung
- Teil 14 : Neue Elemente
- Teil 15 : Theorie
- Teil 16 : Anwendungen und FAQs

---

## Hinweis zur Anwendung

Ab der Version 2005 sind im Manual Hyperlinks eingeführt. Dies bedeutet eine erhebliche Vereinfachung beim Suchen von Befehlen/Elementen etc.. Die volle Funktionalität des FEAP–Manuals ist daher nur bei der Betrachtung des PDF-Files verfügbar. Bei den Kapiteln **MESH**, **MACRO**, **PLOT**, **ELMT** ist zu Beginn jeweils eine Liste der möglichen Befehle sowie eine Liste von möglichen Stichworten sowie anschließend eine Tabelle der möglichen Befehle angeordnet. Darüberhinaus können von jeder Seite über die Fußzeile die einzelnen Kapitel erreicht werden. Alle Kapitel können aus dem Inhaltsverzeichnis **CONTENTS** direkt erreicht werden. Gleiches ist auch möglich über die im Acrobat–Reader verfügbare Liste der Lesezeichen.

## Copyright

- Ohne Genehmigung des Autors ist es nicht gestattet, dieses Heft ganz oder teilweise auf fotomechanischem Wege (Fotokopie, Mikrokopie) zu vervielfältigen.
  - Stand: July 1, 2015
  - © Prof. Dr.–Ing. W. Wagner  
Institut für Baustatik  
Karlsruhe Institute of Technology  
Postfach 6980  
76128 Karlsruhe
- Telefon: (0721) 608–42280  
Telefax: (0721) 608–46015  
E-mail: info@ibs.kit.edu  
Internet: <http://www.ibs.kit.edu>

# Contents

<b>1</b>	<b>Introduction to Users Manual</b>	<b>13</b>
1.1	Allgemeine Beschreibung . . . . .	13
1.2	Eingabekommandos . . . . .	14
1.3	Macrokommandos . . . . .	14
1.4	Graphische Ausgabe . . . . .	15
1.5	Elemente . . . . .	15
1.6	Prinzipbeispiel . . . . .	15
1.7	FEAP–Editor . . . . .	15
1.8	FEAP–Netzgenerierung NEGE . . . . .	15
1.9	FEAP–Netzgenerierung GMESH . . . . .	16
1.10	FEAP–Netzgenerierung CYLT . . . . .	16
1.11	Beispielsammlung . . . . .	16
1.12	Hinweise zum Hinzufügen neuer Elemente . . . . .	16
1.13	Theorie . . . . .	16
<b>2</b>	<b>Mesh Commands</b>	<b>17</b>
2.1	Available Macros . . . . .	17
2.2	Overview on commands . . . . .	18
2.3	Overview on actions . . . . .	20
2.4	Aufbau FEAP–Eingabefile . . . . .	22
2.5	Startoptionen für FEAP . . . . .	22
2.5.1	Ausführung als Windowsprogramm . . . . .	22
2.6	Anwendung im reinen Batchbetrieb . . . . .	22
2.7	Macros in detail . . . . .	23
<b>3</b>	<b>Macro Commands</b>	<b>125</b>
3.1	Available Macros . . . . .	125
3.2	Overview on commands . . . . .	126
3.3	Overview on actions . . . . .	128
3.4	Macros in detail . . . . .	130

<b>4 Plot Commands</b>	<b>251</b>
4.1 Available Macros . . . . .	251
4.2 Overview on commands . . . . .	252
4.3 Overview on actions . . . . .	254
4.4 Macros in detail . . . . .	256
<b>5 3D-Material library</b>	<b>354</b>
<b>6 Elements in Student Version</b>	<b>362</b>
6.1 Available elements . . . . .	362
6.2 Element numbers . . . . .	364
6.3 Short description of elements . . . . .	366
6.4 Detailed description of elements . . . . .	369
6.4.1 ELMT01 . . . . .	369
6.4.2 ELMT02 . . . . .	370
6.4.3 ELMT03 . . . . .	372
6.4.4 ELMT04 . . . . .	374
6.4.5 ELMT05 . . . . .	376
6.4.6 ELMT06 . . . . .	378
6.4.7 ELMT07 . . . . .	379
6.4.8 ELMT08 . . . . .	381
6.4.9 ELMT09 . . . . .	383
6.4.10 ELMT10 . . . . .	386
6.4.11 ELMT11 . . . . .	387
6.4.12 ELMT12 . . . . .	390
6.4.13 ELMT13 . . . . .	395
6.4.14 ELMT14 . . . . .	397
6.4.15 ELMT15 . . . . .	398
6.4.16 ELMT16 . . . . .	400
6.4.17 ELMT17 . . . . .	401
6.4.18 ELMT18 . . . . .	403
6.4.19 ELMT19 . . . . .	405
6.4.20 ELMT20 . . . . .	407
6.4.21 ELMT21 . . . . .	408
6.4.22 ELMT22 . . . . .	411
6.4.23 ELMT23 . . . . .	412
6.4.24 ELMT24 . . . . .	413
6.4.25 ELMT25 . . . . .	415

6.4.26 ELMT26 . . . . .	416
6.4.27 ELMT27 . . . . .	417
6.4.28 ELMT28 . . . . .	417
6.4.29 ELMT29 . . . . .	417
6.4.30 ELMT30 . . . . .	418
6.4.31 ELMT31 . . . . .	419
6.4.32 ELMT32 . . . . .	419
6.4.33 ELMT33 . . . . .	420
6.4.34 ELMT34 . . . . .	422
6.4.35 ELMT35 . . . . .	422
6.4.36 ELMT36 . . . . .	422
6.4.37 ELMT37 . . . . .	422
6.4.38 ELMT38 . . . . .	422
6.4.39 ELMT39 . . . . .	422
6.4.40 ELMT40 . . . . .	423
6.4.41 ELMT41 . . . . .	423
6.4.42 ELMT42 . . . . .	423
6.4.43 ELMT43 . . . . .	423
6.4.44 ELMT44 . . . . .	423
6.4.45 ELMT45 . . . . .	424
6.4.46 ELMT46 . . . . .	427
6.4.47 ELMT47 . . . . .	427
6.4.48 ELMT48 . . . . .	427
<b>7 Principal example</b>	<b>428</b>
<b>8 F E A P - E D Editor</b>	<b>432</b>
<b>9 F E A P - NEGE Mesher</b>	<b>433</b>
<b>10 F E A P - GMESH Mesher</b>	<b>445</b>
<b>11 F E A P - ISOGEO</b>	<b>452</b>
11.1 Overview . . . . .	452
11.2 Terminology . . . . .	452
11.3 Known Errors . . . . .	452
<b>12 F E A P – CYLT Mesh Generator for Yield-Line Predictions</b>	<b>453</b>

<b>13 F E A P - Beispielsammlung</b>	<b>463</b>
13.1 Beispiele für 2D/3D - Fachwerkelement . . . . .	464
13.1.1 Beispiel 1 - Dachbinder . . . . .	464
13.1.1.1 System und Belastung . . . . .	464
13.1.1.2 Eingabedatensatz (file: it01) . . . . .	464
13.1.1.3 Ergebnisse . . . . .	465
13.1.1.4 Vergleichslösung . . . . .	465
13.1.1.5 verformte Struktur (20-fach überhöht) . . . . .	465
13.1.2 Beispiel 2 - Kran . . . . .	466
13.1.2.1 System und Belastung . . . . .	466
13.1.2.2 Eingabedatensatz (file: it02) . . . . .	466
13.1.2.3 Ergebnisse . . . . .	467
13.1.2.4 Vergleichslösung . . . . .	467
13.1.2.5 verformte Struktur (20-fach überhöht) . . . . .	468
13.1.3 Beispiel 3 - Brückenbogen . . . . .	469
13.1.3.1 System und Belastung . . . . .	469
13.1.3.2 Eingabedatensatz (file: it03) . . . . .	469
13.1.3.3 Ergebnisse . . . . .	470
13.1.3.4 Struktur und Verschiebungen . . . . .	471
13.1.4 Beispiel 4 - Raumfachwerk . . . . .	472
13.1.4.1 System und Belastung . . . . .	472
13.1.4.2 Eingabedatensatz (file: it04) . . . . .	472
13.1.4.3 Ergebnisse . . . . .	473
13.1.4.4 Struktur und Verschiebungen . . . . .	474
13.2 Beispiele für 2D - Stabelement . . . . .	475
13.2.1 Beispiel 1 - Zweigelenkrahmen . . . . .	475
13.2.1.1 System und Belastung . . . . .	475
13.2.1.2 Eingabedatensatz für Lastfall 1 (file: ib2d1a) . . . . .	475
13.2.1.3 Eingabedatensatz für Lastfall 2 (file: ib2d1b) . . . . .	475
13.2.1.4 Eingabedatensatz für Lastfall 3 (file: ib2d1c) . . . . .	475
13.2.1.5 Ergebnisse . . . . .	476
13.2.1.6 Vergleichslösung . . . . .	476
13.2.2 Beispiel 2 - Rahmentragwerk . . . . .	477
13.2.2.1 System und Belastung . . . . .	477
13.2.2.2 Eingabedatensatz (file: ib2d2) . . . . .	477
13.2.2.3 Ergebnisse . . . . .	477
13.2.2.4 Vergleichslösung . . . . .	477

13.2.2.5 Schnittkraftlinie . . . . .	477
13.2.3 Beispiel 3 - elastisch gebetteter Balken . . . . .	479
13.2.3.1 System und Belastung . . . . .	479
13.2.3.2 Eingabedatensatz (file: ib2d3) . . . . .	479
13.2.3.3 Ergebnisse . . . . .	479
13.2.3.4 Vergleichslösung . . . . .	479
13.2.4 Beispiel 4 - Halbrahmen . . . . .	480
13.2.4.1 System und Belastung . . . . .	480
13.2.4.2 Eingabedatensatz (file: ib2d4) . . . . .	480
13.2.4.3 Ergebnisse . . . . .	480
13.2.4.4 Vergleichslösung . . . . .	480
13.3 Beispiele für 3D - Stabelement . . . . .	481
13.3.1 Beispiel 1 - räumliche Rahmenecke . . . . .	481
13.3.1.1 System und Belastung . . . . .	481
13.3.1.2 Eingabedatensatz (file: ib3d1) . . . . .	481
13.3.1.3 Ergebnisse . . . . .	481
13.3.1.4 Vergleichslösung . . . . .	481
13.3.2 Beispiel 2 - räumlicher Rahmen . . . . .	482
13.3.2.1 System und Belastung . . . . .	482
13.3.2.2 Eingabedatensatz (file: ib3d2) . . . . .	482
13.3.2.3 Ergebnisse . . . . .	482
13.3.2.4 Vergleichslösung . . . . .	482
13.4 Beispiele für Scheibenelemente . . . . .	483
13.4.1 Beispiel 1 - Kragarm mit Einzellast . . . . .	483
13.4.1.1 System und Belastung . . . . .	483
13.4.1.2 Eingabedatensatz für Beispiel 1 (file: is1) . . . . .	483
13.4.1.3 Ergebnisse . . . . .	484
13.4.1.4 Vergleichslösung . . . . .	484
13.4.2 Beispiel 2 - Wandscheibe mit Gleichstreckenlast auf zwei Stützen . . . . .	485
13.4.2.1 System und Belastung . . . . .	485
13.4.2.2 Eingabedatensatz (file: is2) . . . . .	486
13.4.2.3 Ergebnisse . . . . .	487
13.4.2.4 Vergleichslösung . . . . .	488
13.4.3 Beispiel 3 - Wandscheibe mit Loch . . . . .	489
13.4.3.1 System und Belastung . . . . .	489
13.4.3.2 NEGE-Eingabedatensatz (file: is3.neg) . . . . .	490
13.4.3.3 Ergebnisse – verformtes Netz . . . . .	491

13.4.3.4 Vergleichslösung . . . . .	491
13.5 Beispiele für Plattenelement . . . . .	492
13.5.1 Beispiel 1 - Platte mit verschiedenen Lagerbedingungen . . . . .	492
13.5.1.1 Systeme und Belastung . . . . .	492
13.5.1.2 Eingabedatensatz . . . . .	493
13.5.1.3 Ergebnisse . . . . .	494
13.5.1.4 Vergleichslösungen . . . . .	495
13.5.1.5 Bemessung der unterstützten Platte . . . . .	496
13.6 Beispiele für Rotationsschalenelement . . . . .	497
13.6.1 Beispiel 1 - Zylinder mit Randlast . . . . .	497
13.6.1.1 System und Belastung . . . . .	497
13.6.1.2 Eingabedatensatz (file: isr-1) . . . . .	497
13.6.1.3 Ergebnisse . . . . .	498
13.6.2 Beispiel 2 - Zylinder mit Wasserfüllung . . . . .	499
13.6.2.1 System und Belastung . . . . .	499
13.6.2.2 Eingabedatensatz (file: isr-2) . . . . .	499
13.6.2.3 Ergebnisse . . . . .	500
13.6.3 Beispiel 3 - Zylinder mit gleichmäßiger Temperaturbelastung . . . . .	501
13.6.3.1 System und Belastung . . . . .	501
13.6.3.2 Eingabedatensatz (file: isr-6) . . . . .	501
13.6.3.3 Ergebnisse . . . . .	501
13.6.4 Beispiel 4 - Zylinder mit ungleichmäßiger Temperaturbelastung . . . . .	502
13.6.4.1 System und Belastung . . . . .	502
13.6.4.2 Eingabedatensatz (file: isr-7) . . . . .	502
13.6.4.3 Ergebnisse . . . . .	502
13.6.5 Beispiel 5 - Kegel mit Randmoment . . . . .	503
13.6.5.1 System und Belastung . . . . .	503
13.6.5.2 Eingabedatensatz (file: isr-3) . . . . .	503
13.6.5.3 Ergebnisse . . . . .	504
13.6.6 Beispiel 6 - Kugel mit Innendruck . . . . .	505
13.6.6.1 System und Belastung . . . . .	505
13.6.6.2 Eingabedatensatz (file: isr-4) . . . . .	505
13.6.6.3 Ergebnisse . . . . .	506
13.6.7 Beispiel 7 - Kreisplatte mit Flächenlast . . . . .	507
13.6.7.1 System und Belastung . . . . .	507
13.6.7.2 Eingabedatensatz (file: isr-5) . . . . .	507
13.6.8 Beispiel 8 - Kreisplatte mit Loch unter Temperaturbelastung . . . . .	508

13.6.8.1 System und Belastung . . . . .	508
13.6.8.2 Eingabedatensatz (file: isr-8) . . . . .	508
13.7 Beispiele für allgemeines Schalenelement . . . . .	509
13.7.1 Beispiel 1 - Eingespannter T-Träger . . . . .	509
13.7.1.1 System und Belastung . . . . .	509
13.7.1.2 Systemdaten für eine analytische Vergleichslösung . . . . .	509
13.7.1.3 Eingabedatensatz (file: isg-1) . . . . .	510
13.7.2 Beispiel 2 - Beidseitig eingespannter Zylinder mit Einzellasten . . . . .	511
13.7.2.1 System und Belastung . . . . .	511
13.7.2.2 Eingabedatensatz (file: isg-2) . . . . .	511
13.7.2.3 Ergebnisse und verformtes Netz . . . . .	512
13.7.3 Beispiel 3 - Membrangelagertes Tonnendach . . . . .	513
13.7.3.1 System und Belastung . . . . .	513
13.7.3.2 Eingabedatensatz (file: isg-3) . . . . .	513
13.7.3.3 Schnittkraftverläufe und Ergebnisvergleich . . . . .	514
<b>14 Adding elements</b>	<b>515</b>
14.1 General information . . . . .	515
14.1.1 Name of Subprogram, parameters . . . . .	515
14.1.2 Control Switch Parameter Values . . . . .	518
14.2 Common Blocks . . . . .	519
14.3 Treatment of history terms . . . . .	522
14.3.1 Assigning amount of storage for each element . . . . .	522
14.3.2 Accessing history data for each element . . . . .	523
14.3.3 Typical 2D-element framework for history data . . . . .	524
14.4 Typical element framework . . . . .	525
14.5 Use of Parameters and Expression Inputs in New Elements . . . . .	526
14.6 Error analysis . . . . .	527
14.7 Logical Flags . . . . .	527
14.8 3D-Material library . . . . .	528
14.9 Adding elements for FE <sup>2</sup> . . . . .	529
14.10 Plot of user defined data . . . . .	530
<b>15 Theory Manual</b>	<b>531</b>
15.1 Programming structure of FEAP . . . . .	531
15.1.1 General structure . . . . .	531
15.1.2 Calculating arrays on element level . . . . .	532
15.1.3 Displacement arrays . . . . .	532

15.1.4 Making FEAP faster . . . . .	533
15.2 Stability analysis . . . . .	534
15.3 Time integration procedures . . . . .	537
15.3.1 Newmark–method . . . . .	537
15.3.2 Overview Implicit Schemes . . . . .	538
15.3.3 Explicit Schemes . . . . .	538
15.3.4 Energy-conserving algorithm . . . . .	539
15.3.5 Implicit composite scheme - Bathe . . . . .	540
15.3.6 Automatic time stepping procedure . . . . .	540
15.4 PCG-Methods . . . . .	543
15.5 PGMRES-Methods . . . . .	545
15.6 Error Analysis . . . . .	546
15.7 Eigenvalue Computations . . . . .	547
15.7.1 Subspace Iteration . . . . .	547
15.7.2 Lanczos Iteration . . . . .	548
15.8 Augmented Lagrange Formulation . . . . .	550
15.9 Theory of element formulations . . . . .	551
15.9.1 3D-Material library . . . . .	551
15.9.1.1 Material model 1: . . . . .	551
15.9.1.2 Material model 2: . . . . .	552
15.9.1.3 Material model 3: . . . . .	552
15.9.1.4 Material model 4: . . . . .	553
15.9.1.5 Material model 5: . . . . .	553
15.9.1.6 Material model 6: . . . . .	554
15.9.1.7 Material model 7: . . . . .	555
15.9.1.8 Material model 8: . . . . .	555
15.9.1.9 Material model 9: . . . . .	563
15.9.1.10 Material model 10: . . . . .	563
15.9.1.11 Material model 11: . . . . .	563
15.9.1.12 Material model 12: . . . . .	564
15.9.1.13 Material model 13: . . . . .	564
15.9.1.14 Material model 14: . . . . .	564
15.9.1.15 Material model 15: . . . . .	565

<b>16 Applications and FAQs</b>	<b>567</b>
16.1 FAQs . . . . .	567
16.1.1 Crash within start procedure . . . . .	567
16.1.2 Mixing elements . . . . .	567
16.2 Some details using FEAP . . . . .	568
16.2.1 Use of macro prop . . . . .	568
16.2.2 Use of macro newf . . . . .	570

# Chapter 1

## Introduction to Users Manual

### 1.1 Allgemeine Beschreibung

Mit der vorliegenden Studentenversion des Programmes FEAP stellt das Institut für Baustatik der Universität Karlsruhe ein Finite-Element-Programm zur linearen Berechnung allgemeiner Strukturen für Bauingenieurstudenten bereit. Neben Fachwerken, 2D- und 3D-Stäben können Scheiben, Platten und Schalentragwerke berechnet werden. Das Programm wurde als ein Tool im Forschungs- und Entwicklungsbereich sowie für die Lehre entwickelt. Die Stärken des Programmes liegen im unmittelbaren Zugriff, der Möglichkeit neueste Forschungsergebnisse zu implementieren, sowie der einfachen Einbindung von Elementen, was häufig in Studienarbeiten geschieht. Aufgrund der damit verbundenen ständigen Modifikationen kann für eine Fehlerfreiheit des Programmes nicht garantiert werden. Damit ergibt sich der 'natürliche Zwang', die berechneten Ergebnisse kritisch zu prüfen, Überschlags- und einfache Vergleichsrechnungen durchzuführen. Zu diesem Zweck ist insbesondere auf den grundlegenden Vorlesungen der Baustatik aufzubauen.

Aufgrund der allgemeinen Einsetzbarkeit des Programmes, aufgrund der ständigen Modifikationen und des wissenschaftlichen Versuchscharakters können und wollen die Programmierer nicht mit kommerziellen Finite-Element-Programmen konkurrieren, insbesondere den Spezialprogrammen im Baubereich.

Das Programm ist so ausgelegt, daß – auf der Basis einer mittels Editor erstellten Eingabedatei – die Bearbeitung eines Problems, d.h. die Berechnung und die graphische Auswertung interaktiv erfolgt. Dabei steht eine 'On-Line'-Dokumentation aller zu verwendenden Befehle zur Verfügung.

Die Beschreibung der Dateneingabe, der Berechnung und der graphischen Darstellung ist in den Teilen 2-4 beschrieben. Teil 5 enthält die Beschreibung der zur Verfügung stehenden Elemente, während in Teil 6 eine umfangreiche Beispieldatenbank zur Verfügung steht.

Die nachfolgenden Teile 2 - 6.1 sind in englischer Sprache dargestellt, was in der direkten Zusammenarbeit mit Prof. Taylor aus Berkeley begründet ist. Man betrachte dies nicht als Nachteil, sondern vielmehr als Ansporn, sich mit einer anderen Sprache auseinanderzusetzen und als ersten Schritt, auch wissenschaftliche Literatur in Fremdsprachen zu lesen.

## 1.2 Eingabekommandos

Dieser Teil des Handbuchs enthält **Macrobefehle**, die dazu dienen, ein Finite–Element–Netz für die Analyse mit FEAP zu spezifizieren. Die Eingabe eines Problems, das mit finiten Elementen gelöst werden soll, beinhaltet im wesentlichen die Definition des Finite–Element–Netzes mit den entsprechenden Rand– und Belastungsbedingungen und die Auswahl der zu verwendenden Elemente. Diese Eingabedaten sind in einem Editor zu erstellen. Durch die Struktur der Eingabe sind z. B. auch Generierungen möglich, so daß die reine Dateneingabe auf ein Minimum beschränkt ist.

Die Dateneingabe läßt sich wesentlich vereinfachen, wenn zu Beginn des Datensatzes unter dem Macro **para** Parameter belegt werden, die bei der späteren Eingabe berücksichtigt werden. So können einem Knoten beispielsweise die Koordinaten  $x = l, y = 0.5 \cdot l, z = (l + 3) \cdot 2$  zugewiesen werden. Das Beispiel zeigt, daß auch arithmetische Ausdrücke zugelassen sind.

Die wesentliche Daten, die zur Definition eines Netzes benötigt werden, sind in der folgenden Tabelle aufgeführt:

Eingabe	Macro
* Globale Problemdefinition	<b>feap</b>
* Parameterdefinition	<b>para</b>
* Knotenkoordinaten	<b>coor</b>
* Elementtopologie (Zusammenhang des FE-Netzes)	<b>elem</b>
* Randbedingungen	<b>boun</b>
* Belastungen	<b>load</b>
* Materialeigenschaften, Elementzuordnung	<b>mate</b>

In FEAP werden diese einzelnen Datengruppen jeweils nach einem Macro (Abkürzung für den Befehl z.B. **coor** für die Eingabe der Knotenkoordinaten) in der Eingabedatei abgelegt. Die Reihenfolge dieser Datengruppen ist beliebig. Neben der Eingabe dieser Grunddaten gibt es noch einige weitere Möglichkeiten zur Spezifizierung des FE-Netzes und eventueller Anfangsbedingungen, sowie Generierungsmöglichkeiten. Diese sind in der nachfolgenden Tabelle enthalten.

Eingabe	Macro
* Generierung von 2D-Netzen	<b>bloc</b>
* Generierung von 2D-Netzen	<b>nege</b>
* Generierung von 2D-Netzen	<b>gmesh</b>
* Generierung von Randbedingungen	<b>ebou</b>
* Generierung von Randbedingungen	<b>edge</b>
* Generierung von Randlasten	<b>eloa</b>
* Umrechnung von Polar- auf Kartesisches KOS	<b>pola</b>

Im Teil 2 des Manuals werden die oben angegebenen Macros in alphabetischer Reihenfolge näher beschrieben und an Beispielen gezeigt, wie die jeweilige Eingabe bei einem Macro vorzunehmen ist.

## 1.3 Macrokommandos

Dieser Teil des Handbuchs enthält die Befehle, die dazu dienen, dem Programm FEAP die Algorithmen vorzugeben und weiterhin die Ergebnisse darzustellen. Diese Kommandos können interaktiv vorgegeben werden. Die **Macrobefehle** sind im interaktiven Mode direkt am Bildschirm einzugeben. Eine Macrobefehlsfolge wird innerhalb der Kommandos **inte** sowie **exit** oder **quit** vorgegeben. Hierbei steht **inte** im Input–File nach **end!** **exit** gibt das Ende der Macrobefehlsfolge an, wenn eine RESTART - Datei erstellt werden soll. **quit** gibt das Ende der Macrobefehlsfolge an, wenn keine RESTART - Datei erstellt werden soll. **exit** bzw. **quit** werden im interaktiven Mode am Bildschirm eingegeben.

## 1.4 Graphische Ausgabe

In diesem Teil des Manuals werden die graphischen Ausgabemöglichkeiten des Programmes FEAP beschrieben. Das Ansprechen der Graphik geschieht wieder über **Macrobefehle** analog den Kommandos auf der Macroebene. Ein typischer Plotbefehl lautet z.B.

**load**, n1, n2, n3 = load, -1, 1, 2

Damit werden alle Lasten auf das verformte Netz (n1) mit einem Vergrößerungsfaktor (n3) gezeichnet. Die Spitze der Lastvektoren zeigt dabei auf den Knoten (n2). Es können die Knoten mit Nummern, die Systeme, die Elementnummern, die Randbedingungen, die Belastungen, die Reaktionskräfte sowie die unterschiedlichen Materialtypen dargestellt werden.

Für Stäbe kann der Schnittgrößenverlauf, für Scheiben, Platten und Schalen die flächenhaften Verteilung von Schnittgrößen (Spannungen) und Verschiebungen dargestellt werden. Alle Größen können in der Regel bezogen auf das unverformte als auch bezogen auf das verformte System geplottet werden.

Weiterhin können Spannungs- und Verschiebungsverläufe bei Scheiben, Platten und Schalen in Schnitten berechnet und geplottet werden.

## 1.5 Elemente

Dieser Teil des Manuals enthält die Beschreibung der zur vorliegenden Studentenversion FEAP gehörenden finiten Elemente. Es werden die Eingabedaten, die möglichen Macro-daten sowie die Plotoptionen beschrieben.

In der vorliegenden **Elementbibliothek** befinden sich 12 (in der Regel lineare) Elemente aus allen Bereichen der Statik. Neben einem allgemeinen 2D/3D-Fachwerkelement sind zwei 2D-Balkenelemente sowie ein 3D-Balkenelement enthalten. Für Scheiben werden 2 Elemente, die sich hinsichtlich der Anwendungsbereiche unterscheiden, zur Verfügung gestellt. Neben einem Plattenelement werden 2 Schalenelemente - eines für Rotationsschalen unter rotationssymmetrischer Belastung sowie eines für eine allgemeine Schalengeometrie - angeboten. Weiterhin ist ein Feder/Masse-element hinzugefügt, mit dem knotenweise Steifigkeiten bzw. Punktmassen addiert werden können. Das 11. Element erlaubt die Beschreibung exzentrisch angeordneter 3-D-Stäbe und ist auch im geometrisch nichtlinearen Fall einsetzbar. Schließlich ist das letzte Element für Wärmeleitungs-, Torsions-, sowie Grundwasserströmungsprobleme einsetzbar. Darüberhinaus lassen sich mit diesem Element alle notwendigen Systemwerte beliebiger dickwandiger Querschnitte ermitteln.

## 1.6 Prinzipbeispiel

Hier wird ein **Beispiel** exemplarisch detailliert erklärt.

## 1.7 FEAP–Editor

In diesem Abschnitt werden Informationen zum **FEAP–Editor** gegeben.

## 1.8 FEAP–Netzgenerierung NEGE

In diesem Abschnitt werden Informationen zur 2D–Netzgenerierung mit **NEGE** gegeben.

## 1.9 FEAP–Netzgenerierung GMESH

In diesem Abschnitt werden Informationen zur 2D/3D–Netzgenerierung mit **GMESH** gegeben.

## 1.10 FEAP–Netzgenerierung CYLT

In diesem Abschnitt werden Informationen zur 2D–Netzgenerierung mit **CYLT** gegeben.

## 1.11 Beispieldaten

Dieser Teil des Manuals enthält eine Sammlung von durchgerechneten **Beispielen** zu den einzelnen Elementtypen. Die Beispiele sind so einfach wie möglich gehalten, um die jeweiligen Grundsätze der Dateneingabe zu demonstrieren. Die angegebenen Datensätze der Beispiele brauchen nicht abgetippt zu werden. Sie stehen auf den Rechnern im CIP-Pool zur Verfügung.

## 1.12 Hinweise zum Hinzufügen neuer Elemente

Dieser Teil des Manuals enthält Programmierinformationen zum Hinzufügen von neuen **Elementen**.

## 1.13 Theorie

Der letzte Teil des Manuals enthält ausgewählte **theoretische Hintergründe** zu Algorithmen und einzelnen Elementen.

# Chapter 2

## Mesh Commands

### 2.1 Available Macros

The following entries are available for the control information and input of the mesh:

feap	aloa	angl	back	base	blco	bloc	blox
boun	btem	cmod	coor	curv	cylt	debu	disp
eang	ebou	edge	edis	elem	elfr	eloa	el3b
end	epsq	fixe	gbou	gcor	gele	geom	gmesh
icon	impf	inte	icor	isec	jint	knv1	k nv2
link	load	loa0	macro	mate	mesn	nege	neco
ndvi	nmpq	nopa	nopr	opti	para	pars	poin
pola	pres	prin	qloa	rbou	regi	rndm	rot
rsum	segm	sloa	solv	sphe	stop	temp	tie
tran	trib	vang	vbou	ybou	ycon	yedg	yloa
ynod							

Each of these commands describes a specific function to be used in defining the problem to be solved. A short overview on commands is given in section 2.2, see macro [Index](#) in FEAP, whereas possible actions can be found in section 2.3, see macro [Action](#) in FEAP.

## 2.2 Overview on commands

Macro	Aufgabe
<b>aloa</b>	Lasten auf Gebieten
<b>angl</b>	Basis lokal: Knoten
<b>back</b>	Hintergrundnetz bei Generierung
<b>blco</b>	Knoten+Elemente: Generierung Interface Zone
<b>bloc</b>	Knoten+Elemente: 2D+3D Generierung
<b>base</b>	Dreibeinberechnung
<b>blox</b>	Knoten+Elemente: 2D Generierung
<b>boun</b>	Randbedingungen an Knoten
<b>btem</b>	Temperaturen: 2D+ 3D Generierung
<b>cmod</b>	Koordinaten: spezielle Modifikationen
<b>coor</b>	Koordinaten
<b>curv</b>	Definition von Kurven
<b>cylt</b>	Titelzeile + Generierungstyp bei CYLT
<b>debu</b>	Debug - Option
<b>eang</b>	Basis lokal: Linie
<b>ebou</b>	Randbedingungen für eine feste Koord.
<b>edge</b>	Randbedingungen für eine Linie
<b>edis</b>	Werte für Randbedingungen für eine feste Koord.
<b>el3b</b>	Elemente fuer 3D - Stab
<b>elem</b>	Elemente
<b>elfr</b>	Elemente, freie Numerierung
<b>eloa</b>	Linienlasten
<b>end</b>	Ende Eingabedaten
<b>epsq</b>	Verzerrungslasten
<b>feap</b>	Titelzeile + Problemgrößen
<b>fixe</b>	Randbedingungen an Segmenten bei Generierung
<b>gbou</b>	Koordinaten bei Generierung GMESH
<b>gcor</b>	Elemente bei Generierung GMESH
<b>gele</b>	Koordinaten der Segmente bei Generierung
<b>geom</b>	Titelzeile + Problemgroessen bei GMESH
<b>gmesh</b>	
<b>icon</b>	Kontaktdateneingabe
<b>icor</b>	Koordinaten: Imperfektionen
<b>impf</b>	Imperfektionen
<b>inte</b>	Interaktive Eingabe der Macros
<b>isec</b>	Verschneidungen
<b>jint</b>	$J_2$ -Integraleingabe
<b>knv1</b>	Knotenvektoren $\Xi^1$ für FEAP - ISOGEO
<b>knv2</b>	Knotenvektoren $\Xi^2$ für FEAP - ISOGEO
<b>link</b>	Verbindung von Freiheitsgraden
<b>load</b>	Knotenlasten
<b>loa0</b>	ständige Knotenlasten F0

Macro	Aufgabe
<b>macr</b>	Macro - Modus
<b>mate</b>	Materialdaten
<b>mesn</b>	Anwender Eingabe Macro
<b>ndvi</b>	Unterteilung bei Generierung GMESH
<b>neco</b>	Definition von Konstanten bei Generierung
<b>nege</b>	Titelzeile + Problemgrößen bei NEGE
<b>nmpq</b>	Geometrieparameter f. FEAP - ISOGEO
<b>nopa</b>	Dateneingabe nur Zahlen
<b>nopr</b>	Datenausgabe: aus
<b>opti</b>	Knotennummernoptimierung
<b>optn</b>	Knotennummernoptimierung
<b>para</b>	Definition von Konstanten
<b>pars</b>	Dateneingabe allgemeine Ausdrücke
<b>poin</b>	Randbedingungen an einem Punkt
<b>poin</b>	Knotenlasten an einem Punkt
<b>pola</b>	Koordinaten: polar → kartesisch
<b>pres</b>	Linienlasten auf Segmenten bei Generierung
<b>prin</b>	Datenausgabe: ein
<b>qloa</b>	Elementlasten
<b>rbou</b>	Randbedingungen in einem Zylinderbereich
<b>rndm</b>	Koordinaten: Zufallsveränderung
<b>rsum</b>	Reaktionskräfte bei mehreren Knoten
<b>regi</b>	Regionen bei Generierung
<b>rot</b>	Koordinaten: Rotation um Achse
<b>segm</b>	Segmente bei Generierung
<b>sloa</b>	Oberflächen- bzw. Folgelasten
<b>solv</b>	Gleichungsloeser
<b>sphe</b>	Koordinaten: kugel → kartesisch
<b>stop</b>	Ende der Berechnung/ letzte Karte Eingabe
<b>temp</b>	Temperaturen
<b>tie</b>	Verbindung von Knoten mit gl. Koordinaten
<b>trans</b>	Koordinaten: Translation
<b>trib</b>	Knoten/Elemente 2D+3D Generierung Dreiecke
<b>vang</b>	Basis lokal: Region
<b>vbou</b>	Randbedingungen in einer Region
<b>ybou</b>	Randbedingungen bei Fliesslinien
<b>ycon</b>	Definition von Konstanten bei Generierung CYLT
<b>yedg</b>	Kantengenerierung bei Generierung CYLT
<b>yloa</b>	Lasteingabe bei Fliesslinien
<b>ynod</b>	Knoteneingabe bei Generierung CYLT

## 2.3 Overview on actions

Aufgabe	Macro
Anwender Eingabe Macro	<code>mesn</code>
Basis lokal: Knoten	<code>angl</code>
Basis lokal: Linie	<code>eang</code>
Basis lokal: Region	<code>vang</code>
Datenausgabe: ein	<code>prin</code>
Datenausgabe: aus	<code>nopr</code>
Dateneingabe: allgemeine Ausdrücke	<code>pars</code>
Dateneingabe: nur Zahlen	<code>nopa</code>
Debug - Option	<code>debu</code>
Definition von Konstanten	<code>para</code>
Definition von Konstanten bei Fliesslinien	<code>ycon</code>
Definition von Konstanten bei Generierung	<code>neco</code>
Definition von Kurven	<code>curve</code>
Dreibeinberechnung	<code>base</code>
Elemente	<code>elem</code>
Elemente, freie Numerierung	<code>elfr</code>
Elemente bei Generierung GMESH	<code>gele</code>
Elemente fuer 3D - Stab	<code>el3b</code>
Elementlasten	<code>qloa</code>
Ende der Berechnung/ letzte Karte Eingabe	<code>stop</code>
Ende Eingabedaten Mesh	<code>end</code>
Geometrieparameter f. FEAP - ISOGEO	<code>nmpq</code>
Gleichungslöser	<code>solv</code>
Hintergrundnetz bei Generierung	<code>back</code>
Imperfektionen	<code>impf</code>
Interaktive Eingabe der Macros	<code>inte</code>
$J_2$ -Integraleingabe	<code>jint</code>
Kantengenerierung bei Fliesslinien	<code>yedg</code>
Knoteneingabe bei Fliesslinien	<code>ynod</code>
Knoten+Elemente: 2D+3D Generierung	<code>bloc</code>
Knoten+Elemente: 2D Generierung	<code>blox</code>
Knoten+Elemente: 2D Interface Zone	<code>blco</code>
Knotenvektoren $\Xi^1$ für FEAP - ISOGEO	<code>knv1</code>
Knotenvektoren $\Xi^2$ für FEAP - ISOGEO	<code>knv2</code>
Koordinaten bei Generierung GMESH	<code>geor</code>
Knoten+Elemente: 2D+3D Generierung Dreiecke	<code>trib</code>
Knotenlasten	<code>load</code>
Knotenlasten (ständig)	<code>loa0</code>
Knotenlasten an einem Punkt	<code>poin</code>
Knotennummernoptimierung	<code>opti</code>
Knotennummernoptimierung	<code>optn</code>
Kontaktdateneingabe	<code>icon</code>
Koordinaten	<code>coor</code>

Aufgabe	Macro
Koordinaten der Segmente bei Generierung Koordinaten: Imperfektionen Koordinaten: kugel → kart. Koordinaten: polar → kart. Koordinaten: Rotation um Achse Koordinaten: Translation Koordinaten: spezielle Modifikationen Koordinaten: Zufallsveränderung	geom icor sphe pola rot tran cmod rndm
Lasteingabe bei Fliesslinien Lasten auf Gebieten Linienlasten Linienlasten auf Segmenten bei Generierung	yloa aloa eloa pres
Macro - Modus Materialdaten	macr mate
Oberflächen- bzw. Folgelasten	sloa
Randbedingungen an Knoten Randbedingungen an einem Punkt Randbedingungen an Segmenten bei Generierung Randbedingungen bei Fliesslinien Randbedingungen fuer eine feste Koord. Randbedingungen fuer eine Linie Randbedingungen in einer Region Randbedingungen in einem Zylinderbereich Randbedingungen: Werte fuer eine feste Koord. Reaktionskräfte bei mehreren Knoten Regionen bei Generierung	boun poin fixe ybou ebou edge vbou rbou edis rsum regi
Segmente bei Generierung	segm
Temperaturen Temperaturen: 2D+ 3D Generierung Titelzeile + Generierungstyp bei Gen. CYLT Titelzeile + Problemgrössen Titelzeile + Problemgroessen bei Gen. GMESH Titelzeile + Problemgroessen bei Gen. NEGE	temp btem cylt feap gmesh nege
Unterteilung bei Generierung GMESH	ndvi
Verbindung von Freiheitsgraden Verbindung von Knoten mit gl. Koordinaten Verzerrungs'lasten' Verschneidungen	link tie epsq isec

## 2.4 Aufbau FEAP–Eingabefile

<b>Feap</b> oder <b>Gmesh</b> oder <b>Nege</b>	Kopfzeile
↑ <b>MESH</b> –Macros ↓ <b>end</b>	
< <b>opti</b> > < <b>tie</b> >	Eingabe, falls nötig
< <b>macr</b> > ↑ <b>MACRO</b> –Macros ↓ <b>end</b>	Optional: Eingabe von Macro-Befehlen, die im Batch-Modus automatisch abgearbeitet werden
↑ Eingabewerte für <b>MACRO</b> –Macros in entsprechender Reihenfolge ↓	zugehörige Eingabedaten <b>genaue Zeilenanzahl erforderlich!!</b>
<b>inte</b>	Optional: interaktive Eingabe
<b>stop</b>	Ende Eingabe

## 2.5 Startoptionen für FEAP

### 2.5.1 Ausführung als Windowsprogramm

Ausführung rein interaktiv mit **inte**

oder

zunächst mit Macro-Befehlen **macr** und dann interaktiv mit **inte**

## 2.6 Anwendung im reinen Batchbetrieb

Start mit

```
feap -iIfile <-oOfile -rRfile -sSfile -pPfile> -dId
```

und reinen Macro-Befehlen **macr**. Dazu im Inputfile **nopr** und bei den Macrobefehlen **nopr** verwenden.

Ifile = Eingabefile etc. Pfade sind hinzuzufügen.

Idev = 3 INTEL, Idev = 4 SALFORD.

## 2.7 Macros in detail

### FEAP

---

**feap** [ title of problem for printouts, etc.]  
numnp,numel,nummat,ndm,ndf,nen,nad,ndd

---

Each problem to be solved by FEAP must start with a single record which contains the characters **feap** as the first entry; the remainder of the record (columns 5–80) may be used to specify a problem title. The title will be printed with each “page” of output as the first line.

Immediately following the **feap** record, the ‘control’ information describing characteristics of the problem to be solved must be given. The ‘control’ information describes the characteristics of the finite element problem to be solved. The data entries are:

numnp	— total number of nodal points in the problem.
numel	— total number of elements in the problem.
nummat	— number of material property sets in the problem.
ndm	— number of spatial coordinates needed to define mesh.
ndf	— maximum number of degrees-of-freedom on any node.
nen	— maximum number of nodes on any element.
nad	— increases size of element arrays to ndf*nen + nad
ndd	— maximum number of parameters for element properties (default 50)

The number of spatial coordinates needed to define the finite element mesh (ndm) must be 1, 2, or 3. The maximum number of degrees-of-freedom on any node must be between 1 and 6. The maximum number of the other quantities is limited only by the size of the dynamically dimensioned array used to store all the data and solution parameters. This is generally quite large and, normally, should not be exceeded. If the error message that memory is exceeded appears the data should be checked to make sure that no errors exist which could cause large amounts of memory to solve the problem (e.g., if the error occurs when the **tang**, **utan**, or **cmas** solution macro statements are encountered, the profile of the matrix should be checked for very large column heights — appropriate renumbering of the mesh can often significantly reduce the storage required). If necessary, the main subprogram — ‘program feap’ can be recompiled with a larger dimension for the array m(\*) in blank common.

For simplicity of input files it is allowed to start **FEAP** without defining the number of nodes ('numnp'), the number of elements ('numel'), and the number of materials('nummat'). In this case **FEAP** counts these values automatically. Results for 'numnp', 'numel' and 'nummat' are shown in the output file. For the correct calculation of 'nummat' it is necessary to define each material with a separate **mate**-card. Otherwise 'nummat' is standard input.

**Usage in FEAP - ISOGEOM:**

In FEAP - ISOGEOM input files 'numnp' and 'numel' have to be given in all cases. 'numnp' is calculated by  $n \times m$  for every patch and then summed up, 'numel' is determined by the sum of  $(n - p) \times (m - q)$  of all patches. 'ndm' has to be set to 4 as control points are given in 4D-projective coordinates. For 2D cases the z-coordinate should be set to zero. 'nen' is the maximum number of control points having influence on one element. So all variables are chosen equivalent to the Finite Element Analysis counterparts.

**ALOA**


---

**aloa**  
 xmin,xmax,ymin,ymax,<zmin,zmax>  
 idof,etyp  
 qtyp,q1,q2,...  
 <etc., terminate with a blank record>

---

With **aloa** an external load **q** is set on all elements between xmin/ ymin,<zmin> xmax/ymax<zmax>. An element will be loaded for degree of freedom **idof**, if the center of the element is within these bounds. **etyp** defines the element type. Elements should be defined isoparametrically.

etyp	element
4	4-node plate/shell element
9	9-node plate/shell element
8	8-node brick element, loaded on top ( $z = +\frac{\ell_z}{2}$ )
-8	8-node brick element, loaded on bottom ( $z = -\frac{\ell_z}{2}$ )
27	27-node brick element, loaded on top ( $z = +\frac{\ell_z}{2}$ )
-27	27-node brick element, loaded on bottom ( $z = -\frac{\ell_z}{2}$ )

**qtyp** defines the type of loading whereas  $q_1, q_2, \dots$  are the associated load terms. Currently the following load types are implemented:

qtyp	$q_1$	$q_2$	$q_3$	$q_4$	$q_5$	Name
1	$q_0$					constant load $q_0$
2	$q_0$	$\ell_x$	$\ell_y$			$q = q_0 \cdot \cos(\frac{\pi}{2l_x}x) \cdot \cos(\frac{\pi}{2l_y}y)$

**Remarks**

1. For the Cos–Load case the load  $q_0$  is the maximum value at (0,0),  $\ell_x$  is the quadrant length of plate in x-direction and  $\ell_y$  the quadrant length of plate in y-direction.
2. Macro could not be used with more than one element types.
3. Brick elements only along with plates in x-y-plane.
4. **aloa** and **reac** lead to wrong results for associated dofs at boundaries.

## ANGL

---

### angl

angl1, angl2,itrot  
node1, ngen1,  $\alpha(\text{node1})$   
node2, ngen2,  $\alpha(\text{node2})$   
<etc., terminate with blank record>

---

The **angl** command is used to specify the angles for sloping nodal boundary conditions in two directions.

angl1 : first axis of rotated basis (default=1)
angl2 : second axis of rotated basis (default=2)
itrot : 0/1 with/without rot. dofs 4-6 (default=0)

For each node to be specified a record is entered with the following information:

node : the number of the node to be specified.
ngen : the increment to the next node, if generation is used, otherwise 0.
$\alpha(\text{node})$ : value of angle in degrees that new 1-coord. makes with $x(1,\text{node})$ .

When generation is performed, the node number sequence will be (for node1–node2 sequence shown above):

node1, node1+ngen1, node1+2\*ngen1, ..., , node2

The values for each angle generated will be a linear interpolation between  $\alpha(\text{node1})$  and  $\alpha(\text{node2})$ .

The degree-of-freedoms associated with the sloping boundary may differ from element to element as described in the 'ELMTnn' manuals.

The angl is positive from first axis to second axis which means counter clockwise.

### Remarks:

1. Only one choice of directions angl1, angl2 and itrot is possible.
2. A transformation will be done for dofs angl1 and angl2.
3. In case of ndf=6 the same transformation is chosen for dofs angl1+3 and angl2+3 for itrot=0.
4. The associated dofs of the used element can be chosen via the macro **mate**, parameters idfg. As example the angles of the DKQ-plate element can be transformed via  
**mate**  
1,7,2,3,1
5. Results are printed and plotted for nodes with  $\text{angl} \neq 0$  in directions angl1,angl2. Thus mixed vectors occur.
6. Results which base on a smoothing procedure are plotted in global directions. Thus results for nodes with  $\text{angl} \neq 0$  are transformed to global directions. Ex.: **disp**, **resi**, **eigv**,...

**BACK****back**

```
nelm1, (ix(i,nelm1),i=1,3)
nelm2, (ix(i,nelm2),i=1,3)
<etc., terminate with blank record>
node1, (x(i,node1),i=1,2), (d(i,node1),i=1,4)
node2, (x(i,node2),i=1,2), (d(i,node1),i=1,4)
<etc., terminate with blank record>
```

---

The **back** command is used to specify a background mesh of triangular elements.

nelm	– the number of the element to be specified.
ix(1,nelm)	– node-1 number attached to element.
ix(2,nelm)	– node-2 number attached to element.
ix(3nelm)	– node-3 number attached to element.

node	– the number of the node to be specified.
x(1,node)	– value of coordinate in 1-direction.
x(2,node)	– value of coordinate in 2-direction.
d(1,node)	– required element size.
d(2,node)	– required stretching.
d(3,node)	– 1-value of direction cosine of stretching direction.
d(4,node)	– 2-value of direction cosine of stretching direction.

**Remarks:**

1. A background mesh with triangular elements will be defined to monitor the element density of the mesh. If **back** is used the parameter 'esiz' (used in **nege**) will not be used.
2. For mesh generation with NEGE the following order of statements is necessary: **nege**, **[neco]**, **[back]**, **geom**, **segm**, **regi**, **[pres]**, **[fixe]**, further FEAP-macros. Note: macros in brackets are optional.

**BASE****base**

n1,x2,x3,x4

&lt;terminate with blank record&gt;

Calculate the nodal base vectors at all nodes due to the chosen type 'n1'. For some fields the additional value 'x2-x4' are necessary. Actually the following base definitions are possible:

n1	free form for each global node	x2	x3	x4
0	basis for each global node + averaging process, see element description			
n1	free form for each element node	x2	x3	x4
1	basis for each element node, see element description			
2	normal vector $e_3$ for each element node, see element description, arbitrary vector $t_1$ input, vectors $e_1 = t_1 / \ t_1\ $ , $e_2 = e_3 \times e_1$ calculated	$t_{1x}$	$t_{2x}$	$t_{3x}$
n1	analytical solution for each global node	x2	x3	x4
3	plate x-y plane $e_1 = e_x, e_2 = e_y, e_3 = e_z$			
4	plate y-z plane $e_1 = e_y, e_2 = e_z, e_3 = e_x$			
5	plate x-z plane $e_1 = e_x, e_2 = e_z, e_3 = -e_y$			
6	plate x-y plane polar			
7				
8	cylinder(x=axis) $y = y_0 + y, z = z_0 + z$	$y_0$	$z_0$	
9	cylinder(y=axis) $x = x_0 + x, z = z_0 + z$	$x_0$	$z_0$	
10	cylinder(z=axis) $x = x_0 + x, y = y_0 + y$ , outside	$x_0$	$y_0$	
11	cylinder(z=axis) $x = x_0 + x, y = y_0 + y$ , inside	$x_0$	$y_0$	
12	sphere $x = x_0 + x, y = y_0 + y, z = z_0 + z$	$x_0$	$y_0$	$z_0$
13	hypar $z = (4l_z/(l_x l_y)) xy$	$l_x$	$l_y$	$l_z$
14	rot. hyperboloid $r = r_0/c \sqrt{c^2 + z^2}$	$r_0$	$c$	
15	rot. paraboloid $r = r_0 \sqrt{2} z$	$r_0$		
16	parabola $z = a x^2$	$a$		
17	twisted beam	$a$	=	
		$l_x$		

For these geometries the following base vectors are calculated:

	System	Base vectors		
		$\mathbf{t}_1$	$\mathbf{t}_2$	$\mathbf{t}_3$
3	plate x-y plane	[1, 0, 0]	[0, 1, 0]	[0, 0, 1]
4	plate y-z plane	[0, 1, 0]	[0, 0, 1]	[1, 0, 0]
5	plate x-z plane	[1, 0, 0]	[0, 0, 1]	[0, -1, 0]
6	plate x-y plane polar	$[\cos \varphi, \sin \varphi, 0]$	$[-\sin \varphi, \cos \varphi, 0]$	[0, 0, 1]
8	cylinder (x-axis) $1 = x, 2 = -\varphi, 3 = r$	[1, 0, 0]	[0, $\sin \varphi, -\cos \varphi$ ]	[0, $\cos \varphi, \sin \varphi$ ]
9	cylinder (y-axis) $1 = -\varphi, 2 = y, 3 = r$	$[\sin \varphi, 0, -\cos \varphi]$	[0, 1, 0]	$[\cos \varphi, 0, \sin \varphi]$
10	cylinder (z-axis) outside $1 = \varphi, 2 = z, 3 = r$	$[-\sin \varphi, \cos \varphi, 0]$	[0, 0, 1]	$[\cos \varphi, \sin \varphi, 0]$
11	cylinder (z-axis) inside $1 = \varphi, 2 = z, 3 = r$	$[\sin \varphi, -\cos \varphi, 0]$	[0, 0, 1]	$[-\cos \varphi, -\sin \varphi, 0]$
12	sphere (z-axis) $1 = \theta, 2 = \varphi, 3 = r$	$\begin{bmatrix} -\sin \varphi \\ \cos \varphi \\ 0 \end{bmatrix}$	$\begin{bmatrix} -\cos \theta & \cos \varphi \\ -\cos \theta & \sin \varphi \\ \sin \theta \end{bmatrix}$	$\begin{bmatrix} \sin \theta & \cos \varphi \\ \sin \theta & \sin \varphi \\ \cos \theta \end{bmatrix}$
13	hypar shell $z = axy$ $a = 4l_z/(l_x l_y)$	$[1, 0, ay]/t$ $t = \sqrt{1 + ax^2}$	$[0, 1, ax]/t$ $t = \sqrt{1 + ay^2}$	$[-ay, -ax, 1]/t$ $t = \sqrt{1 + a(x^2 + y^2)}$
		corrected by lamina base		
14	rot. hyperboloid $(t = \sqrt{1 + r_{,z}^2})$	$\begin{bmatrix} -\sin \varphi \\ \cos \varphi \\ 0 \end{bmatrix}$	$1/t \begin{bmatrix} r_{,z} & \cos \varphi \\ r_{,z} & \sin \varphi \\ 1 \end{bmatrix}$	$1/t \begin{bmatrix} \cos \varphi \\ \sin \varphi \\ -r_{,z} \end{bmatrix}$
15	rot. paraboloid $(t = \sqrt{1 + r_{,z}^2})$	$\begin{bmatrix} -\sin \varphi \\ \cos \varphi \\ 0 \end{bmatrix}$	$1/t \begin{bmatrix} r_{,z} & \cos \varphi \\ r_{,z} & \sin \varphi \\ 1 \end{bmatrix}$	$1/t \begin{bmatrix} \cos \varphi \\ \sin \varphi \\ -r_{,z} \end{bmatrix}$
16	parabola $(t = \sqrt{1 + 2ax^2})$	$1/t \begin{bmatrix} 1 \\ 0 \\ 2ax \end{bmatrix}$	$\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$	$1/t \begin{bmatrix} -2ax \\ 0 \\ 1 \end{bmatrix}$
17	twisted beam $(t = \sqrt{1 + \pi^2 r^2 / (4a^2)})$	$1/t \begin{bmatrix} 1 \\ -r\pi/a \sin \varphi \\ r\pi/a \cos \varphi \end{bmatrix}$	$\begin{bmatrix} 0 \\ \cos \varphi \\ \sin \varphi \end{bmatrix}$	$\mathbf{t}_3 = \mathbf{t}_1 \times \mathbf{t}_2$

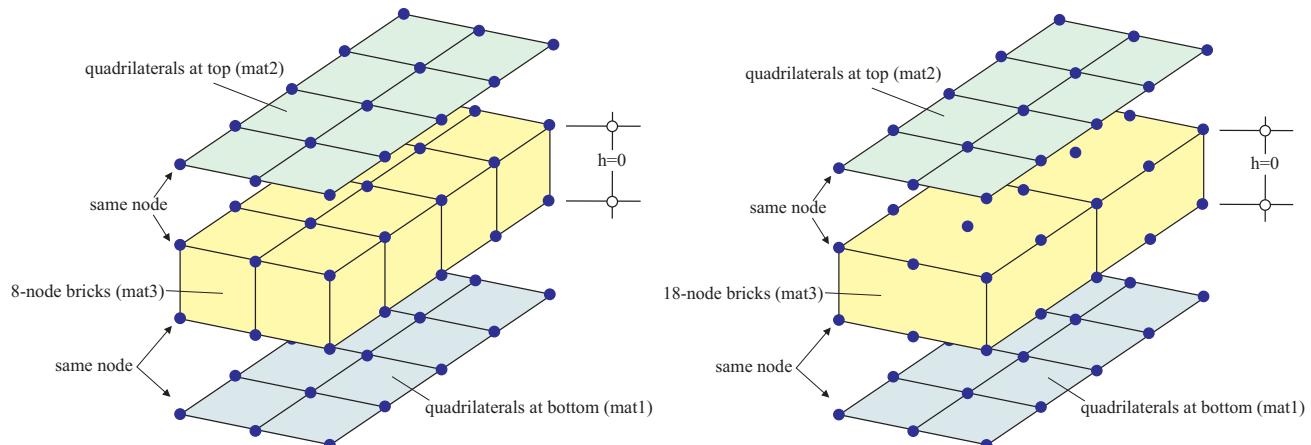
The nodal base vectors can be plotted within **plot** state under the macro **base**.

**BLCO****blco**

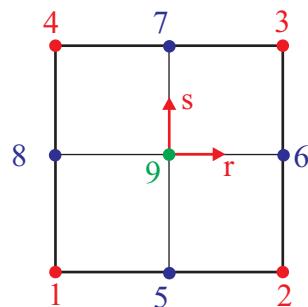
nodes,r-inc,s-inc,node1,elmt1-,mat1,mat2,mat3,<mtyp>  
 1, x1, y1, z1 (only ndm coordinates required)  
 2, x2, y2, z2  
 etc., until all 'nodes' records are input.

The **blco** command is used to generate a mesh for an interface zone and is very similar to the **bloc** macro.

The macro generates at first a 2D-mesh for quadrilaterals which are 'at the bottom' of the structure. In a second step a similar mesh is generated 'at the top' of the structure. These meshes have different nodes, the nodes themselves have the same coordinates. In between a third mesh of 8-node trilinear elements (mtyp=1,default) of 18-node biquadratic(Lagrange)/linear elements (mtyp=2) or of 16-node biquadratic(Serendipity)/linear elements (mtyp=3) of thickness  $h=0$  is generated at the end.



The patch of nodes/elements defined by **blco** is developed from a master element which is defined by an isoparametric 4-9 node mapping function in terms of the natural coordinates  $r$  and  $s$ . The node numbers on the master element of each patch defined by **blco** are specified according to the following figure. The four corner nodes of the master element must be specified, the mid-point and central nodes are optional.



The spacing between the  $r$ -increments and  $s$ -increments may be varied by a proper specification of the mid-side and central nodes. Thus, it is possible to 'concentrate' nodes/elements into one corner of the patch generated by **blco**. The mid-nodes must lie within the central-half of the  $r$ - or  $s$ -directions to keep the isoparametric mapping single valued for all  $(r, s)$  points.

The data parameters are defined as:

nodes	— number of master nodes needed to define the patch.
r-inc	— number of nodal increments to be generated along r-direction of the patch.
s-inc	— number of nodal increments to be generated along s-direction of the patch.
node1	— number to be assigned to first generated node in patch
elmt1	— number to be assigned to first element generated in patch
mat1	— material number to be assigned to generated quadrilaterals at the bottom
mat2	— material number to be assigned to generated quadrilaterals at the top
mat3	— material number to be assigned to generated 8 node interface elements
mtyp	— type of interface elements: mtyp=1: 8-node, mtyp=2: 18-node, mtyp=3: 16-node

r-inc, s-inc must be even numbers for mtyp=2,3!

**BLOC****bloc**

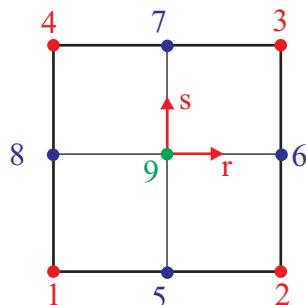
nodes,r-inc,s-inc,node1,[elmt1-,mat,r-skip,b-type]  
 1, x1, y1, z1 (only ndm coordinates required)  
 2, x2, y2, z2  
 etc., until all 'nodes' records are input.

The **bloc** command acts on 2-D meshes for  $0 < btype < 9$  or  $btype = 16$ , see A and on 3-D meshes for  $10 < btype < 19$ , except  $btype = 16$ , see B.

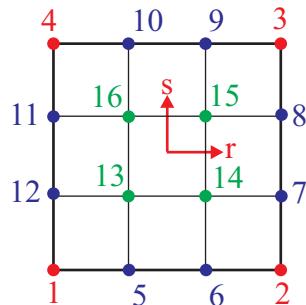
**A: 2-D mesh generation**

The **bloc** data input segment is used to generate a regular one or two dimensional patch of nodes. Alternatively, nodes together with 4-node quadrilateral elements may be generated for two dimensional patches or three dimensional surfaces.

The patch of nodes/elements defined by **bloc** is developed from a master element which is defined by an isoparametric 4-9 node mapping function in terms of the natural coordinates  $r$  and  $s$ . The node numbers on the master element of each patch defined by **bloc** are specified according to the following figure. The four corner nodes of the master element must be specified, the mid-point and central nodes are optional.



The spacing between the  $r$ -increments and  $s$ -increments may be varied by a proper specification of the mid-side and central nodes. Thus, it is possible to 'concentrate' nodes/elements into one corner of the patch generated by **bloc**. The mid-nodes must lie within the central-half of the  $r$ - or  $s$ -directions to keep the isoparametric mapping single valued for all  $(r, s)$  points.



Patches may be interconnected, in a restricted manner, by using the "r-skip" parameter judiciously. In addition, the **tie** macro may be used to 'connect' any nodes which have the same coordinates.

The data parameters are defined as:

nodes	– number of master nodes needed to define the patch.
r-inc	– number of nodal increments to be generated along r-direction of the patch.
s-inc	– number of nodal increments to be generated along s-direction of the patch.
node1	– number to be assigned to first generated node in patch (default = 1). First node is located at same location as master node 1. Last generated node (i.e., node1 - 1 + (r-inc * s-inc)) is located at same location as master node 3.
elmt1	– number to be assigned to first element generated in patch; if zero no elements are generated (default = 0)
matl	– material number to be assigned to all generated elements in patch (default = 1)
r-skip	– number of nodes to skip between end of an r-line and start of next r-line (may be used to interconnect blocks side-by-side) (default = 1)
b-type	<ul style="list-style-type: none"> <li>= 0 – nodes and 4-node elements on patch; sequentially along consecutive r-lines</li> <li>= 1 – 3-node triangles (diagonals in 1–3 direction of block)</li> <li>= 2 – 3-node triangles (diagonals in 2–4 direction of block)</li> <li>= 3 – 3-node triangles (diagonals alternate 1–3 then 2–4)</li> <li>= 4 – 3-node triangles (diagonals alternate 2–4 then 1–3)</li> <li>= 5 – 3-node triangles (diagonals in union-jack pattern)</li> <li>= 6 – 3-node triangles (diagonals in inverse union-jack pattern)</li> <li>= 7 – 6-node triangles (diagonals in 1–3 direction of block)</li> <li>= 8 – 8-node quadrilaterals (N.B. Interior node generated but not used, rinc and sinc must be even numbers).</li> <li>= 9 – 9-node quadrilaterals (rinc and sinc must be even numbers).</li> <li>= 16 – 16-node quadrilaterals (rinc and sinc must be multiples of three).</li> </ul>

## B: 3–D mesh generation

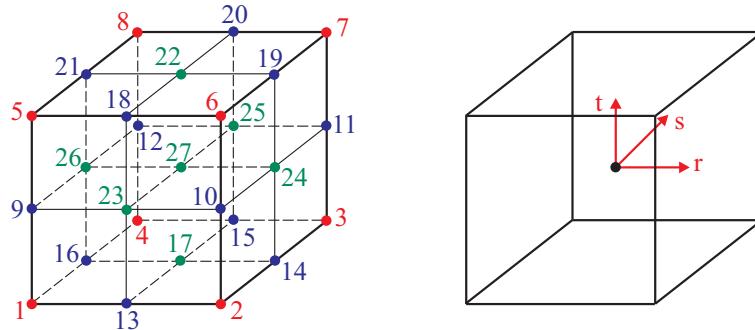
---

```
bloc  
nodes,r-inc,s-inc,t-inc, node1,elmt1,mat,b-type  
1, x1, y1, z1  
2, x2, y2, z2  
etc., until all 'nodes' records are input.
```

---

The **bloc** data input segment is used here to generate a regular 3 dimensional patch of nodes for 8/20/27/64-node 3–D brick elements and 4–node tetrahedron elements. The 20 node brick element can be used with nen=20 or nen=21. In the latter case an additional unused node is generated. Don't use this version together with the sparse matrix solver 2 (?!).

The patch of nodes/elements defined by **bloc** is developed from a master element which is defined by an isoparametric 8/27 node mapping function in terms of the natural coordinates  $r$ ,  $s$  and  $t$ . The node numbers on the master element of each patch defined by **bloc** are specified according to the following figure



The eight corner nodes of the master element at ( $t = \pm 1$ ) must be specified, the mid-point nodes and the central nodes are optional.

The spacing between the  $r$ ,  $s$ ,  $t$ -increments may be varied by a proper specification of the mid-side and central nodes. Thus, it is possible to 'concentrate' nodes/elements into one corner of the patch generated by **bloc**. The mid-nodes must lie within the central-half of the  $r$ -,  $s$ -,  $t$ -directions to keep the isoparametric mapping single valued for all  $(r, s, t)$  points.

Patches may be interconnected by the **tie** command to connect any nodes which have the same coordinates.

**Bug:** - combination of nodes 9-12/13-16/18-21 with 23-26 leads to errors.

**Bug:** - Hide does not work for tetrahedron.

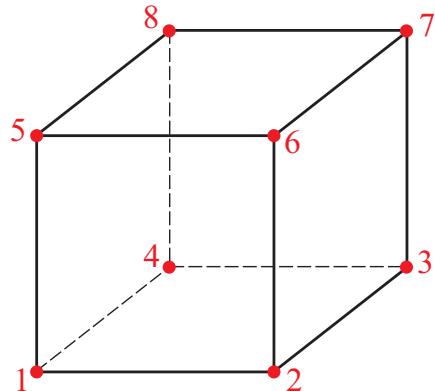
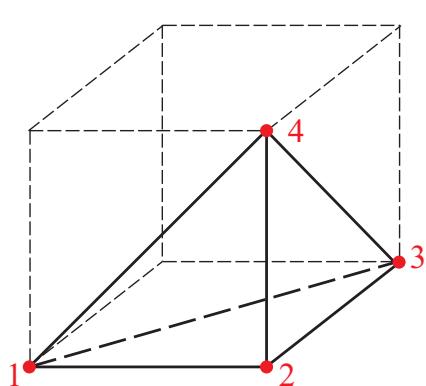
**b-type = 10,12,13,14,15,19**

This option generates 8-/20-/21-/27- and 64- node brick elements.

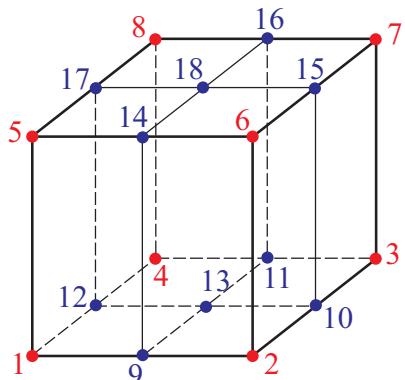
The data parameters are defined as:

nodes	— number of master nodes needed to define the patch.
r-inc	— number of nodal increments to be generated along <i>r</i> -direction of the patch.
s-inc	— number of nodal increments to be generated along <i>s</i> -direction of the patch.
t-inc	— number of nodal increments to be generated along <i>t</i> -direction of the patch.
node1	— number to be assigned to first generated node in patch (default = 1). First node is located at same location as master node 1. Last generated node (i.e., node1 - 1 + (r-inc * s-inc * t-inc)) is located at same location as master node 3.
elmt1	— number to be assigned to first element generated in patch; if zero no elements are generated (default = 0)
matl	— material number to be assigned to all generated elements in patch (default = 1)
b-type = 10	— 8-node brick elements
= 12	— 20-node brick elements (Interior nodes generated but not used, nen=21 but element node 17 not used! ), rinc, sinc and tinc must be even numbers
= 13	— 27-node brick elements, rinc, sinc and tinc must be even numbers
= 14	— 20-node brick elements (continuous node numbering, interior nodes generated but not used, rinc, sinc and tinc must be even numbers
= 15	— 18-node brick elements (biquadratic/linear, continuous node numbering, rinc, sinc must be even numbers, tinc=2
= 19	— 64-node brick elements, rinc, sinc and tinc must be multiples of three

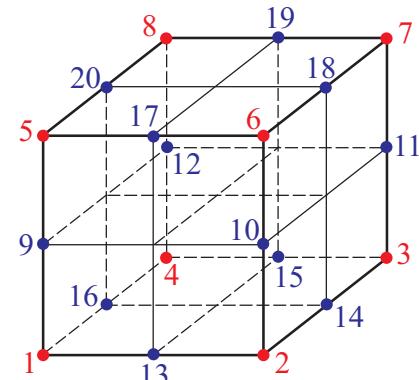
The positions of the local nodes of the generated elements are illustrated in the following figures.



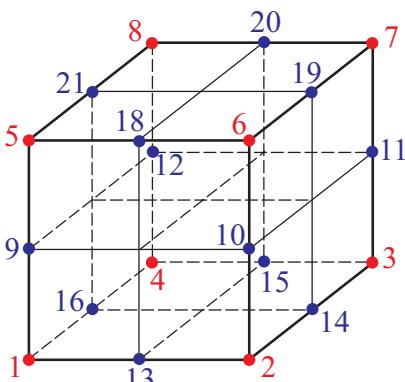
4-nodes



8-nodes

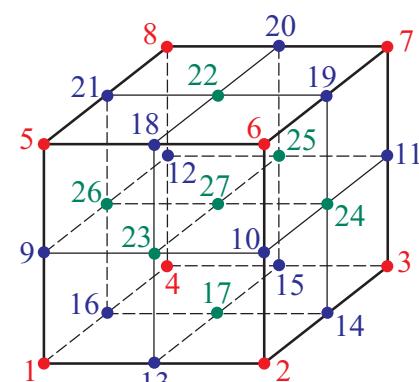


18-nodes

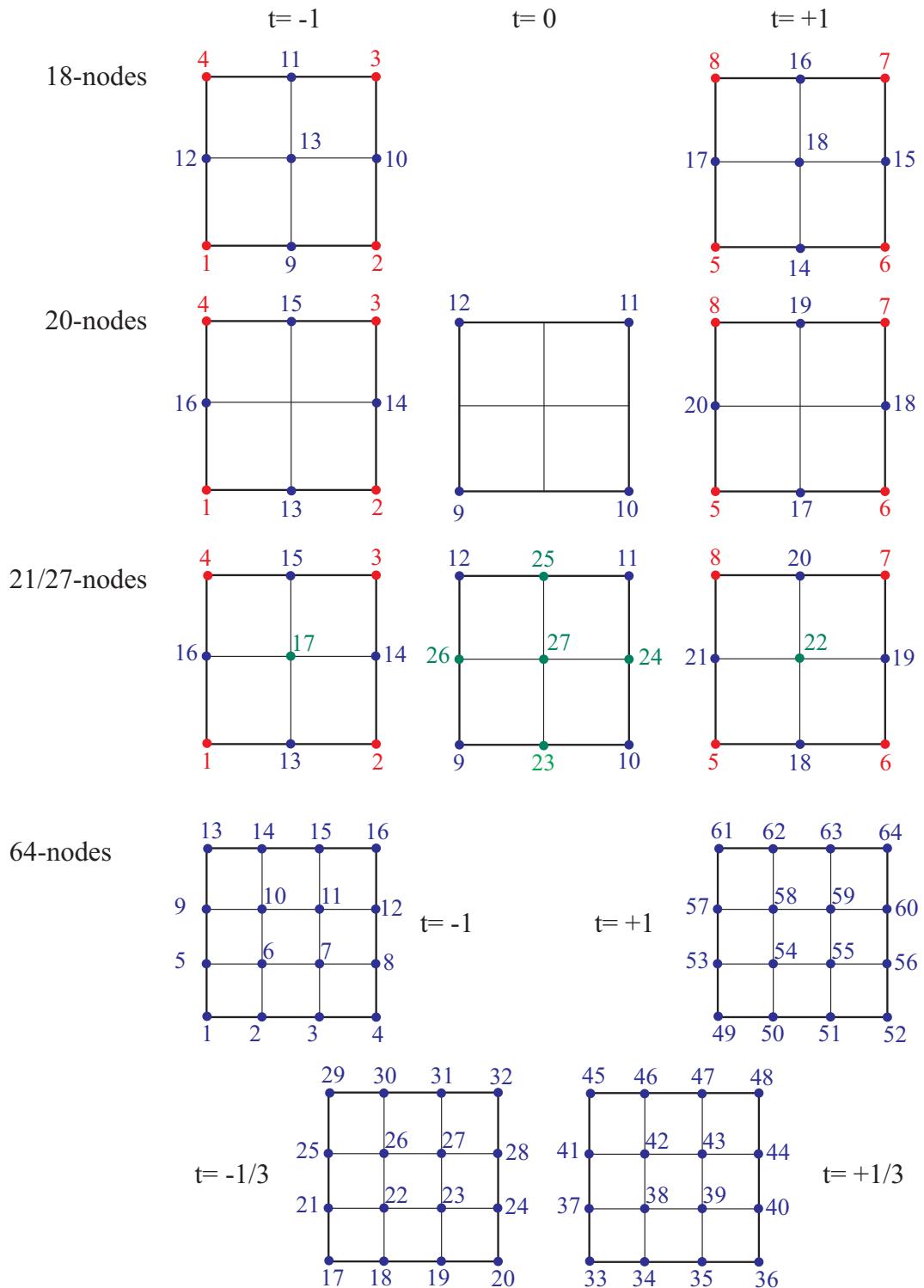


21-nodes (node 17 not used)

20-nodes



27-nodes



**b-type = 11**

This option generates 4-node tetrahedron elements by sub-dividing one brick element into 6 tetrahedrons. Thus one has to specify six-times the number of elements ( $r\text{-inc} * s\text{-inc} * t\text{-inc} * 6$ ). The data parameters are defined as:

nodes	— number of master nodes needed to define the patch.
r-inc	— number of nodal increments to be generated along $r$ -direction of the patch.
s-inc	— number of nodal increments to be generated along $s$ -direction of the patch.
t-inc	— number of nodal increments to be generated along $t$ -direction of the patch.
node1	— number to be assigned to first generated node in patch (default = 1). First node is located at same location as master node 1. Last generated node (i.e., $\text{node1} - 1 + (\text{r-inc} * \text{s-inc} * \text{t-inc})$ ) is located at same location as master node 3.
elmt1	— number to be assigned to first element generated in patch; if zero no elements are generated (default = 0)
matl	— material number to be assigned to all generated elements in patch (default = 1)
r-skip	— number of nodes to skip between end of an $r$ -line and start of next $r$ -line (may be used to interconnect blocks side-by-side) (default = 1)
b-type	— type of patch to be generated = 11

## BLOX

---

### **blox**

```
nodes,r-inc,s-inc,node1,[elmt1-,mat,r-skip,b-type]
1, x1, y1, z1 (only ndm coordinates required)
2, x2, y2, z2
etc., until 4
dx1,dx2,dx3, ... r-times (16 per line!)
dy1,dy2,dy3, ... s-times
mx1,mx2,mx3, ... r-times
my1,my2,my3, ... s-times
```

---

The **blox** data input segment is used to generate a nonregular 2D patch of nodes on a rectangular(!) mesh similar to **bloc**.

The patch of nodes/elements defined by **blox** is developed from a master element which is defined by an 4 node mapping function. The node numbers on the master element of each patch defined as in **bloc**.

The data parameters are defined as:

nodes	— number of master nodes needed to define the patch.
r-inc	— number of nodal increments to be generated along r-direction.
s-inc	— number of nodal increments to be generated along s-direction.
node1	— number to be assigned to first generated node.
elmt1	— number to be assigned to first element generated in patch.
matl	— dummy input
r-skip	— number of nodes to skip between end of an r-line and start of next r-line (default = 1)
b-type	— like <b>bloc</b>
dx(i=1,r-inc)	— increments in x-direction
dy(i=1,s-inc)	— increments in y-direction
mx(i=1,r-inc)	— material number for element dx(i)
my(i=1,s-inc)	— material number for element dy(i)
The maximum of mx(i),my(i) will be used for element dx(i),dy(i).	

## BOUN

---

### boun

```
node1, ngen1, (id(i,node1),i=1,ndf)
node2, ngen2, (id(i,node2),i=1,ndf)
<etc., terminate with blank record>
```

---

The **boun** command is used to specify the values for the boundary restraint conditions. For each node to be specified a record is entered with the following information:

node	– the number of the node to be specified
ngen	– the increment to the next node, if generation is used, otherwise 0.
id(1,node)	– value of boundary restraint for 1-dof for 'node'
id(2,node)	– value of boundary restraint for 2-dof for 'node'.
	etc. until to 'ndf' directions

The boundary restraint codes are interpreted as follows:

- $\text{id}(i,\text{node}) = 0$  i-dof has unknown displacement,  $f(i,\text{node})$  will be an applied load.
- $\text{id}(i,\text{node}) \neq 0$  i-dof has known displacement,  $f(i,\text{node})$  is the displacement value (default = 0.).

When generation is performed, the node number sequence will be (for node1-node2 sequence shown above):

node1, node1+ngen1, node1+2\*ngen1, ..., , node2

The values for each boundary restraint will be as follows:

- $\text{id}(i,\text{node}1) = 0$  or positive  $\rightarrow \text{id}(i,\text{node}1+\text{ngen}1) = 0$
- $\text{id}(i,\text{node}1) = \text{negative}$   $\rightarrow \text{id}(i,\text{node}1+\text{ngen}1) = -1$

With this convention the value of 'id(i,node2)' will be negative whenever the value of id(i,node1) starts negative. Accordingly, it is necessary to assign a positive value for the restraint code to terminate a generation sequence (e.g., when it is no longer desired to set a dof to be restrained). Alternatively, an 'i-dof' may be eliminated for all nodes by using the generation sequence:

	Degree of Freedom			
node	1	...	i	...
1,	1,	...	-1,	...
numnp,	0,	...	, +1,	...

Subsequent records may then be used to assign the values to the other degree-of-freedoms.

### Remarks:

1. Up to 16 values are possible on each input record.
2. **boun** sets the boundary conditions whereas **ebou** adds boundary conditions to existing values.

For the specification of prescribed displacements within a calculation, see macro command **disp**.

## BTEM

---

### btem

```
nodes,r-inc,s-inc,t-inc,node1,[r-skip]  
1, temp1  
2, temp2  
etc., until all 'nodes' records are input.
```

---

The **btem** data input segment is used to generate temperatures for a regular two or three dimensional patch of nodes.

The patch of nodes/elements defined by **btem** is developed from a master element which is defined by an isoparametric 4–9 node mapping function in terms of the natural coordinates  $r$ ,  $s$  and  $t$ . The node numbers on the master element of each patch defined by **btem** are specified according to Fig. 24.25 on page 748 of THE FINITE ELEMENT METHOD, 3RD. ED., by O. C. Zienkiewicz. The corner nodes of the master element must be specified, the mid-point and central node are optional.

The node numbers on the master element of each patch are specified according to the following set up  
**2-D-generation**

Coordinate	corner nodes	mid nodes	center node
$t = 0$	1 2 3 4	5 6 7 8	9

### 3-D-generation

Coordinate	corner nodes	mid nodes	center node
$t = -1$	1 2 3 4	13 14 15 16	17
$t = 0$	9 10 11 12	23 24 25 26	27
$t = +1$	5 6 7 8	18 19 20 21	22

The spacing between the  $r$ ,  $s$  and  $t$ -increments may be varied by a proper specification of the mid-side and central nodes. Thus, it is possible to 'concentrate' nodes/elements into one corner of the patch generated by **btem**. The mid-nodes must lie within the central-half of the  $r$ ,  $s$  or  $t$ -directions to keep the isoparametric mapping single valued for all  $(r, s, t)$  points.

Patches may be interconnected, in a restricted manner, by using the "r-skip" parameter judiciously. In addition, the **tie** macro may be used to 'connect' any nodes which have the same coordinates.

The data parameters are defined as:

nodes	— number of master nodes needed to define the patch.
r-inc	— number of nodal increments to be generated along r-direction of the patch.
s-inc	— number of nodal increments to be generated along s-direction of the patch.
t-inc	— number of nodal increments to be generated along t-direction of the patch (default = 1).
node1	— number to be assigned to first generated node in patch (default = 1). First node is located at same location as master node 1. Last generated node (i.e., node1 - 1 + (r-inc * s-inc)) is located at same location as master node 3.
r-skip	— number of nodes to skip between end of an r-line and start of next r-line (may be used to interconnect blocks side-by-side) (default = 1)
tempi	— temperature at node i

**CMOD****cmod**

type,node1,node2,parameters  
 <terminate with blank record>

The **cmod** command may be used to modify coordinates which have been defined before by **coor**, **bloc** etc. Coordinates from node1 to node2 are treated.

Up to now the following types are possible

Type	Parameter	Result	Formula	Remarks
1	r	sphere	$z = \sqrt{r^2 - x^2 - y^2}$	
2	r,f	parabola	$z = -\frac{f}{r^2} (x^2 + y^2) + f$	
3	a	hypar	$z = a xy$	
4	a,c	hyperboloid	$r = \frac{a}{c} \sqrt{c^2 + z^2}$	polar coor.
5	r,a	hyperboloid	$z = r(\sqrt{1 + x^2/a^2} - 1)$	
6	r,a	sphere	$z = z + \sqrt{r * r - y * y} - a$	
7	a	twisted beam	$y = y \cos \varphi, z = y \sin \varphi, \varphi = \pi/(2a) x$	
8	r,m	sphere-3 blocs	correct x,y,z to real spherical coordinates	use GMESH for meshing 3 blocs with $m \times m$ elements

Be careful with the use of this macro. Due to the modification of the coordinates macros like **eloa**, **edge** or **tie** may not work.

## COOR

### coor

```
node1, ngen1, (x(i,node1),i=1,ndm)
node2, ngen2, (x(i,node2),i=1,ndm)
<etc., terminate with blank record>
```

The **coor** command is used to specify the values for nodal coordinates. For each node to be specified a record is entered with the following information:

node	– the number of the node to be specified
ngen	– the increment to the next node, if generation is used, otherwise 0.
x(1,node)	– value of coordinate in 1-direction for 'node'
x(2,node)	– value of coordinate in 2-direction for 'node', etc. to 'ndm' directions

When generation is performed, the node number sequence will be (for node1–node2 sequence shown above):

```
node1, node1+ngen1, node1+2*ngen1, .... , node2
```

The values for each coordinate will be a linear interpolation between x(node1) and x(node2).

Input of node numbers is not sequential, gaps in numbering are possible.

### Usage in FEAP - ISOGEOM:

The **coor** command works as in standard feap. Control points **B** have to be given in unweighted 4D-projective coordinates ( $x, y, z, w$ ). Generation may not be used, gaps in numbering are not allowed. Input of node numbers is advised to be sequential. The enumeration has to follow a fixed scheme as the sequence of the control points defines the form of the mesh. There is no further possibility to build kind of an 'element-node'-relation. This is fixed in the ordering of the control points. The generation of NURBS is usually done with a 3D-modelling software. Export into an IGES-textfile gives the correct order of control points. These IGES-files have to be converted by a preprocessor to fit the FEAP - ISOGEOM input convention. **coor** has to stand before macro **nmpq**. Not more than 10 digits for the base may be used, the total length of one line may not exceed 80 characters. The exponent can be started with e or d.

<b>Enumeration scheme:</b>
<pre>int=0 for patch=1:n_patch   for j=1:m(patch)     for i=1:n(patch)       int = int + 1       CP(int)=CP(i,j,patch)     end   end end</pre>

## CURV

---

**curv**

n1,n2,n3  
p1,p2,...,p8

---

Description of a curve.

To use the **curve**-option the complete mesh must first be defined. **curv** must be inserted before the **end** command for the mesh definition.

n1 the unique curve number (may be any integer number)  
(the maximum number of curves is 80)

n2 type of the desired curve (see below)

n3 parameter to describe the intention of the curve:  
e.g. mesh refinement and mesh smoothing

The parameters are :

n3=15 curved surface elements

n3=16 curved boundary function for plane surfaces

n3=17 curved boundary function for curved surfaces

n3=18 linear boundary function for curved surfaces

n3=20+ndf functions, which define boundary conditions

n3=30+ndf functions, which define load conditions

p1 to p8 - up to 8 parameters needed to specify the curve (see below)

---

1 to 29 for 2-dimensions

30 to 49 - for 3-dimensions

51 to 99 user defined curves (**SR cvuser**)

n2 = 1 Linear function defined by two points

p1 = x1, p2 = y1, p3 = x2, p4 = y2

n2 = 2 Linear function defined by one point and the slope

p1 = x1, p2 = y1, p3 = slope

n2 = 3 half circle defined by midpoint (x1/y1) and radius

p1 = radius

p2 = +1 or -1 for upper or lower part of the circle

p3 = x1, p4 = y1

n2 = 30 Linear function defined by two points

p1 = x1, p2 = y1, p3 = z1,

p4 = x2, p5 = y2, p6 = z2.

n2 = 31 sphere with radius r and mid-point x1,y1,z1

p1 = radius, p2 = x1, p3 = y1, p4=z1

n2 = 32 plane function defined by  $(x_0 \cdot x + y_0 \cdot y + z_0 \cdot z + c_0) = 0$

p1 = x0, p2 = y0, p3 = z0, p4 = c0

The possible values for n2 are :

## CYLT

---

**cylt** [title of problem for printouts, etc.]  
typ

---

Each problem to be solved by **FEAP** using the yield-line mesh generator **CYLT** must start with a single record which contains the characters **cylt** as the first entry. The remainder of the record (columns 5-80) may be used to specify a problem title. Immediately following the **cylt** record, the 'control' information describing characteristics of the problem to be solved must be given. The 'control' information describes the characteristics of the problem to be solved. The data entries are:

typ – 0 = yield-line mesh  
– 1 = 1st step of triangulation  
– 2 = 2nd step of triangulation  
– etc.

### Remarks:

1. It is recommended to test the data input with 'typ' = 0. The achieved yield-line mesh will not be able to produce calculation results (therefore use 'typ' = 1). Depending on the geometry it is possible that several yield-line mesh solutions are found. The user has to choose the considered mesh number after the generation. The number does not depend on the most likely solution; it may be possible that different numbers content the same mesh, see CYLT-manual.
2. 'typ' > 1 yields higher triangulated meshes which will not be useful for a yield-line calculation performed by FEAP. The possibility of higher triangulation is optional to other programs.
3. A FEAP-input file will be generated called ifeap.tmp (always).
4. For mesh generation with CYLT the following order of statements is necessary: **cylt** , **[ycon]**, **ynod**, **yedg**, further FEAP-macros (including **ybou**).  
Note: macros in brackets are optional.
5. For further information see the **CYLT**-manual.

## **DEBU**

---

**debu**

n1

---

The **debu** macro command can be used to debug the program code.

- n1 = 0 - set the debug option to false (default value)
- n1 = 1 - set the debug option to true

## DISP

---

### disp

```
node1, ngen1, (d(i,node1),i=1,ndf)
node2, ngen2, (d(i,node2),i=1,ndf)
<etc., terminate with blank record>
```

---

The **disp** command is used to specify the values for nodal prescribed displacements (forced conditions). For each node to be specified a record is entered with the following information:

node	– the number of the node to be specified.
ngen	– the increment to the next node, if generation is used, otherwise 0.
d(1,node)	– value of displ for 1-dof for 'node'
d(2,node)	– value of displ for 2-dof for 'node'
	etc., to 'ndf' directions

When generation is performed, the node number sequence will be (for node1–node2 sequence shown above):

node1, node1+ngen1, node1+2\*ngen1, .... , node2

The values for each displacement will be a linear interpolation between d(node1) and d(node2).

Up to 16 values are possible on each input record.

A displacement is prescribed when together with the **disp** command also a boundary condition is set for the same degree of freedom with **boun** or **ebou**.

## EANG

---

### eang

angl1,angl2,itrot  
i-dir,xi-value,angle  
<etc., terminate with a blank record>

---

**eang** define a local base system along a line. The angles may be set along any set of nodes which has a constant value of the 'i-direction' coordinate (e.g., 1-direction (or x), 2-direction (or y), etc.). The following data are necessary:

angl1	— first axis of rotated basis (default=1)
angl2	— second axis of rotated basis (default=2)
itrot	— 0/1 with/without rot. dofs 4-6 (default=0)
i-idir	— the direction of the coordinate (i.e., 1 = x, 2 = y, etc. to <i>ndm</i> ) to be searched.
xi-value	— the value of the i-direction coordinate to be used during the search (a tolerance of about 1/1000 of the mesh size is used during the search, any coordinate within the tolerance is assumed to have the specified value).
angle	— value of angle new 1-coordinate makes with x(1,node).

### Remarks:

1. Only one choice of directions angl1, angl2 and itrot is possible.
2. A transformation will be done for dofs angl1 and angl2.
3. In case of ndf=6 the same transformation is chosen for dofs angl1+3 and angl2+3 for itrot=0.
4. The associated dofs of the used element can be chosen via the macro **mate**, parameters idfg. As example the angles of the DKQ-plate element can be transformed via  
**mate**  
1,7,2,3,1
5. Results are printed and plotted for nodes with  $\text{angl} \neq 0$  in directions angl1,angl2. Thus mixed vectors occur.
6. Results which base on a smoothing procedure are plotted in global directions. Thus results for nodes with  $\text{angl} \neq 0$  are transformed to global directions. Ex.: **disp**, **resi**, **eigv**,...

## EBOU

---

### **ebou,<gap>**

<vgap, only in case of ebou,gap>  
i-dir,xi-value,(ibc(j),j=1,ndf)  
<etc., terminate with a blank record>

---

The boundary restraint conditions may be set along any set of nodes which has a constant value of the 'i-direction' coordinate (e.g., 1-direction (or x), 2-direction (or y), etc.). The data to be supplied during the definition of the mesh (or in macro execution using the **mesh** command) consists of:

<vgap>	– Input only in case of <b>ebou,gap</b> : user defined absolute tolerance for search.
i-idir	– the direction of the coordinate (i.e., 1 = x, 2 = y, etc. to <i>ndm</i> ) to be searched.
xi-value	– the value of the i-direction coordinate to be used during the search (a tolerance of about $1/1000 \cdot \sqrt{numnp}$ of the mesh size is used during the search, any coordinate within the tolerance is assumed to have the specified value). In case of problems use <b>ebou,gap</b>
ibc(1)	
ibc(2)	restraint conditions for all nodes with the
...	value of the search (0 = active dof).
ibc(ndf)	> 0 or < 0 denotes a <i>fixed</i> dof).

### Remarks:

1. Up to 16 values and up to 80 characters are possible on each input record.
2. **boun** sets the boundary conditions whereas **ebou** adds boundary conditions to existing values.
3. For the specification of prescribed displacements within a calculation, see macro commands **edis** or **disp**.

## EDGE

---

**edge**  
 $a_1, a_2, (a_3), b_1, b_2, (b_3), c_1, c_2, (c_3)$ , tol, add, rad  
(ibc(j), j=1, ndf)  
<etc., terminate with a blank record>

---

- The boundary restraint conditions may be set on a line between points on a straight line, on a parabola or on a circle.
- The straight line is defined by  $P_a(a_1, a_2, (a_3))$ ,  $P_b(b_1, b_2, (b_3))$  and  $P_c(0, 0, (0))$ .
- The parabola is defined by  $P_a(a_1, a_2, (a_3))$  (begin),  $P_b(b_1, b_2, (b_3))$  (end) and the third point  $P_c(c_1, c_2, (c_3))$ .
- The circle is defined by  $P_a(a_1, a_2, (a_3))$  (begin),  $P_b(b_1, b_2, (b_3))$  (end), the center point  $P_c(c_1, c_2, (c_3))$  and a nonzero value 'rad' (radius of circle). Circles are only possible in the 1-2-plane. Thus differences in 3-direction are ignored. Angles to the points  $P_a$  and  $P_b$  are measured from the 1-axis counterclockwise. Angles are allowed only in the range:  $0^0 \leq \varphi \leq 360^0$ . Thus a circle between for example  $300^0$  and  $450^0$  has to be described by two circles:  $300^0 \leq \varphi \leq 360^0$  and  $0^0 \leq \varphi \leq 90^0$ .
- **edge** adds the boundary conditions for add = 0 and set the boundary conditions for add = 1.
- 'Tol' is a value to define the convergence radius around the line. With the default value 'tol' = 1000 usually the associated nodes are found. In special situations (fine discretization) a higher value may be necessary to find only the nodes exact on the given line.
- Proof if all nodes are correct found with **boun** in the plot mode.
- 'Rad' is the radius of the circle. Input for other types 'Rad' = 0.
- The boundary data are defined by:

ibc(1)  
ibc(2) restraint conditions for all nodes with the  
... value of the search (0 = active dof).  
ibc(ndf) > 0 or < 0 denotes a *fixed* dof).

### Remarks:

1. Up to 16 values and up to 80 characters are possible on each input record.

## EDIS

---

### edis

i-dir,xi-value,(dbc(j),j=1,ndf)  
<etc., terminate with a blank record>

---

Prescribed values for boundary restraint conditions may be set along any set of nodes which has a constant value of the 'i-direction' coordinate (e.g., 1-direction (or x), 2-direction (or y), etc.). The data to be supplied during the definition of the mesh (or in macro execution using the **mesh** command) consists of;

i-idir	— the direction of the coordinate (i.e., 1 = x, 2 = y, etc. to <i>ndm</i> ) to be searched.
xi-value	— the value of the i-direction coordinate to be used during the search (a tolerance of about 1/1000 of the mesh size is used during the search, any coordinate within the tolerance is assumed to have the specified value).
dbc(1)	
dbc(2)	prescribed values for restraint conditions for all nodes
...	
dbc(ndf)	

### Remarks:

1. Up to 16 values and up to 80 characters are possible. Only one input record could be used at the moment, thus the constraint is  $ndf \leq 14$ .
2. The boundary conditions have to be set with appropriate macro commands, typically **ebou**. Otherwise these values are interpreted as loads.
3. Control of input is possible, if **ebou** is inactive. Then values of **edis** could be seen as loads.

**ELEM****elem**

```
nelm1, matl1, (ix(i,nelm1),i=1,nen),ngen1
nelm2, matl2, (ix(i,nelm2),i=1,nen),ngen2
<etc., terminate with blank record, or when elem2 = numel>
```

The **elem** command is used to specify the values of nodal numbers which are attached to each element. For each element to be specified, a record is entered with the following information:

nelm	– the number of the element to be specified.
matl	– the material number for the element, this will determine the element type.
ix(1,nelm)	– node–1 number attached to element.
ix(2,nelm)	– node–2 number attached to element. etc., to 'nen' nodes.
ngen	– value to increment each node–i value when generation is used (default = 1).

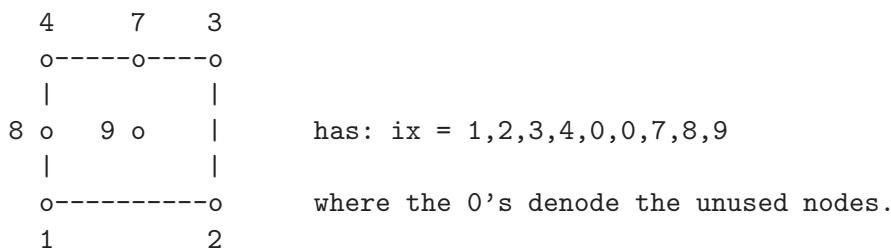
When generation is performed, the element number sequence will be in increments of 1 from 'nelm1' to 'nelm2'; the nodes which are generated for each intermediate element will be as follows:

$$\text{ix}(i,\text{nelm1}+1) = \text{ix}(i,\text{nelm1}) + \text{ngen1}$$

except

$$\text{ix}(i,\text{nelm1}+1) = 0 \text{ whenever } \text{ix}(i,\text{nelm1}) = 0$$

The program will assume that any zero value of an  $\text{ix}(i,\text{nelm})$  indicates that no node is attached at that point. This option is especially useful when isoparametric elements with midside nodes form a transition to elements without midside nodes. In this case it is often necessary to specify an element whose sides have mixed order. For example:



In case 'numel' is defined in the input it is not necessary to describe the last element number. A blank record will cause the program to generate elements up to 'numel' using the node increment specified on the last input record.

Input of element numbers is sequential! Free input of element numbers is possible with macro **elfr**.

**ELFR(EE)**

---

**elfr**

nelm1, matl1, (ix(i,nelm1),i=1,nen)  
nelm2, matl2, (ix(i,nelm2),i=1,nen)  
<etc., terminate with blank record

---

The **elfr** command is used to specify the values of nodal numbers which are attached to each element. For each element to be specified, a record is entered with the following information:

nelm	– the number of the element to be specified.
matl	– the material number for the element, this will determine the element type.
ix(1,nelm)	– node-1 number attached to element.
ix(2,nelm)	– node-2 number attached to element.
	etc., to 'nen' nodes.

No generation is possible. In this case use macro **elem**.

Input of element numbers is not sequential, gaps in numbering are possible.

## **EL3B**

---

### **el3b**

```
nelm1, matl1, (ix(i,nelm1),i=1,nen),ngen1  
nelm2, matl2, (ix(i,nelm2),i=1,nen),ngen2  
<etc., terminate with blank record, or when elem2 = numel>
```

---

The **el3b** command is identical to the **elem** command. The only difference is that the node-generation does not act on the third node which is used for 3-D beams to specify the the orientation of local axis.

## ELOA

---

### eloa

$a_1, a_2, (a_3), b_1, b_2, (b_3), c_1, c_2, (c_3)$ , tol, bou, rad  
etyp, ltyp, td(i)... load values due to load case  
<etc., terminate with a blank record>

---

Loads may be set on a line between points. Here, a straight line, a parabola or a circle are possible up to now.

- The straight line is defined by  $P_a(a_1, a_2, (a_3))$  and  $P_b(b_1, b_2, (b_3))$  and  $P_c(0, 0, (0))$ .
- The parabola is defined by  $P_a(a_1, a_2, (a_3))$  (begin),  $P_b(b_1, b_2, (b_3))$ (end) and the third point  $P_c(c_1, c_2, (c_3))$ .
- The circle is defined by  $P_a(a_1, a_2, (a_3))$  (begin),  $P_b(b_1, b_2, (b_3))$ (end), the center point  $P_c(c_1, c_2, (c_3))$  and a nonzero value 'rad' (radius of circle). Circles are only possible for x-y-systems (2D). Angles to the points  $P_a$  and  $P_b$  are measured from the x-axis counterclockwise. Angles are allowed only in the range:  $0^0 \leq \varphi \leq 360^0$ . Thus a circle between for example  $300^0$  and  $450^0$  has to be described by two circles:  $300^0 \leq \varphi \leq 360^0$  and  $0^0 \leq \varphi \leq 90^0$ .
- 'Tol' is a value to define the convergence radius around the line. With the default value 'tol' = 1000 usually the associated nodes are found. In special situations (fine discretization) a higher value may be necessary to find only the nodes exact on the given line.
- 'bou': For bou = 0(default) external loads are calculated only if no boundary condition is set. On the other hand, for bou = 1, the given load is set directly on the node as a single load! Use for this purpose load case 1! Thus, if boundary conditions are set, the load is assumed as given displacement.
- Proof if all nodes are correct loaded with

- # **load** (nodes and load vectors)
- # **load**,,-1 (only nodes marked) in the plot mode.
- **eloa** adds the loads to the nodal force vector.
- 'Rad' is the radius of the circle. Input for other types 'Rad' = 0.
- 'etyp' is the parameter to define on which type of elements loads are generated:
  - # etyp = 1: linear elements (2 nodes on boundary)
  - # etyp = 2: quadratic elements (3 nodes on boundary)
- 'ltyp' is the parameter to define the special loading parameter due to the following table

load case	ltyp	d(1)	d(2)	d(3)	d(4)	d(5)	d(6)	d(7)	d(8)	d(9)	d(10)	d(11)	d(12)
constant loads	1	$q_{0x}$	$ip_x$	$q_{0y}$	$ip_y$	$q_{0z}$	$ip_z$	$m_{0x}$	$ip_x$	$m_{0y}$	$ip_y$	$m_{0z}$	$ip_z$
linear loads	2	$q_{ax}$	$q_{bx}$	$ip_x$	$q_{ay}$	$q_{by}$	$ip_y$	$q_{az}$	$q_{bz}$	$ip_z$			

- Loads acts in global 1 – 3 direction on specified length

ip = 0	$ds = \sqrt{(dx^2 + dy^2 + dz^2)}$	element length e.g. load case dead load
ip = 1	$ds = dx$	e.g. load case snow
ip = 2	$ds = dy$	
ip = 3	$ds = dz$	
ip = 12	$ds = \sqrt{(dx^2 + dy^2)}$	
ip = 13	$ds = \sqrt{(dx^2 + dz^2)}$	
ip = 23	$ds = \sqrt{(dy^2 + dz^2)}$	

- constant loads are defined as

$$\begin{aligned} q_x(s) &= q_{0x} & m_x(s) &= m_{0x} \\ q_y(s) &= q_{0y} & m_y(s) &= m_{0y} \\ q_z(s) &= q_{0z} & m_z(s) &= m_{0z} \end{aligned}$$

- linear loads are defined as

$$\begin{aligned} q_x(s) &= q_{ax} + \frac{s}{l_{ab}} \cdot (q_{bx} - q_{ax}) \quad \text{with } l_{ab} = ds \text{ specified above.} \\ q_y(s) &= q_{ay} + \frac{s}{l_{ab}} \cdot (q_{by} - q_{ay}) \\ q_z(s) &= q_{az} + \frac{s}{l_{ab}} \cdot (q_{bz} - q_{az}) \end{aligned}$$

#### • Remarks:

- 1.) Do not use this macro for Bernoulli-type elements.
- 2.) The macro can be used in the axisymmetric case with  $q_{axi} = q \cdot 2\pi \cdot r$ .
- 3.) Proof that the loads acts correctly on quadratic elements.
- 4.) The intermediate node for quadratic elements must be on a straight line and the MIDpoint.
- 5.) Force input may be used as displacements if boundary conditions are set. Thus, these points could not have loads! If loads are generated from **eloa**, delete them by using the macro **load**.
- 6.) Up to now use macro only for straight lines.
- 7.) Use the parameter 'bou' for given displacements.
- 8.) **eloa** and **reac** lead to wrong results for associated dofs at boundaries.

**END**

---

**end**

---

The last command in the mesh input must be **end**. Then special actions like searching for double nodes with **tie** or linking conditions with **link** are done. It follows the execution of the program, which is controlled by **macro** commands. The execution of these commands can be done in batch-modus **macro** or in interactive modus **inte**. In the first case the commands are defined in the input-file after the **macro** command, in the second case the commands are defined interactively.

**stop** is the final command in the input file, which stops the execution.

**EPSQ**

---

**epsq**

iswm,nss,natyp,fx,fy  
( epsq(i),i=1,nss)

---

Input of transfer data

1. line

iswm control switch parameter  
nss no. of strain parameter: 6 (3D), 8 (Shell), 10 (Shell ext.), 6 (3D-beam), 15 (3D-beam ext.)  
natyp 1 (3D), 2 (shell), 3 (shell ext.), 4 (3D-beam), 5 (3D-beam ext.)  
fx factor in  $\kappa_x = \frac{1}{1 + f_x(\ell/h)^2}$   
fy factor in  $\kappa_y = \frac{1}{1 + f_y(\ell/h)^2}$

2. line

Input of strain array for

3D nss=6  $\mathbf{E} = [\varepsilon_{11}, \varepsilon_{22}, \varepsilon_{33}, 2\varepsilon_{12}, 2\varepsilon_{13}, 2\varepsilon_{23}]^T$   
inextensible shell nss=8  $\mathbf{E} = [\varepsilon_{11}, \varepsilon_{22}, 2\varepsilon_{12}, \kappa_{11}, \kappa_{22}, 2\kappa_{12}, 2\varepsilon_{13}, 2\varepsilon_{23}]^T$   
extensible shell nss=10  $\mathbf{E} = [\varepsilon_{11}, \varepsilon_{22}, 2\varepsilon_{12}, \kappa_{11}, \kappa_{22}, 2\kappa_{12}, 2\varepsilon_{13}, 2\varepsilon_{23}, \varepsilon_{33}^0, \varepsilon_{33}^1,]^T$   
inextensible beam nss=6  $\mathbf{E} = [\varepsilon_{11}, 2\varepsilon_{12}, 2\varepsilon_{13}, \vartheta, \kappa_{11}, \kappa_{22}]^T$   
extensible beam nss=15  $\mathbf{E} = [\varepsilon_{11}, 2\varepsilon_{12}, 2\varepsilon_{13}, \vartheta, \kappa_{11}, \kappa_{22}, \varepsilon_{22}^0, \varepsilon_{22}^1, \varepsilon_{22}^2, \varepsilon_{33}^0, \varepsilon_{33}^1, \varepsilon_{33}^2, \varepsilon_{23}^0, \varepsilon_{23}^1, \varepsilon_{23}^2]^T$

Strain data are input for the calculation of prescribed displacements  $\mathbf{V} = \mathbf{AE}$  within macro **epsq**.

**FIXE****fixe**

segm1, (id(i,segm1),i=1,ndf)  
segm2, (id(i,segm2),i=1,ndf)  
<etc., terminate with blank record>

---

The **fixe** command is used to specify the values for the boundary restraint conditions at segments. For each segment to be specified a record is entered with the following information:

segm	– the number of segment to be specified
id(1,segm)	– value of boundary restraint for 1-dof for 'segm'
id(2,segm)	– value of boundary restraint for 2-dof for 'segm'. etc. until to 'ndf' directions

The boundary restraint codes are interpreted as follows:

- id(i,segm) = 0     i-dof has unknown displacement,  
id(i,segm) = 1     i-dof has known displacement 0.

**Remarks:**

1. For mesh generation with NEGE the following order of statements is necessary: **nege** , **[neco]**, **[back]**, **geom**, **segm**, **regi**, **[pres]**, **[fixe]**, further FEAP-macros. Note: macros in brackets are optional.
2. Compare the macro **boun**.

## **GBOU**

---

**gbou**

---

This macro is up to now not documented.

## GCOR

---

```
gcor
node1, (x(i,node1),i=1,ndm),ityp
node2, (x(i,node2),i=1,ndm),ityp
<etc., terminate with blank record>
```

---

The **gcor** command is used to specify the values for nodal coordinates for the generation program GMESH. The only difference to **coor** is that no generation increment can be used. For each node to be specified a record is entered with the following information:

node	– the number of the node to be specified
x(1,node)	– value of coordinate in 1-direction for 'node'
x(2,node)	– value of coordinate in 2-direction for 'node'
<x(3,node)	– value of coordinate in 3-direction for 'node'>
<ityp	– 0=cart.,1=pola(1-2),2=pola(2-3),3=pola(1-3),4=sphe>

Input could be given in different coordinate systems, see 'ityp'.

Output is always in cartesian coordinates.

## GELE

---

### gele

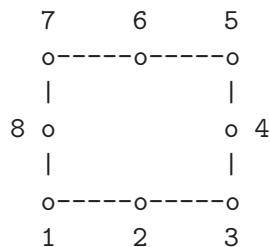
```
nelm1, matl1, (ix(i,nelm1),i=1,8)
nelm2, matl2, (ix(i,nelm2),i=1,8)
<etc., terminate with blank record>
```

---

The **gele** command is used to specify the values of nodal numbers which are attached to each element for program GMESH. The only differences to **elem** are that no generation is available and always 8 nodes are used. For each element to be specified, a record is entered with the following information:

nelm	– the number of the element to be specified.
matl	– the material number for the element, this will determine the element type.
ix(1,nelm)	– node-1 number attached to element.
ix(2,nelm)	– node-2 number attached to element, etc., to '8' nodes.

Node numbering of bloc is in contrast to **elem** of FEAP: corner-node, mid-side-node, corner-node, ... etc. anti-clockwise and is shown below:



## GEOM

---

**geom**

node1, (x(i,node1),i=1,2)  
node2, (x(i,node2),i=1,2)  
<etc., terminate with blank record>

---

The **geom** command is used to specify the values for nodal coordinates for segment nodes. For each node to be specified a record is entered with the following information:

node	— the number of the node to be specified
x(1,node)	— value of coordinate in 1-direction for 'node'
x(2,node)	— value of coordinate in 2-direction for 'node'

**Remark:**

For mesh generation with NEGE the following order of statements is necessary: **nege** , **[neco]**, **[back]**, **geom**, **segm**, **regi**, **[pres]**, **[fixe]**, further FEAP-macros. Note: macros in brackets are optional.

## GMESH

---

**gmesh** [ title of problem for printouts, etc.]  
node,ndf,ndm,iopt,ityp,iprin

---

Each problem to be solved by **FEAP** and using the mesh generator **GMESH** must start with a single record which contains the characters **gmesh** as the first entry; the remainder of the record (columns 5-80) may be used to specify a problem title. The title will be printed with each “page” of output as the first line. Immediately following the **gmesh** record, the ‘control’ information describing characteristics of the problem to be solved must be given. The ‘control’ information describes the characteristics of the finite element problem to be solved. The data entries are:

node	— number of nodes per element.
	— 3 node triangle, 4/8 node quadrilateral.
ndf	— maximum number of degrees-of-freedom on any node.
ndm	— maximum number of dimensions.
iopt	— 0/1 without/with node numbering optimization.
ityp	— 0 = generate mesh, — 1 = check input data.
iprin	— 0 = no action, — 1 = print additional information.

### Remarks:

1. A FEAP-input file will be generated called ifeap.tmp (always).
2. Control information of generation can be found in the file GMESH.log.
3. For mesh generation with GMESH the following order of statements is necessary:
4. **gmesh**, [**neco**], **gcor**, **gele**, further FEAP-macros.  
Note: macros in brackets are optional.
5. For further information see the **GMESH**-manual.

## ICON

---

**icon**

```
nsl,nsntl,nmntl,naxi
nsln(i),nmln(i),isl(i),ifla(i),sfacn(i),sfact(i),ityp(i),itan(i) ↑
beta
na,nsvn(na)
nb,nsvn(nb)
...
na,msrn(na)
nb,msrn(nb)
...
...
```

$i = 1 \text{ to } ns$

---

The **icon** macro is used to define surfaces which may interact through contact during the analysis of solids. It is necessary to specify:

nsl	— the total number of potential contact surfaces.
nsntl	— the total number of nodes on all “slave” surfaces.
nmntl	— the total number of nodes on all “master” surfaces.
naxi	— flag (0 = plane, 1 = axisymmetric problem.)

When looking along a potential contact surface the “slave” surface is on the left side and the “master” surface is on the right side. Nodes defined on surfaces must always be specified with the “master” surface on the right as the sequence is input. For each contact surface ( $i = 1 \text{ to } ns$ ), the following slideline control information is required:

nsln	— number of slave nodes per slide line
nmln	— number of master nodes per slide line
isl	— slide-line type: 1 = sliding only; 2 = slide-line tied; 3 = slide + void; 4 = friction + void.
ifla	— flag for contact algorithm 1 = 1-pass algorithm 0 = 2-pass algorithm (change role of master and slave surface during 2 <sup>nd</sup> pass)
sfacn	— stiffness scaling factor normal (penalty)
sfact	— stiffness scaling factor tangential (penalty)
ityp	— type of solution method : 0 = Penalty; 1 = Augmented Lagrangian; simultaneous iteration (implemented for frictionless contact only); 2 = Augmented Lagrangian, nested iteration, symmetrized treatment of friction; 3 = Augmented Lagrangian, nested iteration, non-symmetric treatment of friction.
itan	— type of solution tangent matrix : 0 = Consistent; 1 = Nonsymmetric contribution to tangent omitted for frictional problems (not ordinarily recommended); 2 = Linearized.

After the above slideline control information, the following contact node data is required:

beta	– friction coefficient on this surface
na	– local number of slave node on this surface
nsvn(na)	– global (mesh) node number associated with na.
.....	repeat until all nodes are input (or generated).
na	– local number of master node on this surface
msrn(na)	– global (mesh) node number associated with na.
.....	repeat until all nodes are input (or generated).

When specifying nodes on a master or slave surface the local nodes must be input in an increasing sequence (i.e., 1,2,5,...). If a node is missing in the sequence the global node will be generated according to the following relationship:

$$\text{nsvn}(i+1) = (\text{nsvn}(nb) - \text{nsvn}(na)) / (nb - na) + \text{nsvn}(i)$$

It is also necessary to indicate on the macro program level with the macro **cont** that a contact problem is being solved. If a solution is desired without using the contact option it is necessary to reset the contact solution using **cont,off**. It is also possible to reset the value of the penalty parameters during execution using the **cont** macro.

#### Further remarks:

The augmented Lagrangian solution schemes available are one solution to difficulties posed by the penalty method; i.e., potential ill-conditioning on one hand and underpenalization of constraints on the other. The simultaneous technique (option 1) performs the augmentations during the equilibrium iterations, and has the drawback that it does not preserve quadratic rates of convergence. The nested techniques (options 2 and 3) are identical in the frictionless case, but differ in their treatments of friction. The symmetrized option only enforces the Coulomb condition in the augmentation phase, while the nonsymmetric option enforces it both in the solution phase and in the augmentation phase. The term 'nested' refers to the manner in which augmentations are performed; within a time step, they are done after equilibrium has been enforced using the current Lagrange multipliers. That is, solutions (with the associated Newton iterations) and augmentations are done recursively and completely separately until convergence of the multipliers is achieved. In the nested schemes, the augmentations must be requested by the user at the time they are desired by the use of the CAUG macro; no such request need be made in the simultaneous treatment.

#### References:

- P. Wriggers, J.C. Simo: A Note on Tangent Stiffnesses for Fully Nonlinear Contact Problems, Comm. Appl. Num. Meth., **1**, 199–203, 1985.
- J. C. Simo, P. Wriggers, K. Schweizerhof, R. L. Taylor: Postbuckling Analysis involving Inelasticity and Unilateral Constraints, Int. J. Num. Meth. Engng. **23**, 779–800, 1986.

## **ICOR**

---

**icor**  
(xperf(i),i=1,ndm),(ximperf(i),i=1,ndm)  
<etc., terminate with blank record>

---

The **icor** command is used to modify perfect coordinates due to (e.g. measured) imperfections.

This macro should be used at the end of the input file. Due to the modification of coordinates macros like **eloa** or **edge** may not work.

## IMPF

---

```
impf  
α  
node1, ngen1, (impf(i,node1),i=1,ndm)  
node2, ngen2, (impf(i,node2),i=1,ndm)  
<etc., terminate with blank record>
```

---

The **impf** command is used to modify the values for nodal coordinates. The handling is identical to the macro **coor**.

The coordinates are modified due to

$$\mathbf{x} = \mathbf{x} + \alpha \cdot \mathbf{impf}$$

Be careful with the use of this macro. Due to the modification of the coordinates macros like **eloa**, **edge** or **tie** may not work. Thus use for example **eloa** before **impf**.

The vector  $\bar{\mathbf{u}} = \alpha \cdot \mathbf{impf}$  can be plotted using **dimp** directly after start of FEAP. Later **dimp** plots  $\bar{\mathbf{u}} + \mathbf{u}$ .

## **INTE**

---

### **inte**

---

The last command in the mesh input must be **end**. Then special actions like searching for double nodes with **tie** or linking conditions with **link** are done. It follows the execution of the program, which is controlled by **macro** commands. The execution of these commands can be done in batch-modus **macro** or in interactive modus **inte**. In the first case the commands are defined in the input-file after the **macro** command, in the second case the commands are defined interactively.

**stop** is the final command in the input file, which stops the execution.

## ISEC

---

**isec**

$a_1, a_2, a_3, b_1, b_2, b_3$ ,tol  
<etc., terminate with a blank record>

---

- This macro sets dof6 = 1 for shells with 5 dofs and release this dof (dof6 = 0) at intersections.
- The intersection line is straight and defined by  $P_a(a_1, a_2, a_3)$ , and  $P_b(b_1, b_2, b_3)$ .
- 'Tol' is a value to define the convergence radius around the line. With the default value 'tol' = 1000 usually the associated nodes are found. For fine discretizations a higher value may be necessary to find only the nodes exact on the given line.
- Proof if all nodes are correct found with **isec** in the plot mode.

**JINT**

---

**jint**

njint

nel1,nel2,nel3,nel4,nel5,nel6,nel7,nel8

...

...

nel1,nel2,nel3,nel4,nel5,nel6,nel7,nel8

...

 $i = 1 \text{ to } njint$ 

The **jint** macro is used to input a series of elements which define the path along which the  $J$ -integral has to be evaluated. It is necessary to specify:

njint – the total number of elements along the path.
nel1–nel8 – the element numbers (8 per record) .

When looking along the path along which the  $J$ -integral has to be computed the series of elements have to be put in "anti-clock-wise".

**KNV1**

---

```
knv1
 $\Xi_1^1(\xi_1^1)$ 
 $\Xi_1^1(\xi_2^1)$ 
...
 $\Xi_1^1(\xi_{n+p+1}^1)$ 
 $\Xi_2^1(\xi_1^1)$ 
...
 $\Xi_2^1(\xi_{n+p+1}^1)$ 
...
 $\Xi_{n\_patch}^1(\xi_{n+p+1}^1)$ 
```

---

**knv1** defines the knot vector in direction of parameter  $\Xi^1$ . After the keyword **knv1** every line contains one knot. The total number of lines after **knv1** must be

$$\sum_{i=1}^{n\_patch} n_i + p_i + 1$$

due to NURBS conventions (as open knot vectors have to be used). The line with the entry

$$\Xi_i^1(\xi_j^1)$$

contains the  $j$ -th knot of the  $i$ -th patch in  $\Xi^1$ -direction. The rules regarding digits and exponents are the same as for macro **coor**. **knv1** and **knv2** have to be placed after **nmpq**.

**KNV2**

---

```
knv2
 $\Xi_1^2(\xi_1^2)$ 
 $\Xi_1^2(\xi_2^2)$ 
...
 $\Xi_1^2(\xi_{m+q+1}^2)$ 
 $\Xi_2^2(\xi_1^2)$ 
...
 $\Xi_2^2(\xi_{m+q+1}^2)$ 
...
 $\Xi_{n\_patch}^2(\xi_{m+q+1}^2)$ 
```

---

**knv2** defines the knot vector in direction of parameter  $\Xi^2$ . After the keyword **knv2** every line contains one knot. The total number of lines after **knv2** must be

$$\sum_{i=1}^{n\_patch} m_i + q_i + 1$$

due to NURBS conventions (as open knot vectors have to be used). The line with the entry

$$\Xi_i^2(\xi_j^2)$$

contains the  $j$ -th knot of the  $i$ -th patch in  $\Xi^2$ -direction. The rules regarding digits and exponents are the same as for macro **coor**. **knv1** and **knv2** have to be placed after **nmpq**.

## LINK

---

### link

```
1,node1,node2,inc,(idl(i),i=1,ndf)
or
2,node1,idm ,x0 ,(idl(i),i=1,ndf),eps
or
3,node_is,node_ie,inc_i,node_js,node_je,inc_j,(idl(i),i=1,ndf)
or
4,idm ,x1 , x2, (idl(i),i=1,ndf),eps
or
5,idmx ,x1 , x2, idmy ,y1 , y2,(idl(i),i=1,ndf),eps
or
6,idmx ,x1 , x2,(idl(i),i=1,ndf),eps
or
7,lx1, lx2, lx3,(idl(i),i=1,ndf)
```

<terminate with a blank record>

A mesh may be generated in FEAP in which it is desired to let the values at more than one node share the same 'displacement' unknown. For example, in repeating structures the value of the dependent variable will be the same at each repeating interval. In a finite element model it is necessary to specify the repeating condition by 'linking' the degree-of-freedoms at these nodes to the same unknown in the equations. The **link** command is used for this purpose.

To use the **link** option the complete mesh must first be defined. After the **end** command for the mesh definition and before the **macro**, **batch** or **interactive** command for defining a solution algorithm, the use of a **link** statement together with the list of affected nodes and degree-of-freedoms will cause the program to search for all conditions that are to be connected together. A tolerance is defined to find the nodes. If this is not successful a user defined value 'eps' can be used, default 'eps=0'.

The input data are interpreted as follows:

#### Type 1: link nodes from node 1 to node 2 to master node 1

node1	— first node in a sequence to be linked = master node
node2	— last node in a sequence to be linked
inc	— increment to generate intermediate nodes affected. (default = node2-node1)
idl(1)	— linking condition, 0 = link, 1 = do not link this dof.
idl(2)	— linking condition, 0 = link, 1 = do not link this dof. etc. for 'ndf' degree of freedoms

**Type 2: link nodes along const. coordinate to master node 1**

node1	— master node to which other nodes are linked
idm	— direction of coordinate (i.e., 1=x,2=y,3=z)
x0	— the value of the idm-direction coordinate
idl( )	— see above
eps	— user defined search tolerance

**Type 3: link nodes j to nodes i (master) , where nodes i and j are out of sequences.**

Thus, type 3 is an automatic repetition of type 1 for given sequences of nodes.

node_is	— first node in a sequence of master nodes
node_ie	— last node in a sequence of master nodes
inc_i	— increment to generate intermediate nodes affected. (default = node_ie-node_is)
node_js	— first node in a sequence to be linked with node_is
node_je	— last node in a sequence to be linked with node_ie
inc_j	— increment to generate intermediate nodes affected. (default = node_je-node_js)
idl(1)	— linking condition, 0 = link, 1 = do not link this dof.
idl(2)	— linking condition, 0 = link, 1 = do not link this dof. etc. for 'ndf' degree of freedoms

**Type 4: link nodes along coordinate x2 to nodes along coordinate x1**

idm	— direction of coordinate (i.e., 1=x,2=y,3=z)
x1	— the value x1 of the idm-direction coordinate
x2	— the value x2 of the idm-direction coordinate
idl( )	— see above
eps	— see above

Thus for idm=1 nodes ( $x_2, y_i, z_i$ ) are linked to nodes ( $x_1, y_i, z_i$ ) for same  $y_i, z_i$ .

**Type 5: link nodes along line  $x_2, y_2$  to nodes along line  $x_1, y_1$**

idmx	— direction of coordinate (i.e., 1=x,2=y,3=z)
x1	— the value $x_1$ of the idmx-direction coordinate
x2	— the value $x_2$ of the idmx-direction coordinate
idmy	— direction of coordinate (i.e., 1=x,2=y,3=z)
y1	— the value $y_1$ of the idmy-direction coordinate
y2	— the value $y_2$ of the idmy-direction coordinate
idl( )	— see above
eps	— see above

Thus for idmx=1,idmy=2 nodes ( $x_2, y_2, z_i$ ) are linked to nodes ( $x_1, y_1, z_i$ ) for same  $z_i$ .

**Type 6: link nodes along coordinate  $x_2$  to nodes along coordinate  $x_1$  cross-wise  
 $y_2(x_2)=-y_1(x_1)$ , only for x,y-direction!**

The origin of the coordinate system must be at least (0,0,z).

idm	— direction of coordinate (i.e., 1=x,2=y)
x1	— the value $x_1$ of the idm-direction coordinate
x2	— the value $x_2$ of the idm-direction coordinate
idl( )	— see above
eps	— see above

Thus for idm=1 nodes ( $x_2, +y_i, z_i$ ) are linked to nodes ( $x_1, -y_i, z_i$ ) for same  $y_i, z_i$ .

This macro is only used for FE<sup>2</sup> on the RVE.

**Type 7: link border nodes of a RVE with length lx1, lx2, lx3. The origin of the coordinate system must be (0,0,0). All border nodes of a plane [ $x_1, x_2, x_3$ ] are linked to a node [-lx1/2, -lx2/2,  $x_3$ ].**

lx1	— length lx1 of the RVE
lx2	— length lx2 of the RVE
lx3	— length lx3 of the RVE
idl( )	— see above

This macro is only used for FE<sup>2</sup> on the RVE.

**Remarks:**

1. Termination of input occurs with a blank record.
2. Whenever it is desired to only connect 'node1' to 'node2', 'inc' need not be specified (it may be blank or zero).
3. Be extremely careful using this macro, because input errors could be seen very difficult! A graphical control with **link** is strongly recommended!
4. Be extremely careful using this macro with multiple records due to possible dependencies.
5. Circle links (like node1 → node2 and then node2 → node1) are identified. Otherwise necessary dofs are deleted and FEAP do not return after a **tang**-macro .
6. In case loads should be specified for the linked nodes, only one node is specified with the resulting load of all nodes. Output for **reac** and **resi** is not correct at linked nodes!
7. In case of boundary conditions should be specified for the linked nodes, only the b.c. for node1 has to be specified.
8. In case of boundary conditions with prescribed displacements **link** should not be used! Use **boun**, **ebou** etc. for setting the b.c. and **disp** or **edis** for setting the prescribed values for all nodes.
9. **link** could be used together with **tie**. Thus it is allowed to use the 'original' node numbers of tied nodes.
10. **Linking conditions could be set for a DOF of a node only once!!** If this rule is violated an error message occur in FEAP. The desired linking condition is not executed! Nevertheless linking conditions for different DOFs of a node are possible in more than one call of **link**.

## LOAD

---

**load**

```
node1, ngen1, (f(i,node1),i=1,ndf)
node2, ngen2, (f(i,node2),i=1,ndf)
<etc., terminate with blank record>
```

---

The **load** command is used to specify the values for nodal boundary loads. For each node to be specified a record is entered with the following information:

node	— the number of the node to be specified.
ngen	— the increment to the next node, if generation is used, otherwise 0.
f(1,node)	— value of load for 1-dof for 'node'
f(2,node)	— value of load for 2-dof for 'node'
	etc., to 'ndf' directions

When generation is performed, the node number sequence will be (for node1–node2 sequence shown above):

node1, node1+ngen1, node1+2\*ngen1, .... , node2

The values for each load will be a linear interpolation between f(node1) and f(node2).

Up to 16 values are possible on each input record.

The **load**-macro is identical to the old **force**-macro.

## LOA0

---

### loa0

```
node1, ngen1, (f(i,node1),i=1,ndf)
node2, ngen2, (f(i,node2),i=1,ndf)
<etc., terminate with blank record>
```

---

The **loa0** command is used to specify the values for nodal loads which are not time dependent. For each node to be specified a record is entered with the following information:

node	— the number of the node to be specified.
ngen	— the increment to the next node, if generation is used, otherwise 0.
f(1,node)	— value of load for 1-dof for 'node'
f(2,node)	— value of load for 2-dof for 'node'
	etc., to 'ndf' directions

When generation is performed, the node number sequence will be (for node1–node2 sequence shown above):

node1, node1+ngen1, node1+2\*ngen1, ..., node2

The values for each load will be a linear interpolation between f(node1) and f(node2).

Up to 16 values are possible on each input record.

On macro level these values could be set via the macro **newf**.

## MATE

---

### **mate**

ma,iel,<idfg1,idfg2,....,idfgn>  
<parameters for material set 'ma' of element type 'iel'>

---

The **mate** command is used to specify the parameters for each material set in the analysis, as well as to specify the particular element type associated with the properties.

Optional it is possible to change the position of degrees of freedoms in the problem. Thus the idfg-array define the global dofnumbers for the used element.

**Example.:** A plate element (e.g. iel=7) with 3 dofs ( $w, \varphi_x, \varphi_y$ ) can be combined with a 3D-beam element (e.g. iel=11) with 6 dofs ( $u, v, w, \varphi_x, \varphi_y, \varphi_z$ ) in the following way

Plate	local number	1( $w$ )	2( $\varphi_x$ )	3( $\varphi_y$ )
	global number	3	4	5

### FEAP input:

mate

1,11

Beam specific data .....

2,7, 3,4,5

Plate specific data .....

The specific parameters to be input are described in each element manual (e.g., see manual for 'elmt01' for the parameters associated with 'iel = 1', etc.).

## MACR

---

xxxx,yyyy,v1,v2,v3

(where **xxxx** is selected from the following list)

acce	arcl	augm	auto	back	bfgs	chec	cmas	cont	conv
copy	crit	curv	damp	data	debu	detk	dibc	disp	dt
dplo	eigi	eigk	eigl	else	end	endi	epsq	erro	exit
ext	feas	form	four	fsol	fsum	geom	help	hist	iden
if	iimp	init	jint	lamb	lan	line	lmas	loop	macn
man	mate	mesh	newf	next	nonl	nopr	para	parv	paus
pbcg	pgmr	plot	pola	post	prin	prco	proc	prop	quit
reac	read	reme	rest	rinp	save	show	sigq	smoo	solv
splo	stre	subs	summ	tang	tec	time	tol	tplo	trans
ueig	umas	updh	utan	velo	writ	yang	yevo	ygra	ymsh
ytab	ytry								

The solution algorithm used by FEAP to solve problems is defined by a “macro statement program”. By properly specifying the macro program a very wide range of applications may be addressed — including both linear and nonlinear, as well as, steady state and transient applications.

The description for the macro statements may be obtained from the manual entry for each command, (e.g., use manual TANG to obtain all options and actions for the **tang** command).

## MESN

---

### **mesn**

---

The mesh macro commands **mesn** ( $n = 1, \dots, 5$ ) are currently not implemented. These commands refer to subroutine calls UMESHn in SR pmesh. They can be used by a programmer who wants to add new input macro features to FEAP.

## NDVI

---

### **ndvi**

```
nelm1, ndvix1, <(dx1(i),i=1,ndvix1)>
nelm1, ndviy1, <(dy1(i),i=1,ndviy1)>
nelm2, ndvix2, <(dx2(i),i=1,ndvix2)>
nelm2, ndviy2, <(dy2(i),i=1,ndviy2)>
<etc., terminate with a blank record>
```

---

The **ndvi** command is used to specify the values of subdivisions of all blocs defined for the mesh generation with GMESH. For each bloc to be specified, two records are entered with the following information:

nelm	– the number of the bloc to be specified.
ndvix1	– number of subdivisions in 1-direction.
ndviy1	– number of subdivisions in 2-direction.
<dx1(i)	– ndvix1*subdivisions in 1-direction.>
<dy1(i)	– ndvix1*subdivisions in 2-direction.>

Remarks:

- 1) At connection lines of different blocs the same subdivision has to be used.
- 2) The input of values dx1,dy1 etc. is only necessary for a non-constant division.

Then:

- 3) Values can be chosen arbitrary. The subdivision bases on relative results from the input values.
- 4) Up to 16 values are possible on each input record.

## NEGE

---

**nege** [ title of problem for printouts, etc.]  
ntyp,node,ndf,ncorr,esiz

---

Each problem to be solved by **FEAP** and using the mesh generator **NEGE** must start with a single record which contains the characters **nege** as the first entry; the remainder of the record (columns 5-80) may be used to specify a problem title. The title will be printed with each “page” of output as the first line. Immediately following the **nege** record, the ‘control’ information describing characteristics of the problem to be solved must be given. The ‘control’ information describes the characteristics of the finite element problem to be solved. The data entries are:

ntyp	– 1 = background mesh (control). – 2 = mesh from segments (control). – 3 = mesh from regions (control). – 4 = generates FE mesh.
node	– number of nodes per element. – 3/6 node triangle, 4/8/9 node quadrilateral.
ndf	– maximum number of degrees-of-freedom on any node. – (max for NEGE is 3!).
ncorr	– 0 = region with straight boundaries, – 1 = region with curved boundaries.
esiz	– approximate element length in diagonal direction within the surrounded rectangle, 10-0.5 ... (rough-fine).

### Remarks:

1. It will be recommended to test data input with ntyp =(1),2 to 4. ’Ntyp’ = (1),2,3 is only used to control the input data for mesh generation. In these cases the plot-options **node**, **elem**, **mesh**, **load**, **boun** are still working.
2. ’Ndm’ must be 2 (number of spatial coordinates).
3. A FEAP-input file will be generated called ifeap.tmp (always).
4. Control information of generation can be found in the file NEGE.log.
5. For mesh generation with NEGE the following order of statements is necessary: **nege** , [**neco**], [**back**], **geom**, **segm**, **regi**, [**pres**], [**fixe**], further FEAP-macros. Note: macros in brackets are optional.
6. For further information see the **NEGE**-manual.

## **NECO**

---

### **neco**

---

This macro is similar to the macro **para**. All definitions which are defined by **neco** hold for the mesh generation and FEAP.

#### **Remark:**

For mesh generation with NEGE the following order of statements is necessary: **nege** , **[neco]**, **[back]**, **geom**, **segm**, **regi**, **[pres]**, **[fixe]**, further FEAP-macros. Note: macros in brackets are optional.

## NMPQ

---

### nmpq

n\_patch

n1, m1, p1, q1, mat1

n2, m2, p2, q2, mat2

<etc., terminate with blank record>

---

The **nmpq** command is used to specify the shape of the mesh and the order of the basis functions. In the first line the number of patches 'n\_patch' has to be stated. As the control points are entered via a linear list, the dimensions of the control point 'matrices' have to be given. For all patches the following constants have to be given. *n* is the number of control points in  $\Xi^1$ -direction, *m* in  $\Xi^2$ -direction. *p* and *q* are the corresponding orders. 'mat' gives the element type (at the moment only element 60 is available, which is derived from element 5). Every line except the first line creates one patch. **nmpq** has to appear after **coor** and before **knv1** and **knv2**.

n_patch	– the number of patches
n	– the number of control points in $\Xi^1$ -direction
m	– the number of control points in $\Xi^2$ -direction
p	– the order of basis functions in $\Xi^1$ -direction
q	– the order of basis functions in $\Xi^2$ -direction
mat	– element type

## **NOPA**

---

**nopa**

---

**nopa** allows all input data only numerical. Thus parameters are not allowed in input data. The opposite command is **pars** which allows all input data as expressions.

## **NOPR**

---

**nopr**

---

The use of the **nopr** macro command will discontinue the output of all subsequent mesh data (except for material data printed in each element). The use of **prin** will cause the output of mesh information to again be reported. The default value is **prin** at start of each problem execution.

## OPTI

---

### opti, optn

---

A mesh may be generated in **FEAP** without looking on a good node numbering which leads to small differences for the node numbers. This may be essential for the bandwith of the stiffness matrix and therefore for the necessary memory and for the solution time of solving equations.

Thus two optimization procedures are available which lead to an internal renumbering of the nodes.

To use the **opti** option the complete mesh must first be defined. After the **end** command for the mesh definition and before the **macro** or **interactive** command for defining a solution algorithm, **opti** is used to optimize the input data.

Optimization procedures:

- **opti** - bases on Minimum front criteria (Hoit-Wilson)
- **optn** - bases on the 'Cuthill-McKee Algorithm'

Note that the optimization procedure bases on the chosen basic node numbering. Furthermore it is not clear which algorithm leads to better results, thus try both ones and compare the results shown in the first FEAP-output.

**PAGE**

---

**page**  
character

---

The **page** command may be used to specify the 'character' which produces a page feed on a line printer for the printout of the results. In 'character', 4 single characters may be defined.

## PARA

---

### para

---

Data input specifications in FEAP may be in the form of numerical constants, parameters, or expressions. Constants are conventional forms for specifying input data. Parameters are single character lower case letters to which values are assigned. Expressions are combinations of constants and parameters and functions.

Here, the **para** macro command may be used to assign values to letter parameters.

A letter parameter is defined immediately following the **para** macro (several may follow terminating with a blank record) according to the following:

x = input data

where 'x' may be any of the single letters (a-z), any group of two letters (aa-zz), or any letter and a numeral (a0-z9) followed by the equal sign. All alphabetic input characters are automatically converted to lower case, hence there are 962 unique parameters permitted at any one time.

The input data may be

numbers (floating point numbers can contain an 'e' or a 'd' exponent format),  
previously defined letter constants,  
expressions.

Expressions can be calculated from numbers, parameters and functions. The expression is processed left to right and can contain **one** set of parentheses to force groupings.

The following arithmetic operations +, -, \*, / or ^ are allowed.

The following functions may appear in an expression, a statement, or a parameter definition:

sin	sind	asin	asind	sinh	abs	int	
cos	cosd	acos	acosd	cosh	exp	log	sqrt
tan	tand	atan	atand	atanh	inc	dec	

The trigonometric and inverse trigonometric functions which end in d involve values of angles in degrees; whereas, the ones without involve values in radians. Each function has one argument which is contained between parenthesis (which counts as the one level of depth). The argument may be an expression but may not contain any parenthesis or additional functions.

The function inc(val+) increases a value by val+, reset to 0 is given by val+ = 0, whereas the function dec(val-) increases a value by val-, reset to 0 is given by val- = 0.

A hierarchical evaluation is performed according to the rules defined below

Order	Operation	Notation
1.	Parenthetical expressions	(...)
2.	Functions	^
3.	Exponentiation	*
4.	Multiplication or Division	/
5.	Addition or Subtraction	+

#### Examples:

a = 3. bb = 14/3.45 p = 4 \* atan(1)  
c = f + 1.03e-4\*a/b c1 = a + 3.23/bb i = 4 \* cos(a+b)

## Remarks

- The macro **para** replaces the macro **cons**.
- The macro **para** can be used also as macro command **para**.
- Note that the expressions do not have more than 75 characters and contain not more than 32 entries (+,-,numbers,parameter) for a correct input!
- Note that internal computations are all performed in double precision arithmetic (e.g., as REAL\*8 variables).
- At the present time only **one level** of parenthesis may appear in any expression.
- Solve the above problems by a repetition of the definitions.

Example:  $q1 = \tan(1. / (3. + aa))$  is not a legal expression at the present time. It should be replaced by the pair of statements

$q1 = 1. / (3. + aa)$

$q1 = \tan(q1)$

- As a consequence of these repetition parameters may have their values redefined as many times as needed by using the **para** data command followed by other commands and data using the values of assigned parameters.
- In interactive mode of execution, the current set of parameters may be output by entering **list** while in **parameter** input mode. After listing input of additional parameters may be continued.
- It is possible to use expressions containing the parameters in any input mode which uses the utility input routine 'dinput'. Users who write their own programs should try to use this routine to perform all inputs – 'dinput' will input from either the input file or from the keyboard, depending on which is the active input unit.

## Consequences for all FEAP input data

The most powerful form of data input in FEAP is the use of parameters as described above. Once defined by **para** they can be used at any place of the data input.

Furthermore expressions containing of constants, parameters and functions are allowed at any place of input.

Accordingly, a load may be assigned as

**load**       $1,,a/12.$     +     $3.$

is permitted. Another example are the node and element numbers. After defining n,e,m,x,y,p by **para** the following input is possible

**bloc**

$4,4,4,n,e,m$

$1,0.+x,0.+y$

$2,5.+x,0.+y$

$3,5.+x,5.+y$

$4,0.+x,5.+y$

**coor**

n+25,0,5.5+x,2.5+y

**elem**

e+16,p,n+14,n+15

to input a block of nodes and elements. Note that **here** the expressions do not have more than 10 characters to have a correct input!

## PARS

---

### **pars**

---

**pars** allows all input data as expressions. Thus parameters could be used in input data. **pars** is set automatically by **para**. The opposite command is **nopa** which allows all input data only numerical.

## POIN

---

### poin

```
( xp(i),i=1,ndm),<fact>*
( fp(i),i=1,ndf)
(bcp(i),i=1,ndf)
<etc., terminate with a blank record>
```

---

At a point with coordinates xp loads and boundary conditions can be set with **poin**.

FEAP is searching for points within a certain circle around the point with coordinates xp.

In case of a very fine discretization, it may occur that more than one point will be found. In that case a warning occur during the data input process of FEAP. With a value of 'fact' (default=1) larger than 1 the radius of the circle can be reduced.

### Remarks:

- \* Input for a point with **xp=0**: fact have to be set explicitly!
- Loads fp and bc-conditions bcp are 'added'.
- Nodes have to be defined before.
- Be careful in case of tied nodes to avoid multiple loads, see macro **tie**.
- Input possible only for ndf  $\leq 16$ .

## POLA

---

### pola

node1,node2,inc, $x_{0i}$ ,  $x_{0j}$ ,ntype  
<terminate with blank record>

---

The **pola** command may be used to convert any coordinates  $(r, \theta)$  which have been specified in polar (or cylindrical) form, to cartesian coordinates  $x_i$ ,  $x_j$ . The conversion is performed using the table below where 'ntype' denotes the coordinates  $i$  and  $j$  for which the conversion takes place. 'ntype = 0' is the default value for two-dimensional meshes.

ntype	=	0 or 12	$x_1 - x_2$ -plane
$r$	=	$x(1,\text{node})$	input value of radius
$\theta$	=	$x(2,\text{node})$	input value of angle in degrees
$x(1,\text{node})$	=	$x_{01} + r \cdot \cos(\theta)$	
$x(2,\text{node})$	=	$x_{02} + r \cdot \sin(\theta)$	
ntype	=	13	$x_1 - x_3$ -plane
$r$	=	$x(1,\text{node})$	input value of radius
$\theta$	=	$x(3,\text{node})$	input value of angle in degrees
$x(1,\text{node})$	=	$x_{01} + r \cdot \cos(\theta)$	
$x(3,\text{node})$	=	$x_{03} + r \cdot \sin(\theta)$	
ntype	=	23	$x_2 - x_3$ -plane
$r$	=	$x(2,\text{node})$	input value of radius
$\theta$	=	$x(3,\text{node})$	input value of angle in degrees
$x(1,\text{node})$	=	$x_{02} + r \cdot \cos(\theta)$	
$x(2,\text{node})$	=	$x_{03} + r \cdot \sin(\theta)$	

As can be seen from the equations in the table above,  $x_{0i}$  and  $x_{0j}$  are the cartesian coordinates of the origin of the polar coordinate system.

A sequence of nodes may be converted by specifying non-zero values for 'node1', 'node2', and 'inc'. The sequence generated will be:

node1, node1+inc, node1+2\*inc, ... , node2

Several records may follow the **pola** command. Execution terminates with a blank record.

## PRES

---

**pres**

segm1, (p1(i,segm1),i=1,ndf), (p2(i,segm1),i=1,ndf)  
segm2, (p1(i,segm2),i=1,ndf), (p2(i,segm2),i=1,ndf)  
<etc., terminate with blank record>

---

The **pres** command is used to specify the values for loads at segments. For each segment to be specified a record is entered with the following information:

segm	– the number of segment to be specified
p1(1,segm)	– value of load for 1-dof at 'node 1'
p1(2,segm)	– value of load for 2-dof at 'node 1' – etc. until to 'ndf' directions
p2(1,segm)	– value of load for 1-dof at 'node 1'
p2(2,segm)	– value of load for 2-dof at 'node 2' – etc. until to 'ndf' directions

**Remarks:**

1. A trapezoidal load is defined from the nodal values.
2. Loads are acting in the direction of defined dofs of freedom.
3. The reference length is the real length of the segment.
4. The load case warping is defined by p1(1,1)=-999.
5. For mesh generation with NEGE the following order of statements is necessary: **nege** , **[neco]**, **[back]**, **geom**, **segm**, **regi**, **[pres]**, **[fixe]**, further FEAP-macros. Note: macros in brackets are optional.

## **PRIN**

---

### **prin**

---

The use of the **prin** macro will cause the description of all information produced during MESH description to be written on the output file. The use of **nopr** will discontinue the output of mesh information (except for data printed in elements). The default value is **prin**.

## QLOA

---

**qloa**  
ma,q01,q02,q03,q04,q05,q06,q07,q08,q09,q10  
elema,elemb,incr  
<etc., terminate element input with a blank record>  
<etc., terminate load value input with a blank record>

---

With **qloa** up to 10 load parameters are set which act on the specified elements associated with material number *ma*. Elements are generated from element *a* until element *b* with increment *incr*. This input could be repeated until all elements are defined. Each element could be loaded more than once (if this makes sense!).

This macro could be used only **after** the definition of elements and materials! The definition and use of the load parameters depend on the used element and can be found in the element description:

elmt01 : -  
elmt02 : *ma, q<sub>1</sub>, q<sub>2</sub>, n<sub>1</sub>, n<sub>2</sub>, ifol, T<sub>u</sub>, T<sub>o</sub>*  
elmt03 : *ma, q<sub>1</sub>, q<sub>2</sub>, n<sub>1</sub>, n<sub>2</sub>*  
elmt04 : *ma, q<sub>xL</sub>, q<sub>yL</sub>, q<sub>zL</sub>, q<sub>xG</sub>, q<sub>yG</sub>, q<sub>zG</sub>*  
elmt05 : *ma, b<sub>1</sub>, b<sub>2</sub>*  
elmt06 : -  
elmt07 : *ma, q, ΔT*  
elmt08 : -  
elmt09 : *ma, b<sub>1</sub>, b<sub>2</sub>, b<sub>3</sub>, b<sub>x</sub>, b<sub>y</sub>, b<sub>z</sub>*  
elmt10 : -  
elmt11 : *ma, q<sub>x</sub>, q<sub>y</sub>, q<sub>z</sub>*  
elmt12 : -  
elmt13 : *ma, q*  
elmt14 : -  
elmt15 : *ma, y<sub>p</sub>, z<sub>p</sub>, q<sub>X</sub>, q<sub>Y</sub>, q<sub>Z</sub>*  
elmt16 : -  
elmt17 : *ma, q*  
elmt18 : *ma, q*  
elmt19 : *ma, q*  
elmt20 : *ma, q<sub>1</sub>, q<sub>2</sub>*  
elmt21 : *ma, q<sub>x</sub>, q<sub>y</sub>, q<sub>z</sub>, ΔT*  
elmt22 : *ma, q<sub>x</sub>, q<sub>y</sub>*  
elmt30 : *ma, q<sub>1</sub>, q<sub>2</sub>, q<sub>3</sub>, ltyp*

For details please check the [element library](#).

An associated load vector will be calculated on element level(isw=22). In nonlinear cases the actual load factor **prop** is used. Nonlinear effects like e.g. dependency on displacements or temperature (modification of stiffness matrix) are possible if coded on element level. In such cases the macro **sloa** could also be used.

**RBOU****rbou**

$r_{min}, r_{max}, \phi_{min}, \phi_{max}, < z_{min}, z_{max} >, (\text{ibc}(j), j=1, \text{ndf})$   
 <etc., terminate with a blank record>

This macro is used if coordinates are given in cartesian directions. Otherwise use macro **vrou**

The boundary restraint conditions may be set in a two/three dimensional region which is defined by

$r_{min} \leq r \leq r_{max}$
$\phi_{min} \leq \phi \leq \phi_{max}$
$< z_{min} \leq z \leq z_{max} >$

The data to be supplied during the definition of the mesh (or in macro execution using the 'mesh' command) consists of:

$r_{min}$	-	minimum value of radius $r = \sqrt{(x^2 + y^2)}$
$r_{max}$	-	maximum value of radius $r = \sqrt{(x^2 + y^2)}$
$\phi_{min}$	-	minimum value of angle $\phi$ in degree with $\phi_{min} \geq 0$
$\phi_{max}$	-	maximum value of angle $\phi$ in degree with $\phi_{max} \geq 0$
$< z_{min}$	-	minimum value of coordinate 3=z >
$< z_{max}$	-	maximum value of coordinate 3=z >
ibc(1)		
ibc(2)		restraint conditions for all nodes with the
...		value of the search (0 = active dof).
ibc(ndf)		> 0 or < 0 denotes a <i>fixed</i> dof).

**Remarks:**

- Up to 16 values and up to 80 characters are possible on each input record.
- boun** sets the boundary conditions whereas **ebou**, **vrou**, **gbou** and **rbou** adds boundary conditions to existing values.
- There is no tolerance in the defined block values. Thus nodes defined by block may lay not in the block. In this case modify the max and min values.

For the specification of prescribed displacements within a calculation, see macro commands **disp**, **load**.

## RNDM

---

### rndm

val, ntype  
node1, ngen1  
node2, ngen2  
<etc., terminate with blank record>

---

The **rndm** command is used to modify the values for nodal coordinates with a random parameter ( $0 < rndm < 1$ ) times a given value.

X=X+2\*(rndm-0.5)\*val

For each node to be specified a record is entered with the following information:

val	— value to be added on coordinates by function random
ntype	— 1 - produce every time different random numbers(def.) — 2 - produce every time same random numbers
node1	— the number of first node to be specified
node2	— the number of last node to be specified
ngen	— the increment to the next node, if generation is used, otherwise 0.

When generation is performed, the node number sequence will be (for node1-node2 sequence shown above):

node1, node1+ngen1, node1+2\*ngen1, .... , node2

Be careful with the use of this macro. Due to the modification of the coordinates macros like **eloa**, **edge** or **tie** may not work. Thus use for example **eloa** before **rndm**.

## RSUM

---

```
rsum
dofrsum,<rstyp>
for rstyp=1
node1, node2, ngen1
nodei, nodek, ngeni
<etc., terminate with blank record>
OR
for rstyp=2
i-dir,xi-value
```

---

The **rsum** command is used to specify the values for a sum of nodal reactions for the degree of freedom 'dofrsum'. **Alternatively** 2 options are possible

- **rstyp=1**(default)

For each node to be specified a record is entered with the following information:

node1 – the first number of node to be specified
node2 – the last number of node to be specified
ngen – the increment to the next node

node2 and ngen1 are optional.

When generation is performed, the node number sequence will be (for node1-node2 sequence shown above):

node1, node1+ngen1, node1+2\*ngen1, .... , node2

The intermediate nodes will be a linear interpolation between 'node1' and 'node2'.

- **rstyp=2**

i-idir – the direction of the coordinate (i.e., 1 = x, 2 = y, etc. to ndm) to be searched.
xi-value – the value of the i-direction coordinate to be used during the search (a tolerance of about $1/1000 \cdot \sqrt{\text{numnp}}$ of the mesh size is used during the search.)

A graphical control is possible using the macro **rsum** in Plot-modus.

The resulting reaction force R is always printed using the macro **reac**. When using the macro **tplo** this resulting force R is plotted versus displacement or time in case of reac or reat.

## REGI

---

### regi

regi1, nregi1, (segm1(i),i=1,nregi1)  
regi2, nregi2, (segm2(i),i=1,nregi2)  
<etc., terminate with blank record

---

The **regi** command is used to specify the segments which are attached to each region. For each region to be specified, a record is entered with the following information:

regi1	— the number of region to be specified.
nregi1	— the number of segments for the region to be specified.
segm1(1)	— segment 1 attached to region regi.
segm1(2)	— segment 2 attached to region regi.

### Remarks:

1. Each region has its own material. If a segment is used against its defined direction, this segment has to be marked by a Minus sign. Each segment must occur for the first time positive! Thus, modify the segment definition if necessary.
2. Regions are defined as a number of sections along its boundaries. **Outside**-boundaries have to be defined **counterclockwise**, whereas **Inside**-boundaries have to be defined **clockwise**!
3. There must not be a direct connection between segments.
4. Up to 16 values are possible on each input record, then use continuation line.
5. For mesh generation with NEGE the following order of statements is necessary: **nege** , **[neco]**, **[back]**, **geom**, **segn**, **regi**, **[pres]**, **[fixe]**, further FEAP-macros. Note: macros in brackets are optional.

## ROT

---

**rot**

node1,node2,inc,iax,ang  
<terminate with blank record>

---

The **rot** command may be used to change cartesian coordinates by a defined rotation ang(deg) around axis iax.

$$x(i, node) = \mathbf{T}(iax, ang) \cdot x(i, node)$$

A sequence of nodes may be converted by specifying non-zero values for node1, node2, and inc. The sequence generated will be:

node1, node1+inc, node1+2\*inc, ... , node2

Several records may follow the **rot** command. Execution terminates with a blank record.

A translation of coordinates is possible with **tran**.

**SEGM****segm**

segm1, (ix(i,segm1),i=1,2), < mtyp, (g(i,segm1),i=1,4) >  
 segm2, (ix(i,segm2),i=1,2), < mtyp, (g(i,segm2),i=1,4) >  
 <etc., terminate with blank record>

The **segm** command is used to specify the values of nodal numbers which are attached to each segment. For each segment to be specified, a record is entered with the following information:

segm	– the number of the segment to be specified.
ix(1,segm)	– node-1 number attached to segment.
ix(2,segm)	– node-2 number attached to segment.

only for curved segments:

mtyp	– 2 = circle (midpoint $P_x$ , $P_y$ , radius $R = R_x$ ). – 3 = ellipse (midpoint $P_x$ , $P_y$ , radii $R_x$ , $R_y$ ).
g(1,segm)	– $P_x$ .
g(2,segm)	– $P_y$ .
g(3,segm)	– $R_x$ .
g(4,segm)	– $R_y$ (not used for mtyp=3).

mtyp	– 6 = circle like 2, but automatically finding of coordinates of end points of the segment and of midpoint of the circle. Assume for this purpose in <b>geom</b> that both nodes have the coordinates of the intersection of the segments before and after the circle.
g(1,segm)	– $P_1$ node number on segment before circle.
g(2,segm)	– $P_2$ node number on segment after circle.
g(3,segm)	– $R$ radius of circle.

**Remarks:**

1. Each segment has a 'direction' from node 1 to node 2. This has to be taken into account in the definition of regions. See the macro **regi**.
2. For mesh generation with NEGE the following order of statements is necessary: **nege**, **[neco]**, **[back]**, **geom**, **segm**, **regi**, **[pres]**, **[fixe]**, further FEAP-macros. Note: macros in brackets are optional.

## SLOA

---

**sloa**

iel,ns,nv,nl  
(ixl(i),i=1,ns), (p(i),i=1,nv), ngen, lgen  
<etc., terminate with blank record>

---

The **sloa** command is used to specify the values for surface loading quantities. Only traction quantities are considered (e.g., no surface displacement distributions may be specified by **sloa**). The nodal values for the loads are determined by each element subprogram (i.e., in 'ELMTnn' with the isw = 7. Here for example the associated load vector and the load stiffness matrix have to be computed).

Datas are specified as follows:

iel	– element subprogram which generates surface loads (only one routine may be given for a problem).⇒ FEAP element number
ns	– number of nodes on surface of element.
nv	– number of parameters defining distributed loading.
nl	– loading type (generally only one type is currently included in elements and 'nl' is ignored – default may be 0).
	Next input for all loaded elements
ixl(i)	– list of nodes on element surface.
p(i)	– list of parameters defining loading.
ngen	– increment for node generation (default 0).
lgen	– number of repetitions (default 0).

A maximum of 8 items can appear on each record. If more than 8 items are required continue on the next record.

## SOLV

---

### solv

n0,<n1,n2,n3>

---

Choose an appropriate solver for **FEAP** solutions by defining 'n0' For the iterative solvers input of 'n1'- 'n3' is possible. Otherwise the default values are be used. These values could also be changed using the macros **pbcg** and **pgmr** in **macro**-mode.

n0		solver	storage technique	n1	n2	n3	n4
0	sym/unsym.	standard profile solver [default]	columnheight				
1	symmetric	sparse matrix solver (SM) (without minimum degree)	CSR CSR				
2	symmetric	sparse matrix solver (SM) (with minimum degree)	CSR				
3	sym/unymm.	SUPER-LU (sequential)	CSR				
4	sym/unsym.	PARDISO direct (parallel)	CSR	isym			
5	sym/unsym.	PBCG	CSR	niter	tol	itol	isym
6	sym/unsym.	PGMRES	CSR	niter	tol	im	isym
7	sym/unsym.	PGMRES	CSR	niter	tol	im	isym
8	symmetric	PARDISO ML (sequential)	CSR	niter	tol	impro	
9		Simplex optimization solver					

The standard **solver 0** is used for small problems (small neq) and allows symmetric **and** un-symmetric stiffness matrices based on an Gaussian elimination with a  $\mathbf{L}^T\mathbf{D}\mathbf{L}$  decomposition. A column-height technique is used, thus the efficiency of the solver depends strongly on the nodal and element numbering. The speed of the solution process can be improved using a node number optimization with **opti**-macro.

The other solvers are advantageous to solve large systems of equations and do not depend on node and element numbering.

For **symmetric problems** the **sparse matrix solver(SM)** (**n1=1,2**) or the **Pardiso solver**(n1=4) are preferable and very fast. Here a  $\mathbf{L}^T\mathbf{D}\mathbf{L}$  decomposition is used. When storing the stiffness matrix only non zero entries are taken into account. This called a compressed sparse row format (CSR). In addition the Pardiso solver is running parallel and is furthermore to deal with **unsymmetric problems**.

For very large systems direct solvers could not compete with iterative solvers. For this purpose two solvers are implemented. These solvers do not need a triangular decomposition, which is the most time consuming part in a solution process of a large system of equations. As a consequence the system of equations is not solved directly but in a set of iterations. The number of iterations depends strongly on the preconditioning (**prco**)and can be in the range of 10 to 2000. Furthermore - to be efficient - a CSR sparse storage procedure is used while calculating the stiffness matrix. These solvers could be used for **symmetric and un-symmetric** matrices without any time differences!

**Solver 3** is the direct SUPER-LU solver. This is a symmetric and unsymmetric sparse matrix solver. The solver package for the **LU** decomposition is written in C and use its own storage management. For storage efficiency the CSR-storage technique is used. The solver is available only for the SALFORD-Version.

**Solver 4** is the PARDISO **direct parallel** (shared memory) solver. This is a symmetric and unsymmetric sparse matrix solver. The solver package for the **LU** decomposition use its own storage management. For storage efficiency the CSR-storage technique is used. The solver is available only for the INTEL-Version or the INTEL-Basel-Version.

Input is

- The parameter 'n1' = 'isym' (Def = 1) sets the type of matrix: 1 = symmetric , 2 = unsymmetric.

**Solver 5** is based on the Pre-conditioned bi-Conjugated Gradient method (PBCG). Three versions of preconditioning are available with the macro **prco**.

Input is

- The parameter 'n1' = 'niter' (Def = 150) denotes the maximum numbers of iterations. Typical values are in the range of 50-1000.
- The parameter 'n2' = 'tol' (Def =  $1 \cdot 10^{-8}$ ) sets the tolerance value.
- The parameter 'n3' = 'itol' (Def = 1) sets the type of tolerance criterion (1-4).
- The parameter 'n4' = 'isym' (Def = 1) sets the type of matrix: 1 = symmetric , 2 = unsymmetric.  
Parameters 'n1', 'n2', 'n3' can be reset with the macro **pbcg,n1,n2,n3**.

**Solver 6** is based on the Pre-conditioned Generalized Minimum Residual Method (PGMRES) and should be more robust but needs more work space for the iterative solution. Three versions of preconditioning are available with the macro **prco**.

- The parameter 'n1' = 'niter' (Def = 150) denotes the maximum numbers of iterations. Typical values are in the range of 10-500.
- The parameter 'n2' = 'tol' (Def =  $1 \cdot 10^{-8}$ ) sets the tolerance value.
- The parameter 'n3' = 'im' (Def = 50) sets the number of Krylov iteration vectors. Typical values are in the range of 50-100.
- The parameter 'n4' = 'isym' (Def = 1) sets the type of matrix: 1 = symmetric , 2 = unsymmetric.  
Parameters 'n1', 'n2' can be reset with the macro **pgmr,n1,n2**.

**Solver 7** is an alternative version of the PGMRES solver.

**Solver 8** is the PARDISO ML (multi-level) solver. This is a symmetric **iterative sequential** solver. The solver package for the **LU** decomposition use its own storage management. For storage efficiency the CSR-storage technique is used. The solver is available only for the INTEL-Version and PARDISO-Basel.

Input is

- The parameter 'n1' = 'niter' (Def = 300) denotes the maximum numbers of iterations. Typical values are in the range of 50-1000.
- The parameter 'n2' = 'tol' (Def =  $1 \cdot 10^{-6}$ ) sets the tolerance value.
- The parameter 'n3' = 'impro' (Def = 25) sets the maximum numbers of iterations without improvement. Then the solver will stop with error 101.

**Solver 9** is an optimization solver based on the SIMPLEX-algorithm.

**Remarks:**

1. Do not use **opti** together with SM/SUPERLU/PARDISO/PBCG/PGMRES-solvers.
2. Solvers 1,2,8 need symmetric matrices, solvers 0,3,4,5,6,7 allow also un-symmetric matrices.
3. Recommendations: use for symmetric small problems solver 0 with **opti** or for large systems solver 2,3,4 and for un-symmetric problems solver 0 with **opti** or for large systems solver 3,4. In case of 3d-structures use always for large problems the iterative solvers 5,6,7,8 or the parallel solver 4.
4. solvers 1-8 not allowed with **cont**act, **ext**ended system,
5. calculation of  $\det \mathbf{K}_T$  and negative diagonals

n0		solver	$\det \mathbf{K}_T$	neg. diags	
0	sym/unsym.	standard profile solver [default]	x	x	
1	symmetric	sparse matrix solver (SM) (without minimum degree)	x	x	
2	symmetric	sparse matrix solver (SM) (with minimum degree) (fast version of 1)	x	x	
3	sym/unsym.	SUPER-LU (sequential)	x	x	
4	sym/unsym.	PARDISO direct(parallel)		x	Intel
4	sym/unsym.	PARDISO direct(parallel)	x	x	Basel
5	sym/unsym.	PBCG	-	-	
6	sym/unsym.	PGMRES	-	-	
7	sym/unsym.	PGMRES	-	-	
8	symmetric	PARDISO ML (sequential)	-	-	Basel

6. solvers 5,6: all free nodes need boundary conditions!! (solvers 4,9 not tested)
7. Infos SUPERLU: <http://crd.lbl.gov/~xiaoye/SuperLU/>
8. Infos PARDISO: <http://www.pardiso-project.org/>
9. Infos PBCG: <http://www.nr.com/> Numerical recipes
10. Infos PGMRES: <http://www-users.cs.umn.edu/~saad/books.html>

## SPHE

---

### sphe

node1,node2,inc, $x_{01}$ ,  $x_{02}$ ,  $x_{03}$   
<etc., terminate with blank record>

---

The **sphe** command may be used to convert any coordinates  $(r, \theta, \beta)$  which have been specified in spherical form, to cartesian coordinates  $(x_1, x_2, x_3)$ . The conversion is performed using the following relations:

$r$	=	$x(1,\text{node})$	— input value of radius
$\theta$	=	$x(2,\text{node})$	— input value of angle in degrees
$\beta$	=	$x(3,\text{node})$	— input value of angle in degrees
$x(1,\text{node})$	=	$x_{01} + r \cdot \cos(\theta) \cdot \sin(\beta)$	
$x(2,\text{node})$	=	$x_{02} + r \cdot \sin(\theta) \cdot \sin(\beta)$	
$x(3,\text{node})$	=	$x_{03} + r \cdot \cos(\theta)$	

Here  $x_{01}$ ,  $x_{02}$  and  $x_{03}$  are the cartesian coordinates of the origin of the spherical coordinate system.

A sequence of nodes may be converted by specifying non-zero values for node1, node2, and inc. The sequence generated will be:

node1, node1+inc, node1+2\*inc, ..., node2

Several records may follow the **sphe** command. Execution terminates with a blank record.

## **STOP**

---

**stop**

---

The last command in the mesh input must be **end**. Then special actions like searching for double nodes with **tie** or linking conditions with **link** are done. It follows the execution of the program, which is controlled by **macro** commands. The execution of these commands can be done in batch-modus **macro** or in interactive modus **inte**. In the first case the commands are defined in the input-file after the **macro** command, in the second case the commands are defined interactively.

**stop** is the final command in the input file, which stops the execution.

## TEMP

---

**temp**

node1, ngen1, t(node1)  
node2, ngen2, t(node2)  
<etc., terminate with blank record>

---

The **temp** command is used to specify the values for nodal temperatures. For each node to be specified a record is entered with the following information:

node	– the number of the node to be specified
ngen	– the increment to the next node, if generation is used, otherwise 0.
t(node)	– value of temperature for 'node'.

When generation is performed, the node number sequence will be (for node1–node2 sequence shown above):

node1, node1+ngen1, node1+2\*ngen1, .... , node2

The values for each temperature will be a linear interpolation between t(node1) and t(node2).

## TIE

---

```
tie,      ,<tol>
tie,all  ,<tol>
tie,node,<tol>,n1,n2
tie,mate,<tol>,n1,n2
tie,dir  ,<tol>,n1,n2
tie,line ,<tol>      and continuation line: x1,y2,<z1>,x2,y2,<z2>
tie,mlin,<tol>,n1,n2 and continuation line: x1,y2,<z1>,x2,y2,<z2>
tie,prin
tie,nopr
```

---

A mesh may be generated in FEAP in which are more than one node with the same coordinates. The **tie** command may be used to “connect” these nodes so that the same values of the solution will be produced at all nodes with the same initial coordinates. Repetitions of the command are allowed to define e.g. the connection of nodes on different lines.

Special options are possible to connect only nodes on a line, nodes in a direction etc. For the input of the necessary data tol,n1,n2 **no parameter input is allowed!!**

With ‘tol’ a user defined tolerance for **tie** could be set. The tolerance is defined as  $|\mathbf{x}_{max} - \mathbf{x}_{min}| \cdot 'tol'$ .  
The default value for ‘tol’ is  $10^{-3} \frac{1}{\sqrt{\text{numnp}}}$ .

With **tie,node**, the **tie** command acts only on nodes between node numbers ‘n1’ and ‘n2’.

With **tie,mate**, the **tie** command acts only on nodes of elements with material numbers ‘n1’ and ‘n2’.

With **tie,dir**, the **tie** command acts only on nodes which have the coordinate ‘n2’ in direction ‘n1’.

With **tie,line**, the **tie** command acts on a line between points 1 and 2. Input data for the line must follow directly on a continuation line with: x1,y2,<z1>,x2,y2,<z2>.

With **tie,mlin**, the **tie** command acts on a line between points 1 and 2 but only for materials ‘n1’ and ‘n2’. Input data for the line must follow directly on a continuation line with: x1,y2,<z1>,x2,y2,<z2>.

With **tie,prin**, all **tie** output active. The macro acts like **prin**.

With **tie,nopr**, all **tie** output is suppressed. The macro acts like **nopr**.

**Comment:** The summary of input data -as a final result of the meshing procedure- on the screen it is necessary that **prin** is activated.

To use the **tie** option the complete mesh must be defined first. After the **end** command for the mesh definition and before the **macro** or **interactive** command for defining a solution algorithm, the use of a **tie** statement will cause the program to search for all coordinates that are to be connected together.

When nodes are connected any specified, restrained boundary condition will be assigned to all interconnected nodes. Thus, it is only necessary to specify restrained boundary conditions for one of the nodes.

If tied nodes are loaded then the sum of all loads is set on all nodes. This can occur e.g. for **aloa**.

$F_i = F_1, \quad F_k = F_2 \rightarrow$  loads are added and set:  $F_i = F_1 + F_2, \quad F_k = F_1 + F_2$

Tied nodes are printed when starting FEAP. A further control is possible in **plot** modus with macro **tie**.

## TRAN

---

**tran**  
node1,node2,inc, $dx_1, dx_2, dx_3$   
<terminate with blank record>

---

The **tran** command may be used to change cartesian coordinates by a defined translation.

$$x(i, \text{node}) = x(i, \text{node}) - dx_i$$

A sequence of nodes may be converted by specifying non-zero values for node1, node2, and inc. The sequence generated will be:

node1, node1+inc, node1+2\*inc, ... , node2

Several records may follow the **tran** command. Execution terminates with a blank record.

A rotation around a certain axis is possible with **rot**.

## TRIB

---

### trib

nodes,r-inc,node1,[elmt1-,mat]  
1, x1, y1, z1 (only ndm coordinates required)  
2, x2, y2, z2  
3, x3, y3, z3

---

The **trib** data input segment is used to generate a regular triangular patch consisting of triangular elements for two dimensional patches or three dimensional surfaces.

The patch of nodes/elements defined by **trib** is developed from a master triangle which is defined by an isoparametric 3-node mapping function in terms of the natural coordinates.

The spacing between the  $r$ -increments is equal.

Patches may be interconnected. This can be done by using the **tie** macro to 'connect' any nodes which have the same coordinates. The data parameters are defined as:

nodes	— number of master nodes needed to define the patch.
$r$ -inc	— number of nodal increments to be generated along $r$ -direction and $s$ -direction of the patch.
node1	— number to be assigned to first generated node in patch (default = 1). First node is located at same location as master node 1. Last generated node (i.e., $\text{node1} - 1 + (r\text{-inc} * (r\text{-inc} + 1)) / 2$ ) is located at same location as master node 3.
elmt1	— number to be assigned to first element generated in patch; if zero no elements are generated (default = 0)
matl	— material number to be assigned to all generated elements in patch (default = 1)

The generated number of elements is:  $\text{numel} = r^2$ . The generated number of nodes is:  $\text{numnp} = (r + 1) \cdot (r + 2) / 2$ .

**VANG****vang**

angl1,angl2,itrot,inpc  
 $x_{1min}, x_{1max}, x_{2min}, x_{2max}, < x_{3min}, x_{3max} >$   
<etc., terminate with a blank record>

---

**vang** define a local base system in a region. Angles may be set for the following directions

angl1	– Number of first axis of rotated basis (default=1)
angl2	– Number of second axis of rotated basis (default=2)
itrot	– 0/1 with/without rot. dofs 4-6 (default=0)
inpc	– 1=cartesian input calculate $\phi$ (default=1)
inpc	– 2=cartesian input calculate $\phi - 90$
inpc	– 3=polar input calculate $\phi$
inpc	– 4=polar input calculate $\phi - 90$

in a two/three dimensional region which is defined by

$x_{1min}$	$\leq$	$x_1$	$\leq$	$x_{1max}$
$x_{2min}$	$\leq$	$x_2$	$\leq$	$x_{2max}$

**Remarks:**

1. Only one choice of axis angl1, angl2 and itrot is possible.
2. A transformation will be done for dofs angl1 and angl2.
3. In case of ndf=6 the same transformation is chosen for dofs angl1+3 and angl2+3 for itrot=0.
4. The associated dofs of the used element can be chosen via the macro **mate**, parameters idfg. As example the angles of the DKQ-plate element can be transformed via  
mate  
1,7,2,3,1
5. Results are printed and plotted for nodes with  $\text{angl} \neq 0$  in directions angl1,angl2. Thus mixed vectors occur.
6. Results which base on a smoothing procedure are plotted in global directions. Thus results for nodes with  $\text{angl} \neq 0$  are transformed to global directions. Ex.: **disp**, **resi**, **eigv**,...

## VBOU

---

### vrou

$x_{1min}, x_{1max}, x_{2min}, x_{2max}, < x_{3min}, x_{3max} >, (ibc(j), j=1, ndf)$   
<etc., terminate with a blank record>

---

The boundary restraint conditions may be set in a two/three dimensional region which is defined by

$$\begin{array}{l} x_{1min} \leq x_1 \leq x_{1max} \\ x_{2min} \leq x_2 \leq x_{2max} \\ < x_{3min} \leq x_3 \leq x_{3max} > \end{array}$$

The data to be supplied during the definition of the mesh (or in macro execution using the 'mesh' command) consists of:

$x_{1min}$	—	minimum value of coordinate 1
$x_{1max}$	—	maximum value of coordinate 1
$x_{2min}$	—	minimum value of coordinate 2
$x_{2max}$	—	maximum value of coordinate 2
$< x_{3min}$	—	minimum value of coordinate 3 >
$< x_{3max}$	—	maximum value of coordinate 3 >
ibc(1)		
ibc(2)		restraint conditions for all nodes with the
...		value of the search (0 = active dof).
ibc(ndf)		> 0 or < 0 denotes a <i>fixed</i> dof).

### Remarks:

1. Up to 16 values and up to 80 characters are possible on each input record.
2. **boun** sets the boundary conditions whereas **ebou** and **vrou** adds boundary conditions to existing values.
3. There is no tolerance in the defined block values. Thus nodes defined by block may lay not in the block. In this case modify the max and min values.
4. Coordinates  $x_i$  may be defined in any coordinate system.

**YBOU**

---

**ybou**

$a_1, a_2, (a_3), b_1, b_2, (b_3)$ , bc,  $m_{pl,l}, m_{pl,u}$   
<etc., terminate with a blank record>

---

**ybou** defines boundary constraints in FEAP for yield line calculations.

- Each boundary restraint condition is defined by a line between  $P_a(a_1, a_2, (a_3))$  and  $P_b(b_1, b_2, (b_3))$ .
- 'bc' defines the boundary condition of the plate's edge. 'bc' = 0 signifies a free edge, 'bc' = 1 is a supported edge.
- ' $m_{pl,l}$ ' is the lower plastic moment, ' $m_{pl,u}$ ' is the upper plastic moment at the considered edge. I.e. the caused tension cracks will appear at the upper side of the plate when ' $m_{pl,u}$ ' is meant.

**Remarks:**

1. Additionally the usual degrees of freedom belonging to the boundary have to be given by the regular FEAP command **edge**. In this way the degrees of freedom for the optimization search will be directed.
2. Even when a free edge exists ('bc' = 0), plastic moments can be defined. In this way it is possible to map symmetrical problems being cut along the axis of symmetry.

## **YCON**

---

### **ycon**

---

This macro is used for mesh generation with CYLT and is similar to the macro **para**. All definitions which are defined by **ycon** hold for the mesh generation and FEAP.

#### **Remark:**

For mesh generation with CYLT the following order of statements is necessary: **cylt** , **[ycon]**, **ynod**, **yedg**, further FEAP-macros (including **ybou**).

Note: macros in brackets are optional.

## YEDG

---

### yedg

edge1, node1, node2, ang  
<etc., terminate with blank record>

---

The **yedg** command is used within mesh generation with CYLT to specify the values of nodal numbers which are attached to each edge of a yield-line plate. For each edge to be specified, a record is entered with the following information:

edge1	– the number of the edge to be specified.
node1	– node–1 number attached to edge.
node2	– node–2 number attached to edge.
ang	– rotation of plane: 'ang' = 1 – angle is 45° for supported edges 'ang' = 0 – angle is 90° for free edges

### Remarks:

1. The angles defined by 'ang' specify the rotation of the planes defining the lines of intersection and ruling the yield-lines, see **CYLT**-manual.

**YLOA**

---

**yloa** $a_1, a_2, b_1, b_2, c_1, c_2$  $q_1, q_2, q_3$ 

&lt;etc., terminate with blank record&gt;

The **yloa** command is used to specify the values of loads on triangular yield-line elements for points, lines and areas.

- The first line per entry defines the nodes of the load, e.g.  $P_a(a_1, a_2)$ ,  $P_b(b_1, b_2)$  and  $P_c(c_1, c_2)$  defining a triangular area.
- The second line gives the extrapolated loads for each node.

**Remarks:**

1. The current definition is a preliminary version. The loads have to be extrapolated by the user.
2. As zero loads are not allowed, FEAP is able to check the zero load entries in the second line and defines the type of load:  
 $P_a$  with  $q_a$  defines a single load,  
 $P_a, P_b$  with  $q_a, q_b$  defines a linear load,  
 $P_a, P_b, P_c$  with  $q_a, q_b, q_c$  defines an area load.
3. It must be assured that the triangulated yield-line mesh provides a node at the required load point.

**YNOD**

---

**ynod**

node1, (x(i,node1),i=1,2)  
node2, (x(i,node2),i=1,2)  
<etc., terminate with blank record>

---

The **ynod** command is used within mesh generation with CYLT to specify the values of nodal coordinates for edge nodes of yield-lines (using **yedg**). For each node to be specified a record is entered with the following information:

node	– the number of the node to be specified
x(1,node)	– value of coordinate in 1-direction for 'node'
x(2,node)	– value of coordinate in 2-direction for 'node'

# Chapter 3

## Macro Commands

### 3.1 Available Macros

The following entries `xxxx,yyyy,v1,v2,v3` are available for the control of the calculation: (where `xxxx` is selected from the following list)

acce	arcl	augm	auto	back	bfgs	chec	cmas	cont	conv
copy	crit	curv	damp	data	debu	detk	dibc	disp	dt
dplo	eigi	eigk	eigl	else	end	endi	epsq	erro	exit
ext	feas	form	four	fsol	fsum	geom	help	hist	iden
if	iimp	init	jint	lamb	lan	line	lmas	loop	macn
man	mate	mesh	newf	next	nonl	nopr	para	parv	paus
pbcg	pgmr	plot	pola	post	prin	prco	proc	prop	quit
reac	read	reme	rest	rinp	save	show	sigq	smoo	solv
splo	stre	subs	summ	tang	tec	time	tol	tplo	trans
ueig	umas	updh	utan	velo	writ	yang	yevo	ygra	ymsh
ytab	ytry								

The values `v1,v2,v3` can be predefined parameters. This can be done e.g. on input-level via `para` or on macro-level via `para`. Furthermore calculations for `v1,v2,v3` are possible, each up to 15 characters. For details see the description how to define `parameter`.

A short overview on commands is given in section 3.2, see macro `Index` in FEAP, whereas possible actions can be found in section 3.3, see macro `Action` in FEAP.

The solution algorithm used by FEAP to solve problems is defined by a “macro statement program”. By properly specifying the macro program a very wide range of applications may be addressed – including both linear and nonlinear, as well as, steady state and transient applications.

The description for the macro statements may be obtained from the manual entry for each command, (e.g., use manual `tang` to obtain all options and actions for the `tang` command).

### **3.2 Overview on commands**

Makro	Aufgabe
<b>acce</b>	Beschleunigungen
<b>arcl</b>	Bogenlängenverfahren
<b>augm</b>	Augmented Lagrange Verfahren
<b>auto</b>	Zeitschrittsteuerung automatisch
<b>back</b>	Zeitschritt rückgängig machen
<b>bfgs</b>	BFGS - Verfahren
<b>chec</b>	System - Test
<b>cmas</b>	Massenmatrix - konsistent
<b>cont</b>	Kontakt
<b>conv</b>	Konvergenzabfrage
<b>copy</b>	Kopiere Versch.vektor in Lastvektor
<b>crit</b>	Schädigung CFK
<b>curv</b>	Kurven für Lasten und Lagerungen
<b>damp</b>	Dämpfungsmatrix
<b>data</b>	Aenderung von Daten
<b>debu</b>	Debug - Option
<b>detk</b>	Determinante der Steifkeitsmatrix
<b>disp</b>	Verschiebungsausgabe
<b>dt</b>	Zeitschritt setzen
<b>dplo</b>	Setze Linie für Ausgabe
<b>eigi</b>	Eigenvektor (inverse Iteration)
<b>eigk</b>	Eigenvektor (alle mit RSG)
<b>eig1</b>	Alternative IF/ELSE/ENDIF (Batch)
<b>else</b>	Ende Makrokommmandos (Batch)
<b>end</b>	Ende IF/ELSE/ENDIF (Batch)
<b>endi</b>	Eingeprägte Verschiebungen aus E
<b>epsq</b>	Fehlerberechnung
<b>erro</b>	Ende Makrokommmandos (interaktiv)
<b>exit</b>	Extended System
<b>feas</b>	Eigenvektorberechnung FEAST
<b>form</b>	Lastvektor
<b>four</b>	Fourierreihe
<b>fsol</b>	Fourierlösung
<b>fsum</b>	Fouriersummation
<b>geom</b>	Geometrische Matrix
<b>help</b>	Hilfe
<b>hist</b>	History, Ablauf der Makros
<b>iden</b>	Einheitsmatrix

Makro	Aufgabe
<b>if</b>	Alternative IF/ELSE/ENDIF (Batch)
<b>iimp</b>	Felder auf Elementebene
<b>init</b>	Dynamische Rechnung: Anfangswerte
<b>jint</b>	Materielle Kräfte, J - Integral
<b>lamb</b>	Beullastfaktor
<b>lan</b>	Eigenvektorberechnung LANCZOS
<b>line</b>	Lineare Berechnung
<b>lmas</b>	Massenmatrix - Lumped
<b>loop</b>	Schleifen - Anfang
<b>macn</b>	vom Anwender definierte Macros
<b>man</b>	Manual
<b>mesh</b>	Netzänderung
<b>mate</b>	Materialänderung
<b>newf</b>	Laständerung
<b>next</b>	Schleifen - Ende
<b>nonl</b>	Nichtlineare Berechnung
<b>nopr</b>	Datenausgabe deaktivieren
<b>para</b>	Definition von Konstantn
<b>parv</b>	Postprozessor PARAVIEW
<b>paus</b>	Pause
<b>pbcg</b>	PBCG - Verfahren
<b>pgmr</b>	PGMRES - Verfahren
<b>plot</b>	Graphik
<b>pola</b>	Polarkoordinaten
<b>post</b>	Postprozessor
<b>prin</b>	Datenausgabe aktivieren
<b>prco</b>	Vorkonditionierung
<b>proc</b>	Prozedur erstellen, abrufen
<b>prop</b>	Last - Zeit - Funktionen
<b>quit</b>	Programm - Abruch
<b>reac</b>	Reaktionskräfte
<b>read</b>	Datensatz lesen
<b>reme</b>	Netzverfeinerung
<b>rest</b>	Restart
<b>save</b>	Datensicherung
<b>show</b>	Zustandsinformationen
<b>sigq</b>	S, C für eingeprägte Verschiebungen aus E
<b>smoo</b>	Netzglättung
<b>solv</b>	Lösung
<b>splo</b>	Setze Linie für Ausgabe
<b>stre</b>	Spannungsausgabe
<b>subs</b>	Eigenvektorberechnung SUBSPACE
<b>summ</b>	Ueberlagerung von Versch./Sp.

Makro	Aufgabe
<b>tang</b>	Tang. Steifigkeitsmatrix (Symm.)
<b>tec</b>	Postprozessor TECPLOT
<b>time</b>	Zeit ändern
<b>tol</b>	Toleranz des Iterations - Fehlers
<b>tplo</b>	Kurvendiagramme
<b>trans</b>	Zeitabhängige Verfahren Anfangswerte
<b>ueig</b>	
<b>umas</b>	Massenmatrix - unsymmetrisch
<b>utan</b>	Tang. Steifigkeitsmatrix - unsymmetrisch
<b>u pdh</b>	Update $H_2 \rightarrow H_1$ for convergence
<b>velo</b>	Geschwindigkeiten
<b>writ</b>	Datensatz schreiben
<b>yang</b>	Knickwinkelausgabe Fliesslinien
<b>yevo</b>	Evolutionsstrategie Fliesslinien
<b>ygra</b>	Gradientenermittlung Fliesslinien
<b>ymsh</b>	Netz aktualisieren Fliesslinien
<b>ytab</b>	Simplex-Tableau lösen Fliesslinien
<b>ytry</b>	direkte Suche Fliesslinien

### 3.3 Overview on actions

Aufgabe	Makro	Aufgabe	Makro
Änderung von Daten	<b>data</b>	Hilfe	<b>help</b>
Anwender -Macros	<b>macn</b>	History, Ablauf der Makros	<b>hist</b>
Augmented Lagrange Verfahren - Update	<b>augm</b>	<b>IF/ELSE/ENDIF (Batch)</b>	<b>if</b>
Beschleunigungen	<b>acce</b>	<b>IF/ELSE/ENDIF (Batch)</b>	<b>else</b>
Beullastfaktor	<b>lamb</b>	<b>IF/ELSE/ENDIF (Batch)</b>	<b>endi</b>
BFGS - Verfahren	<b>bfgs</b>	J - Integral, Materielle Kräfte	<b>jint</b>
Bogenlängenverfahren	<b>arcl</b>	Knickwinkelausgabe Fliesslinien	<b>yang</b>
Dämpfungsmatrix	<b>damp</b>	Kontakt	<b>cont</b>
Datenausgabe aktivieren	<b>prin</b>	Konvergenzabfrage	<b>conv</b>
Datenausgabe deaktivieren	<b>nopr</b>	Kopiere Versch.vektor in Lastvektor	<b>copy</b>
Datensatz lesen	<b>read</b>	Kurvendiagramme	<b>tplo</b>
Datensatz schreiben	<b>writ</b>	Kurven für Lasten und Lagerungen	<b>curv</b>
Datensicherung	<b>save</b>	Laständerung	<b>newf</b>
Debug - Option	<b>debu</b>	Lastvektor	<b>form</b>
Definition von Konstanten	<b>para</b>	Last - Zeit - Funktionen	<b>prop</b>
Determinante der Steifikeitsmatrix	<b>detk</b>	Lineare Berechnung	<b>line</b>
direkte Suche Fliesslinien	<b>ytry</b>	Linie für Ausgabe	<b>dplo</b>
Dynamische Rechnung: Anfangswerte	<b>init</b>	Linie für Ausgabe	<b>splo</b>
Eigenvektorberechnung SUBSPACE	<b>subs</b>	Lösung	<b>solv</b>
Eigenvektorberechnung LANCZOS	<b>lan</b>	Massenmatrix - konsistent	<b>cmas</b>
Eigenvektorberechnung FEAST	<b>feas</b>	Massenmatrix - lumped	<b>lmas</b>
Eigenvektorberechnung RSG	<b>eig1</b>	Massenmatrix - unsymmetrisch	<b>umas</b>
Eigenvektor (inverse Iteration)	<b>eigi</b>	Manual	<b>man</b>
Einheitsmatrix	<b>iden</b>	Materialänderung	<b>mate</b>
eingeprägte Verschiebungen aus E	<b>epsq</b>	Netz aktualisieren Fliesslinien	<b>ymsh</b>
S, C für eingeprägte Verschiebungen aus E	<b>sigq</b>	Netzänderung	<b>mesh</b>
Ende Makrokommmandos (Batch)	<b>end</b>	Netzglättung	<b>smoo</b>
Ende Makrokommmandos (interaktiv)	<b>exit</b>	Netzverfeinerung	<b>reme</b>
Evolutionsstrategie Fliesslinien	<b>yevo</b>	Nichtlineare Berechnung	<b>nonl</b>
Extended System	<b>ext</b>	Pause	<b>paus</b>
Fehlerberechnung	<b>erro</b>	PBCG - Verfahren	<b>pbcg</b>
Felder auf Elementebene	<b>iimp</b>	PGMRES - Verfahren	<b>pgmr</b>
Fourierlösung	<b>fsol</b>	Polarkoordinaten	<b>pola</b>
Fourierreihe	<b>four</b>	Postprozessor	<b>post</b>
Fouriersummation	<b>fsum</b>	Postprozessor PARAVIEW	<b>parv</b>
Geometrische Matrix	<b>geom</b>	Postprozessor TECPLOT	<b>tec</b>
Geschwindigkeiten	<b>velo</b>	Programm - Abruch	<b>quit</b>
Gradientenermittlung Fliesslinien	<b>ygra</b>	Prozedur erstellen, abrufen	<b>proc</b>
Graphik	<b>plot</b>	Reaktionskräfte	<b>reac</b>
		Restart	<b>rest</b>

Schädigung CFK	<code>crit</code>
Schleifen - Anfang	<code>loop</code>
Schleifen - Ende	<code>next</code>
Simplex-Tableau lösen Fliesslinien	<code>ytab</code>
Spannungsausgabe	<code>stre</code>
System - Test	<code>chec</code>
Tang. Steifigkeitsmatrix (Symm.)	<code>tang</code>
Tang. Steifigkeitsmatrix (Unsymm.)	<code>utan</code>
Toleranz des Iterations - Fehlers	<code>tol</code>
Ueberlagerung von Versch./Spann.	<code>summ</code>
Verschiebungsausgabe	<code>disp</code>
Vorkonditionierung	<code>prco</code>
Update $H_2 \rightarrow H_1$	<code>updh</code>
Zeit ändern	<code>time</code>
Zeitabhängige Verfahren Anfangswerte	<code>trans</code>
Zeitschritt setzen	<code>dt</code>
Zeitschritt rückgängig machen	<code>back</code>
Zeitschrittsteuerung automatisch	<code>auto</code>
Zustandsinformationen	<code>show</code>

## 3.4 Macros in detail

### **ACCE**

---

**acce,,<n1,n2,n3>**  
**acce,all**

---

The macro command **acce** may be used to print the current values of the “acceleration” vector as follows:

1. Using the macro command:

**acce,,n1,n2,n3**

prints out the current acceleration vector for nodes 'n1' to 'n2' at increments of 'n3' (default = 1). If 'n2' is not specified only the value of node 'n1' is output. If both 'n1' and 'n2' are not specified only the first nodal acceleration is reported.

2. If the macro command is specified as:

**acce,all**

prints all nodal quantities.

In order to output a solution vector it is first necessary to specify macro commands to compute the desired values, i.e., a dynamic analysis.

## ARCL

---

**arcl,<xxxx>,n1,n2,n3**

---

The **arcl** macro computes an arclength solution with the following options:

xxxx	n1	n2	n3
	0 to 5	0 or 1	
		1	
modi			
add	e1	$\tau$	
impf	e1	$\xi$	
on			
off			
check	e1		
step	Id [6]	mono[1]	rm [0.5]

With **arcl,,n1,n2** the arclength method is initialized. 'n1' options area defined as follows:

n1 = 0	Normal plane, modified newton solution
n1 = 1	Updated normal plane, modified newton solution
n1 = 2	Normal plane, full newton solution
n1 = 3	Updated normal plane, full newton solution
n1 = 4	Displacement control, modified newton solution
n1 = 5	Displacement control, full newton solution
n2 = 0	Use current values for arclength and load direction (Initial default is calculated by first solution step).
n2 = 1	Change current values for arclength and load direction

With **arcl,modi** or **arcl,,1** the iteration parameters could be modified, due to the chosen version.

With **arcl,add** a scaled eigenvector is added. 'e1' denotes the number of the eigenvector to be included and  $\tau$  is a scaling factor such that

$$\mathbf{u} \leftarrow \mathbf{u} + \frac{|\mathbf{u}|}{|\boldsymbol{\phi}_{e1}|} \frac{1}{\tau} \boldsymbol{\phi}_{e1}$$

where  $\mathbf{u}$  is the current solution and  $\boldsymbol{\phi}_{e1}$  is the  $e1^{th}$  eigenvector.

With **arcl,impf** an eigenvector is added in the sense of an imperfection. 'e1' denotes the number of the eigenvector to be included and  $\xi$  is the maximum amplitude such that

$$\mathbf{u} \leftarrow \mathbf{u} + \xi \boldsymbol{\phi}_{e1}$$

where  $\mathbf{u}$  is the current solution and  $\boldsymbol{\phi}_{e1}$  is the  $e1^{th}$  eigenvector. Often  $\xi$  is chosen as thickness 'h'.

**Remarks:**

- **arcl** has to be called once at the beginning of each series of macro commands, when a nonlinear problem is to be solved using this method. With this call all flags will be set to perform an arc length solution. It should be noted that you can reset this flags to continue e.g. with pure newton method when you type **arcl,off**. The command **arcl,on** then switches back to the arc length procedure.
- **arcl** requires at this stage of implementation that the time increment **dt** is set to 1.0. This is done automatically. Thus no input of **dt** is necessary.
- For full and modified Newton methods the standard procedures for one step are:

time	time
loop,,N	tang,,1
tang,,1	loop,,N
next	form
	solv
	next

- In case of divergence make restart with the macro **back,,1**.
- For the calculation of load deflection curves we need to specify proportional load with **prop**. However you may use the default values of **prop**, since the actual load level is computed by **arcl**.
- If you like to perform a branch-switching you must calculate the eigenvectors associated with the bifurcation load first (You may use a shift on the tangent, see **tang**).
- The command **arcl,chec** tells you whether the stability point is a limit point or a bifurcation point. In case of a bifurcation point the value returned by **arcl,chec** should be zero up to a tolerance).

$$\varphi^T \mathbf{P} = \begin{cases} = 0 & \dots \text{Bifurcation point} \\ \neq 0 & \dots \text{Limit point} \end{cases}$$

- The branch-switching is initiated by the command **arcl,add,n1,n2** which adds the  $n1^{th}$  eigenvector  $\phi_{n1}$  to the current displacement field as shown above. n2 is the scaling factor  $\tau$ . In case n2 is zero a scaling factor is automatically computed using the formula

$$\tau = 100 \frac{\mathbf{u}^T \phi_{n1}}{\sqrt{(\mathbf{u}^T \mathbf{u})(\phi_{n1}^T \phi_{n1})}} + 1$$

- Note, that the command **arcl,add** can be used generally. Thus one has not to specify **arcl,kfl** before applying this command. This means that the command can be used also with e.g. standard load control analysis.
- After the addition of an eigenvector  $\phi_{n1}$  to the displacement field  $\mathbf{u}$  a new equilibrium state has to be computed on the secondary branch. This is simply obtained by the following macro series

loop,,N	time
tang,,1	tang,,1
next	next

- Within arclength solution techniques the load increments are modified automatically. Modifications can be initialized by **arcl,step**. The current increment size is

$$ds_0 = \alpha * ds_0(i - 1)$$

where  $ds_0(i - 1)$  is the increment size of the previous time step.

The modification factor  $\alpha$  is determined by

$$\alpha = (Id/I(i - 1))^{sp}$$

where Id (=n1) is a user defined number of iteration within a time step to achieve convergence, and I(i-1) is the number of iterations needed in the previous time step. The exponent sp is set to 0.5 [recommended  $0.5 \leq sp \leq 1.0$ ]. A large value for Id (=n1) will result in larger step sizes.

Within the iteration a monotone test is performed using the ratio of the incremental displacement norms of two subsequent iteration steps.

$$\theta_k = \|\Delta v_{(i+1)}\| / \|\Delta v_{(i)}\|.$$

If this ratio is larger than an user defined number mono(=n2) the time step is restarted with a reduced ds0 with the reduction factor

$$\alpha = \sqrt{\frac{rm*n2}{\theta_k}}.$$

The number rm is a safety factor (rm approx. 0.5) For certain problems (e.g. stability points) it might be necessary to use an large number for n2 (n2=3 to 5) in order to prevent very small step sizes.

The step control can be turned off by the macro sequence

**arcl,off**

**arcl,on**

**References** for theoretical background: J. C. Simo, P. Wriggers, K.- H. Schweizerhof, R. L. Taylor: Postbuckling Analysis involving Inelasticity and Unilateral Constraints, Int. J. Num. Meth. Engng. **23**, 779–800, 1986.

W. Wagner, P. Wriggers: A Simple Method for the Calculation of Secondary branches, Engineering Computations, **5**, 103–109, 1988.

## **AUGM**

---

### **augm**

---

The macro command **augm** updates the Lagrange parameter in a nested augmented iteration on element level using (isw = 10). This features is element depending. It can be used e.g. for incompressible rubber materials to improve incompressibility or for contact formulations.

## AUTO

```
auto
auto,load
auto,init,<htol,dtmax>
auto,para,<dtdo,dtup1,dtup2>
auto,impl,<eta, xsi, gamma >
```

Using the macro command **auto** leads to a variable length of the time increment within a dynamic analysis. Currently this macro is implemented for the time stepping algorithms **Newm**, **HHT** and **Alpha**.

The necessary parameter are defined as follows:

parameter	abbreviation	description	default
htol	$T$	tolerance value for residual	0.1
dtmax	dtmax	maximum desired time step length	<b>dt</b>
dtdo	$D_A$	reduction factor	0.85
dtup1	$D_G$	increasing factor 1	0.80
dtup2	$D_M$	increasing factor 2	1.25
eta	$\eta$	upper limiting factor	1.1
xsi	$\xi$	accuracy parameter	0.005
gamma	$\gamma$	lower limiting factor	0.5

**Theory:** The algorithm is based on the so called "Half-step-residual".

### General Remarks:

The macro has to be initialized with **auto, init**.

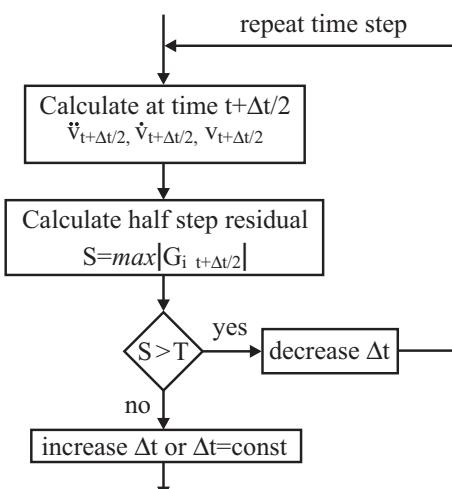
The tolerance htol is typically  $0.1 \cdot P_{max}$ . Here  $P_{max}$  is a maximum load value and could be calculated with **auto, load**.

With **auto, para,<dtdo,dtup1,dtup2>** the default defined parameters could be changed.

In a dynamic analysis the length of the next time increment will be checked by adding the **auto**-command at the end of each converged time step.

For small errors the **next** time step will be started with an unchanged or (conservatively) increased time increment, whereas for larger errors the time step has to be repeated with a reduced time increment.

### Main steps:



initialisation	time steps
<b>cmas</b> or <b>lmas</b> hline <b>dt,,xx</b> <b>prop</b> <b>trans</b> <b>auto,init</b> <b>&lt;auto,para&gt;</b>	<b>loop,,n</b> <b>time</b> <b>loop,,m</b> <b>tang,,1</b> <b>next(m)</b> <b>auto</b> <b>next(n)</b>

A more detailed description could be found in the [theory part](#) of this manual.

With **auto**, **impl,<eta,xsi,gamma>** an a priori time step length calculation for the IMPL-EX scheme (damage/plasticity) is performed according to

$$\Delta t_{n+1}^2 \leq \xi \alpha^{ref} \Delta t_n^2 \min \frac{1}{\left\| \Delta \lambda_n(x) - \frac{\Delta t_n}{\Delta t_{n-1}} \Delta \lambda_{n-1}(x) \right\|}$$

with  $\alpha^{ref} = \frac{\sigma_u}{\sqrt{E}}$  ( $= \alpha_0 = q_0$ ) in case of damage and  $\alpha^{ref} = \frac{\sigma_u}{E}$  ( $=$  uniaxial elastic strain) in case of elasto-plastic models. The increment of the internal variable is  $\Delta \lambda_n$  at time step  $n$ . The accuracy parameter  $\xi$  directly affects the next time step size. While the limiting parameter  $\eta$  affects the next time step size in case of an increasing step size according to

$$\Delta t_{n+1} \leq \eta \Delta t_n, \quad \Delta t_{n+1} \leq \Delta t_{max}$$

with  $\Delta t_{max}$  as the manually chosen value by the macro **dt,,**, **Δt**.

The last parameter  $\gamma$  performing a back step if the calculated time step size  $\Delta t_{n+1} \leq \gamma \Delta t_n$ .

#### Usage:

Only the above given 'time steps'-box is necessary.

#### Reference:

J. Oliver, A.E. Huespe, J.C. Cante: An implicit/explicit integration scheme to increase computability of non-linear material and contact/friction problems, Comput. Methods Appl. Mech. Engrg. 197 (2008) 1865-1889

**BACK**

---

**back,,<dtnew>**

---

The use of the **back** macro command will decrement the current time by **dt**, the current time increment. In addition, the previous value of the proportional loading will be recomputed, if necessary. The value of the current time and proportional loading are reported in the output (or to the screen). The **back** macro also will recompute the dynamic state at the old time for every time integration of the equations of motion, as well as, restore the stress data base for any elements with non-linear constitutive equations which require variables other than the displacement state to compute a solution.

Time steps before the current time step could not be reached. Thus only **ONE back** is allowed.

As an option, it is possible to specify a new time increment for integrations to be continued. The value of 'dtnew' is then used to perform the updates on the solutions in the same way as if the command **dt,,dtnew** were given. See manual on **dt** macro command for additional details.

Thus **back,,dtnew** is equal to the sequence

```
# back  
# dt,,dtnew
```

**Remark:**

- In case of an active macro **tplo** reaction forces have to be recalculated via **reac,all** for a correct plot of reaction forces.

## BETA

---

### **beta**

---

The macro command **beta** is obsolete. Please use **trans**.

## **BFGS**

---

**bfgs,,<n1,v2>**

---

The **bfgs** macro can be used to solve a nonlinear solutions step iteratively. The iterative procedure is based on a (BFGS) quasi Newton method which applies to problems with **symmetric** tangent matrices. The method involves updating the initial tangent matrix and provides a secant approximation to the tangent matrix. The convergence behavior is superlinear.

### **Remarks:**

1. Before executing the **bfgs** macro the initial tangent matrix must be computed via the **tang** macro.
2. The parameter 'n1' denotes the maximum number of iteration vectors (default and absolute maximum= 15).
3. The parameter 'v2' is the line search tolerance and may be chosen between 0.5 and 0.9. The default 0.8 will generally produce good performance.

## **CHEC**

---

### **chec**

---

The **chec** macro command requests a check of the mesh consistency. It is necessary for elements to have checking capability for the “isw = 2” option in order for **chec** to report results. Typical tests may include jacobian tests at nodes, tests on node sequencing, etc.

## **CMAS**

---

### **cmas**

---

The macro command **cmas** is used to compute a consistent (i.e., a non-diagonal) “mass” matrix. Each element computes a contribution to the consistent mass in the array ‘s’ when ‘isw’ is 5 and ‘imtyp’ eq.1. A consistent mass or a lumped mass (see macro command **lmas**) may be used for transient solutions computed using the Newmark method (see macro command **trans**). Both may also be used for eigencomputations (see macro command **subs**).

## CONT

---

**cont**,<off>,n,pen-n,pen-t

---

The **cont** macro is used to activate the contact logic during macro execution. It is necessary to describe the surfaces which may come into contact during the analysis when specifying the mesh data (i.e., see **icon** in the FEAP MESH USERS MANUAL).

The contact logic may be skipped during execution of a macro program (even though the contact surfaces are defined using **icon**) by:

1. never specifying a **cont** macro instruction, or
2. by specifying the **off** option in the second field.

The program will automatically readjust the profile of the equations during each computation of a 'tangent' matrix to account for the current configuration of contacting surfaces.

During execution it is possible to reset the values of the penalty parameters on any contact surface, 'n', to the values of 'pen-n' and 'pen-t' in normal and tangential direction. This permits the adjustment of the penalty parameter from a smaller to larger value during iterations. For problems in which large deformations occur the convergence to a solution may lead to a large number of iterations when large penalty parameters are involved. On the other hand, the use of a lower penalty parameter may result in unacceptable large penetrations across the contact surface. In these situations, it is recommended that the penalty parameter be adjusted to larger values during the iteration process in each load.

## References

Theoretical background:

J. C. Simo, P. Wriggers, K. Schweizerhof, R. L. Taylor: Postbuckling Analysis involving Inelasticity and Unilateral Constraints, Int. J. Num. Meth. Engng. **23**, 779–800, 1986.

Chosing the right penalty parameter:

B. Nour-Omid, P. Wriggers: A Note on the Optimum Choice for Penalty Parameters, Comm. Appl. Num. Meth, **3**, 581–585, 1987.

## CONV

---

```
conv,,< ξ >
conv,aini,<itinc,itdec,dtmin>
conv,auto,< ξdec,ξinc >
conv,aunr,<dtol,dtmin,dtmax>
```

---

The macro command **conv** may be used to check the convergence of a solution. If no convergence is achieved within a user defined number of equations the calculation is automatically set back to the original values via the **back**-macro without **time**. For a new time step the value of **dt** is reduced to ' $dt_{new} = \xi \cdot dt'$ , with  $0 < \xi < 1$ . If no value of  $\xi$  is set, the program asks the value in interactive mode.

With the macro command **conv,stop** the current loop is terminated.

**Example:**

```
# loop,,20
# time
# loop,iter,10
# tang,,1
# next
# conv,,0.5
# next
```

Within 20 time steps a nonlinear calculation is performed. '**conv,,0.5**' leads to a recalculation with half length time step, if no convergence is achieved within 10 increments.

**Remark:** For arclength method **conv** do not set 'dt' but the constant arclength ds0.

With the macro command **conv** it is possible to do a **bisection method** to find a limit point. Hereby the load step is only reduced if no convergence is achieved within a user defined number of equations. A consecutive bisection can be done in the following manner using to 2 blocks.

<b>block 1</b>	<b>block 2</b>
# loop,,20	# loop,,20
# time	# dt,,,0.5
# loop,iter,10	# loop,iter,10
# tang,,1	# tang,,1
# next	# next
# conv,,0.5	# conv,,1
# next	# next

The first block is used to follow the nonlinear solution path until the first divergence occurs. Then with the 2nd block the limit point can be found. For this purpose the step-length 'dt' is reduced in each time step. 'No convergence' is here defined as 'no solution within 10 iterations'.

When using the standard Newton-Raphson scheme, then the macro **conv** can be used to dynamically adjust the load/time step increment dt. Two possibilities are implemented. The first one is analogue to the implementation in Abaqus. The step size is chosen based on the convergence behavior of the previous step.

initialization	time steps
<b>dt</b> , <b>dt</b>	<b>loop</b> , <b>n</b> <b>time</b> <b>loop</b> , <b>m</b> <b>tang</b> , <b>1</b> <b>next</b> ( <b>m</b> ) <b>conv</b> , <b>auto</b>
<b>conv</b> , <b>aini</b>	<b>next</b> ( <b>n</b> )

To use this scheme for dynamic time/load step size, it is necessary to initialize the scheme with **conv,aini,<itinc,itdec,dtmin>**. With itinc the scheme increases **dt** to  $\xi_{inc} \cdot dt$  if the number of iterations of the previous step was smaller than itinc. Vice versa the scheme decreases **dt** to  $\xi_{dec} \cdot dt$  if the previous step needed more than itdec iterations.

To check the conditions it is necessary to perform **conv,auto,< $\xi_{dec},\xi_{inc}$ >** after each iteration loop. If the Newton-Raphson scheme do not converge and **dt** gets smaller than dtmin, the scheme tries the step again with dtmax = 2·dt (dt set by the **dt** macro). If it is still unsuccessful and dtmin is reached again, the load step loop will be terminated. The default parameters are:

**conv,aini,4,10,dt/1000**  
**conv,auto,0.75,1.5**

#### Reference:

Abaqus 6.12: Analysis User's Manual Volume II: Analysis, p. 7.2.3

The second method is using **conv,aunr**. Here no initialization is necessary.

time steps
<b>loop</b> , <b>n</b>
<b>time</b>
<b>loop</b> , <b>m</b>
<b>tang</b> , <b>1</b>
<b>next</b> ( <b>m</b> )
<b>conv,aunr</b>
<b>next</b> ( <b>n</b> )

The scheme chooses a value dt based on a norm of the displacement rates. The rates are specified by  $V_n = \Delta U_n / \Delta t_n$ . With these rates an error is defined as  $E_{t+\Delta t} = \frac{\Delta t}{2} \|\mathbf{V}_{t+\Delta t} - \mathbf{V}_t\|$  after this an relative error is defined by  $R_{t+\Delta t} = E_{t+\Delta t} / \|\mathbf{U}_{t+\Delta t}\|$ . If  $R \geq dtol$  a backstep is performed and dt is set  $q \cdot dt$ , with  $q = \max(0.8\sqrt{dtol/R_{t+\Delta t}}, 0.1)$ . In case of no convergence in the current step also a backstep is performed and dt is set to 0.25·dt. If the step converged and  $R < dtol$  then q is set to  $q = \min(0.8\sqrt{dtol/R_{t+\Delta t}}, 2.0)$  and dt is set again to  $q \cdot dt$ . The only default value for this scheme is dtol =  $10^{-3}$ . Additionally a maximum value for dt (dtmax) can be chosen to prevent unlimited growth of dt. And a minimum value for dt (dtmin) can be set, so the scheme stops the current loop if either the norm is not satisfied or the Newton-Raphson scheme do not converge and dt is already reduced to dtmin.

#### Reference:

Daichao Sheng, Scott W. Sloan, and Andrew J. Abbo: An Automatic Newton-Raphson Scheme, The International Journal of Geomechanics, Volume 2, Number 4, 471-502 (2002)

## COPY

---

**copy**

---

The **copy** macro may be used to transfer the current solution vector for displacements into the load vector. This option may be used in conjunction with results from a previous analysis and a problem with all degree-of-freedoms restrained to combine solutions for graphics outputs, etc.

## **CRIT**

---

### **crit**

---

The **crit** macro is up to now not documented.

## **CURV**

---

**curv**

---

The macro command **curv** is described under MESH.

## DAMP

---

**damp,xxxx**

---

This macro computes the damping matrix. A lumped damping, a consistent or an unsymmetric consistent damping may be used for transient solutions computed using a dynamic algorithm (see macro command **trans**).

The permissible values for **xxxx** are:

**xxxx = lump**

**xxxx = cons** (default)

**xxxx = ucon**

Each element computes a contribution to the damping matrix in the arrays 's' and 'p' when 'isw' is 12.

The specification of **lump** is used to compute a lumped (i.e., a diagonal) “damping” matrix.

The specification of **cons** is used to compute a consistent “damping” matrix.

The specification of **ucon** is used to compute an unsymmetric consistent “damping” matrix together with **utan** in transient analysis and only for the standard solver.

## DATA

---

### **data,xxxx**

---

During macro execution it is sometimes desirable to progressively change parameters, e.g., the time step size or the solution tolerance accuracy or the actual number + multiplier for a fourier solution. This could become cumbersome and require an excessive number of macro commands if implemented directly. Accordingly, the **data** command may be used in instances when the time step or tolerance or the fourier solution is to be varied during a **loop** execution.

The permissible values for **xxxx** are:

**xxxx = tol**

**xxxx = dt**

**xxxx = four**

The actual values of the tolerance or time step size or the fourier solution are given after the **end** macro statement using the data inputs specified in the **tol**, **dt** or **four** manuals. For example, to vary time steps during a loop the commands:

```
loop,time,3  
data,dt
```

```
time
```

```
...
```

```
...
```

```
next,time
```

```
....
```

```
...
```

```
end
```

```
dt,,0.1
```

```
dt,,0.2
```

```
dt,,0.4
```

could be given to indicate three time steps with  $\Delta t = 0.1, 0.2,$  and  $0.4$  respectively.

In interactive mode the program prompts 'input **xxxx** macro >' and the input for the specified macro has to be done, see the manuals on **dt**, **tol** and **four**.

## DEBU

---

**debu,xxxx**

---

The **debu** macro command can be used to debug the program code. The parameter **xxxx** is defined as follows

- '....' set the debug option to true
- **on** set the debug option to true
- **off** set the debug option to false (default value)

### Implemented

- Prints all informations set in PSETA
- Prints informations for history fields
- Userdefined values on element level

### Further options

- **par** opens files *foutpar\_0i* with i=1,8 on units 51-58.

These files could be used via the subroutine MPRINTP, which writes data with respect to the actual THREAD-number(in parallel processing).

Other data could be written directly with '*write(51 + ip,\*)xxxx*',

where '*ip = OMP\_GET\_THREAD\_NUM()*' defines the active THREAD-number (ip=0,1,2,3,...)

.

## **DETK**

---

**detk**  
**detk,init**

---

The macro command **detk** is used to print the current value of the determinant of the stiffness matrix. The determinant is set to 1 at the end of the first time step of a nonlinear calculation.

**detk,init** set the initial value to 1.

The determinant is always calculated in Arc–Length–Method. To get a correct value the tangent matrix have to be factored. For a plot of the determinant versus a single displacement use before calculation the macro **tplo**.

## DIBC

---

**dibc,,n1**

---

The macro command **dibc** is currently not documented.

## DISP

---

```
disp,,<n1,n2,n3>
disp,all
disp,line
disp,eigv,<n1,n2,n3>
disp,evex,<n1,n2,n3>
disp,eigi,<n1,n2,n3>
disp,cart
disp,pola,n1
```

---

The macro command **disp** may be used to print the current values of the “displacement” vector as follows:

1. Using the macro command:

**disp,,n1,n2,n3**

prints out the current displacement vector for nodes 'n1' to 'n2' at increments of 'n3' (default = 1). If 'n2' is not specified only the value of node 'n1' is output. If both 'n1' and 'n2' are not specified only the first nodal displacements is reported.

2. If the macro command is specified as:

**disp,all**

all terms in the displacement vector are printed.

3. If the macro command is specified as:

**disp,line**

displacements are printed at nodes on a line defined by **dplo** or **splo**.

4. If the macro command is specified as:

**disp,eigv,n1,n2,n3**

then the eigenvectors from 'n1' to 'n2' are reported at increments of 'n3' (default = 1).

Note that it is not possible to print certain nodes in each eigenvector — the entire vector is output to the screen/output file!

5. If the macro command is specified as:

**disp,evex,n1,n2,n3**

then the eigenvector from extended system is printed from 'n1'(def.=1) to 'n2'(def.=numnp) are printed at increments of 'n3' (def.= 1).

6. If the macro command is specified as:

**disp,eigi,n1,n2,n3**

then the eigenvector from inverse iteration is printed from 'n1'(def.=1) to 'n2'(def.=numnp) are printed at increments of 'n3' (def.= 1).

7. If the macro command is specified as:

**disp,pola,n1**

switches the output to polar directions. 'n1'=12,13,23 means the plane i/k in which the transformation acts. Within the next print macro the displacements i and k will then be printed in polar coordinates ( $u_i = u_{\text{radial}}$  and  $u_k = u_{\text{tangential}}$ ).

8. If the macro command is specified as:

**disp,cart**

switches the output to cartesian directions (default!). Within the next print macro all displacements will then be printed in cartesian coordinates.

In order to output a solution vector it is first necessary to specify macro commands to compute the desired values: either normal solutions for a static or dynamic analysis, or the eigenvectors.

## **DPLO**

---

**dplo**,set  
**dplo**

---

The macro command **dplo** is used to print nodal displacements or nodal stresses along a line.

With **dplo**,set coordinates of starting point and end point of line are set.

**dplo** reset all values to zero.

With **disp**,line nodal displacements are printed.

With **stre**,line nodal stresses are printed.

## DT

---

**dt**,v1,v2

---

The **dt** macro command specifies the value of the time step for time dependent problems (i.e., transient or quasistatic problems). The value of 'v1' indicates the time step to be used and should be greater or equal to zero (only for dynamic problems) . Generally, it is necessary to use a **time** macro command, in conjunction with the **dt** command, to advance the time and compute proportional loading values if necessary.

With **dt**,v2 the current time increment is multiplied by 'v2'. Thus it holds  $\text{dt}_{\text{new}} = \text{dt}_{\text{old}} \cdot v2$ .

## EIGI

---

**eigi,,n1,n2**

---

**eigi** compute the lowest eigenvalue and eigenvector of the current tangent matrix.

- n1 = max. number of iteration steps ( default = 50 )
- n2 = terminating tolerance ( default = 1.e-8 )
- In most times a few iterations are sufficient.
- This macro is convenient in a path following calculation when the lowest eigenvalue is required in each load step.
- This computation is equal to the subspace iteration done with :

**iden**

**subs,,1**

solving the eigenvalue problem  $[K_T - \omega\mathbf{1}]\phi = 0$

- The inverse iteration is usually much faster than subspace iteration.
- The eigenvector is not saved in a restart file.

N.B: The term 'lowest eigenvalue' means, the eigenvalue closest to Zero. There may be also some bigger negative eigenvalues. This can be checked using an 'storm sequence test'. ( not implemented yet ) or looking at the number of negative diagonals.

## EIGK

---

**eigk**,<xxxx>,n1,n2,n3

---

The **eigk** macro computes the lowest eigenvalue of the problem  $(\mathbf{K}_T - \lambda \mathbf{M}) \boldsymbol{\varphi} = \mathbf{0}$  via the coordinate overrelaxation method (COR). This method is designed for the detection of dynamic stability points (c-stability) during an implicit solution with the Newmark-scheme.

The following table defines the input options

xxxx	n1	n2	n3
	maxit	$\omega$	$\eta$
chec			
off			
init			

In the above table the parameters have the following meaning:

- maxit maximum number of iterations, default n1 = 20.  
 $\omega$  Overrelaxation parameter; default  $\omega = 1.4$ .  
Can be chosen in the range of  $1.1 < \omega < 1.8$ .  
 $\eta$  Convergence tolerance for algorithm; default:  $\eta = 10^{-5}$ .

<xxxx> = chec: Initializes COR to compute the eigenvalue of the above system of equations for a converged solution within the incremental Newmark scheme. This leads to a printout of the eigenvalue within the time stepping algorithm.

<xxxx> = off: Stops computation of eigenvalue via COR.

<xxxx> = init: Initializes the COR method with a previous computed eigenvector from **subs** and computes the eigenvalue. The associated commands in **subs** are:

**subs,,1**

**subs,init.**

*REMARK:* Before the coordinate overrelaxation method is started use subspace to compute an eigenvector as a starting vector for COR. Otherwise COR will not run efficiently for detecting singular points.

The following macro commands are an example for the computation of singular point during dynamic calculations.

<b>dt,,Δt</b>	Time increment $\Delta t$ .
<b>prop,,1</b>	Proportional load.
<b>lmas</b>	Lumped mass matrix.
<b>tang</b>	Tangent stiffness matrix.
<b>subs,,1</b>	Compute starting vector for COR.
<b>subs,init</b>	Move first eigenvector to COR.
<b>eigk,che</b>	Initialize COR method with default values.
<b>trans</b>	Initialize Newmark method with default values.
<b>loop,time,m</b>	Perform M-time steps.
<b>time</b>	Advance time.
<b>loop,iter,n</b>	Iteration for nonlinear dynamic solution.
<b>tang,,1</b>	
<b>next</b>	
<b>next</b>	

**References** for theoretical background: P. Wriggers, C. Carstensen: In preparation

## EIG1

---

**eig1,,n1**

---

'**eig1**' calculates all eigenvalues and all eigenvectors for one element. Thus an input-file with one element, no loads and no boundary conditions is necessary.

Results for the eigenvalues are printed directly, results for eigenvectors are printed directly only in case of 'n1=2'.

Furthermore results can be printed via **disp,eigv** and plotted via **eigv**

## ELSE

---

**else**,expression

---

The **ELSE** command may be used with a matching pair of **IF-ENDIF** commands. The expression is optional and is used to control the actions taken during the solution. If the expression is absent the commands between the **ELSE** and **ENDIF** are executed. If the expression evaluates to be positive then the commands contained between the IF and the **ELSE** or **ENDIF** are executed, otherwise solution continues with a check of the next **ELSE**.

At present expression is restricted to 4 characters! An extension to 15 characters is projected!

For example, the sequence

```
ZEROA  
...  
IF,10-a  
tang,,1  
ZEROA  
ELSE  
form  
solv  
ENDIF  
INCRA  
...
```

would compute a tangent, residual, and solution increment if 10-a is positive; otherwise the solution increment is computed using a previous tangent. The parameter a may be computed using a function command. For example,

```
FUNCTION ZEROA  
a = 0  
END  
would zero the counter a.  
FUNCTION INCRA  
a = a + 1  
END
```

would define a function which increments a.

## **END**

---

**end,,< n1 >**

---

The last batch macro command must be **end** or **quit**. This terminates the macro execution and returns the program to subprogram “pcontr”, which may then perform additional tasks on the same data, enter a new problem, or “stop” execution. The use of **end** causes a restart file to be updated for subsequent resumptions of execution with the current status preserved.

If ‘n1’ is  $> 0$  the restart–file is in ascii–mode. Thus the file can be transformed to another machine.

Immediately following the **end** macro command any data required by statements in the “macro program” should appear when a ‘macro’ execution is performed (i.e., “batch” executions).

**ENDIF****endif**,expression

The **ENDIF** command is used with a matching **IF** command to terminate the control construction.

For example, the sequence

```
ZEROA  
...  
IF,10-a  
tang,,1  
ZEROA  
ELSE  
form  
solv  
ENDIF  
INCRA  
...
```

would compute a tangent, residual, and solution increment if 10-a is positive; otherwise the solution increment is computed using a previous tangent. The parameter a may be computed using a function command. For example,

```
FUNCTION ZEROA  
a = 0  
END  
would zero the counter a.  
FUNCTION INCRA  
a = a + 1  
END
```

would define a function which increments a.

## **EPSQ**

---

**epsq,< n1 >**

---

Calculation of prescribed displacements  $\mathbf{V} = \mathbf{AE}$ .

n1 = 0 read strain data from file defined in **material 8**,

n1 > 0 read strain data from input macro **epsq**.

## **ERRO**

---

```
erro,,< v1 >  
erro,save,< v1 >
```

---

The macro command **erro** calculates results of all implemented error indicators of the finite element analysis. Currently these are 'energy-based'(1) and 'L<sub>2</sub>-based(2)'. Results are printed here and can be plotted with **erro**.

The theoretical background is described in the [Theory Manual](#).

The results are given with respect to a user defined error value 'v1' in percent (default: 5 %).

With option **save** it is possible to write the indicator values on a file **fres.err** for further handling within a refinement strategy.

## **EXIT**

---

**exit,,< n1 >**

---

The last interactive macro command must be **exit** or **quit**. This terminates the macro execution and returns the program to subprogram “pcontr”, which may then perform additional tasks on the same data, enter a new problem, or “stop” execution. The use of **exit** causes a restart file to be updated for subsequent resumptions of execution with the current status preserved.

If ‘n1’ is  $> 0$  the restart–file is in ascii–mode. Thus the file can be transformed to another machine.

For interactive execution, using **inte**, any additional data will be requested as needed.

**EXT****ext**,<xxxx>,n1,n2,n3

---

The **ext** macro computes a singular point using an extended system of equations. The following table defines the input options

xxxx	n1	n2	n3
	mext	$\epsilon$	iev(mext=5)
<b>eps</b>	exeps	kex	
<b>on</b>	mext	$\epsilon$	iev(mext=5)
<b>off</b>			

In the above table the parameters have the following meaning:

- mext: initialization of extended system with the following approximation of the eigenvector:  
mext = 1:  $\phi = \frac{\{1,1,1,\dots,1\}}{\|\{1,1,1,\dots,1\}\|}$   
mext = 2:  $\phi = \frac{\mathbf{u}}{\|\mathbf{u}\|}$   
mext = 3:  $\phi = \mathbf{K}_T^{-1} \mathbf{I}$   
mext = 4:  $\phi = \{\mathbf{K}_T^{-1} \mathbf{I}\}^{10}$   
mext = 5:  $\phi = \phi_{iev} = ev.$  no.  $iev$  from subspace iteration
- $\epsilon$ : Perturbation factor for the approximate computation of the derivatives of the tangent stiffness  $\mathbf{K}_T$ . Default:  $\epsilon = 0 \implies \epsilon = 10^{-7}$
- $iev$ : No. of eigenvector calculated from subspace iteration
- exeps: Penalty parameter for the stabilization of the extended system near a singular point.
- kex: Equation number to be stabilized.

With  $<\text{xxxx}> = off$  the extended system is switched off.  $<\text{xxxx}> = on$  reinitializes the extended system.  $<\text{xxxx}> = eps$  initializes the stabilization using the penalty approach. Here, one should use an equation number  $kex$  which is associated with a negative diagonal element. If there is none, a good choice is to take the last equation number of the f.e. mesh. The penalty factor  $exeps$  can be set in most cases to 1. However, depending on the material data used, a different value may be appropriate. *REMARK:*

Before the extended system is initiated one step using arclength control has to be performed, see **arcl**.

Thus the following macro commands should be used for example to compute a singular point.

```
dt,,1      Time increment, should be 1 for arcl.  
prop,,1    Proportional load.  
arcl,,2    Initialize arclength method.  
time       Advance time.  
loop,,n    Iteration for one step of arclength control.  
tang,,1  
next  
arcl,off   stop arc-length-method  
ext,,3     Initialize extended system with option 3.  
time       Advance time.  
loop,,n    Iteration to compute singular point.  
tang,,1  
next  
ext ,off   stop extended system  
arcl,on    start arc-length-method  
...
```

**References** for theoretical background: P. Wriggers, W. Wagner, C. Miehe: A Quadratically Convergent Procedure for the Calculation of Stability Points in Finite Element Analysis, Comp. Meth. Appl. Mech. Engng., **70** (1988), 329–347.  
P. Wriggers, J. C. Simo: A General Procedure for the Direct Computation of Stability Points, Int. J. Num. Meth. Engng., **30** (1990), 155–176.

## FEAS

---

**feas**,,**<n1,n2,n3>**

---

- The **feas** macro command requests the solution of '**n1**' eigenvalues of a problem about the current state.
- '**n1**' eigenvalues and eigenvectors are calculated in the interval '**n2**' $\leq$  '**m**' eigenvalues $\leq$  '**n3**'. To decide which values could be used compare the output of error norms. To be successful typically '**n1**' should be '**n1**' > 1.5-2.'**m**'.
- For the efficiency it is absolutely critcial to know the limits '**n2**' $\leq$  '**m**' eigenvalues $\leq$  '**n3**'!
- The macro could be used only in the INTEL-version and only for **solv**,**4** (Pardiso-Solver). Thus the solution is parallel and could be used for large systems.
- The macro is based on the FEAST-algorithm, for details see Eric Polizzi: Density-matrix-based algorithm for solving eigenvalue problems, PHYSICAL REVIEW B 79, 115112 2009, <http://www ecs umass edu/~polizzi/feast/>
- If the solution has not converged it is possible to choose more eigenvalues for the first time using **feast**, **subspace**/ or **lanczos**.
- Within a calculation the number '**n1**' of eigenvalues is limited to a maximum value. This value is defined by the '**n1**' of the first **feast** or **subspace**/**lanczos** iteration.
- The **feas** macro must be preceded by the specification of the tangent stiffness array using a **tang** command, and a second matrix. This can be either the mass matrix ( a lumped mass by **lmas** or a consistent mass by **cmas**) or an identiyy.

Thus one of the following eigenvalue problems can be solved:

$$[\mathbf{K}_T - \omega^2 \mathbf{M}] \boldsymbol{\varphi} = \mathbf{0} \quad [\mathbf{K}_T - \omega^2 \mathbf{M}_D] \boldsymbol{\varphi} = \mathbf{0} \quad [\mathbf{K}_T - \omega \mathbf{1}] \boldsymbol{\varphi} = \mathbf{0}$$

- The first and second matrix must be symmetric and stored in CSR-technique, in addition the second matrix must be positive definite.
- If '**n1**' is larger than the number of non-zero mass diagonals it is truncated to the actual number that exist. Whenever '**n1**' is close to the number of nonzero mass diagonals one should compute the entire set since convergence will be attained in one iteration (this applies primarily to small problems).
- Eigenvectors are scaled that the maximum entry is one.

## FORM

---

**form**  
**form,acel**  
**form,ener**  
**form,expl**

---

The **form** macro computes the residual for the current time and iteration of a solution. FEAP is a fully nonlinear program and computes a residual for each solution by subtracting from any applied loads the force computed for the stresses in each element, called the “stress divergence” or “internal force” term, and if the problem is dynamic the inertia forces.

At the end of each computation FEAP reports the value of the current residual in terms of the Euclidean norm, which is the square root of the sum of squares of each component of force.

If the **acel**-option is present an acceleration is computed by solving the equation:

$$\mathbf{M}\mathbf{a}_0 = \mathbf{G}_0$$

where **M** is a consistent mass (e.g., **cmas**) or a lumped mass (**lmas**) which must be available before the specification of the **form** command. **G** is the residual calculated by the **form** macro.

This option must be used to compute consistent accelerations for starting a transient analysis (**trans**) using the Newmark (or similar)-integration algorithms when the initial force or the initial displacements/velocities are specified.

If the **ener**-option is present an external energy is computed by solving the equation:

$$\pi_a = \mathbf{u}^T \mathbf{F} = \mathbf{v}^T \int_{\Omega} \mathbf{N}^T \bar{\mathbf{t}} d\Omega$$

where **u** is the displacement vector and **F** the load vector. This could be used e.g. for the evaluation of influence areas for plate structures.

If the **expl**-option is present this is a part of an explicit time integration procedure, see **trans,expl** or **trans,expa**. In this case the equation:

$$\mathbf{M}\mathbf{a}_{n+1} = \mathbf{G}_{n+1}$$

is solved, where **M** is a lumped mass (**lmas**) matrix.

## FOUR

---

**four**,nf,factor

---

The macro **four** is a set up for a fourier series solution.

'nf' defines the number of the calculated fourier harmonic. The sign of 'nf' influences the calculation of the displacements:

sign (nf)	$u_1$	$u_2$	$u_3$
$nf > 0$	$\cos nf\theta$	$\sin nf\theta$	$\cos nf\theta$
$nf = 0$	1	1	1
$nf < 0$	$\sin nf\theta$	$\cos nf\theta$	$\sin nf\theta$

The solution can be multiplied by a value defined by 'factor'. The default value is 1.

Within a fourier calculation the program uses the files four\_dis and four\_str.

The fourier solution needs the following macros **fsol**, **fsum**.

## **FSOL**

---

**fsol**,,factor

---

The macro **fsol** saves the results of a fourier series solution for the current harmonic.

The solution can be multiplied by a value defined by 'factor'. The default value is 1.

Displacements may be calculated by a macro **tang**,1 (?) whereas stresses are calculated automatically und this macro. Displacement field and stress field are stored for the actual harmonic on the files four\_dis and four\_str.

The fourier solution needs the following macros **four**, **fsum**.

## **FSUM**

---

**fsum**, $\theta$

---

The macro **fsum** is used to sum the displacements and stresses at a given angle  $\theta$  (input in degree)  
The fourier solution needs the following macros **four**, **fsol**.

In total a fourier calculation is given by

Loop over all harmonics

**four**,nf,factor

**tang**,,1 (?)

**fsol**,nf,factor

Results can be calculated at a given angle  $\theta$

**fsum**, $\theta$

**Note:** The macros **four**, **fsol**, **fsum** are not tested. Thus additional work has to be done—especially for harmonic loading and associated element formulation. There exists an old 3–9 node axisymmetric FEAP–element.

## GEOM

---

### geom

---

The macro command **geom** is used to compute a geometrical matrix for buckling analysis via the formula  $[\mathbf{K}_0 + \Lambda(\mathbf{K}_U + \mathbf{K}_G)]\phi = \mathbf{0}$

- Algorithm for classical buckling analysis

```
# tang,,1 to calculate displacements  
# geom  
# subs,,n  
#  $\mathbf{P}_c = \Lambda_1 \cdot \mathbf{P}$ 
```

Be sure that the displacements are small!!

- Algorithm for linear buckling analysis

```
# based on an equilibrium state  
# geom  
# subs,,n  
#  $\mathbf{P}_c = \Lambda_1 \cdot \mathbf{P}$ 
```

- Remarks:

1. Do not use in combination with single displacement control!
2. Find the lowest positive eigenvalue (?)
3. Note that the stiffness matrix is destroyed after **geom**. Thus go on with e.g. **tang,,1**.

## **HELP**

---

**help**

---

- The use of the **help** macro will produce a list of the currently implemented macro commands.
- This feature is useful only in an interactive mode of macro execution.
- If additional information is required for a specific command it is necessary for the user to consult the MACRO command users manual or to type

**help, macroname**

where **macroname** is the name of the command for which information is required. The available information will be given on the screen.

## HIST

---

**hist,<clab,n1,n2>**

---

The use of the **hist** macro permits the user to keep a history of the previously executed macro commands and use this history to reexecute specific commands. The **hist** macro has several different modes of use which permit easy control of the execution of macro commands while in an interactive mode (use is not recommended in a batch **macr** execution). The following options are available:

clab	n1	n2	Description
<b>read</b>			Input the list of macro commands which were 'saved' in a previous execution. Warning, this command will destroy all items currently in the 'history' list, hence it should be the first command when used.
<b>save</b>			Save the previous 'history' of macro commands which have been 'added' to the 'history' list on the file named 'Feap.his'.
<b>add</b>			Add all subsequent macro commands executed for the current analysis to the 'history' list. (default)
<b>noad</b>			Do not add subsequent macro commands executed to the 'history' list.
<b>list</b>	'n1'	'n2'	List the current 'history' of macro statements. 'n1' to 'n2', (default is all in list).
<b>edit</b>	'n1'	'n2'	Delete items 'n1' to 'n2' from current 'history' list.
<b>xxxx</b>	'n1'	'n2'	Reexecute macro commands 'n1' to 'n2' in the current 'history' list. (note: <b>xxxx</b> may be anything not defined above for <b>clab</b> including a blank field.

Use of the **hist** option can greatly reduce the effort in interactive executions of MACRO programs. Since it is not possible to name the file which stores the 'history' of macro commands, it is necessary for the user to move any files needed at a later date to a file other than 'Feap.his' before starting another analysis for which a 'history' will be retained. Prior to execution it is necessary to restore the list to file 'Feap.his' before a **hist,read** command may be issued.

Note that the 'history' of macro commands will not be saved in 'Feap.his' unless a macro command **hist,save** is used. It is, however, possible to use the **hist** option without any **read** or **save** commands.

### Remark:

With the input '!' it is possible to repeat the last macro.

## IDEN

---

**iden**,,*<n1,n2>*

---

The **iden** macro command is used to specify an identity matrix.

- In general it may be used in conjunction with an eigen computation to compute the eigenpairs of a stiffness matrix for example to test the correct rank of the stiffness matrix. When used in this mode all boundary restraints must be omitted and a shift used to compute any “zero” eigenvalues.
- The macro **iden** is often used for nonlinear buckling analysis.
- $[\mathbf{K}_T - \omega \mathbf{1}] \boldsymbol{\varphi} = \mathbf{0}$ .
- An alternative buckling analysis can be found under the macro command **geom**.
- When ‘n1’ and ‘n2’ are specified they indicate the node range (i.e., ‘n1’ to ‘n2’) for which the identity matrix is to be specified.

**IF****if**,expression

The **IF** command must be used with a matching **ENDIF** command. Optionally, one or more **ELSE** commands may be included between the **IF-ENDIF** pair. The expression is used to control the actions taken during the solution. If the expression evaluates to be positive then the commands contained between the **IF** and the **ELSE** or **ENDIF** are executed, otherwise solution continues with a check of the next **ELSE**.

At present expression is restricted to 4 characters! An extension to 15 characters is projected!

For example, the sequence

```
ZEROA  
...  
IF,10-a  
tang,,1  
ZEROA  
ELSE  
form  
solv  
ENDIF  
INCRA  
...
```

would compute a tangent, residual, and solution increment if 10-a is positive; otherwise the solution increment is computed using a previous tangent. The parameter a may be computed using a function command. For example,

```
FUNCTion ZEROA  
a = 0  
END  
would zero the counter a.  
FUNCTion INCRA  
a = a + 1  
END
```

would define a function which increments a.

## IIMP

---

### iimp

---

The macro command **iimp** is used to generate user-defined values/fields on element level. Each element computes a contribution when 'isw' is 20.

For example a stochastical imperfection field can be calculated to disturb perfect homogeneous situations.

## INIT

---

**init,disp**  
**init,rate**  
**init,acce**

---

Non-zero initial displacements or rates (e.g., velocities) for a dynamic solution may be specified using the **init** macro command. The values for any non-zero vector are specified after the **end** macro command for **batch** executions and may be generated in a manner similar to nodal generations in the mesh input. For interactive execution prompts are given for the corresponding data. Accordingly, the vectors are input as:

n1,ng1,v1-1, . . . ,v1-ndf

n2,ng2,v2-1, . . . ,v2-ndf

etc.

where, 'n1' and 'n2' define two nodes;

- 'ng1' defines an increment to node 'n1' to be used in generation;
- 'v1-1','v2-1' define values for the first degree of freedom at nodes 'n1', 'n2', respectively; etc. for the remaining degree of freedoms.
- Generated values are linearly interpolated using the 'v1' and 'v2' values; etc. for the remaining degree of freedoms.
- Note that 'ng2' is used for the next pair of generation records.
- If a value of 'ng1' or 'ng2' is zero or blank, no generation is performed between 'n1' and 'n2'.
- Do not generate over nodes with boundaries.

Initial accelerations are basically calculated using the macro **form,acel**. For special cases it could be necessary to prescribe these values directly. This could be done via **init,acce**. Input is similar to **init,rate**.

## JINT

---

**jint,,<n1,n2,n3>**  
**jint,all**  
**jint,eps,n1,n2**

---

- Nodal material forces may be computed for all nodes in the problem and reported for nodes 'n1' to 'n2' at increments of 'n3' (default = 1). If 'n2' is not specified then only the value for node 'n1' is output. When both 'n1' and 'n2' are not specified only values for node '1' is output.
- All material forces may be output with the **jint,all** command.
- With the **jint,eps** command material forces for a node n are printed only if  $|jint(n1, n)| > n2$ .

## LAMB

---

### **lamb,,n1**

---

The **lamb** macro is used to compute approximately buckling load factors  $\Lambda$  from eigenvalues  $\omega$ .

With the parameter 'n1' the number i of eigenvalue has to be specified (default=1).

For ' $n1 \leq 0$ ' buckling loads are calculated for all available eigenvalues.

The calculation bases on the formula

$$\Lambda_i = \frac{\varphi_i^T \mathbf{K}_L \varphi_i}{\varphi_i^T \mathbf{K}_L \varphi_i - \omega_i \varphi_i^T \varphi_i}$$

**Reference** for theoretical background: W. Wagner:

A Note on FEM Buckling Analysis, Comm. Num. Meth. Engng. **11**, 1995, p.149–158.

## LAN

---

**lan**,,  
**lan,prin**,  
<n1,n2,n3>

---

- The **lan** macro command requests the solution of 'n1' eigenpairs of a problem about the current state.
- Typically 2.'n1' eigenvalues and eigenvectors are calculated. Only the **first** 'n1' results could be used(print and plot), see the output of error norms.
- To find a solution a working space of 'n2'(default = 50) eigen vectors is used.
- All eigenvalues are computed until two subsequent iterations produce values which are accurate to the current defined tolerance. The iteration behaviour can be controlled by the parameters 'tol's (tolerance for Lanczos iteration) by 'n3' as follows

'n3' = 0	'tol's = 1.d-12 (default)
'n3' > 0	'tol's = <b>tol</b>

The default value for **tol** is 1.d-16, see the **tol** macro.

- If the solution has not converged it is possible to compute further iterations (in the interactive version).
- The **lan** macro must be preceded by the specification of the tangent stiffness array using a **tang** command, and a second matrix. This can be either the mass matrix ( a lumped mass by **lmas** or a consistent mass by **cmas**) or a 'buckling' matrix ( an identity matrix by **iden** ). Thus one of the following eigenvalue problems can be solved:

$$[\mathbf{K}_T - \omega^2 \mathbf{M}] \boldsymbol{\varphi} = \mathbf{0} \quad [\mathbf{K}_T - \omega \mathbf{1}] \boldsymbol{\varphi} = \mathbf{0}$$

- Note that the smallest 'n1' eigenvalues and eigenvectors are computed with reference to the current "shift" specified on the **tang** command.
- If 'n1' is larger than the number of non-zero mass diagonals it is truncated to the actual number that exist. Whenever 'n1' is close to the number of nonzero mass diagonals one should compute the entire set since convergence will be attained in one iteration (this applies primarily to small problems).
- When a large number of eigenvalues should be calculated the subspace iteration may be very time consuming. Then the **lanczos** iteration is advantageous.
- In case of large systems often iterative **solvers** are used. Then the **subspace** iteration is very time consuming whereas the **lanczos** iteration can be used without any problems.
- Use of the **prin** option in second place produces additional output for testing. This option is recommended for small problems only.
- Eigenvectors are scaled that the maximum entry is one.

## **LINE**

---

### **line**

---

The **line** macro command can be used to restrict parts of the code to linear calculations or to switch between linear and nonlinear calculations only by setting a logical variable 'linear'. The default value of 'linear' is false. Thus nonlinear calculations are permitted.

Up to now the execution of the convergence test by **tol** is implemented.

A change of the variable 'linear' can be done with the macro **nonl**.

Thus:

**line** allows linear calculations whereas

**nonl** allows nonlinear calculations (default).

## **LMAS**

---

### **lmas**

---

The macro command **lmas** is used to compute a lumped (i.e., a diagonal) “mass” matrix. Each element computes a contribution to the lumped mass in the array ’p’ when ’isw’ is 5 and ’imtyp’ is 1.

A lumped mass or a consistent mass (see macro command **cmas**) may be used for transient solutions computed using the Newmark method (see macro command **trans**). Both may also be used for eigencomputations (see macro command **subs**).

## LOOP

---

**loop**,<xxxx>,n1

---

The **loop** macro command is used with the **next** command to repeat the set of macro commands between them 'n1' times.

The four characters 'xxxx' may be used to describe the action taken by the loop, e.g., **loop**,time,5 can indicate that the looping is for time steps and is to be performed for 5 steps. Alternatively, the 'xxxx' may be left blank.

During interactive executions, **loop** — **next** commands are not executed until the **next** macro command is issued. In this way a set of macro statements may be grouped together and 'batch' executed.

The **loop** — **next** commands may be nested to a depth of 8.

## MACN

---

**macn,,n1,n2,n3**

**macn,name**

---

The macro commands **macn** ( $n = 1, \dots, 5$ ) are currently not implemented. These commands refer to subroutine calls in SR pmacr. They can be used by a programmer who wants to add new macro features to FEAP.

With the command **macn,name** the macro name **macn** is reset to **name** which can then be used during the analysis. **name** is any four character name. It is not allowed to use already existing macro names like e.g. **tang**.

## **MAN**

---

**man**

---

The complete documentation is available using the macro **man**.

## MATE

---

**mate,def**  
**mate,save**  
**mate,org**  
**mate,new,<v1,n2,n3>**

---

The macro command **mate** can be used to modify the material relations of each element. Several options are possible.

- **mate,def** defines rules for a modification of the material relations. This option is up to now not active.
- **mate,save** makes a copy of the actual material relations. This is done at the beginning by default.
- **mate,org** makes a reset of material relations based on **mate,save**.
- **mate,new,<v1,n2,n3>** modifies the material relations. To do this, an error calculation has to be done in advance via **erro**.

'v1' defines the value to decide for a change (def.=1.0), 'n2' defines the no. of norm to be used (def.=1).

Typically material '1' is modified to material '2'. Here, the **actual** material number is changed to material number 'n3' (def.=2).

## MESH

---

### **mesh**

---

The use of the **mesh** macro command permits the redefinition of some of the mesh data.

Nodal forces may be redefined during macro execution to consider additional loading distributions. In addition, nodal coordinates, values of temperatures, angles of sloping boundaries, constants, material numbers on elements, and material properties may be redefined.

It is not permitted to change the boundary restraint codes or the element connection data as these change the profile of the resulting equations and this is not recomputed to permit the necessary redistribution of memory use.

The description of data input can be found in the MESH command users manual. In interactive mode online help will be given in the MESH mode with the commands **help** or **help**, macroname.

## **NEWF**

---

### **newf**

---

The use of the **newf** macro command will set a fixed pattern of nodal forces and displacements to the values of the current pattern in “forced” boundary loadings plus the previous “fixed” pattern. That is:

$$f0(i,n) \leftarrow f(i,n)*prop(t) + f0(i,n)$$

where  $f0(i,n)$  is the “fixed” pattern loads,  $f(i,n)$  is the pattern specified in “forced” boundary loads, and  $prop(t)$  is the current value of the proportional loading at the current time ‘ $t$ ’.

#### **Remarks:**

- When execution is initiated the values in  $f0(i,n)$  are all zero.
- Initial values could be set on input level via the macro **loa0**.
- NOTE at restart  $f0(i,n)$  again will have only the initial values set via **loa0**. Caution must be exercised at any restart where **newf** had been used in generating the results.

## **NEXT**

---

**next,<xxxx>**

---

The **next** macro command must be used in conjunction with a **loop** command.

The **loop** — **next** pair are used to repeat the execution of a set of macro commands. The **loop** appears first, followed by one or more macro commands' then a **next** command.

If one makes a mistake and start the loop with the macro **next**, or type one **next** more than necessary, FEAP waits on a macro to close the group of commands. In this situation type **loop**. Thus the mistake can be corrected without any calculation.

The **loop** — **next** commands may be nested to a depth of 8.

If desired, the 'xxxx' may be used to describe the type of **next** which is being closed, i.e., **next,time** would indicate the end of a time loop.

During interactive executions, **loop** — **next** commands are not executed until the **next** macro command is issued.

In this way a set of macro statements may be grouped together and 'batch' executed.

## **NONL**

---

### **nonl**

---

The **nonl** macro command can be used to switch between linear and nonlinear calculations by setting a logical variable 'linear'. The default value of 'linear' is false which allows nonlinear calculations.

A change of the variable 'linear' can be done with the macro **line**.

Thus:

**line** allows linear calculations whereas

**nonl** allows nonlinear calculations (default).

## **NOPR**

---

### **nopr**

---

- The use of the **nopr** macro command will discontinue the output of a statement (or statements) describing action taken during each macro command execution. However the residual which shows the convergence behaviour in nonlinear iterations is still output. This holds also for the load level reported by the arc length scheme. Furthermore, plot results and element outputs will still be reported.
- The use of **prin** will cause the output of macro execution descriptions to again be reported.
- The default value is **prin** at start of each macro program execution.

## PARA

---

**para**  
**para,list**

---

The use of the **para**meter command permits the input of data parameters during **execution**.

Typically these parameters are set during the data input phase to vary the input values. For example, parameters may be set and used during proportional loading table inputs. Only 1 or 2 character parameters are permitted and should be lower case letters and numerals (first character must be a letter) only. All further details of the macro **para** are described in **MESH-Input** by the mesh-macro **para**.

In Macro-mode two options for the use of the **parameter** command are allowed.

- Directly in Macro-modus

Examples:

PARA,a=200	sets the parameter a to the value of 200.
BATCh .... LOOP,,25 .... PARA,a=a+1 .... NEXT END	resets the command repeatedly during the loop.

**Note:** At present the expression after macro **para** in Version 2 is restricted to 4 characters! An extension to 15 characters is projected!

Use of LIST will display parameters and values for all letters set previously to non-zero values.

- in Batch -Modus

Use of

```
BATCh  
....  
PARA  
....  
END  
a = 12  
b = a/9  
list
```

inputs a parameter set from data after the END command.

## PARV

---

**parv,init,n1,<n2>**  
**parv,next,<n1>**  
**parv,eigv,n1,n2**

---

The macro command **parv** can be used to save data for postprocessing with the program PARAVIEW, see <http://www.paraview.org/>.

Alternatively the program **TECPLOT** could be used, using the macro **tec**.

More than one material is possible. Associated elements must have the same type and could differ only in the number of nodes.

- **parv,init, n1,<n2>**

opens a file **Rname.pvd** including a group of files **Rname.mxx.tyyyy.vtu** for the postprocessing data. Here mxx defines different materials with  $1 \leq xx \leq 99$  and tyyyy results at time-steps  $1 \leq yyyy \leq 9999$ . A separation into materials is introduced for  $n2 = 0$  (default). No separation is chosen automatically for nummat > 99.

Furthermore initial data like nodal coordinates and elements are written. For consistency it is necessary to initialize nodal stresses by **stre,node!**

With  $n1$  the type of used element is defined.

No.	Element type	ndf	ndm	nel	comment	VTK_type
					List of available elements	
2	beam-2d	3	2	2		03
2	axishell	3	2	2		03
9	beam-3d	6	3	2		03
11	beam-3d	7	3	2	incl. warping	03
1	plane stress	2	2,3	3,4,8,9		05,09,23,28
4	plane strain	3	2	4,8,9	3 dofs for cosserat theory	09,23,28
13	plane strain	1	2	4,8,9	2D Phase field	09,23,28
14	plane strain	5	2	4,8,9	2D Phase field	09,23,28
3	plate	3	2,3	3,4,8,9		05,09,23,28
8	shell-5	5	3	4,9		09,23,28
10	shell-6	6	3	4,9		09,23,28
12	shell-7	7	3	4,9	incl. warping dof of beam	09,23,28
5	solid	3	3	8,20,27		12,25
6	solid	4	3	8,20,27	Add. dof for th.-mech. coupling	12,25
7	solid	6	3	8,20,27	6 dofs for cosserat theory	12,25
15	solid	7	3	8,20,27	3D Phase field	12,25
16	solid	3	3	4		10

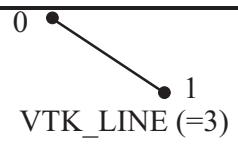
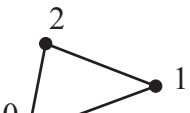
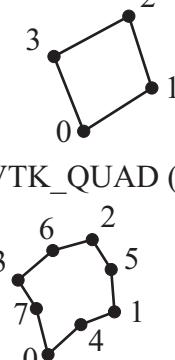
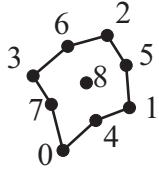
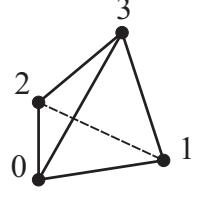
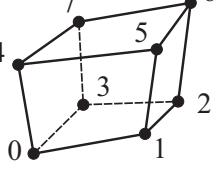
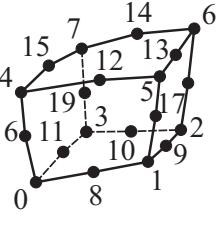
- **parv,next,<n1>**

writes all current data on files **Rname.mxx.tyyyy.vtu** in each time step. These are: nodal coordinates, elements, nodal displacements, (velocities, accelerations in case of **trans**), deformed mesh (displacements  $u_1, u_2, [u_3]$ ) and nodal stresses which have to be calculated by **stre,node** before!

$n1 = 0$ (default)writes results for the **next** step, whereas  $n1 = t$  writes results for step  $t + 1$ .

- **parv,eigv,n1,n2** opens a file **Rname.EVxx.vtu** for the postprocessing of eigenvector  $xx = n2$  and element type  $n1$ . Components of the eigenvector can be plotted similar to nodal displacements, the shape of the eigenvector is presented as deformed mesh (displacements  $u_1, u_2, [u_3]$ ).
- In case of polar coordinates, **pola** has to be used in plot-mode before.

- The node numbering (starting from 0, but similar to FEAP) of the associated elements in Paraview is presented in the following table.

1D	Linear	 VTK_LINE (=3)	
2D	Linear	 VTK_QUAD (=9)	 VTK_QUADRATIC_QUAD (=23)
			 VTK_BIQUADRATIC_QUAD (=28)
3D	Linear	 VTK_TETRA (=10)	 VTK_HEXAHEDRON (=12)
			 VTK_QUADRATIC_HEXAHEDRON (=25)

- A non-linear calculation could be written completely to PARAVIEW with

**stre**,node  
**parv**,init,n1

```
prop,,1
dt,,1
loop,,n-time-steps
  time
  loop,,m-iteration steps
    tang,,1
    next
    stre,node
    parv,next,,10
  next
```

## **PAUS**

---

### **paus**

---

The **paus** macro command can be used in interactive macro mode to stop the iteration behaviour in case of a diverging solution. Within each iteration step the actual internal energy is compared to the value at beginning of iteration. If the actual value is 100 times larger then the first one FEAP asks you if the iteration should be continued:

- 'y' or 'Y' continues iteration
- 'n' or 'N' terminates loop

An associated sequence of macro solutions looks like:

- **time**
- **loop,,n**
- **tang,,1**
- **paus**
- **next**

## PBCG

---

**pbcg**,<xxxx>,<n1,v2,n3>

---

With the command **pbcg** the parameter of the Pre-conditioned bi-Conjugated Gradient method (PBCG) for the iterative solution of a linear system of equations could be modified. This solver is chosen in the input file via **solv**,<sup>5</sup>

With <xxxx> = iter or 'blank' the parameters n1,n2,v3 are set.

- The parameter 'n1' (Def = 150) denotes the maximum numbers of iterations. Typical values are in the range of 50-1000.
- The parameter 'v2' (Def =  $1 \cdot 10^{-8}$ ) sets the tolerance value.
- The parameter 'n3' (Def = 1) sets the type of tolerance criterion (1-4).

Note that for the successful use of the solver a preconditioning is necessary based on the macro **prco**.

## **PGMR**

---

**pgmr,<xxxx>,n1,v2**

---

With the **pgmr** command the parameter of the Pre-conditioned Generalized Minimum Residual method (PGMRES) for the iterative solution of a linear system of equations could be modified. This solver is chosen in the input file via **solv**,<sup>6</sup>

With <xxxx> = iter or 'blank' the parameter n1,v2 are set.

- The parameter 'n1' (Def = 150) denotes the maximum numbers of iterations. Typical values are in the range of 10-500.
- The parameter 'v2' (Def =  $1 \cdot 10^{-8}$ ) sets the tolerance value.

Note that for the successful use of the solver a preconditioning is necessary based on the macro **prco**.

## PLOT

---

```
plot,xxxx,<n1,n2,n3>
or
plot
xxxx,<n1,n2,n3>
....
end
```

---

- In FEAP, screen and some types of hard copy plots may be made for several quantities of interest. A **plot** statement must be specified to initiate graphics outputs.
- Two possibilities to use the **plot** macros exist.

```
# 1.) Type plot and FEAP switches to graphics mode ('Plot mode') and the prompt 'Plot > will be displayed. At this time, all quantities xxxx and the 'n1', 'n2', and 'n3' values may be specified. A return to the macro mode can be done by xxxx = end or = q.  
# 2.) Alternatively, a plot,xxxx,n1,n2,n3' command may be issued while in macro execution mode (this is the only option for batch executions).
```

- On terminals where only one plane is visible (text or plot) the plot image may disappear after performing a plot action (this is what happens when using a GraphOn). In this case it may be necessary to press a key on the terminal to switch between graphics and text mode – the image is retained between changes in mode! If FEAP runs on a PC the switch is done automatically.

The following values may be used for the quantity **xxxx**:

aacc	acce	adis	aeig	aeve	angl	avel	axis
back	base	bord	boun	cart	ceig	cent	clip
colo	copy	defm	defo	dimp	disp	dmag	dplo
draw	eigi	eigv	elem	end	eplo	erro	evan
evex	fact	flux	forc	fram	hide	hids	hmsh
init	isec	ints	isom	jint	line	link	load
logo	magn	man	mate	matn	maxi	mesh	mono
move	movi	ndii	node	outl	pele	pers	pdis
plof	pnod	pola	prin	pris	prof	quit	reac
rmsh	resi	rot0	rot1	rot2	rot3	rotm	rplo
rsum	scal	sect	show	size	slee	splo	stre
str1	symm	text	tie	tplo	traj	ueig	velo
wipe	xsc	zoom					

- The action to be taken by each command is described in the following sections of the PLOT Comand Users Manual. Furthermore in the interactive Plot mode online help is available by typing **help** which gives information on the available plot macros and **help,xxxx** which gives information on the plot macro **xxxx**.

## POLA

---

**pola,,n1**

---

Displacements, velocities, accelerations and eigenvectors can be plotted in cartesian coordinates ( $n1 = 0$ , default) or in polar coordinates ( $n1 = ik = 12,13,23$ ).  $i$  and  $k$  describe the plane in which the transformation should be done. Afterwards the component  $i$  describes the radial e.g. displacement whereas the component  $k$  describes the e.g. tangential displacement.

The plot is performed in standard manner via e.g. the **disp** – macro.

## POST

---

```
post,init,<n1>
post,disp
post,reac
post,eigv,n1
post,stre
post,warp,<n1>
post,clos
```

---

The macro command **post** may be used to save data for postprocessing.

- **post,init**

opens a file **Rname.pos** for the postprocessing data. Furthermore general informations, the nodal coordinates and element connections are written on **Rname.pos**.

When **tie** is used different output is possible.

In case  $n1 = 0$  the output is

- 1) list of numnp nodes. Tied nodes have coordinates  $x1=-999.0$
- 2) list of numel elements. Tied nodes are here not used.

Thus the resulting mesh has only additional(tied) nodes.

In case  $n1 = 1$  nodes are renumbered. Total number of nodes is  $numnlp = numnp - \text{number of tied nodes}$ . Furthermore the list of nodes for each element is modified.

- 1) list of renumbered  $numnlp$  nodes. Tied nodes do not occur.
- 2) list of numel elements with new node numbers. Tied nodes do not occur.

- **post,disp**

writes the current nodal displacements on file **Rname.pos**.

- **post,reac**

writes the current nodal reactions on file **Rname.pos**.

Calculate reactions directly before **post,reac**.

- **post,eigv,n1**

writes the calculated eigenvector  $n1$  on file **Rname.pos**.

- **post,stre**

writes the current nodal stresses on file **Rname.pos**.

- **post,warp,n1**

writes the current nodal warping values on file **Rname.pos**. These values are calculated with **Element 12** using **stre,node**.

$n1=0,1$  store  $\tilde{\mathbf{w}}$

$n1=2$  store  $\bar{\mathbf{w}}$

- **post,clos**

closes the file **Rname.pos**.

## PRIN

---

**prin**  
**prin,<xxxx>**

---

- The use of the **prin** macro will cause a statement (or statements) describing action taken during each macro command execution to be reported on the output.
- The use of **nopr** will discontinue the output of macro execution descriptions (except actual plots).
- The default value is **prin**.
- The specification of:

```
# xxxx = tang
# xxxx = utan
# xxxx = cmas
# xxxx = lmas
# xxxx = geom
# xxxx = damp
# xxxx = resi
```

will output the **diagonal** entries for the specified array for **solv,0**. Other solvers may give output for the first n stored elements. This may be useful in debugging elements, etc.

- Be sure that you have calculated the specified arrays directly before! For example with the macros

```
# iden
# lmas
# prin,iden
# prin,lmas
```

only the elements of **lmas** are printed.

- The specification of:

```
# xxxx = eigv
```

will print the actual eigenvalues.

- The specification of:

```
# xxxx = off
```

will suppress the following output

- \* Echo the macro command and the associated options
- \* Echo the output: computation time / prop. load value
- \* Echo the output: residual / actual iteration / max no. of iterations

```
# This option can be redefined (default) by
```

```
xxxx = on
```

## PRCO

---

**prco**,<xxxx>,n1,v2,n3

---

With the **prco** command the type of preconditioning for the iterative solvers (**PBCG**) and (**PGMRES**) could be chosen or modified.

With ith <xxxx> = iter or 'blank' the parameter n1,v2,n3 are set.

- The parameter 'n1' (Def = 3) defines the method of preconditioning. At present four methods are available:

```
# 'n1'=1 uses a unit matrix. M = 1. This option should not be used.  
# 'n1'=2 uses the diagonal entries of the stiffness matrix. M = [1/Kii]. This option is very fast  
and works well only for 3D-problems with same length values.  
# 'n1'=3 uses an incomplete LU factorization of K with dropout and without pivoting which  
could be very time consuming for large scale problems.  
# 'n1'=4 uses an incomplete LU factorization of K with level k fill-in which could be very time  
consuming for large scale problems.
```

- With 'v2' and 'n3' parameters for preconditioner 3 are set.

```
# With 'v2' a threshold value tol for L and U is set. Any element whose size is less than some  
tolerance (relative to the norm of current row) is dropped. As recommendation use values in  
the range of  $1 \cdot 10^3 - 5 \cdot 10^3$  (Def =  $1 \cdot 10^3$ ).  
# With 'n3' the number of elements lfil in a row will be defined. The higher lfil the more reliable  
the code is but one looses efficiency. Recommendation is lfil = 5 to 10 (Def. = 5).
```

- With 'n3' parameters for preconditioner 4 are set.

```
# With 'n3' the number of fill-ins will be defined. The higher lfil the more reliable the code is  
but one looses efficiency. Recommendation is lfil = 2 to 3 (Def. = 5).
```

Remarks:

- In the limit with lfil = neq and tol = 0 preconditioner 3 will yield the same factors as a Gaussian elimination without pivoting. Thus no iteration is necessary for the iterative solver for the solution which is nonsense.
- Use always preconditioner 3,4 especially for 2D-problems.

## PROC

---

**proc**,name,<n1,n2,n3>

---

The procedure command is used for the repeated input of certain macro sequences. It can be seen as a user defined macro which contains a number of **FEAP**-macros, which can be defined before an analysis. Every time this macro is used the whole sequence of macros is done.

A procedure is created during a macro analysis by entering the command:

**proc**,name,n1,n2,n3

- **name** is any 1-8 character alphanumeric identifier which specifies the procedure name (**the first 4 characters must not be the same as an existing macro command name!**)
- **n1,n2,n3** are any 1 to 4 character parameter names for the procedure, which can be used everywhere. The parameters are optional and may be blank.
- A procedure is saved in a file with the extender '.pcd'.
- On HP-WorkStations the procedure has to be written in the FEAP dialog window. Here, the procedure is terminated by using an **end** macro command.
- On PC's a procedure editor supports the input of the procedure. Here, the first line of the procedure has to be v1,v2,v3 (see above), **end** is not necessary.
- Furthermore the procedure can be written with any editor.
- FEAP assumes that procedures are in the same directory as the input file of the current problem. With the command **proc,path** this can be changed to any other directory.

- **Example:**

For example the procedure for a load step in a nonlinear static analysis - named step.pcd - may be defined by:

Macros  
-----  
n1,n2,n3  
**time**  
**loop**,,n1  
**tang**,,1  
**next**  
**disp**,,n2  
**stre**,,n3  
**reac**,all  
**(end)**

Thus **step**,n1,n2,n3 leads to the following actions: With **time** the load is updated; then 'n1' iterations will be performed. Finally the displacements are shown for node 'n2', the stresses for element 'n3' and the reactions for all nodes.

## PROP

---

**prop,,<n1>**

---

- In the solution of transient or quasi-static problems in which the **time** command is used to describe each new time state the loading may be varied proportionally. At each time the loading applied will be computed from:

$$F(i,t) = f_0(i) + f(i) \cdot \text{prop}(t)$$

where  $f_0(i)$  is a fixed pattern which is initially zero but may be reset using **newf** (see macro manual on **newf**);  $f(i)$  are the “forced” nodal conditions defined during mesh input or revised during a ‘MESH’ macro command; and  $\text{prop}(t)$  is the value of the proportional loading at time ‘ $t$ ’.

- The specific proportional loading is defined by specifying one record for each of the ‘ $n1$ ’ **prop**-cards (default for ‘ $n1$ ’ is 1, maximum is 10) with the following data:

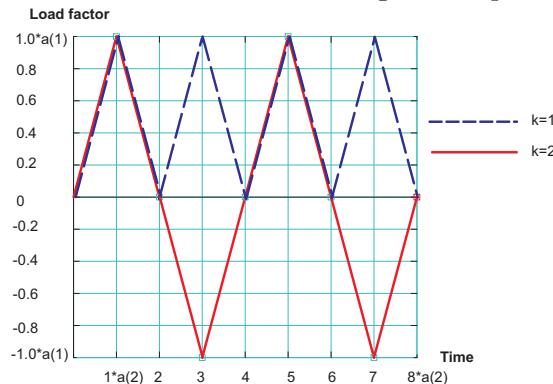
type,  $k$ ,  $t_{min}$ ,  $t_{max}$ ,  $a(i)$ ,  $i = 1, 4$

- Currently three different types of proportional loading may be specified for all time values between  $t_{min}$  and  $t_{max}$ :

- Type 1 defines a function by:

$$\text{prop}(t) = a(1) + a(2) \cdot (t - t_{min}) + a(3) \cdot \sin[a(4) \cdot (t - t_{min})]^k$$

- Type 2 defines a “sawtooth” loading with input data:  $k$ ,  $a(1)$ ,  $a(2)$ , see the following figure.



- Type 3 defines a polynomial by:

$$\text{prop}(t) = a(1) + a(2) \cdot (t - t_{min}) + a(3) \cdot (t - t_{min})^2 + a(4) \cdot (t - t_{min})^3$$

- The default values are

n1 = 1	
type = 1	k = 1
$t_{min} = 0$	$t_{max} = 1.e8$
$a(1) = 0$	$a(2) = 1$
	$a(3) = 0$
	$a(4) = 0$

- Thus the default load function

$$f(i,t) = f_0(i) + f(i) \cdot t$$

is given by the input **prop**, <cr>

- Within the defined time interval  $t_{min} \leq t \leq t_{max}$  the load factor is calculated from all defined ‘ $n1$ ’ **prop**-cards:  $\text{prop}(t) = \sum_{i=1}^{nprop} \text{prop}_i(t)$ .
- An example how to use is presented in the **application manual** part **prop**

## **QUIT**

---

**quit**

---

The last macro command may be **quit**, or just **q**. This terminates the macro execution and returns the program to subprogram “pcontr”, which may then perform additional tasks on the same data, enter a new problem, or “stop” execution.

The **quit** macro causes termination of execution without writing the restart files (they remain the same as at the beginning of execution).

## REAC

---

**reac,,<n1,n2,n3>**  
**reac,all**  
**reac,eps,n1,n2**

---

- Nodal reactions may be computed for all nodes in the problem and reported for nodes 'n1' to 'n2' at increments of 'n3' (default = 1). If 'n2' is not specified then only the value for node 'n1' is output. When both 'n1' and 'n2' are not specified only values for node '1' is output.
- All reactions may be output with the **reac,all** command.
- With the **reac,eps** command reactions for a node n are printed only if  $|reac(n1, n)| > n2$ .  
With the **reac,sigq** command stresses are calculated on a RVE.  
Results are only valid for equilibrium state! Macro is implemented only for 3D-case.

$$\mathbf{S} = \frac{1}{V} \mathbf{F}_b$$

- In addition to the “reaction” at each degree of freedom an equilibrium check is performed by summing the values for each degree of freedom over all nodes in the analysis. The sum of the absolute value of the reaction at each degree of freedom is also reported to indicate the accuracy to which equilibrium is attained.
- NOTE that problems with rotational degrees of freedom or in curvilinear coordinates may not satisfy an equilibrium check of this type. For example, the sum for the radial direction in an axisymmetric analysis will not be zero due to the influence of the “hoop stresses”.
- In addition to sums over all the nodes a sum is computed for only the nodes output. This permits the check of equilibrium on specified series of nodes, for example nodes with boundary constraints , or the computation of the applied load on a set of nodes in which motions or restraints are specified.
- **reac** prints the following values:  
t=0: loads from mate + single loads  
after tang: reactions
- If the macro **rsum** is used during input, the resulting reaction force is always printed for the specified nodes.
- In case of a dynamic analysis dynamic forces are included.

## READ

---

### **read,xxxx**

---

The **read** macro command may be used to input the values of displacements and nodal stresses previously computed and saved using the **writ** macro command – it is primarily used for plots related to deformations or nodal stresses. It is not intended for a restart option (see **rest**) but may be used to restore displacement states of linear and non-linear elastic elements (or other elements with no data base requirements) for which reactions, stresses, etc. may then be computed.

The values of **xxxx** are used to specify the file name (4-characters only), manipulate the file, and write out displacements. The values permitted are:

<b>xxxx</b> = <b>wind</b>	rewind the current file (note in a file is normally opened and positioned at the end-of-file mark).
<b>xxxx</b> = <b>clos</b>	close the current output file.
<b>xxxx</b> = <b>disp</b>	read a displacement state from the current file.
<b>xxxx</b> = <b>stre</b>	read a nodal stress state from the current file.
<b>xxxx</b> = anything else	will be used to name the current file. Only four characters are permitted and only one file may be opened at any time. Files may be opened and closed several times during any run to permit the use of more than one file name.

A **read** input is created using the **writ** macro command which has identical options for **xxxx**.

## REME

---

**reme,unif**  
**reme,adap,<n1,n2>**  
**reme,new**  
**reme,old**  
**reme,rest**

---

- The macro command **reme** may be used to refine a mesh of 4-node elements.

The remeshing options shall be specified as:

- **reme,unif** - refine complete mesh (not regarding any error indicator), number of elements in the generated mesh will be four times the original number.
- **reme,adap** - refine mesh (using an error indicator)
  1. n1 - indicates the percentage of error norm (default 5.)
  2. n2 - specifies the type of error norm (1 - energy - norm, (default) 2 - L2 - norm )
- **reme,new** - read the new input file (switch i.. to n..) only in the first step
- **reme,old** - read the input file again after the first step
- Remarks:
  1. To calculate problems efficiently, provide the input file with the following options:
  2. use always the macro **opti**, if the standard solver is used
  3. use the macro **nopr**, to avoid large output files, due to the fact that nodal values are written for each node separately!
  4. Only meshes with 4-node elements can be generated.
  5. To use the option **adap**, your element must provide the desired error indicator (isw=9).
  6. The new input file will be written into a new file (ifile  $\Rightarrow$  nfile).
  7. Nodal markings are necessary for the restart, they will be written on the 'nfile.nrm'.
  8. The 'new' mesh can be viewed in plot with **rmsh**.
  9. Boundary and load conditions must be generated with **edge**, **eloa**,....

- A macro-sequence for using the adaptivity option is given by the following macros (e.g. as procedure ada1.pcd):

1. start program
  - **tang,,1** : calculate a solution
  - **reme,unif**: uniform mesh refinement
2. for each adaptive step
  - **reme,new** : read the new input file (first step) or
  - **reme**, old : read the new input file (all further steps)
3. then (e.g. as procedure ada2.pcd):
  - (**nonl** : only for nonlinear analysis)
  - (**reme,hist** : only for nonlinear analysis)
  - **tang,,1** : calculate a solution
  - **stre,node** : calculate nodal stresses
  - **reme,adap,n1,n2** : adaptive mesh refinement

## REST

---

**rest,<fileext,n1>**

---

- A restart may be made using the terminal results of a previous analysis ( which are retained on the restart read file specified at the start of each analysis). After entering the macro program the restart may be specified. If the previously computed problem was “dynamic”, it is necessary to specify the **trans** macro command prior to issuing a **rest** command in order to restore the terminal velocity and acceleration state. If the previous problem was static and the new analysis is to be continued as a dynamic calculation, the **rest** is issued before the **trans** macro command (since the previous analysis did not write a velocity or acceleration state to the restart file. Also specify **arcl** first when the problem which was analysed previously used the arc-length method.
- If the **save** command has been used during the analysis a restart at the solution state saved on the standard restart file or on files with the extension ‘fileext’ is possible by specifying **rest,<fileext>**. The extension is restricted to 4 characters. This command can be used in batch and interactive mode. The File *Filename.Fileext* is loaded.
- The use of the restart option requires considerable care to ensure that the previous results used are proper. At the termination of any analysis which computes a solution state a new file is saved on the restart write file specified at the start of the analysis. If the last analysis performed is for a different problem than the current one an error will result.
- If no new solution state is computed during macro execution (e.g., only plotting is performed) no restart file is written to the specified fileset – the previous restart file is retained on the original fileset.
- If ‘n1’ is > 0 the restart file is in ascii-mode. Thus the file can be checked explicitly or read by FEAP from a version running on another computer. This may be dangerous due to a loss of accuracy by missing digits.

## RINP

---

**rinp**  
**rinp,new**

---

- **FEAP** can be used as preprocessor under Windows and Unix with the macro **rinp**. Thus, open an input file with an editor and make some input. Save but do not leave the editor. Start **FEAP**, look for error messages and use the plot mode to control the input data.
- Every modification of the input data can be seen if the input file is saved in the editor and if in **FEAP** the macro **rinp** is used.
- With **rinp,new** a new set of files can be defined.

## SAVE

---

```
save,<fileext,n1>
save,init,<n1>
save,next,<n1>
```

---

- The **save** macro command may be used to save the current solution state and history data for use as a restart file. At the current state of implementation the **save** command saves also not converged states. Thus it cannot be used without caution in iteration loops. The problem may then be initiated from the saved state by the **rest** command and continued.
- During an analysis more than one state may be saved. In the solution of complicated non-linear problems where difficulties are expected in achieving convergence (e.g., a solution step may produce an overflow which terminates execution, in conjunction with the arc-length method or in elasto-plastic calculations when the maximum load increment or step-length is not known) a restart state may be saved on the disk for each converged state.

This can be done by using a file extension within the saving procedure.

This **fileext** is optional and may be any 1–4 characters and is appended to the name of the current restart save file which was named when the problem was started if specified. In interactive mode should a name be given which already exists the user receives a prompt for an alternate name or given the opportunity to write over the old file.

- If 'n1' is > 0 the save file is in ascii-mode. Thus the file can be checked explicitly or read by FEAP on another computer. This may be dangerous due to a loss of accuracy by missing digits.
- If for example restart files for all load steps are necessary then it is useful to mark the restart files with numbers. For this purpose the macro **save,init** set the first number (called 'icount') to 'n1'.

Next each command **save,next** updates 'icount' and writes the next restart file. Thus a sequence

Filename.0001, Filename.0002, Filename.0003,....

is stored.

## SHOW

---

**show,xxxx,n1,n2,n3**

---

This command shows different values of the system and gives information on the currently chosen parameters on the macro level.

The permissible values and actions for **xxxx** are:

**xxxx = coor**

Coordinates are shown for node n1 to node n2 with increments n3.

**xxxx = boun**

Boundary conditions are shown for node n1 to node n2 with increments n3.

**xxxx = forc**

Forces are shown for node n1 to node n2 with increments n3.

**xxxx = elem**

Associated nodes are shown for element n1 to element n2 with increments n3.

**xxxx = node**

Coordinates,forces and b.c. are shown for node n1.

**xxxx = memo**

Length and position of the currently used arrays in common M

Length of all allocated arrays

**xxxx =**

The currently chosen parameters on the macro level are shown.

## SIGQ

---

**sigq**

---

Calculate on a RVE :

$$\begin{aligned}\mathbf{S} &= \frac{1}{V}(\mathbf{F}_b - \mathbf{L}^T \mathbf{K}^{-1} \mathbf{F}_a) \\ \mathbf{C} &= \frac{1}{V}(\mathbf{M} - \mathbf{L}^T \mathbf{K}^{-1} \mathbf{L})\end{aligned}$$

Associated 'Load'vector is defined with macro **epsq**.

## **SMOO**

---

**smoo**,xxxx,n1,n2,n3

---

The macro command **smoo** optimizes geometrically an given mesh.

The permissible values and actions for **xxxx** are:

**xxxx=oesp**

simple mesh smoothing without consideration of edge- or boundary-nodes

**xxxx=wesp**

simple mesh smoothing with consideration of edge- or boundary-nodes

All boundaries for plane surfaces must be defined by curves. For spatial meshes the surface has to be prescribed by curves. (in macro **curv** – n3 has to set to 15, 16, 17 or 18)

The parameters are :

**n1** number of the node from which the mesh smoothing should be executed (default=1)

**n2** number of iterations of mesh smoothing (useful 1,2 or 3) (default=1)

**n3** weight factor for equality of sidelengths (first two numbers;w1)  
and 90-degree-angles (third and fourth number;w2)

(for instance w1w2=0210 denotes w1=2 and w2=10)(default=0101)

## **SPLO**

---

**splo,set**  
**splo**

---

The macro command **splo** is used to print nodal displacements or nodal stresses along a user-defined line.

With **splo,set** coordinates of starting point and end point of line are set.

**splo** reset all values to zero.

With **disp,line** nodal displacements are printed.

With **stre,line** nodal stresses are printed.

Furthermore coordinates of starting and end point of line are set for the the **plot**-macros **dplo**, **eplo**, **rplo**, **splo**,

## SOLV

---

**solv**

**solv,<line,v1>**

---

- The macro command **solv** is used to specify when the equations generated by a **tang**, **utan** and/or **form** are to be solved. In FEAP, a direct solution of the equations is performed using a profile storage with a variable band (active column) method of solution.
- In the solution of some nonlinear problems it is possible to obtain convergence for a wider range of loading and time step size using a “line search”. The line search may be requested by placing **line** in the second field of the ’solv’ macro command. The parameter ’v1’ is the required energy reduction to preclude a line search being performed (if the current energy is larger than ’v1’ times the minimum energy in the step so far, a line search is performed). If not specified ’v1’ defaults to 0.8 (recommended values are between 0.6 and 0.9). Line search should never be used in a linear problem since extra evaluations of the residual are required during the line search.
- Note that a ’complete’ calculation can be done by **tang,,1** which is equal to **tang**, **form**, **solv**.

## STRE

---

```
stre,,<n1,n2,v3>
stre,all
stre,line
stre,<ints,n1,n2,n3>
stre,<lay ,n1,n2,v3>
stre,<node,n1,n2,v3>
stre,<pris,n1,n2,m3>
stre,<dmag,n1,n2,v3>
```

---

The **stre** macro command is used to output stress results in elements 'n1' to 'n2' at increments of 'v3' (defaults 'n1' = 1, 'n2' = 'n1', 'v3' = 1 ), or at nodes using 'projected" values. Thus,two options exist for reporting stress values. These are:

1. Stresses may be reported at selected points within each element. The specific values reported are described in each element type. In general elements report values at gauss points. The values at all points are reported when the command **stre,all** is used.
2. For 2/3-dimensional elements results may be reported at nodes using the **stre,node** option. A projection method using stresses at points in each element is used to compute nodal values. In general, nodal values are not always as accurate as stresses within elements. This is especially true for reported 'yield' stresses where values in excess of the limit value result in the projection method employed. For a mesh producing accurate results inside elements this degradation should not be significant.
3. Principal stress 'm3' may be reported at nodes 'n1' to 'n2' (increments of 1) using the **stre,pris** option. Details on principal stresses, especially on the use of 'm3' may be found within macro **pris**.

For specific values reported in each element look into the ELEMENT users manual' for 'ELMTnn' where 'nn' is a number describing the element wanted (between 01 and 40 for FEAP). An online help may be available by 'HELP,ELMTnn'. (up to now not implemented). Note that manuals may not be available for all types in this range.

With the option **stre,line** stresses are printed at nodes on a line defined by **splt** or **dplt**.

With the option **stre,ints** the distribution of stress  $n_2$  at the center of element  $n_1$  over the thickness in a shell type element is printed. For the output  $n_2$  intervals (default  $n_2=10$ ) are used for each layer.

The value of 'v3' can be used also for layered elements. If 'v3' is < 0 than  $\text{abs}('v3')$  is the layer number and the position for which the stresses should be reported. E.g. 'v3=-3.2' prints stresses in layer 3 at position 2. For the definition of layer and position see the associated element manual or formulation. In this case the incrementation between 'n1'and 'n2' by 'v3' does not work (set to 1). Ouput is possible at Gauss-points (**stre,lay**) or at nodes (**stre,node**).

Stresses at **damaged** integration points may be reported within each element. The specific values reported are described in each element type. Here, the command **stre,dmag** prints stresses in this points.

**Remark** on variable istv:

If the variable istv is > 0 the stresses 5-7 are modified to main stresses , otherwise they are not modified, see the macro **plot,stre**. Istv defines the number of plotted stresses, see also the macro **stre,node**

## SUBS

---

```
subs,,<n1,n2,n3>
subs,prin,<n1,n2,n3>
subs,init,<n1,n2,n3>
```

---

- The **subs** macro command requests the solution of 'n1' eigenpairs of a problem about the current state.
- Typically 2.'n1' eigenvalues and eigenvectors are calculated. Only the **first** 'n1' results could be used(print and plot), see the output of error norms.
- An additional number of 'n2' vectors are used to expand the subspace and improve convergence ('n1' plus 'n2' is set to the minimum of the input values or 'n1' plus 8 or 2 times 'n1' or the maximum number of eigenvalues in the problem).
- All eigenvalues are computed until two subsequent iterations produce values which are accurate to the current defined tolerance. The iteration behaviour can be controlled by the parameters 'tol' (tolerance for subspace iteration) and 'nits' (number of iterations for subspace computation) by 'n3' as follows

'n3' = 0	'tol' = 1.d-12	'nits' = 25
'n3' < 0	'tol' = tol	'nits' = 25
'n3' > 0	'tol' = tol	'nits' = 'n3'

Thus, the default values are given in the first row of this table.

The default value for **tol** is 1.d-16, see the **tol** macro.

- If the solution has not converged it is possible to compute further iterations (in the interactive version).
- The **subs** macro must be preceded by the specification of the tangent stiffness array using a **tang** command, and a second matrix. This can be either the mass matrix ( a lumped mass by **lmas** or a consistent mass by **cmas**) or a 'buckling' matrix ( an identity matrix by **iden** or a geometrical matrix by **geom**). Thus one of the following eigenvalue problems can be solved:

$$\begin{aligned} [\mathbf{K}_T - \omega^2 \mathbf{M}] \boldsymbol{\varphi} &= \mathbf{0} \\ [\mathbf{K}_T - \omega \mathbf{1}] \boldsymbol{\varphi} &= \mathbf{0} \\ [\mathbf{K}_L + \Lambda \mathbf{K}_{NL}] \boldsymbol{\varphi} &= \mathbf{0} \\ [\mathbf{K}_L + \Lambda(\mathbf{K}_U + \mathbf{K}_G)] \boldsymbol{\varphi} &= \mathbf{0} \end{aligned}$$

- Note that the smallest 'n1' eigenvalues and eigenvectors are computed with reference to the current "shift" specified on the **tang** command.
- If 'n1' is larger than the number of non-zero mass diagonals it is truncated to the actual number that exist. Whenever 'n1' is close to the number of nonzero mass diagonals one should compute the entire set since convergence will be attained in one iteration (this applies primarily to small problems).

- Use of the **prin** option in second place produces an output of all subspace matrices in addition to the estimates on the reciprocals of the shifted eigenvalues. For large problems considerable output results from a use of this option, and thus it is recommended for small problems only.
- Use of the **init** option initialize the COR–method, see macro command **eigk**
- When a large number of eigenvalues should be calculated the subspace iteration may be very time consuming. Then the **lanczos** iteration is advantageous.
- In case of large systems often iterative **solvers** are used. Then the **subspace** iteration is very time consuming whereas the **lanczos** iteration can be used without any problems.
- Eigenvectors are scaled that the maximum entry is one.

## SUMM

---

**summ,file,fac**  
**summ,zero**

---

The **summ** macro command is used to summarize results from different calculations. The macro works correct, if the results are calculated on the same mesh (only nodes and elements) and if the displacement and stress fields have the same variables. Results in previous calculations have to be saved with the macro **writ**. The values are stored in the actual displacement and stress fields! Thus **FEAP** can be used under **summ** only as a **postprocessor**.

Save data:      **writ**,fil1  
                 **writ**,disp  
                 **writ**,clos  
                 ....

Summarize data: **summ**,fil1,fac1  
                 **summ**,fil2,fac2  
                 ....

Here values of different calculations can be added by different factors fac1,fac2,...  
Thus it holds for example:  
 $v = fac1 * v(fil1) + fac2 * v(fil2) + \dots$

Then all **print-** and **plot-**options of **FEAP** are available.

With **summ,zero** the displacement and stress fields are cleared.

## TANG

---

**tang,,<n1,v2>**  
**tang,line,<n1,v2,v3>**

---

The **tang** macro computes the tangent stiffness matrix about the current value of the solution vector.

- For linear applications the current stiffness matrix is just the normal 'stiffness' matrix.
- If the value of 'n1' is greater than zero a force vector for the current residual is also computed (this is identical to the **form** macro computation) — thus leading to greater efficiency when both the tangent stiffness and a residual force vector are needed.
- If the value of 'v2' is non-zero a "shift" is applied to the stiffness matrix in which the mass matrix (which must already be formed using a **cmas** or **lmas** macro) is multiplied by 'v2' and subtracted from the stiffness matrix. This option may be used with the **subs** pace algorithm to compute the closest eigenvalues to the shift, 'v2'. Alternatively, the shift may be used to represent a forced vibration solution in which all loads are assumed to be harmonic at a value of the square-root of 'v2' (rad/time-unit).
- After the tangent matrix is computed, a triangular decomposition is available for subsequent solutions using **form** and **solv**, **bfgs**, etc. This decomposition is not computed when the value 'n1' is negative.
- Note that a 'complete' calculation can be done by **tang,,1** which is equal to **tang, form, solv**.
- In the solution of non-linear problems, using a full or modified Newton method, convergence from any starting point is not guaranteed. Two options exist within available macro commands to improve chances for convergence.
  - # One is to use a line search to prevent solutions from diverging rapidly. Specification of the command **tang,line** plus options invokes the line search option (it may also be used in conjunction with 'solv,line' in modified Newton schemes). The parameter 'v3' may be chosen between 0.5 and 0.8 and will generally produce good performance.
  - # The second option to improve convergence of non-linear problems is to reduce the size of the load step increments. The macro command **back,,dt** may be used to "back-up" to the beginning of the last time step (all data in the solution vectors is reset and the history data base for inelastic elements is restored to the initial state when the current time is started). Repeated use of the 'back' command may NOT be used. Only one back-up in time is permitted. The loads may then be adjusted and a new solution with smaller step sizes started.
- In combination with **arcl** the backup has to be done by **back,,1!**

## TEC

---

```
tec,init, n1  
tec,write  
tec,eigv,n1,n2  
tec,close
```

---

The macro command **tec** can be used to save data for postprocessing with the program TECPLLOT. see <http://www.tecplot.de>

Alternatively the program **PARAVIEW** could be used, using the macro **parv**.

- **tec,init, n1**

opens a file **Rname.tec** for the postprocessing data.

With n1 the type of used element is defined.

No.	Element type	ndf	ndm	nel	comment
0					List of available elements
2	beam-2d	3	2	2	
2	axishell	3	2	2	
9	beam-3d	6	3	2	
11	beam-3d	7	3	2	incl. warping
1	plane stress	2	2,3	3,4,9	
4	plane strain	3	2	4,9	3 dofs for cosserat theory
13	plane strain	1	2	4,9,16	2D Phase field
14	plane strain	5	2	4,9,16	2D Phase field
3	plate	3	2,3	3,4,9	
8	shell-5	5	3	4,9	
10	shell-6	6	3	4,9	
12	shell-7	7	3	4,9	incl. warping dof of beam
5	brick	3	3	8,27	
6	brick	4	3	8,27	Add. dof for th.-mech. coupling
7	brick	6	3	8,27	6 dofs for cosserat theory
15	brick	7	3	8,27	3D Phase field
16	tetraeder	3	3	4	

- **tec,write**

writes all current data on file **Rname.tec** e.g. in each time step. These are: nodal coordinates, elements (only in first step), nodal displacements and nodal stresses which have to be calculated by **stre,node** before!

**Don't forget to write the initial configuration for nonlinear problems!**

- **tec,eigv,n1,n2**

writes current data on file **Rname.tec** These are: nodal coordinates, elements (only in first step), **here** nodal values of eigenvector 'n1', scaled by factor 'n2' (def.=1) and nodal stresses which have to be calculated by **stre,node** before!

- **tec,close**

close the file **Rname.tec**.

- A non-linear calculation could be written completely to TECPLOT with

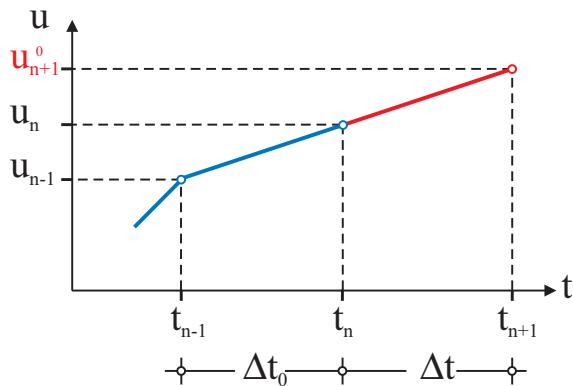
```
tec,init,n1
stre,node
tec,write (initial configuration)
prop,,1
dt,,1
loop,,n-time-steps
  time
  loop,,m-iteration steps
    tang,,1
  next
  stre,node
  tec,write
next
tec,close
```

**TIME****time,,<tmax>****time,start**

- The use of the **time** macro command will increment the current time by the current time increment  $\Delta t$  prescribed by **dt**. In addition, a new value of the proportional loading will be computed, if necessary. The value of the current time and proportional loading are reported in the output (or to the screen).
- The **time** macro also will perform the first update for the Newmark–beta time integration of the equations of motion, as well as, update the stress data base for any elements with non-linear constitutive equations which require variables other than the displacement state to compute a solution. Accordingly, it is imperative to include a time macro for these classes of problems.
- As an option, it is possible to specify the maximum time that integration is to be performed. Accordingly, when a variable time step is employed the 'tmax' value may be used as a convenient stop marker. This also is essential if an automatic time stepping algorithm is implemented.
- As a second option, it is possible to set a start increment with **time, start** as an extrapolation from the last increment. This may lead to less iterations within a nonlinear solution process.

$$\mathbf{u}_{n+1}^0 = \mathbf{u}_n + \frac{\Delta t}{\Delta t_o} \cdot (\mathbf{u}_n - \mathbf{u}_{n-1})$$

Here  $\Delta t$  is the current and  $\Delta t_o$  the last time increment. Time increments can be prescribed with **dt**.



**time, start** is tested for different algorithms

```
# implicit dynamic algorithms: ok
# explicit dynamic algorithms: not allowed
# arcl: should not be used at present
# modified Newton-Raphson: ok
# different time steps Δti: ok
```

Ref: S. Hartmann et.al. Iterative solvers within sequences of large linear systems in non-linear structural mechanics, Z. Angew. Math. Mech. 89(2009), 711-728 (2009)

## **TOL**

---

**tol**,v1

---

The **tol** macro command is used to specify the solution tolerance values to be used at various stages in the analysis by the value 'v1'.

Uses include:

- Convergence of non-linear problems in terms of the norm of energy in the current iterate (the inner dot product of the displacement increment and the solution residual vectors).
- Convergence of the subspace eigenpair solution which is measured in terms of the change in subsequent eigenvalues computed.

The default value of **tol** is

- 1.0d-16 for norm of energy
- 1.0d-12 for eigenvalue calculation.

## TPLO

**tplo,<xxxx>,n1,n2,n3**

The **tplo** macro is used to plot load deflection curves within FEAP.

The following values will be provided for each load step n at node i dof k:

$\lambda^n, t^n, v_{ik}^n, \dot{v}_{ik}^n, \ddot{v}_{ik}^n, R_{ik}^n, \det \mathbf{K}^n, \sigma_{ik}^n, \text{val1}_{ik}^n, \text{val2}_{ik}^n$ ,

Typically values are plotted with respect to displacement  $v_{ik}$  or time  $t$ .

### Initialisation

<b>xxxx = def</b>	define up to 10 nodes for plotting. Do not use this command after restart!!
<b>xxxx = init</b>	initialize macro and define node and dof
<b>xxxx = mark</b>	put marks on load deflection curve
<b>xxxx = set</b>	define active node and values

- **tplo,init,n1,n2,n3** initializes that from node 'n1' the displacement 'n2' will be plotted. The load factor and the nodal reaction force is multiplied by n3 (default = 1). This is helpful in case of symmetry conditions (e.g. double symmetry  $\rightarrow n3 = 4$ ).

If 'n2' is negative a plot is shown versus the negative displacement.

- Alternatively with **tplo,def** up to 10 nodes can be defined for plotting. Up to 3 nodes can be defined by n1,n2 and n3. Otherwise nodes are defined interactively (if n1=0).

Then **tplo,init,n1,n2,n3** is not used. **tplo,set,n1,n2,n3** sets the active node.

The active plotting values n1, n2, n3 can be changed by using **tplo,set,n1,n2,n3**. Further **tplo** macros act on this **active** node, e.g. **tplo,save**.

- **tplo,sset,n1,n2** sets that stress n1 at node n2 will be plotted with respect to values set by **tplo,set,n1,n2,n3**.

- The command **tplo,mark,n1,n2,n3** put marks on the chosen curve in increments 'n3'. The marks are plotted in color 'n1' (default n1=1). Consult the **colo** command for available colors. 'n2' defines the type of mark (default n2=0). The following markers are available:

'n2'	0	1	2	3	4	5	6
marker	small dot	large dot	plus sign	triangle	square	cross	lozenge

All marks can be dropped by **tplo,mark**.

Basically the actual point of the curve is plotted in a separate color(red).

### Plotting of data during Calculations

<b>xxxx = acce</b>	plot acceleration vs time
<b>xxxx = detd</b>	plot determinant vs displacement 2)
<b>xxxx = dett</b>	plot determinant vs time 2)
<b>xxxx = disp</b>	plot displacement vs time
<b>xxxx = load</b>	plot load vs displacement
<b>xxxx = loat</b>	plot load vs time
<b>xxxx = phas</b>	plot velocity vs displacement (phase diagram)
<b>xxxx = reac</b>	plot reaction vs displacement 1,5)
<b>xxxx = reat</b>	plot reaction vs time 1,5)
<b>xxxx = strd</b>	plot stress i at node k vs displacement 3)
<b>xxxx = strt</b>	plot stress i at node k vs time 3)
<b>xxxx = velo</b>	plot velocity vs time
<b>xxxx = tdis</b>	plot time vs displacement
<b>xxxx = us1d</b>	plot valuse1 vs displacement 4)
<b>xxxx = us1t</b>	plot valuse1 vs time 4)
<b>xxxx = us2d</b>	plot valuse2 vs displacement 4)
<b>xxxx = us2t</b>	plot valuse2 vs time 4)

### Provision of data during Calculations

1. The nodal reactions have to be calculated within each time step via **reac**.
2. The determinant is calculated only within arc-length method, extended system, or using SM-solver or if calculated explicitly by the macro **detk**.
3. stress k at node i has to be defined with the macro **tplo,sset**. Furthermore stress has to be calculated via **stre,node** for each time step.
4. Values have to be provided by user. More informations are given in the chapter [Adding elements](#).
5. With **tplo,reac** the reaction force is plotted for the specified dof. If via the macro **rsum** a sum of forces is defined, then this sum is plotted. Same holds for **tplo,reat**.

### **Storage of data and restart options**

<b>xxxx = save</b>	save load deflection data in file rfilename.ldf
<b>xxxx = read</b>	read load deflection data from files rfilename.ldf and rfilename.ldd

- The command **tplo,save** saves load deflection data for the active node in file 'rfilename.ldf'.
- With **tplo,save,n1** the values of the active node are saved in file rfilename\_n1.ldf' ( $0 \leq n1 \leq 9$ ).
- The command **tplo,read** reads load deflection data for the active node from file 'rfilename.ldf'.
- With **tplo,read,n1** the values of the active node are read from file rfilename\_n1.ldf' ( $0 \leq n1 \leq 9$ ).
- Values for nodes defined by **tplo,def** are saved in the file 'rfilename.ldd' automatically. Do not use this command after restart!! Then, these values are automatically set by **tplo,read**. If no macro **tplo,save** has been used, in a restart values are read only from 'rfilename.ldd'. The active node is set to the first defined node.

### **Further options for displaying curves**

<b>xxxx = fact</b>	multiply curve by factors
<b>xxxx = xval</b>	plot between xmin and xmax on screen
<b>xxxx = yval</b>	plot between ymin and ymax on screen
<b>xxxx = conv</b>	define action in case of no convergence
<b>xxxx = incr</b>	set plot increment to n1 (default = 1)

- The command **tplo,fact,n1,n2** multiply loads or reactions by the factor n2 and displacements by  $+(n1 > 0)$  or  $-(n1 < 0)$ .
- The command **tplo,xval,n1,n2** set the plot region on the first axis to  $xmin = n1$  and  $xmax = n2$ ;  $n1=n2=0$ : full region due to curve.
- The command **tplo,yval,n1,n2** set the plot region on the second axis to  $ymin = n1$  and  $ymax = n2$ ;  $n1=n2=0$ : full region due to curve.
- With **tplo,conv,n1** action is defined in case of no convergence in last time step:  $n1=0$ : plot point,  $n1=1$ (default): do not plot point.
- **tplo,incr,n1** plots only every n1-point.

### **Behavior in case of a backstep, see **back****

- After a backstep the nodal reaction force has to be calculated again via **reac,all**.

## TRANS

---

**trans**,name<v1,v2>

---

The use of the macro command **trans** indicates that a transient solution has to be computed. Six options are implemented. The method used depends on the specified **name** in the command.

name	method	type	v1	v2	v3
—	Newmark beta	implicit	$\beta$ (0.25)	$\gamma$ (0.50)	
newm	Newmark beta	implicit	$\beta$ (0.25)	$\gamma$ (0.50)	
back	Euler backward	implicit	-	-	-
hht	HHT-alpha	implicit	$\alpha(0.05)$	-	-
alpha	Generalized-alpha	implicit	$\rho$	$\alpha_m$	$\alpha_f$
ener	Energy momentum conserving	implicit	-	-	
bath	Implicit composite scheme	implicit	$\gamma$	-	
expl	Central differences 1	explicit	-	-	-
expa	Central differences 2	explicit	-	-	-
off	stop, switch to static calc.	-	-	-	-

### General Remarks:

- Algorithms NEWM, BACK, HHT, ALPHA, BATH allow and ENER has always unsymmetric stiffness matrices.
- It is possible to specify nonzero values for the initial velocity using the macro command **init**,rate.
- Same holds for initial displacements (**init**,disp).
- If the initial state is not in equilibrium an initial acceleration must be obtained by using a **form**,accel macro command before initiating any transient state.
- It is also possible to compute self-equilibrating static states with non-zero displacements and then switch to a dynamic solution. Alternatively, a restart mode (**rest**) may be used to start from a previously computed non-zero state.
- A good choice for a time step within implicit methods is  $\Delta t = T/20$ , where T is a period of the vibration of the structure. It can be calculated via: **cmas**, **tang**, **subs**, which leads to values for  $\omega^2$  and  $\omega$ . Then it holds  $T = \frac{1}{f} = \frac{2\pi}{\omega}$ .
- Using the macro command **auto** leads to a variable length of the time step within a dynamic analysis. Currently this macro is implemented for the time stepping algorithms **trans**, NEWM, HHT and ALPHA.

### Remarks on different methods:

- **NEWM** The Newmark-beta step-by-step integration of the equations of motion is a well known reliable method. The values of the Newmark parameters are

$\beta$  - the parameter which primarily controls stability (default is 0.25).

$\gamma$  - the parameter which primarily controls numerical damping ( default is 0.50). Note:  $\gamma \geq 0.50$ .

With  $\beta = 0.25$  and  $\gamma = 0.50$  the method becomes the 'average' acceleration method. The method is absolute stable. Thus large time steps are possible. After a number of time steps oscillations may occur especially for accelerations in the nonlinear range. This may change the total energy dramatically and may lead to divergence. A better numerical damping can be introduced using the HHT-alpha or Generalized-alpha algorithms.

The energy momentum conserving algorithm ENER conserves the total energy of the system.

Macros for algorithm:

initialisation	time steps
<b>cmas</b> or <b>lmas</b> <b>&lt;damp&gt;</b> <b>dt,,xx</b> <b>prop</b> <b>trans</b>	<b>loop,,n</b> <b>time</b> <b>loop,,m</b> <b>tang,,1</b> <b>next(m)</b> <b>next(n)</b>

- **BACK** The Euler-backward difference solution is used for first-order ordinary differential equations such as heat transfer, etc. The method is absolute stable. Thus large time increments are possible.

Macros for algorithm:

initialisation	time steps
<b>cmas</b> or <b>lmas</b> <b>dt,,xx</b> <b>prop</b> <b>trans,back</b>	<b>loop,,n</b> <b>time</b> <b>loop,,m</b> <b>tang,,1</b> <b>next(m)</b> <b>next(n)</b>

- **HHT** The HILBER-HUGHES-TAYLOR (HHT) step-by-step integration of the equations of motion is a modification of the Newmark-beta method. Here the numerical dissipation is handled much better. As input the parameter  $\alpha$  has to be defined, with  $0 \leq \alpha \leq 1/3$ . Newmark method is used for  $\alpha = 0$ .

From  $\alpha$  follows:

$$\beta = 0.25(1 + \alpha)^2$$

$$\gamma = 0.50 + \alpha$$

For  $\alpha = 0$  no numerical damping occurs whereas the maximum damping is introduced for  $\alpha = 1/3$ . A value  $\alpha = 0.05$  is recommended (default).

Macros for algorithm: see Newmark beta

- **ALPHA** The generalized-alpha step-by-step integration of the equations of motion is a modification of the Newmark-beta method. Here the numerical dissipation is handled much better. As input the spectral radius  $\rho$  has to be defined,

From  $\rho$  follows with  $0.5 \leq \rho \leq 1$ :

$$\begin{aligned}\alpha_m &= \frac{2\rho - 1}{\rho + 1} \\ \alpha_f &= \frac{\rho}{\rho + 1}\end{aligned}$$

For  $\rho = 0$  a direct input of  $\alpha_m, \alpha_f$  is possible with  $\alpha_m \leq 0.5, \alpha_f \leq 0.5$ .

It follows

$$\begin{aligned}\beta &= 0.25(1 - \alpha_m + \alpha_f)^2 \\ \gamma &= 0.50 - \alpha_m + \alpha_f\end{aligned}$$

Newmark method is used for  $\alpha_m = \alpha_f = 0$ .

Macros for algorithm: see Newmark beta

- **ENER** The energy momentum conserving step-by-step integration of the equations of motion preserves energy in the nonlinear range in contrast to the Newmark method. Here an unsymmetric stiffness matrix has to be used. The method is absolute stable. Thus large time steps are possible. To use this algorithm the residual G and stiffness matrix  $K_T$  have to be calculated at time  $t_{n+1/2}$ . Check if these features are available for the used element! The main necessary formulas are presented in section Energy-conserving algorithm: element-formulation.

Macros for algorithm:

initialisation	time steps
<b>cmas</b> or <b>lmas</b>	<b>loop,,n</b>
	<b>time</b>
<b>dt,,xx</b>	<b>loop,,m</b>
<b>prop</b>	<b>utan,,1</b>
<b>trans,ener</b>	<b>next(m)</b>
	<b>next(n)</b>

- **BATH** The implicit composite step-by-step integration scheme is able to produce stable solutions of the equations of motion in the nonlinear range by using numerical damping. In contrast to the energy momentum conserving scheme there is no need for modifications on element level. The method is absolute stable. Thus large time steps can be used. But two Newton-Raphson-iterations have to be performed per time step.

#### Remarks:

# A perfect choice for linear calculations:

$$\gamma = 2 - \sqrt{2}$$

Resulting in minimal period elongation with maximum high frequency dissipation. With this choice of  $\gamma$   $K_T$  has to be calculated only one time.

# For nonlinear calculations:

$$\gamma = 0.731$$

resulting in perfect approximations of the accelerations.

Macros for algorithm:

initialisation	time steps
<b>cmas</b> or <b>lmas</b>	<b>loop,,n</b>
	<b>time</b>
<b>dt,,xx</b>	<b>loop,,m</b>
<b>prop</b>	<b>tang,,1</b>
<b>trans,bath</b>	<b>next(m)</b>
	<b>time</b>
	<b>loop,,m</b>
	<b>tang,,1</b>
	<b>next(m)</b>
	<b>next(n)</b>

- **EXPL** - Version RLT: The explicit central difference step-by-step integration of the equations of motion is only stable for small time increments. Thus a large number of time steps are needed. The method is very fast (no iteration, only right hand side terms). A lumped mass and lumped damping matrix has to be used.

**Remarks:**

# No initial calculation is necessary.

# A simple but conservative choice of the critical time step  $\Delta t_c$  is the CFL-condition (Courant, Friedrichs, Lewy):

$$\Delta t_c \leq \frac{L_{min}}{c} = \frac{2}{\omega_{max}}$$

with:  $L_{min}$  = smallest element length,  $c$  = speed of sound  $c = \sqrt{E/\rho}$   $\rho = \gamma/g$ ,  $E$  = elastic modulus.

# An alternate estimate for  $\omega_{max}$  is provided by the Gerschgorin bound :

$$\omega_{max} \leq \max \left( \frac{1}{M_{ii}} \sum_{j=1}^{ndf} K_{ij} \right) \quad i, j = 1, ndf, \quad \Delta t_c = \frac{2}{\omega_{max}}$$

# At present no automatic choice for  $\Delta t_c$  is implemented.

# Typically a large number of solution points occur. Thus, to plot not all steps in a load deflection curve use the macro **tplo,incr,n1**; thus only every n1-step is plotted.

Macros for algorithm:	initialisation	time steps
	<b>lmas</b> <b>&lt;damp,lump&gt;</b> <b>dt,,xx</b> <b>prop</b> <b>trans,expl</b>	<b>loop,,n</b> <b>time</b> <b>form,expl</b> <b>next(n)</b>

- **EXPA** - Version ABAQUS: This algorithm is a modification of EXPL

- an initial calculation is necessary in case of initial accelerations (up to now this is not implemented!).
- implemented is only a version with constant time steps.
- implemented is only a version without damping.

- **OFF** - stops dynamic analysis, FEAP will continue with a static analysis, e.g. **TRANS,HHT** starts dynamic algorithm again. Only one choice of the dynamic algorithm is allowed. Intermediate restart files are different for static or dynamic calculations!

## **UEIG**

---

### **ueig**

---

The macro command **ueig** is used to compute *all* conjugated complex eigenvalues and eigenforms of a non-symmetric matrix generated by the **tang** command. It is not necessary to specify the identity matrix with **iden**.

The eigencomputations uses a special assembly procedure which needs at *isw* = 21 the computation of the non-symmetric matrix within the element routine.

Be careful to use this command for large problems. Since all eigenvalues and -vectors are determined the computing time might be very long.

## **UMAS**

---

### **umas**

---

The macro command **umas** is used to compute a non-symmetric 'mass' matrix. Each element computes a contribution to the consistent mass in the array 's' when 'isw' is 5. A non symmetric 'mass' may occur in transient solutions and can only be used together with **utan** and only for the standard solver.

## UPDH

---

### updh,,n1

---

The **updh** macro command leads to an update of internal variables  $H_2 \rightarrow H_1$  on the micro-problem within a multiscale problem.

With ' $n_1=1$ ' the update is initialized on **Macro**-level.

With ' $n_1=2$ ' the update is performed on **Micro**-level.

#### Remarks:

- The procedure for a time step on macro level must the form

```
n1,0,0  
time      set new time  
loop,,n   begin iteration for solution  
tang,,1   solve problem  
next      end iteration at convergence  
updh,,1   activate update on local level: done by batch,updh on micro-level
```

- The update of internal variables on the micro-problem is done via the procedure batch,updh, see [Material model 8](#).
- The used element on macro level must have a call of the [3D-Material library](#) for isw=15, see chapter [Adding elements](#).

## UTAN

---

**utan**,<n1,v2>  
**utan**,**line**,<n1,v2,v3>

---

The **utan** macro computes the complete tangent stiffness matrix about the current value of the solution vector. The complete tangent stiffness matrix consists of both the upper and lower non-zero profile of the matrix. Accordingly, problems which have an “unsymmetric” tangent stiffness matrix may be considered. For linear applications the current stiffness matrix is just the normal ‘stiffness’ matrix.

If the value of ‘n1’ is non-zero, a force vector for the current residual is also computed (this is identical to the **form** macro computation) — thus leading to greater efficiency when both the tangent stiffness and a residual force vector are needed.

If the value of ‘v2’ is non-zero a “shift” is applied to the stiffness matrix in which the lumped mass matrix (which must already be formed using a **lmas** macro) is multiplied by ‘v2’ and subtracted from the diagonals of the stiffness matrix. This option may be used with the **subspace** algorithm to compute the closest eigenvalues to the shift, ‘v2’. Alternatively, the shift may be used to represent a forced vibration solution in which all loads are assumed to be harmonic at a value of the square-root of ‘v2’ (rad/time-unit).

In the solution of non-linear problems using a full or modified Newton method convergence from any starting point is not guaranteed. Two options exist within available macro commands to improve chances for convergence. One is to use a line search to prevent solutions from diverging rapidly. Specification of the command **utan**,**line** plus options invokes the line search option (it may also be used in conjunction with **solv**,**line** in modified Newton schemes). Line search has been generally used only with symmetric tangents, the algorithm may not be robust for arbitrary unsymmetric tangents. The parameter ‘v3’ may be chosen between 0.5 and 0.8 and will generally produce good performance.

The second option to improve convergence of non-linear problems is to reduce the size of the load step increments. The macro command **back** may be used to “back-up” to the beginning of the last time step (all data in the solution vectors is reset and the history data base for inelastic elements is restored to the initial state when the current time is started). This is the only method which is quite general when unsymmetric tangents are used. Repeated use of the **back** command may NOT be used. Only one back-up in time is permitted. The loads may then be adjusted and a new solution with smaller step sizes started.

After the tangent matrix is computed, a triangular decomposition is performed for subsequent solutions using **form** and **solv**, etc. Note that the **bfgs** algorithm is available only for symmetric tangents and will not work in conjunction with **utan**.

## **VELO**

---

**velo,,<n1,n2,n3>**

**velo,all**

---

The macro command **velo** may be used to print the current values of the “velocity” vector as follows:

1. Using the macro command:

**velo,,n1,n2,n3**

prints out the current velocity vector for nodes 'n1' to 'n2' at increments of 'n3' (default = 1). If 'n2' is not specified only the value of node 'n1' is output. If both 'n1' and 'n2' are not specified only the first nodal velocity is reported.

2. If the macro command is specified as:

**velo,all**

prints all nodal quantities.

In order to output a solution vector it is first necessary to specify macro commands to compute the desired values, i.e., a dynamic analysis.

## WRIT

---

**writ,xxxx**

---

The **writ** macro command may be used to save the current values of displacements and nodal stresses for subsequent use. This option is particularly useful for saving states which are to be plotted later. It is not intended as a restart option (see **rest** for restarting a previously saved problem state).

The values of **xxxx** are used to specify the file name (4-characters only), manipulate the file, and write out states. The values permitted are:

<b>xxxx = wind</b>	rewind the current file (note in a file is normally opened and positioned at the end-of-file mark).
<b>xxxx = clos</b>	close the current output file.
<b>xxxx = disp</b>	write the current displacement state onto the current file.
<b>xxxx = stre</b>	write the current nodal stress state onto the current file.
<b>xxxx = anything else</b>	will be used to name the current file. Only four characters are permitted and only one file may be opened at any time. Files may be opened and closed several times during any run to permit the use of more than one file name.

A **writ** output is reinput using the **read** macro command which has identical options for **xxxx**.

**writ** outputs can be summarized with the **summ** macro command.

## **YANG**

---

**yang**

---

The macro command **yang** may be used to print the angles of the deflected planes after a yield-line calculation step performed by **ytab**.

## **YEVO**

---

**yevo**

---

The macro command **yevo** performs an evolution step during a yield-line optimization process.

## **YGRA**

---

### **ygra**

---

The macro command **ygra** detects the gradient of the search directions during a yield-line optimization process based on the gradient method. The step length is calculated automatically, but it can be defined by using the **dt** command. To reactivate its automatical definition set 'dt' to zero. After a gradient step every new mesh has to be loaded by the **ymsh** command.

For further details concerning the yield-line optimization process using linear programming, see the **ytab** macro command.

## **YMSH**

---

### **ymsh**

---

The macro command **ymsh** is used to reload the yield-line mesh which was modified during the yield-line optimization process based on the gradient method.

For further details concerning the yield-line optimization process using linear programming, see the **ytab** macro command.

## **YTAB**

---

### **ytab**

---

The **ytab** macro computes the simplex optimization tableau during a yield-line calculation based on linear programming. The minimal load factor is solved and printed for the current optimization step.

- Performing the gradient method the **ygra** command has to be used afterwards. Furthermore the modified mesh has to be reloaded by the **ymsh** command before using **ytab** again.
- Performing the direct search method the **ytry** command is used. It is not necessary to reload the mesh.

For both methods the step length may be changed by using the **dt** command.

Note: the selected optimization method should not be changed during the calculation process.

## **YTRY**

---

### **ytry**

---

The macro command **ytry** performs a trial step during a yield-line optimization calculation based on the direct search method. At the beginning of a calculation process the step length is defined by  $\frac{1}{10}$  of the smaller plate dimension along the coordinates. The step length may be modified by using the **dt** command.

For further details concerning the yield-line optimization process using linear programming, see the **ytab** macro command.

# Chapter 4

## Plot Commands

### 4.1 Available Macros

In FEAP, plots may be made for several quantities of interest in 2/3-dimensional problems. A **plot quantity** must be specified to initiate graphics outputs. After entering graphics mode a prompt will be displayed. At this time, **quantity** and the 'n1', 'n2', and 'n3' values may be specified. Alternatively, a **plot,quantity,n1,n2,n3** command may be issued while in macro execution mode (this is the only option for batch executions).

The values n1,n2,n3 can be predefined parameters. This can be done e.g. on input-level via **para** or on macro-level via **para**. Furthermore calculations for n1,n2,n3 are possible, each up to 15 characters. For details see the description how to define **parameter**.

The following may be used for **quantity**:

aacc	acce	adis	aeig	aeve	angl	avel	axis
back	base	bord	boun	cart	ceig	cent	clip
colo	copy	defm	defo	dimp	disp	dmag	dplo
draw	eigi	eigv	elem	end	eplo	erro	evan
evex	fact	flux	forc	fram	hide	hids	hmsh
init	isec	ints	isom	jint	line	link	load
logo	magn	man	mate	matn	maxi	mesh	mono
move	movi	ndii	node	outl	pele	pers	pdis
plof	pnod	pola	prin	pris	prof	quit	reac
rmsh	resi	rot0	rot1	rot2	rot3	rotm	rplo
rsum	scal	sect	show	size	slee	splo	stre
str1	symm	text	tie	tplo	traj	ueig	velo
wipe	xsca	zoom					

A short overview on commands is given in section 4.2, see macro **Index** in FEAP, whereas possible actions can be found in section 4.3, see macro **Action** in FEAP.

The action to be taken by each command is described in the following sections of the PLOT Manual.

## 4.2 Overview on commands

Macro	Aufgabe	Macro	Aufgabe
aacc acce adis aeig aeve angl avel axis	Beschleunigungen (Pfeile) Beschleunigungen Verschiebungen (Pfeile) Eigenvektoren (Pfeile) Eigenvektor Ext.Syst. (Pfeile) Knoten: mit Winkeln Geschwindigkeiten (Pfeile) Koordinatensystem	line link load logo	Liniendarstellung Knoten: gelinkt Kräfte Logo - Farbe
back base bord boun	Hintergrund Dreibein Rahmen Randbedingungen	magn main man mate matn maxi mesh mono move movi	Vergrößerung des Plotbildes Spannungen - Hauptspannungen Manual Materialdarstellung Materialnummer Maximalwerte Versch./Spannungen etc. System Farbe: Farbe/schwarz - weiss Position Bild auf Schirm
cart ceig cent clip colo copy	Cartesische Darstellung  Zentriere Bild Ausschnitt Farbe ändern Bildausgabe (WINDOWS)	ndii node outl	Knoten: negative Diagonalelemente Knoten System: Umrisse
defm defo dimp disp dmag dplo draw	System verformt Plotte auf verformtes System Imperfektionen+Verschiebungen Verschiebungen Schädigung Schnitt: Verschiebungen Zeichne Linien	pele pers pdis ploff pnod pola prin prof quit	Elementnummer per Mausklick Perspektivische Darstellung Vorgegebene Verschiebungen Bildausgabe (CGM) Knotennummer per Mausklick Polarkoordinaten Bildausgabe (PS) Profil der Steifigkeitsmatrix Ende Plot
eigi eigv elem end eplo erro evan evex	Eigenvektor: inverse Iteration Eigenvektor: Subspace Iteration Elementnummern Ende Plot Schnitt: Eigenvektoren Fehlerverteilung Eigenvektor: Animation Eigenvektor: Extended System	reac resi rot0 rot1 rot2 rot3 rotm rplo rsum	Reaktionskräfte Residuum Rotation: Zurücksetzen Rotation: um Achse 1 Rotation: um Achse 2 Rotation: um Achse 3 Rotation: Drehung mit Maus Schnitt: Reaktionskräfte Reaktionskraefte bei mehreren Knoten
fact flux forc fram	Größenfaktor Fluss Schnittgrößen Teilrahmen	scal sect show size slee splo stre str1 symm text tie	Vergrößerungsfaktor Querschnittswerte Zustandsinformationen Schriftgroesse Pause Schnitt: Spannungen Spannungen Spannungen - elementweise konstant Symmetrie Text Knoten: zusammenfallend
hide hids hmsh	Hidden Line 3D Hidden Line Schale System gefüllt		
init ints isec isom jint	Anfangswerte setzen Spannungen über Dicke Verschneidungen Isometrische Darstellung Materielle Kräfte		

Macro	Aufgabe
<b>tplo</b>	Kurvendiagramme
<b>traj</b>	Trajektorien der Spannungen
<b>ueig</b>	Eigenvektor: unsymmetrisch
<b>velo</b>	Geschwindigkeiten
<b>wipe</b>	Lösche Bild
<b>xsca</b>	Skalierung
<b>zoom</b>	Ausschnitt

## 4.3 Overview on actions

Aufgabe	Macro	Aufgabe	Macro
Anfangswerte setzen	init	Manual	man
Ausschnitt	zoom	Materialdarstellung	mate
Ausschnitt	clip	Materialnummer	matn
Beschleunigungen	acce	Materielle Kräfte	jint
Beschleunigungen (Pfeile)	aacc	Maximalwerte Versch./Spannungen	maxi
Bildausgabe (CGM)	plof	Pause	slee
Bildausgabe (PS)	prin	Perspektivische Darstellung	pers
Bildausgabe (WINDOWS)	copy	Plotte auf verformtes System	defo
Cartesische Darstellung	cart	Polarkoordinaten	pola
Dreibein	base	Position Bild auf Schirm	move
Eigenvektor: Subspace Iteration	eigv	Profil der Steifigkeitsmatrix	prof
Eigenvektor: (Pfeile)	aeig	Querschnittswerte	sect
Eigenvektor: Animation	evan	Rahmen	bord
Eigenvektor: Extended System	evex	Randbedingungen	boun
Eigenvektor: Ext.Syst. (Pfeile)	aeve	Reaktionskräfte	reac
Eigenvektor: inverse Iteration	eigi	Residuum	resi
Eigenvektor: unsymmetrisch	ueig	Rotation: Drehung mit Maus	rotm
Elementnummern	elem	Rotation: um Achse 1	rot1
Elemente: Nr. per Maus	pele	Rotation: um Achse 2	rot2
Ende Plot	end	Rotation: um Achse 3	rot3
Ende Plot	quit	Rotation: Zurücksetzen	rot0
Farbe ändern	colo	Reaktionskraefte: Knoten bei Summe	rsum
Farbe: Farbe/schwarz - weiss	mono	Schädigung	dmag
Fehlerverteilung	erro	Schnitt: Eigenvektoren	eplo
Fluss	flux	Schnitt: Reaktionskräfte	rplo
Geschwindigkeiten	velo	Schnitt: Spannungen	splo
Geschwindigkeiten (Pfeile)	avel	Schnitt: Verschiebungen	dplo
Größenfaktor	fact	Schnittgrössen	forc
Hidden Line 3D	hide	Schriftgrösse	size
Hidden Line Schale	hids	Skalierung	xска
Hintergrund	back	Spannungen	stre
Isometrische Darstellung	isom	Spannungen - elementweise konstant	str1
Imperfektionen+Verschiebungen	dimp	Spannungen - Hauptspannungen	main
Koordinatensystem	axis	Spannungen über Dicke	ints
Knoten	node	Symmetrie	symm
Knoten: gelinkt	link	System	mesh
Knoten: negative Diagonalelemente	ndii	System: gefüllt	hmsh
Knoten: mit Winkeln	angl	System: Umrisse	outl
Knoten: zusammenfallend	tie	System: verformt	defm
Knoten: Nr. per Maus	pnod	Trajektorien der Spannungen	traj
Kräfte	load	Teilrahmen	fram
Kurvendiagramme	tplo	Text	text
Liniendarstellung	line		
Lösche Bild	wipe		
Logo - Farbe	logo		

Aufgabe	Macro
Vergrösserung des Plotbildes	<code>magn</code>
Vergrösserungsfaktor	<code>scal</code>
Verschiebungen	<code>disp</code>
Verschiebungen (Pfeile)	<code>adis</code>
Verschneidungen	<code>isec</code>
Vorgegebene Verschiebungen	<code>pdis</code>
Zeichne Linien	<code>draw</code>
Zentriere Bild	<code>cent</code>
Zustandsinformationen	<code>show</code> <code>ceig</code> <code>movi</code>

## 4.4 Macros in detail

### AACC(E)

---

**aacc,<n1,n2,n3>**

---

Plot all nodal accelerations currently calculated. The maximum length will be automatically scaled. All other accelerations will be scaled in proportion to the maximum, accordingly some very small values may scale to be too small to be visible.

In addition the results can be scaled by the value of 'n3'.

If 'n1' is negative accelerations will be positioned relative to the deformed mesh. If 'n2' is nonzero the vector tip will appear next to the node whereas if 'n2' is zero the tail of the acceleration vectors are on the nodes.

## **ACCE**

---

**acce,<v1,v2,v3,v4>**

---

Plot contours for degree of freedom (i.g. acceleration) 'n1'. For negative values of 'n1' the acceleration are plotted on the deformed mesh.

The macro leads to a filled plot for 'n2'= 0. On the other hand contour lines are plotted for 'n2'> 0.

### **Filled plots**

In interactive mode **FEAP** presents a profile of results with maximum and minimum values. Then **FEAP** asks for the number of used colors and the maximum and minimum value. Up to 14 colors are possible. The default value for the number of colors is 14. As default **FEAP** calculates the contour lines automatically between the extremal values, otherwise between the given input values.

### **Contour plots**

In interactive mode **FEAP** presents a profile of results with maximum and minimum values. Then **FEAP** asks for the number of used contours and the maximum and minimum value. Up to 14 contours are possible. The default value for the number of contours is 14. As default **FEAP** calculates the contour lines automatically between the extremal values, otherwise between the given input values.

For contour line plots, a non-zero 'n3' value will suppress numbers near each contour line. With 'n4' the mesh is plotted in color 'n4' additionally.

Compare the handling of this macro with **stre**.

In batch mode, the requisite data described above must appear after the **end** macro command in the order that the program needs inputs.

## **ADIS(P)**

---

**adis,<n1,n2,n3>**

---

Plot all nodal displacements currently calculated. The maximum length will be automatically scaled. All other displacements will be scaled in proportion to the maximum, accordingly some very small values may scale to be too small to be visible.

In addition the results can be scaled by the value of 'n3'.

If 'n1' is negative displacements will be positioned relative to the deformed mesh. If 'n2' is nonzero the vector tip will appear next to the node whereas if 'n2' is zero the tail of the displacement vectors are on the nodes.

## **AEIG(V)**

---

**aeig,<n1,n2,n3>**

---

Plot all nodal displacements of eigenvector 'n1'. The maximum length will be automatically scaled. All other displacements will be scaled in proportion to the maximum, accordingly some very small values may scale to be too small to be visible.

In addition the results can be scaled by the value of 'n3'.

If 'n1' is negative displacements will be positioned relative to the deformed mesh. If 'n2' is nonzero the vector tip will appear next to the node whereas if 'n2' is zero the tail of the displacement vectors are on the nodes.

## **AEVE(X)**

---

**aeve,<n1,n2,n3>**

---

Plot all nodal displacements of eigenvector from extended system. The maximum length will be automatically scaled. All other displacements will be scaled in proportion to the maximum, accordingly some very small values may scale to be too small to be visible.

In addition the results can be scaled by the value of 'n3'.

If 'n1' is negative displacements will be positioned relative to the deformed mesh. If 'n2' is nonzero the vector tip will appear next to the node whereas if 'n2' is zero the tail of the displacement vectors are on the nodes.

## **ANGL**

---

**angl,<n1,n2,n3>**

---

Plot locations of all nodes with angles. Only nodes within the current plot region are displayed.

In case 'n1' is negative the nodes are plotted relative to the deformed mesh.

In case 'n2' is greater than zero the node numbers are plotted too.

In case 'n2' is less than zero only the node with number n2 is plotted.

In case 'n3' is greater than zero the local basis is plotted. The length can be scaled by 'n3' (base=1).

## **AVEL(O)**

---

**avel,<n1,n2,n3>**

---

Plot all nodal velocities currently calculated. The maximum length will be automatically scaled. All other velocities will be scaled in proportion to the maximum, accordingly some very small values may scale to be too small to be visible.

In addition the results can be scaled by the value of 'n3'.

If 'n1' is negative velocities will be positioned relative to the deformed mesh. If 'n2' is nonzero the vector tip will appear next to the node whereas if 'n2' is zero the tail of the displacement vectors are on the nodes.

## **AXIS**

---

**axis,<v1,v2,v3>**

---

A set of axes defining the coordinate directions will be plotted with the origin of the axes placed at coordinates  $x_1='v1'$ ,  $x_2='v2'$ ,  $x_3='v3'$ . The  $x_1$ ,  $x_2$  and  $x_3$  coordinates are in the dimensions of the problem distances.

## **BACK**

---

**back,<n1,n2,n3,n4>**

---

Set the background color if 'n1' = 1.

- WIN

n1 = 0 back = white (default for n1)

n1 = 1 back = black

- GKS

n1 = 0 back = black (default for n1)

n1 = 1 back = white (default,if n1 = 1)

n1 = 1 back = red(n2), green(n3), blue(n4)

## **BASE**

---

**base,<n1,n2,n3>**

---

Plot nodal basis at nodes. The maximum length will be automatically scaled. The results can be scaled by the value of 'n1'(default 1). If the basis is calculated on element level vectors can be plotted from element 'n2' to 'n3' (default 1 to numel). Otherwise if the basis is calculated for global nodes the basis vectors can be plotted from node 'n2' to 'n3' (default 1 to numnp).

## BORD

---

**bord,n1**

---

The command redraws the border in color 'n1'. Consult the **colo** command for tables of available colors. In a similar way the **FEAP**-logo is redrawn by the **logo**-command.

If 'n1' is 0, **bord = wipe**.

If 'n1' is negative, border and logo are not plotted!

## **BOUN**

---

**boun,<n1,n2>**

---

The boundary restraints will be indicated by small triangles drawn at each node in the direction of the imposed restraint. If 'n1' is zero or positive the restraints are drawn through the undeformed position of the node, whereas, if 'n1' is negative the restraints are through the deformed position of the nodes.

The absolute value of 'n1' is used as scaling factor.

With 'n2' it is possible to plot only the boundary conditions for 'dof 2'. The default value is 0, which leads to a plot of all b.c. .

The boundary conditions 1-3 are drawn in red, 4-6 and further conditions in orange.

The boundary conditions 4-6 overwrite the values 1-3.

### **Remarks:**

1. If **angl** is specified in the mesh input data, the boundary codes appear in the sloping direction.
2. Linked dofs/nodes are also plotted when master nodes have boundary conditions! Additional use of the macro **link** plot the link-behaviour.

## CART

---

**cart**

---

All plots will be drawn in a cartesian frame. This is the default view of plots. A plot may also be in a perspective view (see **pers** plot manual) or an isometric view (see **isom** manual entry).

N.B. Problems of centering the isometric view may still exist.

## **CEIG**

---

**ceig**<n1,n2,n3>

---

This macro is up to now not documented.

## CENT

---

**cent**, $\langle v1,v2 \rangle$

---

With the macro **center** it is possible to center the plot output on the screen.  $v1$  and  $v2$  are center coordinates [ $0 < v1,v2 < 1$ ]. For  $v1 = v2 = 0$  the position can be defined with the mouse cursor.  $v1 = v2 = -1$  resets the center coordinates to  $0.5/0.5$ .

## **CLIP**

---

### **clip**

---

Using the **clip** command a closeup view of any part of the mesh may be plotted.

Use the mouse to define two corners of the region to appear in subsequent plots. After the first point is located and marked (by pressing the left mouse button) the region will appear in a 'rubber band' box (still press the left mouse button) until the second position is located and marked (release the left mouse button). Any subsequent plots will be positioned to show only the 'clipped' part of the mesh. To return to the full view of the mesh enter **zoom** as a plot instruction.

The **clip** command may be repeated on any mesh or part as many times as an element is still appearing on the screen.

## COLO

---

**colo,n1,n2**

---

- Change plot color to 'n1' value. This command works not in combination with all macros. If 'n1' is negative the previous plot color is set to the next value in the table. **Colo,0** resets all colors to the initial values.
- Contour plots are plotted in the range *color1* → *color2*.  
**colo,, -1** change the color direction in: *color2* → *color1*.  
**colo,,0** resets the range to the initial direction.
- Color table for **FEAP**

FEAP Table		
n1	Color	change for VGA
1	white	
2	red	
3	blue	
4	cyan	
5	green	
6	orange	
7	yellow	
8	magenta	light cyan
9	light green	
10	light cyan	
11	dark red	
12	dark cyan	
13	med red	
14	dark orange	
15	white	dark blue
16	black	
17-30	filled plots	
31	white	
32	black	

The colors for filled plots are from blue to red (color) or from black to white (b & w).

## COPY

---

**copy**,*<n1>*

---

With the **copy** command the actual graphic window can be copied to the clipboard for further use within standard Windows applications. 'n1=1' (default) copies the whole window whereas 'n1=2' copies a part of the window defined with the mouse. Move mouse to first point and press left button. Continue to press this button while moving mouse to the 2. point. Loose button which terminates the procedure.

**Remark:** Use **back,,1** before to have a white background.

The **copy** command acts only in the Windows-Version.

## **DEFM**

---

**defm,<n1,n2,n3>**

---

This command will plot a deformed mesh with the displacements multiplied by the 'n1' value (default: n1 = 1). If any part of an element in the deformed mesh leaves the plot region, it will not appear in the plot. If nodes or element numbers are added some "unconnected" values may appear on the screen if the deformed plot option is used.

The **scal** command may be used to increase the region of the plot, and thus include all elements in a plot. An alternative to the **defm** command is **mesh,-1**.

The deformed mesh can be plotted only for material 'n2' (default: complete mesh)

The deformed mesh is plotted in color 'n3'. Consult the **colo** command for tables of available colors. The default value is 'n3' = 5 (green).

## **DEFO**

---

**defo,n1**

---

**defo** defines if the plot results acts on the undeformed or the deformed mesh. For 'n1' > 0 all results are plotted on the deformed mesh. Thus in all other plot-macros 'n1' has not to be negative. The macro is active until '**defo,0**' (0= default). Then all results are plotted on the undeformed mesh.

## DIMP

---

**disp,n1,n2,n3,n4**

---

Plot imperfections(introduced in the input-file via **impf** and displacements. Parameter and handling are similar to the macro **disp**.

## **DISP**

---

**disp,n1,n2,n3,n4**

---

Plot contours for degree of freedom (i.g. displacement) 'n1'. For negative values of 'n1' the displacements are plotted on the deformed mesh.

The macro leads to a filled plot for 'n2'= 0. On the other hand contour lines are plotted for 'n2' > 0.

### **Filled/Contour plots**

FEAP calculates 14 contour lines automatically between the extremal values. Values can be changed by **init,2**.

In interactive mode (by **init,1**) FEAP presents a profile of results with maximum and minimum values. Input is the number of used colors and the maximum and minimum value. Up to 14 (default) colors are possible. As default FEAP calculates the contour lines automatically between the extremal values, otherwise between the given input values.

For contour line plots, a non-zero 'n3' value will suppress numbers near each contour line. With 'n4' the mesh is plotted in color 'n4' additionally.

For a further description see the macro **stre**.

## **DMAG**

---

### **dmag**

---

Plot gauss-points where a damage has been found by a user defined criterion. This criterion is has to be defined in each element.

## DPLO

---

**dplo**,n1,<n2,n3>

---

- The '**dplo**' command may be used to construct plots of displacements along any specified line in the mesh for 2D-problems. 3D-problems can be calculated too if 'pers' and 'hide' is used. The line is than on the 3d-element faces.
- The line is scaled to 0-1 for the intersected part of mesh, not between the line ends.
- To define the ends of the line two options are possible:

# Input of cartesian coordinates via the Macro **splo, set** in **Macro-modus**.

# If the first option is not active, the screen coordinates of the ends of the line are defined via the mouse cursor.

For INTEL: Mark the the first point(1) with a left button click of the mouse. Do the same for the second point(2).

For FTN: Mark the the first point(1) by pressing the left button. Hold the button down until the mouse cursor is located at the position of the second point(2). Here, release the left button.

If this option does not work set coordinates to zero via **splo** in **Macro-modus**.

- A 3-d formulation is possible, but note that now hidden line technic is used. Thus all visible intersections are calculated.
- The intersection bases only on the 4 corner nodes, thus 8/9 nodes are possible but the additional nodes are not used.
- Admissible screen macros for view are **move**, **rot**, **isom** and **pers**. **clip** may 'work'.
- The 'n1' parameter is used to specify the displacement component to be plotted. A negative value of 'n1' acts on the deformed mesh.
- The 'n2' parameter is used to specify the scaling process.

n2 = 0 plot new value in diagram

n2 = 1 plot other value in same diagram with rescaling

n2 = 2 plot other value in same diagram without rescaling

- If 'n2' is < 0 the coordinates are not plotted but printed to screen and output-file.
- The 'n3' parameter is not used.

## DRAW

---

**draw,<n1,n2,n3>**

---

Draw a line. Color and line type can be set by **colo** and **line**.

- If 'n1' is = 0, the line is defined between node n2 and node n3.
- If 'n1' is < 0, the line is drawn on the deformed mesh.
- If 'n1' is = 1, the line is drawn interactiveley with the mouse. A point is defined with the left mouse button. The procedure is finished by pressing the right mouse button.
- If 'n1' is = 2, the line is defined by coordinates of begin and end point.

**EIGI****eigi,<n1,n2,n3,n4>**

Plot the deformed mesh which is proportional to the eigenvector from inverse iteration.

The use of 'n2' to 'n3' depends on what type of plot should be made. Three plots are possible.

**Eigenvector as deformed mesh**

ni	action
n1	number of eigenvector (only 1 possible), n1 < 0 plot filled mesh
n2	n2=2 plot from numel to 1, otherwise from 1 to numel
n3	0

**filled plot for one DOF of eigenvector**

ni	action
n1	number of eigenvector (only 1 possible), n1 < 0 plot on defm mesh (=eigi)
n2	$\leq 0$
n3	dof to be plotted

**contour plot for one DOF of eigenvector**

ni	action
n1	number of eigenvector (only 1 possible), n1 < 0 plot on defm mesh (=eigi)
n2	$> 0$ , n2 = number of contour lines (< 8)
n3	dof to be plotted, n3 > 0 : plot with numbers
n3	dof to be plotted, n3 < 0 : plot without numbers

With 'n4' the mesh is plotted in color 'n4' additionally. For a further description see the macros **stre** and **disp**.

**EIGV**

---

**eigv,<n1,n2,n3,n4>**

---

Plot the deformed mesh which is proportional to eigenvector 'n1' (an eigensolution must be performed before attempting a plot!).

The use of 'n1' to 'n3' depends on what type of plot should be made. Three plots are possible.

**Eigenvector as deformed mesh**

ni	action
n1	number of eigenvector, n1 < 0 plot filled mesh
n2	n2=2 plot from numel to 1, otherwise from 1 to numel
n3	0

**filled plot for one DOF of eigenvector**

ni	action
n1	number of eigenvector, n1 < 0 plot on defm mesh (=eigv)
n2	$\leq 0$
n3	dof to be plotted

**contour plot for one DOF of eigenvector**

ni	action
n1	number of eigenvector, n1 < 0 plot on defm mesh (=eigv)
n2	$> 0$ , n2 = number of contour lines ( $< 8$ )
n3	dof to be plotted, n3 > 0 : plot with numbers
n3	dof to be plotted, n3 < 0 : plot without numbers

With 'n4' the mesh is plotted in color 'n4' additionally. Eigenvectors are scaled that the maximum entry is one.

For a further description see the macros **stre** and **disp**.

## **ELEM**

---

**elem,<n1,n2,n3>**

---

Plot numbers in or near the elements appearing in the plot region visible. After a **zoom** some numbers may appear for elements surrounding the plot region even though no lines for element edges are plotted.

For 'n1' less than zero the element numbers are plotted on the deformed mesh.

For 'n2' greater than zero only the element with number n2 is plotted. For 'n2' less than zero element with number n2 is plotted on the deformed mesh.

The element numbers are plotted in color 'n3'. Consult the **colo** command for tables of available colors. The default value is 3 (blue).

**END**

---

**end**

---

This macro terminates the plot modus. The same action is achieved with **quit** or **q**.

## EPLO

---

**eplo**,n1,<n2>,n3

---

- The **eplo** command may be used to construct plots of displacements of the eigenvector n3 along any specified line in the mesh for 2D-problems. 3D-problems can be calculated too if **pers** and **hide** is used. The line is than on the 3d-element faces.
- The line is scaled to 0-1 for the intersected part of mesh, not between the line ends.
- To define the ends of the line two options are possible:

# Input of cartesian coordinates via the Macro **splo**,**set** in **Macro**-modus.

# If the first option is not active, the screen coordinates of the ends of the line are defined via the mouse cursor.

For INTEL: Mark the the first point(1) with a left button click of the mouse. Do the same for the second point(2).

For FTN: Mark the the first point(1) by pressing the left button. Hold the button down until the mouse cursor is located at the position of the second point(2). Here, release the left button.

If this option does not work set coordinates to zero via **splo** in **Macro**-modus.

- A 3-d formulation is possible, but note that now hidden line technic is used. Thus all visible intersections are calculated.
- The intersection bases only on the 4 corner nodes, thus 8/9 nodes are possible but the additional nodes are not used.
- Admissible screen macros for view are **move**, **rot**, **isom** and **pers**. **clip** may 'work'.
- The 'n1' parameter is used to specify the displacement component to be plotted. A negative value of 'n1' acts on the deformed mesh.
- The 'n2' parameter is used to specify the scaling process.

n2 = 0 plot new value in diagram

n2 = 1 plot other value in same diagram with rescaling

n2 = 2 plot other value in same diagram without rescaling

- If 'n2' is < 0 the coordinates are not plotted but printed to screen and output-file.
- The 'n3' parameter defines the number of the plotted eigenvector.

## **ERRO**

---

**erro,< n1,n2 >**

---

The distribution of error indicators is plotted with the macro **erro**.

To do this, these values have to be calculated with **erro**.

See documentation if error analysis is available for used element.

Results can be printed with **erro**.

The theoretical background is described in the [Theory Manual](#).

The results are given with respect to a user defined error value 'n1' in percent (default: 5 %).

'n2' allows to choose the type of error indicator. With 'n2=1' the Energy-based (default) and with 'n2=2' the  $L_2$ -based indicator is chosen.

**EVAN**

---

**evan,<n1,n2,n3>**

---

Plot the eigenvector 'n1' in animation mode. At first the eigenvector is shown by 8 steps which switches between  $\pm 1$  in steps of 0.5. Then 6 steps are shown which switches between  $\pm 1$  in steps of 1.

- For ' $n1 > 0$ ' the mesh is plotted with contours whereas for ' $n1 < 0$ ' the mesh is filled plotted.
- Hidden line can be achieved for ' $n1 < 0$ ' by

' $n2 \geq 0$ ' plot from 1 to numel  
' $n2 < 0$ ' plot from numel to 1

- The speed may depend on the computer on which **FEAP** runs. Thus the program stops ('sleeps') after a mesh is plotted for ' $n3$ ' seconds ( default = 0 sec.).

**EVE<sub>X</sub>****evex**,<n1,n2,n3,n4>

Plot the deformed mesh which is proportional to eigenvector of the extended system (plot can be shown only if **ext** = on).

The use of 'n2' to 'n3' depends on what type of plot should be made. Three plots are possible.

**Eigenvector as deformed mesh**

ni	action
n1	number of eigenvector (only 1 possible), n1 < 0 plot filled mesh
n2	n2=2 plot from numel to 1, otherwise from 1 to numel
n3	0

**filled plot for one DOF of eigenvector**

ni	action
n1	number of eigenvector (only 1 possible), n1 < 0 plot on defm mesh (=evex)
n2	$\leq 0$
n3	dof to be plotted

**contour plot for one DOF of eigenvector**

ni	action
n1	number of eigenvector (only 1 possible), n1 < 0 plot on defm mesh (=evex)
n2	$> 0$ , n2 = number of contour lines (< 8)
n3	dof to be plotted, n3 > 0 : plot with numbers
n3	dof to be plotted, n3 < 0 : plot without numbers

With 'n4' the mesh is plotted in color 'n4' additionally. For a further description see the macros **stre**

and **disp**.

## **FACT**

---

**fact,v1**

---

The entire plot is scaled by the value of 'v1' (default = 1.). Usually is better to scale the plot using **scal** to permit the entire deformed region to appear in the screen area.

## FLUX

---

**flux,n1,<n2,n3,n4>**

---

Flow vectors for 2D- and 3D-elements based on the Laplace-Equation, e.g. heat-flux problems or ground-water problems, can be plotted for 'n1=1'.

The principal stressees can be plotted by 'load'-vectors for 'n1=2'.

The vectors are plotted at each nodal point.

If 'n1' is negative the plot acts on the deformed mesh.

'n2=2'(default) plots for 2D-elements, 'n2=3' plots for 3D-elements (only for Principal Stresses, must be implemented for heat flux!)

With 'n3' the length of the vectors can be multiplied by a factor.

With 'n4' the length of the tip can be multiplied by a factor (only 2D-stress).

Heat fluxes must be stored in elements (see manual)

2D:  $f_x, f_y$

Stresses must be stored in elements (see manual)

2D:  $S_{xx}, S_{xy}, S_{yy}$

3D:  $S_{xx}, S_{yy}, S_{zz}, S_{xy}, S_{xz}, S_{yz}$

A similar action to **flux,n2** is given by the **stre** command. (**stre,27,<n2,n3,n4>**).

## FORC

---

**forc**,n1,<n2,n3>

---

For beam elements a plot of the element forces may be available by the command **forc**. 'n1' defines the number of force to be plotted. If 'n1' is negative the force will be plotted relative to the deformed mesh (if available).

### Filled/Contour plots

FEAP calculates 14 contour lines automatically between the extremal values. Values can be changed by **init**,2.

In interactive mode (by **init**,1) FEAP presents a profile of results with maximum and minimum values. Input is the number of used colors and the maximum and minimum value. Up to 14 (default) colors are possible. As default FEAP calculates the contour lines automatically between the extremal values, otherwise between the given input values.

The force can be scaled by a positive value 'n3'.

For layered beam elements the number of layer for which the force 'n1' should be plotted is specified by '-n3' (default = 1, which is also valid for non layered elements).

Furthermore one can influence the plane of plotting the forces with 'n1'. (This may not work for all beam elements!)

**forc**, 12 leads to plots in the local 1-2 plane.

**forc**, 13 leads to plots in the local 1-3 plane.

For -12 and -13 the plots are multiplied by -1. The initial set is the plane 13.

## **FRAM**

---

### **fram,n1**

---

This macro command defines a window on the screen for a plot.

n1	Region used
0	entire screen used
1	upper left quadrant
2	upper right quadrant
3	lower left quadrant
4	upper right quadrant

The default value is n1 = 0.

Using the various frames a large amount of information may be placed on a single screen. Each 'frame' may be cleared independently for some devices by using the **wipe** command.

## HIDE

---

**hide**,n1

---

**hide** is used to plot 3D elements (8-nodes) in a 3-d space using a hidden line technique. 20,21,27-node elements may be plotted correct, but only with respect to node 1-8. 2D shell elements (4 nodes) may occur. Further combinations with beams are not tested. In a first step the nearest faces of the elements to the viewpoint have to be calculated. In a second step the elements are sorted with respect to the distance from the viewpoint.

'n1' ≠ 0 enables and 'n1' = 0 disables the hidden line plot.

'n1 > 0 acts on the undeformed and 'n1' < 0 acts on the deformed mesh.

For a perspective view the view point is defined within **pers**, whereas for **rot** the viewpoint coordinates are chosen within this macro.

**hide** acts on **mesh,hmsh, eigv, disp, velo, acce, stre** and similar macros.

Partial parts of the mesh are plotted using **matn**.

Remark:

- 1) **hids** is used for 2-d shell-elements.
- 2) Symmetry from **symm** is not taken into account.

## HIDS

---

**hids,n1**

---

**Hids** is used to plot shell elements in a 3-d space using a hidden line technique. Here the elements are sorted with respect to the distance from the viewpoint.

'n1'  $\neq 0$  enables and 'n1' = 0 disables the hidden line plot.

'n1 > 0 acts on the undeformed and 'n1' < 0 acts on the deformed mesh.

For a perspective view the view point is given within **pers**, whereas for **rot** the viewpoint coordinates are chosen within this macro.

**hids** acts on **hmsh**, **eigy**, **disp**, **velo**, **acce**, **stre** and similar macros.

Partial parts of the mesh are plotted using **matn**.

Remark:

- 1) **hide** is used for 3-d brick-elements.
- 2) Symmetry from **symm** is not taken into account.

## HMSH

---

**hmsh,<n1,n2,n3>**

---

Plot the undeformed and deformed mesh in hidden line technique. The plot is correct if the elements are plotted from back to front of the screen. This can be arranged by a special element numbering technique or by using the **rot**-macros.

If abs 'n1' is equal 2 elements are plotted from numel to 1, otherwise elements are plotted from 1 to numel.

Example: Define a mesh in the first quadrant in x,y,z-direction. Use the macros **rot1,-70** and **symm,3** and a complete hidden line mesh of the quadrants 1-4 can be seen.

The undeformed mesh is plotted for ' $n1 \geq 0$ ' (default), whereas the deformed mesh is plotted for ' $n1 < 0$ '.

The mesh is plotted in color 'n2' and 'n3'. 'n3' is the interior color (default values are 4(cyan) for the undeformed mesh and 5(green) for the deformed mesh. 'n2' is the color of the element border. The default value is 32 (black). Consult the 'colo' command for tables of available colors.

## **INIT**

---

**init,n1**

---

- n1 = 0: All plot values will be reset to its initial state.
- n1 = 1: Interactive calculation of contour or filled plots
- n1 = 2: Automatique calculation of contour or filled plots (def.)
  - Input of plot values
    - no of lines/colors (def.=14)
    - min value (def.=min)
    - max value (def.=max)

**INTS**

---

**ints,n1,n2,<n3>**

---

Plot the distribution of stress  $n_2$  at the center of element  $n_1$  over the thickness in a shell type element. For the output  $n_3$  intervals (default  $n_3=10$ ) are used for each layer.

stress	$n_2$
$\sigma_x$	1
$\sigma_y$	2
$\sigma_z$	3
$\tau_{xy}$	4
$\tau_{xz}$	5
$\tau_{yz}$	6
$\varphi_x$	7
$\varphi_y$	8

The stress numbers  $n_2$  are defined in the following table:

These stresses could be printed with the **stre,ints** command.

## ISEC

---

**isec,<n1,n2>**

---

Plot locations of all nodes on intersections. The following options are possible:

In case 'n1' is negative the nodes are plotted relative to the deformed mesh.

In case 'n2' is greater 0 node numbers are plotted too.

## **ISOM**

---

**isom**

---

Plot all views with an 'isometric' reference frame. The other options are a 'cartesian' reference frame (see **cart** plot command) and a 'perspective' view (see **pers** plot command).

N.B. This option does not always center correctly.

## JINT

---

**jint,<n1,n2,n3>**

---

Plot all nodal material forces currently calculated on the deformed mesh.

- Material forces are plotted according to the following input:

n1 = 0	material forces (1-ndm) (default)
n1 = i	material force i
n1 = ndf+1	material forces (1-ndm) (like n1=0)
n1 = ndf+2	material forces (4-ndf) = moments

- If 'n1' is negative material forces will be positioned relative to the deformed mesh.
- If 'n2' is nonzero the vector tip will appear next to the node whereas if 'n2' is zero the tail of the material force vectors are on the nodes.
- In addition the results can be scaled by the value of 'n3'.
- If 'n3' is negative all nodes with material forces are shown.

## LINE

---

**line,n1**

---

Set line type to 'n1' (must be between 1–8) If 'n1' is set negative the line type will increment the previous value by 1.

n1	Type
1	solid
2	short dash
3	dotted
4	dash-dot
5	long dash
6	double dot
7	dot-dot-dash
8	solid

The initial (default) line type is 1 – solid.

## **LINK**

---

**link,<n1,n2,n3>**

---

Plot locations of all linked nodes and linking conditions. Only nodes within the current plot region are displayed.

In case ' $n1 < 0$ ' only the nodes are plotted. Furthermore with ' $n2 > 0$ ' the node numbers are plotted too. Macro nodes are plotted in colour blue whereas slave nodes are plotted in colour green.

With ' $n1 > 0$ ', linked nodes and linking conditions for linking dof ' $n1$ ' are plotted. Repetition of calling this macro for all values of ' $n1$ ' show all linking conditions, where for each ' $n1$ ' a different colour is used.

Results may not be correct when nodes are affected more than once! In any case of doubt check the results using only one linking card when using the macro **link**.

With ' $n1 = 0$ ' all linked nodes and linking conditions are plotted in one colour.

With the value of ' $n3$ ' the linking conditions can be scaled. The default value is 1.

### **Remarks:**

1. Be extremely careful using the macro **link**, because input errors lead to completely wrong results!
2. When linking to fixed dofs/nodes occurs, these dofs/nodes are plotted for the macro **boun**.

## LOAD

---

**load**,

---

Plot all nodal loads (forces and moments) currently specified. The maximum length will be automatically scaled. All other loads will be scaled in proportion to the maximum, accordingly some very small values may scale to be too small to be visible.

Loads are plotted according to the following input:

```
n1 = 0      loads (1-ndm) (default)
n1 = i      load i
n1 = ndf+1  loads (1-ndm) (like n1=0)
n1 = ndf+2  loads (4-ndf) = moments
```

If 'n1' is negative loads will be positioned relative to the deformed mesh. If 'n2' is nonzero the vector tip will appear next to the node whereas if 'n2' is zero the tail of the load vectors are on the nodes.

In addition the results can be scaled by the value of 'n3'.

If 'n3' is negative all nodes with loads from **load** are shown.

Nodes with prescribed displacements are plotted using the **pdis** command.

## LOGO

---

**logo,n1**

---

The command redraws the **FEAP**-logo in color 'n1' Consult the **colo** command for tables of available colors. Note that the logo vanishes for the black color (32). In a similar way the border of the plot-region is redrawn by the **bord**-command.

## MAGN

---

**magn,<v1>**

---

With the macro **magn** the plot output can be magnified on the screen.

- 'v1' is the value of magnification of the mesh size. The default value for 'v1' is 0.05.
- 'v1' > 0 plots the undeformed mesh, 'v1' < 0 the deformed mesh.
- The process is controlled with the mouse buttons:
  - \* left button click : increase plot of mesh
  - \* right button click : decrease plot of mesh
  - \* both button click : finish the process
- The process starts with the actual values of the mesh size. The size can be set back to the initial value with the macro **init**.
- With the macro **move** the plot output can be moved on the screen.

## **MAN**

---

**man**

---

The complete documentation is available using the macro **man**.

## MATE

---

**mate,<n1,n2,n3>**

---

The undeformed or deformed mesh is plotted with respect to material properties. If 'n1' and/or 'n2' are omitted all element regions with different **material** properties are plotted. The program selects appropriate different colors for each set.

If 'n3' is nonzero the **mesh** is added in black.

If 'n1' is nonzero regions are filled belonging to properties set by 'n1'. A value of 'n1' less than zero plots results on the deformed mesh.

Each element region can be filled in the color given by 'n2'. Consult the **colo** command for tables of available colors.

## MATN

---

**matn**, $\langle n1,n2,n3 \rangle$

---

With the command **matn** one can define for which materials stress or contour plots should be performed. The macro can be used if there exists for example shell elements with the same coordinates.

'n1 – n3' defines the material numbers of plotted elements.

- $n1 = 0$  plot for all material numbers (default),
- $n1 < 0$  set all material numbers to zero,
- $n1 > 0$  plot for material numbers 'n1' to 'n2', inc='n3'.

The actual material numbers can be controlled by **show**.

**matn** acts on the macros **disp**, **velo**, **acce**, **dplo**, **eigv**, **eplo**, **evex**, **mate**, **hmsh**, **rpl0**, **splo**, **stre**, **elem**

### Remarks:

- For **stre,disp** etc.: Results and profile are plotted only for active materials. Other profile values could be chosen via **init,1**.
- For **stre**: No averaging is taken into account at material borders. Thus, the element code needs a special modification for isw=8.

## **MAXI**

---

**maxi,<n1>**

---

Plot nodes with minimal/maximal values of last filled plot (e.g. **disp**, **stre**,...)

In case 'n1' is negative the nodes are plotted relative to the deformed mesh.

## MESH

---

**mesh**, $\langle n1,n2,n3 \rangle$

---

Plot current mesh. Alternatively, in macro mode use of just **plot** will produce the undeformed plot.

If ' $n1$ ' is negative the plot is in the deformed configuration at the current value of **scale**.

For ' $n2$ ' zero the entire mesh is shown. If ' $n2$ ' is non-zero only material property set with the value of ' $n2$ ' is shown. On color terminals a different color will be used for each material type if ' $n2$ ' is input as the negative of the material number.

The mesh is plotted in color ' $n3$ '. Consult the **colo** command for tables of available colors. The default value is ' $n3$ ' = 4 (cyan).

## **MONO**

---

**mono,<n1>**

---

**mono,1** leads to a pure black and white screen with greyshading option for IBM,HP(screen and all plotfiles) and the Postscript-file under the macro 'prin'. For the VGA-screen there is no action.

**mono,0** set all colors to the standard values, see the **colo**-macro.

## MOVE

---

**move**,v1,<v2,<v3>

---

With the macro **move** the plot output can be moved on the screen.

- 'v1' is the number of the chosen axis [1,2].
- 'v2' is the associated screen movement [0-1]. The default value for 'v2' is 0.02.
- 'v1' > 0 plots the undeformed mesh, 'v1' < 0 the deformed mesh.
- 'v3' ≤ 0: The process is controlled with the mouse buttons:
  - \* left button click : move to left/down direction
  - \* right button click : move to right/up direction
  - \* both button click : finish the process

This part is not available in the INTEL-Version.

- 'v3' > 0: Only one step with the given values, this can be chosen e.g. in batch-mode.
- Typically the mesh is plotted. This can be suppressed via 'v3'=2.
- One macro **mesh** - defining the initial values - is necessary before the process is ready to start. The position can be set back to the initial position with the macro **init**.
- With the macro **magn** the plot output can be magnified on the screen.

## MOVI

---

### movi

---

This macro can be used to show the time dependency of a deformation process. For this purpose we have to calculate the complete process as a first step. Each deformed mesh has to be saved via the **writ**-macro (see the macro - manual) on the disk. In a second step we pop via the **movi** - macro all deformed meshes on the screen. The result is something which looks like a movie if you use it on a fast machine. The macro works only if the plot macros act on a separate window. Thus the following things have to be done.

1. Calculation of Problem (example)

```
dt,,1  
prop  
writ,test  
loop,,100  
time  
tang,,1  
writ,disp  
next  
writ,clos
```

2. Movie of Problem (example) (plot every fifth deformed mesh)

```
read,test  
loop,,20  
loop,,5  
read,disp  
next  
plot,movi  
next  
read,clos
```

The macro **movi** itself gives the same results as **defm**.

## **NDII**

---

**ndii,<n1>**

---

Plot locations of all nodes with problems within actual solution phase. The following options are possible:

n1	plot result
1	nodes with $D_{ii} < 0$ (default)
2	nodes with $D_{ii} = 0$
3	nodes with $D_{ii} << 1$

In case 'n1' is negative the nodes are plotted relative to the deformed mesh.

## **NODE**

---

**node,<n1,n2,n3>**

---

Plot locations of all nodes. Only nodes within the current plot region are displayed. Occasionally when a mesh is added no element edge will be plotted to each node. This is due to the fact that only elements which have all nodes visible are displayed.

In case 'n1' is negative the nodes are plotted relative to the deformed mesh.

In case 'n2' is greater than zero the node numbers are plotted too.

In case 'n2' is less than zero only the node with number n2 is plotted.

The nodes are plotted in color 'n3'. Consult the **colo** command for tables of available colors. The default value is 7 (yellow).

## **OUTL**

---

**outl**,n1,n2,n3

---

Plot an outline of all parts with material number 'n2'. For 'n2' zero plot outline of entire problem. (default: n2=0).

If 'n1' is negative the outline of the deformed material number 'n2' is plotted. On color terminals the outline for each material 'n2' may be given in different colors if 'n2' is input as the negative material set number. **outl** will be plotted in color 'n3' (default=1).

## **PDIS**

---

**pdis,<n1,n2,n3>**

---

Plot all prescribed nodal displacements currently in the mesh. The maximum length will be automatically scaled. All other displacements will be scaled in proportion to the maximum, accordingly some very small values may scale to be too small to be visible.

In addition the results can be scaled by the value of 'n3'.

If 'n1' is negative displacements will be positioned relative to the deformed mesh. If 'n2' is nonzero the vector tip will appear next to the node whereas if 'n2' is zero the tail of the displacement vectors are on the nodes.

If 'n3' is negative all nodes with prescribed displacements from **pdis** are shown.

Nodes with external loads are plotted using the **load** command.

## **PELE**

---

**pele,<n1>**

---

- Pick up element numbers via left mouse button click and show material number (center of the element should be used for pick up.)
- In case 'n1' is negative the elements are searched relative to the deformed mesh.
- Data for that element can be found in Macro-modus with e.g. **show,...** .
- Admissible screen macros for view are **move**, **rot**, **isom** and **pers**. **clip** may 'work'.
- Right mouse button click stops procedure.

## PERS

---

**pers**,<n1>

---

Plot all views with an 'perspective' view. The other options are a 'cartesian' and an 'isometric' reference frame. Furthermore a rotation around axes can be done. For details see the plot commands **cart**, **isom** and **rot**.

After entering the command **pers** the user is asked interactively to specify the following

Coordinates of view point (X,Y,Z),

Components of vertical vector (X,Y,Z).

After all parameters are specified the user can use the other plot commands to show mesh, stresses etc. However, the first command to be used should be **zoom** (without any parameters ) to reset zoom options and **scal** to initiate **pers** properly.

With 'n1' = 1 hidden line views are set directly for 3D-structures (thus, the macro **hide**,1 is added automatically). With 'n1' = 2 same action is set for shell-structures (**hids**,1).

## PLOF

---

**plof,n1,n2,n3**

---

This macro is used to make plot files under PHIGS (IBM), GKS (HP) or PCX (VGA) via separate programs.

Graphik	n1	plotfile	n2	type of file	n3
GKS	n1	<i>fplt-n1.eps</i>	1	Postscript	1=landscape(default)
					2=portrait
GKS	n1	<i>fplt-n1.cgm</i>	2	CGM(bin)	-
PHIGS	n1	<i>fplt-n1.cgm</i>	1	CGM(bin)	-
PHIGS	n1	<i>fplt-n1.gdf</i>	2	GDF / HPGL	-
VGA	n1	<i>fplt-n1.pcx</i>	1	PCX	-

- **FPLT** is the name of the actual plot file (only characters 1-6 used).
- After opening the plotfile all of the following plot macros acts on screen and the plot–file too. This is shown by the prompt 'Plof'.
- Up to 9 output files of each type are possible within a session.
- The plot–file can be closed via the macro **plof,(0)**.
- Use before **mono,1** to plot black on white!
- The CGM/HPGL/PS-files can be used directly or can be modified with certain programs, e.g. UNIRAS, CGMGKS, TEX(put!), PRESENTATION,...)

## **PNOD**

---

**pnod,<n1,n2,n3>**

---

- Pick up nodal points via left mouse button click and show node number.
- In case 'n1' is negative or **scal** or **defo** is active the nodes are searched relative to the deformed mesh.
- 'n2' = 1 print in addition coordinates, displacement and stresses at that node on screen. More data may be found in Macro-modus with e.g. **disp**, **reac**, **show**,... .
- 'n3' = 1 print data in addition to output file.
- Admissible screen macros for view are **move**, **rot**, **isom** and **pers**. **clip** may 'work'.
- Right mouse button click stops procedure.

## POLA

---

**pola,n1**

---

Displacements, velocities, accelerations and eigenvectors can be plotted in cartesian coordinates ( $n1 = 0$ , default) or in polar coordinates ( $n1 = ik = 12,13,23$ ).  $i$  and  $k$  describe the plane in which the transformation should be done. Afterwards the component  $i$  describes the radial e.g. displacement whereas the component  $k$  describes the e.g. tangential displacement.

The plot is performed in standard manner via e.g. the **disp** – macro.

## PRIN

---

**prin**,n1,n2,n3

---

This macro is used to make plot files.

n1	plotfile	n2	type of file	n3
n1	fplt_n1.eps	1	Postscript	1=portrait(default)
				2=landscape
n1	fplt_n1.pgl	2	HPGL	-

- Existing files are shown by **prin**, n1 with n1 > 99 (only for WIN).
- **FPLT** is the name of the actual plot file (Pname).
- Up to 99 output files of each type are possible within a session by different values of n1.

Postscript-file

$n1 > 0$  leads to a color plot

$n1 < 0$  leads to a black and white plot with grey shading

- After opening the plotfile all of the following plot macros acts on screen and the plot–file too. This is shown by the prompt **Prin**.
- The plot–file must be closed with the macro **prin,(0)**.
- The EPS/HPGL-files can be used directly or their source code can be modified with certain programs. Note that in the EPS- file all information is given for color and(!) grey shading.

## PRIS

**pris,n1,<n2,n3>,n4**

Plot contours of principal/equivalent stresses, where 'n1' is the component to be plotted with respect to nptyp. The same parameter definitions as in **stre** are used.

Plots are only correct if one element type is used!

**Stresses with respect to 'n1'**

'n1'	Stress for nptyp				
	1	2	3	4	5
1	$S_1$	$n_1$	$m_1$	$S_1$	$n_1$
2	$S_2$	$n_2$	$m_2$	$S_2$	$n_2$
3	$\alpha_s$	$\alpha_n$	$\alpha_m$	$S_3$	$\alpha_n$
4			$m_1$		$m_1$
5			$m_2$		$m_2$
6			$\alpha_m$		$\alpha_m$
7	$\sigma_{v2D}(S)$	$\sigma_{v2D}(n)$	$\sigma_{v2D}(m)$		$\sigma_{v2D}(n)$
8					$\sigma_{v2D}(m)$
9					$\sigma_{v2.5D}$
10				$\sigma_{v3D}$	

**Equivalent stresses**

$$\begin{aligned}
 \sigma_{v2D} &= \sqrt{\sigma_x^2 + \sigma_y^2 - \sigma_x\sigma_y + 3\tau_{xy}^2} \\
 \sigma_{v2.5D} &= \sqrt{\sigma_x^2 + \sigma_y^2 - \sigma_x\sigma_y + 3\tau_{xy}^2 + 3\tau_{xz}^2 + 3\tau_{yz}^2} \\
 \sigma_{v3D} &= \sqrt{\sigma_x^2 + \sigma_y^2 + \sigma_z^2 - \sigma_x\sigma_y - \sigma_x\sigma_z - \sigma_y\sigma_z + 3\tau_{xy}^2 + 3\tau_{xz}^2 + 3\tau_{yz}^2} \\
 \sigma_{v2D}(n) &= \sqrt{n_x^2 + n_y^2 - n_x\sigma_y + 3n_{xy}^2} \\
 \sigma_{v2D}(m) &= \sqrt{m_x^2 + m_y^2 - m_x\sigma_y + 3m_{xy}^2} \\
 \sigma_{v2.5D}(m) &= \sqrt{m_x^2 + m_y^2 - m_xm_y + 3m_{xy}^2 + 3q_x^2 + 3q_y^2}
 \end{aligned}$$

**Default position of Stresses in Elements**

nptyp	stress position							
	1	2	3	4	5	6	7	8
1	$S_x$	$S_{xy}$	$S_y$					
2	$n_x$	$n_{xy}$	$n_y$					
3	$m_x$	$m_{xy}$	$m_y$					
4	$S_x$	$S_y$	$S_z$	$S_{xy}$	$S_{xz}$	$S_{yz}$		
5	$n_x$	$n_{xy}$	$n_y$	$m_x$	$m_{xy}$	$m_y$	$q_x$	$q_y$

### **Technical Information for User-Elements**

Data have to be set on element level (isw=1) via include 'prisdat.h' with common /prisdat/nptyp,nprip(8).

- Titles and type has to be set via the parameter nptyp (default: nptyp=1).
- Stress positions have to be set via the array nprip default: (1,2,3,4,5,6,7,8).

## **PROF**

---

**prof,<n1>**

---

The command plots the upper profile of the set of equations of the discussed problem.  
With 'n1 < 0 ' the lower profile is plotted too.

## **QUIT**

---

**quit**

---

This macro terminates the plot modus. The same action is achieved with **q** or **end**.

## REAC

---

**reac,<n1,n2,n3>**

---

Plot all nodal reactions currently calculated on the deformed mesh. Thus in an equilibrium state the boundary forces and e.g. the contact forces are plotted. At intermediate states the residual forces can be seen.

- Reactions are plotted according to the following input:

n1 = 0 reactions (1-ndm) (default)  
n1 = i reaction i  
n1 = ndf+1 reactions (1-ndm) (like n1=0)  
n1 = ndf+2 reactions (4-ndf) = moments

- If 'n1' is negative reactions will be positioned relative to the deformed mesh.
- If 'n2' is nonzero the vector tip will appear next to the node whereas if 'n2' is zero the tail of the reaction vectors are on the nodes.
- In addition the results can be scaled by the value of 'n3'.
- If 'n3' is negative all nodes with reactions are shown.
- **reac** prints the following values:  
t=0: loads from mate + single loads  
after tang: reactions
- In case of a dynamic analysis dynamic forces are included.

## RMSH

---

**rmsh**<n1,n2,n3>

---

Plot current mesh refinement, if the macro 'reme' has been used.

If 'n1' is negative the plot is in the deformed configuration at the current value of **scal**.

For 'n2' zero the entire mesh is shown. If 'n2' is non-zero only material property set with the value of 'n2' is shown. On color terminals a different color will be used for each material type if 'n2' is input as the negative of the material number.

The mesh is plotted in color 'n3'. Consult the **colo** command for tables of available colors. The default value is 'n3' = 5 (green).

## **RESI**

---

**resi,n1,n2,n3,n4**

---

Plot contours for residuum of degree of freedom 'n1'. For negative values of 'n1' the residuum is plotted on the deformed mesh.

The macro leads to a filled plot for 'n2'= 0. On the other hand contour lines are plotted for 'n2'> 0.

### **Filled/Contour plots**

FEAP calculates 14 contour lines automatically between the extremal values. Values can be changed by **init,2**.

In interactive mode (by **init,1**) FEAP presents a profile of results with maximum and minimum values. Input is the number of used colors and the maximum and minimum value. Up to 14 (default) colors are possible. As default FEAP calculates the contour lines automatically between the extremal values, otherwise between the given input values.

For contour line plots, a non-zero 'n3' value will suppress numbers near each contour line. With 'n4' the mesh is plotted in color 'n4' additionally.

For a further description see the macro **stre**.

**ROT<i>**

---

**rot0,,n2   rot1,n1,n2   rot2,n1,n2   roti,n1,n2**

---

Besides the macros **isom** and **pers** it is possible to make a 3-D-view of a system with the **rot** macros. Thus, rotations about the 1,2,3-axis can be performed by the macros **rot1,rot2,rot3**.

'n1' is the rotation angle about the chosen axis in degree. The positive rotation vector shows in the direction of the coordinate axis.

A special 3-D view of a system can be performed by a number of **rot** commands. Thus, a sequence **rot2,90 , rot2,-40 , rot2,55** leads to a rotation angle 105 degree about axis 2.

By the macro **rot0** all rotation angles are set to zero. Thus we return to a cartesian projection.

The values of all angles can be seen by typing the macro **show**.

The rotated axis are plotted on screen for 'n2' > 0.

A combination with the **isom** and the **pers** macro may work but is not reliable. Thus before use these macros return to the cartesian view by the macro **rot0**.

## **ROTM**

---

**rotm,<v1,v2>**

---

With this macro 3-D-views of a mesh or a deformed mesh can be calculated for each mouse click.

- 'v1' is the number of the chosen axis.
- 'v1' > 0 plots the undeformed mesh, 'v1' < 0 the deformed mesh.
- 'v2' is the rotation angle about the chosen axis in degree. (def. value = 5 deg.) The positive rotation vector shows in the direction of the coordinate axis.
- The process is controlled with the mouse buttons:
  - \* left button click : rotation in positive direction
  - \* right button click : rotation in negative direction
  - \* both button click : finish the process
- The process starts with the actual values of the rotation angles. They can be set back to zero with the macro **rot0**.

**RPLO****rplo,n1,<n2,n3>**

---

- The **rplo** command may be used to construct plots of reactions along any specified line in the mesh for 2D-problems. 3D-problems can be calculated too if **pers** and **hide** is used. The line is than on the 3d-element faces.
- The line is scaled to 0-1 for the intersected part of mesh, not between the line ends.
- The line is scaled to 0-1 for the intersected part of mesh, not between the line ends.
- To define the ends of the line two options are possible:

# Input of cartesian coordinates via the Macro **splo, set** in **Macro-modus**.

# If the first option is not active, the screen coordinates of the ends of the line are defined via the mouse cursor.

For INTEL: Mark the the first point(1) with a left button click of the mouse. Do the same for the second point(2).

For FTN: Mark the the first point(1) by pressing the left button. Hold the button down until the mouse cursor is located at the position of the second point(2). Here, release the left button.  
If this option does not work set coordinates to zero via **splo** in **Macro-modus**.

- A 3-d formulation is possible, but note that now hidden line technic is used. Thus all visible intersections are calculated.
- The intersection bases only on the 4 corner nodes, thus 8/9 nodes are possible but the additional nodes are not used.
- Admissible screen macros for view are **move**, **rot**, **isom** and **pers**. **clip** may 'work'.
- The 'n1' parameter is used to specify the reaction component to be plotted. A negative value of 'n1' acts on the deformed mesh.
- The 'n2' parameter is used to specify the scaling process.

n2 = 0 plot new value in diagram

n2 = 1 plot other value in same diagram with rescaling

n2 = 2 plot other value in same diagram without rescaling

- If 'n2' is < 0 the coordinates are not plotted but printed to screen and output-file.
- The 'n3' parameter is not used.

## **RSUM**

---

**rsum,<n1,n2,n3>**

---

Plot locations of all nodes for sum of reaction forces. Only nodes within the current plot region are displayed.

In case 'n1' is negative the nodes are plotted relative to the deformed mesh.

In case 'n2' is greater than zero the node numbers are plotted too.

In case 'n2' is less than zero only the node with number n2 is plotted.

The nodes are plotted in color 'n3'. Consult the **colo** command for tables of available colors. The default value is 2 (red).

## **SCAL**

---

**scal,v1**

---

Displacements are scaled by value 'v1' (generally this operates only on elements where first two dof are in the  $x_1$  and  $x_2$  directions). The default value of 'v1' is 1.0 and need not be specified. After the displacement components are scaled by 'v1' the plot region is resized to permit both the undeformed and the deformed plot to appear on the screen.

There is no way to ensure that subsequent loading steps will appear in the window. Any element which is not in the plot window will not appear. For problems which undergo large motions it is possible to force the plot region to be well defined by using additional "dummy" nodes to define the expected region. These "dummy" nodes have specified coordinates but are not connected to any element. It is desirable to also restrain all degree-of-freedoms for the "dummy" nodes using appropriate boundary restraints.

**SECT****sect,n1,<n2,n3,n4>**

---

This macro is used to plot all quantities for cross sections.

n3 describes the number of cross section (default=1). (not valid for sect,12 and 13)

n1	n2	n3	n4	action
1		<n3>		mesh on line element
2	n2	<n3>		nodes on line element (n2>0 with numbers)
3		<n3>		element numbers on line element
4		<n3>		center of gravity
5		<n3>		center of shear
6		<n3>		reference point
7	n2	<n3>		plot function w on line element; scaling factor n2 (default = 1)
8		<n3>		mesh on area element
9	n2	<n3>		nodes on area element (n2>0 with numbers)
10		<n3>		element numbers on area element
11	n2	<n3>		plot function w on area element, (n2>0 lines, n2<0 fill)
12	n2	n3	n4	plot stress n2 at element n3 and gauss point n4
13	n2	n3	n4	plot fluxes at element and gauss point n4, scale = n4

## **SHOW**

---

**show**

---

This command shows the user the currently chosen parameters for the plot output.

## SIZE

---

**size,n1**

---

Specify the size of text,numbers and labels to be plotted.

n1	Text size
1	small (default)
2	medium
3	large
4	super large

## **SLEEP**

---

**sleep,n1**

---

With this macro command the program waits for n1 seconds. This can be used within demos or presentations.

## SPLO

---

**splo,n1,<n2,n3>**

---

- The **splo** command may be used to construct plots of stresses along any specified line in the mesh for 2D-problems. 3D-problems can be calculated too if **pers** and **hide** is used. The line is than on the 3d-element faces.
- The line is scaled to 0-1 for the intersected part of mesh, not between the line ends.
- The line is scaled to 0-1 for the intersected part of mesh, not between the line ends.
- To define the ends of the line two options are possible:

# Input of cartesian coordinates via the Macro **splo, set** in **Macro-modus**.

# If the first option is not active, the screen coordinates of the ends of the line are defined via the mouse cursor.

For INTEL: Mark the the first point(1) with a left button click of the mouse. Do the same for the second point(2).

For FTN: Mark the the first point(1) by pressing the left button. Hold the button down until the mouse cursor is located at the position of the second point(2). Here, release the left button.

If this option does not work set coordinates to zero via **splo** in **Macro-modus**.

- A 3-d formulation is possible, but note that now hidden line technic is used. Thus all visible intersections are calculated.
- The intersection bases only on the 4 corner nodes, thus 8/9 nodes are possible but the additional nodes are not used.
- Admissible screen macros for view are **move**, **rot**, **isom** and **pers**. **clip** may 'work'.
- The 'n1' parameter is used to specify the stress component to be plotted. A negative value of 'n1' acts on the deformed mesh.
- The 'n2' parameter is used to specify the scaling process.

n2 = 0 plot new value in diagram

n2 = 1 plot other value in same diagram with rescaling

n2 = 2 plot other value in same diagram without rescaling

- If 'n2' is < 0 the coordinates are not plotted but printed to screen and output-file.
- The 'n3' parameter is used to define the layer number for layered elements. With negative values '-n3' the stress 'n1' for layer 'n3' is plotted.

## STRE

---

**stre,n1,n2,v3,n4**

---

Plot contours of stresses, where 'n1' is the component to be plotted. For negative values of 'n1' the stresses are plotted on the deformed mesh.

The macro leads to a filled plot for 'n2'= 0. On the other hand contour lines are plotted for 'n2' > 0.

### Filled/Contour plots

FEAP calculates 14 contour lines automatically between the extremal values. Values can be changed by **init,2**.

In interactive mode (by **init,1**) FEAP presents a profile of results with maximum and minimum values. Input is the number of used colors and the maximum and minimum value. Up to 14 (default) colors are possible. As default FEAP calculates the contour lines automatically between the extremal values, otherwise between the given input values.

### Layered beam/shell elements and layered solid/solid-shell elements with one element in thickness direction

For layered elements, e.g. composite elements or elements with plastic material behavior, it is possible to specify the layer and the position in the layer for which stresses are plotted by a negative value 'v3', the result is a filled plot. E.g. 'v3=-3.2' plots stresses in layer 3 at position 2. For the definition of layer and position see the associated element manual or formulation.

Solids: This makes **no sense** for more than one element in thickness direction!!

For contour line plots, a non-zero 'v3' value will suppress numbers near each contour line.

With 'n4' the mesh is plotted in color 'n4' additionally.

### Principal stresses

The principal stresses can be plotted as filled or contour plots using the command **pris**.

The principal stresses can be plotted by 'load'-vectors. The associated command is **stre,27,<n2,v3,v4>**. 'n2=2'(default) plots for 2D-elements, 'n2=3' plots for 3D-elements

With 'v3' the length of the vectors can be multiplied by a factor.

With 'v4' the length of the tip can be multiplied by a factor (only 2D).

Stresses must be stored in elements (see manual)

2D:  $S_{xx}$ ,  $S_{xy}$ ,  $S_{yy}$

3D:  $S_{xx}$ ,  $S_{yy}$ ,  $S_{zz}$ ,  $S_{xy}$ ,  $S_{xz}$ ,  $S_{yz}$

A similar command is **flux,2**.

Which stresses are plotted depends on the used element. Originally **FEAP** plot the following stresses.

n1	Component (istv>0)	Component (istv<0)
1	11-stress	User generated values
2	12-stress	”
3	22-stress	”
4		”
5	1-principal stress	”
6	2-principal stress	”
7	$\alpha$	”
8-25	User generated values	”
16-25	if <b>3D-Material library</b> is used: Internal variables	if <b>3D-Material library</b> is used: Internal variables
26	actual component of Principle stress	actual component of Principle stress
27	Principle stresses as vectors	Principle stresses as vectors

Details which stresses are plotted are specified in the element descriptions .

## STR1

---

**str1,n1**

---

Plot elementwise constant stresses on undeformed mesh calculated from element center, where 'n1' is the component to be plotted. Thus, jumps occur at each element border. Which stresses are plotted depends on the used element. Stresses must be calculated in element for isw=14.

Currently **str1** is implemented in **elmt05**, **elmt06**, **elmt07** and **elmt09**.

### Filled/Contour plots

FEAP calculates 14 contour lines automatically between the extremal values. Values can be changed by **init,2**.

In interactive mode (by **init,1**) FEAP presents a profile of results with maximum and minimum values. Input is the number of used colors and the maximum and minimum value. Up to 14 (default) colors are possible. As default FEAP calculates the contour lines automatically between the extremal values, otherwise between the given input values.

For contour line plots, a non-zero 'n3' value will suppress numbers near each contour line.

With 'n4' the mesh is plotted in color 'n4' additionally.

For a further description see the macro **stre**.

## SYMM

---

**symm**,n1,n2

---

It is possible to plot all quantities in additional quadrants in case of symmetric problems with the **symm** macro.

The following symmetry conditions are implemented:

n1	symmetry action
0	reset to first quadrant (default)
1	symmetry with respect to x-axis
2	symmetry with respect to y-axis
3	symmetry with respect to x + y-axis
4	symmetry with respect to z in xy-plane (bases on 1.Quadr.)
5	symmetry with respect to z in xy-plane (bases on 1.-4.Quadr.)

For  $n2 \neq 0$  the window will not be rescaled and cleared by this macro.

## TEXT

---

**text**,<n1,n2,v3>

---

Enter text to be placed in plot region. Type the desired text in the text window. To position the text move the mouse cursor in the view window at the desired location. The cursor is represented by an arrow (VGA) or cross-hairs (IBM). Press the left button to print the given text.

Use the 'n1' parameter to specify the color of each text item or use the **colo** command. The default value is 1 (white).

The 'n2' parameter is not used.

The text size can be changed using the macro **size**.

With values '*n3*' > 0 it is possible to write 15 textlines in the right window directly without mouse. Here '*n3*' is the desired line number. Between 1 and 15 lines are possible.

For the IBM-Version the following length of text is available

size	number of characters
1	21
2	14
3	10
4	8

## TIE

---

**tie** ,*<n1,n2,n3>*

---

Plot locations of all tied nodes. Only nodes within the current plot region are displayed. Occasionally when a mesh is added no element edge will be plotted to each node. This is due to the fact that only elements which have all nodes visible are displayed.

In case 'n1' is negative the nodes are plotted relative to the deformed mesh.

In case 'n2' is greater than zero the node numbers are plotted too.

In case 'n2' is less than zero only the node with number n2 is plotted.

The nodes are plotted in color 'n3'. Consult the **colo** command for tables of available colors. The default value is 2 (red).

## TPLO

---

**tplo,n1**

---

The **tplo** macro is used to plot load deflection curves from plot mode. The initialization has to be done with the **macro** command **tplo**. Several options are available.

The parameter **n1** may be defined as follows

<b>1</b>	= <b>load</b>	plot load vs displacement
<b>2</b>	= <b>disp</b>	plot displacement vs time
<b>3</b>	= <b>phas</b>	plot velocity vs displacement (phase diagram)
<b>4</b>	= <b>reac</b>	plot reaction vs displacement 1)
<b>5</b>	= <b>detd</b>	plot determinant vs displacement 2)
<b>6</b>	= <b>velo</b>	plot velocity vs time
<b>7</b>	= <b>acce</b>	plot acceleration vs time
<b>8</b>	= <b>loat</b>	plot load vs time
<b>9</b>	= <b>reat</b>	plot reaction vs time 1)
<b>10</b>	= <b>dett</b>	plot determinant vs time 2)
<b>11</b>	= <b>tdis</b>	plot time vs displacement 4)
<b>12</b>	= <b>strd</b>	plot stress i at node k vs displacement 3)
<b>13</b>	= <b>strt</b>	plot stress i at node k vs time 3)
<b>14</b>	= <b>us1d</b>	plot valuse1 vs displacement 4)
<b>15</b>	= <b>us2d</b>	plot valuse2 vs displacement 4)
<b>16</b>	= <b>us1t</b>	plot valuse1 vs time 4)
<b>17</b>	= <b>us2t</b>	plot valuse2 vs time 4)

1. The nodal reactions have to be calculated within each time step via **reac**.
2. Only within arc-length method, extended system, or using SM-solver or if calculated by the macro **detk**.
3. stress i at node k has to be defined with the macro **tplo,sset**. Furthermore stress has to be calculated via **stre,node** for each time step.
4. Values have to be provided by user. More informations are given in the chapter [Adding elements](#).

Further options are available using the **macro** command **tplo**.

## **TRAJ**

---

**traj,<n1,n2,n3>**

---

This macro plots the trajectories of main stresses by crosses in the nodes.

If 'n1' is not equal zero, abs('n1') is the number of main stress, i.e. 1,2,(3)

If 'n1' is negative the plot acts on the deformed mesh.

'n2' multiplies the length of the lines by a factor.

'n3=2'(default) plots for 2D-elements, 'n3=3' plots for 3D-elements

Stresses must be stored in elements (see manual)

2D:  $S_{xx}$ ,  $S_{xy}$ ,  $S_{yy}$

3D:  $S_{xx}$ ,  $S_{yy}$ ,  $S_{zz}$ ,  $S_{xy}$ ,  $S_{xz}$ ,  $S_{yz}$

## **UEIG**

---

**ueig**<n1,n2,n3>

---

This macro is up to now not documented.

## VELO

---

**velo,<v1,v2,v3,v4>**

---

Plot contours for degree of freedom (i.g. velocity) 'n1'. For negative values of 'n1' the velocity are plotted on the deformed mesh.

The macro leads to a filled plot for 'n2'= 0. On the other hand contour lines are plotted for 'n2'> 0.

### Filled plots

In interactive mode **FEAP** presents a profile of results with maximum and minimum values. Then **FEAP** asks for the number of used colors and the maximum and minimum value. Up to 14 colors are possible. The default value for the number of colors is 14. As default **FEAP** calculates the contour lines automatically between the extremal values, otherwise between the given input values.

### Contour plots

In interactive mode **FEAP** presents a profile of results with maximum and minimum values. Then **FEAP** asks for the number of used contours and the maximum and minimum value. Up to 14 contours are possible. The default value for the number of contours is 14. As default **FEAP** calculates the contour lines automatically between the extremal values, otherwise between the given input values.

For contour line plots, a non-zero 'n3' value will suppress numbers near each contour line.

With 'n4' the mesh is plotted in color 'n4' additionally.

Compare the handling of this macro with **stre**.

In batch mode, the requisite data described above must appear after the **end** macro command in the order that the program needs inputs.

## WIPE

---

### wipe,n1

---

**wipe** clears the screen. With 'n1' it is possible to decide which part of the screen is cleared. The following values are possible for 'n1'

n1	action
0	entire screen
1	clear part of screen between points 1 and 2
2	clear part of screen outside points 1 and 2

The default value is n1 = 0.

#### Remark:

The input of points 1 and 2 is done via the mouse cursor. For 'n1' =1,2 the screen is not really cleared, but filled with the background color (usually black=32).

## **XSCA(L)**

---

**xsc*a(l)*,v1,v2,v3**

---

The system (including deformations) can be scaled independently in the directions  $x_1$ ,  $x_2$  (and  $x_3$ ). The default values are 1.0 for all directions.

After the system is scaled by 'v1, v2, v3' the plot region is resized to permit both the undeformed and the deformed plot to appear on the screen by the macro **scal**.

There is no way to ensure that subsequent loading steps will appear in the window. Any element which is not in the plot window will not appear. For problems which undergo large motions it is possible to force the plot region to be well defined by using additional "dummy" nodes to define the expected region. These "dummy" nodes have specified coordinates but are not connected to any element. It is desirable to also restrain all degree-of-freedoms for the "dummy" nodes using appropriate boundary restraints.

## ZOOM

---

**zoom**,n1,n2,n3

---

The **zoom** feature permits the user to plot part of mesh and or results.

If the node numbers are known then the region to be plotted may be identified by specifying:

n1	— one node defining region to zoom on.
n2	— other node for zoom.
n1 = n2 = 0	— gives entire mesh.

If coordinates are known then the region to be plotted may be identified by specifying: **zoom**,,,1

and add the following values:  $(x, y, z)_1$ ,  $(x, y, z)_2$ .

The **zoom** without numbers is required to return to the entire mesh as the plot region. This option is useful to rescale the plot to scale = 1.0 after it has been scaled to 'v1' by the **scal** macro.

## Chapter 5

# 3D-Material library

- Theory

For some elements, e.g. **ELMT21** exist a library of different 3D-material models. Currently the following models are implemented.

Material	Description
1	linear elastic isotropic
2	linear elastic orthotropic
3	linear elastic transversal isotropic
4	small elasto-(visco-)plastic strains
5	finite elasto-plastic strains $F = F_e F_p$
6	finite elastic strains, Ogden
7	Method of Cells (MOC), Aboudi
8	FE <sup>2</sup>
9	small strain isotropic damage
10	concrete model, Schütt
11	small strain visco-elastic
12	linear piezoelectric
13	dielectric elastomers
14	finite elastic strains, Blatz-Ko
15	transversal isotropic with damage

Details on the derivation of these material models can be found in the [Theory Manual](#).

Details on the implementation in elements can be found in the [Manual for adding elements](#).

- Material input data/ output plot

**Material 1: linear elastic isotropic, (Theory)**

Record 1	linear elastic isotropic	
$E$	$[F/L^2]$	Young's modulus
$\nu$	$[ - ]$	Poisson's ratio

**Material 2: linear elastic orthotropic, (Theory)**

Record 1	linear elastic orthotropic	
$E_{11}$	$[F/L^2]$	Young's modulus for direction 1
$E_{22}$	$[F/L^2]$	Young's modulus for direction 2
$E_{33}$	$[F/L^2]$	Young's modulus for direction 3
$\nu_{12}$	$[ - ]$	Poisson's ratio
$\nu_{13}$	$[ - ]$	Poisson's ratio
$\nu_{23}$	$[ - ]$	Poisson's ratio
$G_{12}$	$[F/L^2]$	Shear modulus
$G_{13}$	$[F/L^2]$	Shear modulus
$G_{23}$	$[F/L^2]$	Shear modulus
$\varphi$	$[^\circ]$	Rotation angle in 1–2 plane, a layered element resets this value for each layer.

**Material 3: linear elastic transversal isotropic, (Theory)**

Record 1	linear elastic transversal isotropic	
$E_1$	$[F/L^2]$	Young's modulus for direction 1
$E_2$	$[F/L^2]$	Young's modulus for transverse directions 2/3 ( $E_3 = E_2$ )
$\nu_{12}$	$[ - ]$	Poisson's ratio ( $\nu_{13} = \nu_{12}$ )
$G_{12}$	$[F/L^2]$	Shear modulus ( $G_{13} = G_{12}$ )
$G_{23}$	$[F/L^2]$	Shear modulus ( $\nu_{23} = E_2/(2G_{23}) - 1$ )
$\varphi$	$[^\circ]$	Rotation angle in 1–2 plane, a layered element resets this value for each layer.

**Material 4: elasto-(visco-)plastic isotropic small strains, (Theory)**

Record 1	elasto-(visco-)plastic isotropic small strains	
$E$	$[F/L^2]$	Young's modulus
$\nu$	$[ - ]$	Poisson's ratio ( $\nu_{13} = \nu_{12}$ )
$Y_0$	$[F/L^2]$	initial yield stress
$Y_\infty$	$[F/L^2]$	yield stress at $t = \infty$
$xk$	$[F/L^2]$	linear hardening modulus
$xd$	$[F/L^2]$	exponential hardening modulus
$\eta$	$[FT/L^2]$	viscosity (only with linear hardening, $xd = 0$ !)

yield condition:  $f = \sqrt{3/2 \mathbf{S}_D : \mathbf{S}_D} - Y_0 + xk \cdot a + (Y_\infty - Y_0) \cdot (1 - \exp(-xd \cdot a))$

with  $S_D$ : deviatoric stresses and  $a$ : equivalent plastic strains

stre,i	Variable	Description
16	$\varepsilon_v = a$ [-]	equivalent plastic strain
17	$  \varepsilon  $ [-]	norm of strains

**Material 5: elasto-plastic isotropic finite strains, (Theory)**

Record 1		elasto-plastic isotropic finite strains
$E$	[ $F/L^2$ ]	Young's modulus
$\nu$	[-]	Poisson's ratio ( $\nu_{13} = \nu_{12}$ )
$Y_0$	[ $F/L^2$ ]	initial yield stress
$Y_\infty$	[ $F/L^2$ ]	yield stress at $t = \infty$
$xk$	[ $F/L^2$ ]	linear hardening modulus
$xd$	[ $F/L^2$ ]	exponential hardening modulus

yield condition:  $f = \sqrt{3/2 \mathbf{S}_D : \mathbf{S}_D} - Y_0 + xk \cdot a + (Y_\infty - Y_0) \cdot (1 - \exp(-xd \cdot a))$

with  $S_D$ : deviatoric stresses and  $a$ : equivalent plastic strains

stre,i	Variable	Description
16	$\varepsilon_v = a$ [-]	equivalent plastic strain

**Material 6: finite strain elastic, Ogden, (Theory)**

Record 1		finite strain elastic, Ogden
$\mu_1$	[ $F/L^2$ ]	Shear modulus
$\mu_2$	[ $F/L^2$ ]	Shear modulus
$\mu_3$	[ $F/L^2$ ]	Shear modulus
$\alpha_1$	[-]	
$\alpha_2$	[-]	
$\alpha_3$	[-]	
$\Lambda$	[ $F/L^2$ ]	Lamé constant

$$\sum_{i=1}^3 (\mu_i \cdot \alpha_i) = 2\mu$$

**Material 7: Method of Cells (MOC), Aboudi, (Theory)**

Record 1	Method of Cells (MOC), Aboudi	
$E_{11}$	$[F/L^2]$	Young's modulus fiber direction 1
$E_{22}$	$[F/L^2]$	Young's modulus fiber direction 2
$E_{33}$	$[F/L^2]$	Young's modulus fiber direction 3
$\nu_{12}$	$[-]$	Poisson's ratio fiber
$\nu_{13}$	$[-]$	Poisson's ratio fiber
$\nu_{23}$	$[-]$	Poisson's ratio fiber
$G_{12}$	$[F/L^2]$	Shear modulus fiber
$G_{13}$	$[F/L^2]$	Shear modulus fiber
$G_{23}$	$[F/L^2]$	Shear modulus fiber
$\alpha_{t11}$	$[1/K]$	coefficient of thermal expansion fiber direction 1
$\alpha_{t22}$	$[1/K]$	coefficient of thermal expansion fiber direction 2
$\alpha_{t33}$	$[1/K]$	coefficient of thermal expansion fiber direction 3
$E$	$[F/L^2]$	Young's modulus matrix
$\nu$	$[-]$	Poisson's ratio matrix
$G$	$[F/L^2]$	Shear modulus matrix
$\alpha_t$	$[1/K]$	coefficient of thermal expansion matrix
Record 2	Method of Cells (MOC), Aboudi	
$C_{11}$	$[F/L]$	spring stiffness direction 1 (Penalty parameter)
$C_{22}$	$[F/L]$	spring stiffness direction 2
$C_{33}$	$[F/L]$	spring stiffness direction 3
$\Delta T$	$[K]$	Temperature change
$V_F$	$[-]$	fiber volume fraction
$Y_{tF}$	$[F/L^2]$	tensile strength fiber
$Y_{cF}$	$[F/L^2]$	compressive strength fiber
$Y_{tM}$	$[F/L^2]$	tensile strength matrix
$Y_{cM}$	$[F/L^2]$	compressive strength matrix
$Y_{sM}$	$[F/L^2]$	shear strength matrix
$n_C$	$[-]$	number of cells, at present only 4
$\varphi$	$[\circ]$	Rotation angle in 1–2 plane, a layered element resets this value for each layer.

**Material 8: FE<sup>2</sup>, (Theory)**

Record 1	FE <sup>2</sup>
fmicro	Path and name of used RVE (micro problem) (name without extensions .i)
Record 2	FE <sup>2</sup>
fbatch	Path and name of batch file to initiate data
Record 3	FE <sup>2</sup>
irtyp	0 = restart files for each GP and element of micro-RVE 1 = one restart file for micro-RVE
irecl	Length of restart file for each GP of micro-RVE in BYTES (only irtyp=1)

Up to 10 different RVE's could be used. For that it is necessary to use material numbers 1-10 in **MATE!**

irecl can be calculated using one **TANG** of the macro-problem. An error will occur, but the length of the restart file could be seen in the output file (irecl) and e.g. the Windows Explorer. FEAP, INTEL- and SALFORD-Compiler produce slightly **different** file lengths. The maximum value should be used.

Alternatively the MICRO-Problem could be started with one **TANG** and **EXIT**.

**Material 9: small strain isotropic damage, (Theory)**

Record 1	small strain isotropic damage	
$E$	[ $F/L^2$ ]	Young's modulus
$\nu$	[ $-$ ]	Poisson's ratio
$\sigma_u$	[ $F/L^2$ ]	ultimate stress
$G_f$	[ $F/L$ ]	fracture energy
implex	[ $-$ ]	0=implizit, 1=implizit-explizit
$\eta$	[ $FT/L^2$ ]	viscosity
$\alpha = 1$	[ $-$ ]	backward Euler method
<b>stre,i</b>	Variable	Description
16	$r$	[ $-$ ]
17	$d$	[ $-$ ]
18	$  \boldsymbol{\varepsilon}  $	[ $-$ ]

For 2D-problems modify characteristic length  $\ell_c$  in subroutine Mate3d09.for.

**Material 10: concrete model, (Theory)**

Record 1	concrete model	
$E$	$[F/L^2]$	Young's modulus
$\nu$	$[-]$	Poisson's ratio
ipla	$[-]$	plasticity on=1 /off=0
$f_{ctm}$	$[F/L^2]$	tensile strength
$f_{cm}$	$[F/L^2]$	compressive strength
$G_f$	$[F/L]$	energy release during cracking
$G_c$	$[F/L]$	energy release rate
$\gamma_1$	$[-]$	fitting parameter
$\gamma_2$	$[-]$	fitting parameter
<b>stre,i</b>	Variable	Description
16	$\alpha_1$	$[-]$ equivalent plastic strain 1
17	$\alpha_2$	$[-]$ equivalent plastic strain 2

**Material 11: small strain visco-elastic, (Theory)**

Record 1	small strain visco-elastic	
$E$	$[F/L^2]$	Young's modulus
$\nu$	$[-]$	Poisson's ratio
$\alpha$	$[1/K]$	Thermal expansion coefficient
nv	-	Number of viscoelastic terms
nm	-	0=deviatoric strains, 1=total strains
nv Records	small strain visco-elastic	
$\mu_i$	-	Viscoelastic shear parameter
$\tau_i$	$[T]$	Viscoelastic relaxation time

$$\mu_0 = 1 - \sum_{i=1}^{nv} \mu_i \geq 1 \text{ with } 0 \leq \mu_i \leq 1$$

<b>stre,i</b>	Variable	Description
16	$\varepsilon_{v1}$	viscoelastic strain 1
17	$\varepsilon_{v2}$	viscoelastic strain 2
18	$\varepsilon_{v3}$	viscoelastic strain 3
19	$\varepsilon_{v4}$	viscoelastic strain 4
20	$\varepsilon_{v5}$	viscoelastic strain 5
21	$\varepsilon_{v6}$	viscoelastic strain 6
22	$\varepsilon_{v7}$	viscoelastic strain 7
23	$\varepsilon_{v8}$	viscoelastic strain 8
24	$\varepsilon_{v9}$	viscoelastic strain 9
25	$\varepsilon_{v10}$	viscoelastic strain 10

**Material 12: linear piezoelectric, (Theory)**

Record 1	linear piezoelectric
$E$ [ $F/L^2$ ]	Young's modulus
$\nu$ [-]	Poisson's ratio
$\epsilon_{13}$ [ $Q/L^2$ ]	piezoelectric coupling modulus
$\epsilon_{33}$ [ $Q/L^2$ ]	piezoelectric coupling modulus
$\epsilon_{15}$ [ $Q/L^2$ ]	piezoelectric coupling modulus
$\varepsilon$ [ $Q^2/FL^2$ ]	dielectric constant

**Material 13: dielectric elastomers, (Theory)**

**Material 14: finite elastic strains, Blatz-Ko, (Theory)**

Record 1	Blatz-Ko
$\mu$ [ $F/L^2$ ]	Shear modulus
$\nu$ [-]	Poisson's ratio ( $0 \leq \nu < 0.5$ )
$f$ [-]	Interpolation parameter ( $0 \leq f \leq 1$ )

$f = 1$ : compressible NEO-HOOKE,  $f < 1$ : compressible MONEY-RIVLIN

**Material 15: Transversal isotropic with damage , (Theory)**

Record 1	transversal isotropic with damage
$E_{11}$ [ $F/L^2$ ]	Young's modulus for direction 1
$E_{22}$ [ $F/L^2$ ]	Young's modulus for transverse directions 2/3 ( $E_3 = E_2$ )
$\nu_{12}$ [-]	Poisson's ratio ( $\nu_{13} = \nu_{12}$ )
$G_{12}$ [ $F/L^2$ ]	Shear modulus ( $G_{13} = G_{12}$ )
$G_{23}$ [ $F/L^2$ ]	Shear modulus ( $\nu_{23} = E_2/(2G_{23}) - 1$ )
Record 2 Pos 1-6	transversal isotropic with damage
iftyp [-]	<b>Failure criterion</b> 1 = Tsai-Wu      2 = Hashin (extended) 3 = Puck            4 = Cuntze
$R_{11}^t$ [ $F/L^2$ ]	Tensile strength fiber direction
$R_{11}^c$ [ $F/L^2$ ]	Compressive strength fiber direction
$R_{22}^t$ [ $F/L^2$ ]	Tensile strength matrix direction
$R_{22}^c$ [ $F/L^2$ ]	Compressive strength matrix direction
$R_{12}$ [ $F/L^2$ ]	Shear strength direction 12

with respect to failure criterion:

Record 2 Pos 7-9	iftyp=1 (Tsai-Wu), 2 (Hashin)
$R_{13}$ [ $F/L^2$ ]	Shear strength direction 13
$R_{23}$ [ $F/L^2$ ]	Shear strength direction 23
Record 2 Pos 7-9	iftyp=3 (Puck)
$p_{\parallel\perp}^+$ [-]	Curve fitting parameter: 0.30-0.35
$p_{\parallel\perp}^-$ [-]	Curve fitting parameter: 0.25-0.30

Record 2	Pos 7-9	iftyp=4 (Cuntze)
$b_{\parallel\perp}$	[ $\cdot$ ]	Curve fitting parameter: 0.05-0.15
$b_{\perp}^{\tau}$	[ $\cdot$ ]	Curve fitting parameter: 1.0-1.6
$m$	[ $\cdot$ ]	Interaction parameter: 2.5-3.5
Record 3	Pos1	Degradation model typ
idtyp	[ $\cdot$ ]	=1: Constant(CRC-ACS model): iftyp=1 Tsai-Wu =2: Constant(Chang & Lessard model): iftyp=2 Hashin =3: Constant: iftyp=3 Puck and iftyp=4 Cuntze =4: Gradual: iftyp=3 Puck =5: Gradual: iftyp=4 Cuntze

	stre,i	Variable	Description
IFTYP=1 TSAI-WU	16	$iv_1$	[ $\cdot$ ] damaged
IFTYP=2 HASHIN	stre,i	Variable	Description
	16	$M_t$	[ $\cdot$ ] tensile matrix failure
	17	$M_c$	[ $\cdot$ ] compressive matrix failure
	18	$FM$	[ $\cdot$ ] matrix shear failure
	19	$MFM$	[ $\cdot$ ] = $M_t + FM$ or = $M_c + FM$
	20	$FF$	[ $\cdot$ ] Fiber fracture
IFTYP=3 PUCK	stre,i	Variable	Description
	16	$\sum \text{Mode}_i$	[ $\cdot$ ] Matrix cracking
	17	Mode A	[ $\cdot$ ] tensile matrix failure
	18	Mode B	[ $\cdot$ ] matrix shear failure
	19	Mode C	[ $\cdot$ ] compressive matrix failure
	20	$FF$	[ $\cdot$ ] Fiber fracture
IFTYP=4 CUNTZE	stre,i	Variable	Description
	16	$\sum IFF_i$	[ $\cdot$ ] Matrix cracking
	17	$IFF1$	[ $\cdot$ ] tensile matrix failure
	18	$IFF2a$	[ $\cdot$ ] matrix shear failure $\sigma_{22} \geq 0$
	19	$IFF2b$	[ $\cdot$ ] matrix shear failure $\sigma_{22} < 0$
	20	$IFF3$	[ $\cdot$ ] compressive matrix failure
	21	$FF$	[ $\cdot$ ] Fiber fracture

For all damage variables:  $0 \leq d \leq 1$

# Chapter 6

## Elements in Student Version

### 6.1 Available elements

El. Nr.	Aufgabe
1	2D/3D-Fachwerkelement
2	2D-Timoshenko - Balkenelement
3	2D-Bernoulli - Balkenelement
4	3D-Bernoulli - Balkenelement
5	3-9 Knoten Scheibenelement (ESZ, EVZ, Axisym)
6	4 Knoten Scheibenelement (ESZ, Pian/Sumihara)
7	4 Knoten DKQ-Plattenelement
8	Achsenymmetrisches Schalenelement
9	3D Schalenelement
10	Punktelement fuer Federn/Massen
11	3D-Timoshenko - Balkenelement - exzentrisch
12	Wärmeleitung o. Grundwasserströmung
13	flache Schale
14	2D-Knoten-Knoten-Kontaktelement
15	3D-Timoshenko - Balkenelement linear/nichtlinear
16	3D Seilelement
17	4 Knoten Bathe/Dvorkin-Plattenelement
18	4/8/9 Knoten SRI-Plattenelement
19	3 Knoten DKT-Plattenelement
20	Trägerrost Bernoulli/St. Venant Element
21	3D Volumenelement
22	4/8/9 Knoten Scheibenelement
23	4 Knoten IDKQ-Plattenelement
24	Schubspannungen aus $Q/M_T$ in dünnwandigen Querschnitten
25	
26	4 Knoten Scheibenelement (EVZ, Axisym) ( $\bar{\mathbf{B}}$ -Formulierung)
27	
28	
29	
30	4/9 Oberflächen-Lastelement
33	4 Knoten Plattenelement mit Stabilisierung

El. Nr.	Aufgabe
34	
35	
36	
37	
38	
39	
40	
41	
42	
43	
44	
45	3D Solidschalenelement
46	
47	
48	

## 6.2 Element numbers

Aufgabe	El. Nr.
Balken 2D-Timoshenko	2
Balken 2D-Bernoulli	3
Balken 3D-Timoshenko	15
Balken 3D-Timoshenko exzentrisch	11
Balken 3D-Bernoulli	4
Fachwerke 2D/3D	1
Flache Schale linear/nichtlinear	13
Knoten-Knoten-Kontakt 2D	14
Oberflächenlasten	30
Platten Bathe/Dvorkin	17
Platten DKT	19
Platten DKQ	7
Platten IDKQ	23
Platten SRI	18
Platten SRI+Stabilisierung	33
Punktfedern	10
Punktmassen	10
Scheiben (ESZ,EVZ,Axisym)	5
Scheiben (ESZ,Pian/Sumihara)	6
Scheiben	22
Scheiben (EVZ,Axisym) ( $\bar{\mathbf{B}}$ )	26
Schalen Achsensymmetrie	8
Schalen 3D	9
Seil 3D	16
Wärmeleitung	12
Grundwasserströmung	12
Trägerrost	20
Volumen, 3D-Körper	21
Schub aus $Q/M_T$ in dünnwandigen Querschnitten	24
	25
	26
Scheiben - $\bar{\mathbf{B}}$ -Formulierung)	27
	28
	29
Oberflächen-Lasten	30
	31
	32
Platten URI mit Stabilisierung	33

Aufgabe	El. Nr.
	34
	35
	36
	37
	38
	39
	40
	41
	42
	43
	44
3D-Solidschalen	45
	46
	47
	48

### 6.3 Short description of elements

Element No.	Element type	Input Data
<b>01</b>	2D/3D Truss element	$E, A, \rho$
<b>02</b>	non-linear 2D Timoshenko-Beam Element	$E, G, A, I, \rho, \eta, \alpha_t, h, ityp$ $q_1, q_2, n_1, n_2$
<b>03</b>	2D Bernoulli beam element	$E, A, I, h, q_1, q_2, n_1, n_2, T2O, num$ $\rho, c, b, \alpha_t, t_N, t_M$
<b>04</b>	3D Bernoulli beam element	$E, G, A, I_x, I_y, I_z, q_{xL}, q_{yL}, q_{zL}, q_{xG}, q_{yG}, q_{zG}, \alpha$ $T2O, \rho$
<b>05</b>	Plane stress/strain element	$E, \nu, \rho, < i, l, k >$ $h, b_1, b_2, \alpha, T_0$ or $T_0 - T_1$
<b>06</b>	P/S plain stress/strain element	$E, \nu, \rho, h, i$
<b>07</b>	DKQ plate element	$E, \nu, h, q, \rho, c, f_{cd}, f_{yd}, d_x, d_y$ $\alpha_t, \Delta T$
<b>08</b>	Axisymmetric shell element	$E, \nu, h, \rho, c_b, \kappa_s$ $g, p_c, p_0, z_0, idir, s$ $\alpha_t, t_b, t_t$
<b>09</b>	General shell element	$E, \nu, \rho, h$ $b_1, b_2, p, b_x, b_y, b_z, t_c, c$ $igeo, a1, a2, a3$
<b>10</b>	Spring/mass/damping element	$K(i) i = 1, ndf$ $M(i) i = 1, ndf$ $D(i) i = 1, ndf$
<b>11</b>	3D eccentrical Timoshenko beam element	$E, G, A, I_x, I_y, I_z, q_x, q_y, q_z, \alpha$ $lin, \rho, I_{yz}, y_S, z_S, y_M, z_M, y_O, z_O, scf$
<b>12</b>	Heat transfer element (and other steady state problems)	$ityp, kat, ksym$ $K_1, K_2, angl1 - x, \rho \cdot h, \rho \cdot c, \rho \cdot g, elev$
<b>13</b>	shallow shell element	$E, \nu, h, q, \rho, lin$
<b>14</b>	node to node contact element	$idof, pen, tol$
<b>15</b>	2–5 node non-linear 3D Timoshenko-Beam Element	$E, G, A, I_y, I_z, I_{yz}, I_T, y_s, z_s, y_m, z_m, \alpha$ $iGeo, iPlo, iscf, \rho$

Element No.	Element type	Input Data
<b>16</b>	3D cable element	$E, A, \gamma, \alpha_t, s_0/\ell_0$ $S_0, q_x, q_y, q_z, \Delta_t$
<b>17</b>	Bathe/Dvorkin plate element	$E, \nu, h, q, \rho, SCF$
<b>18</b>	SRI plate element	$E, \nu, h, q, \rho, nb, ns, SCF$
<b>19</b>	DKT plate element	$E, \nu, h, q$
<b>20</b>	2D Bernoulli/St.Venant Grid beam element	$E, G, I_t, I_y, q_1, q_2$
<b>21</b>	3D solid element	$matn, lin, isym, istr$ $q_x, q_y, q_z, \Delta T, \alpha_t, \rho$ matn=1: $E, \nu$
<b>22</b>	Plane stress element	$E, \nu, h, \alpha_T, \Delta T, lin, Y_0, cp$
<b>23</b>	Improved DKQ plate element	$E, \nu, h, q$
<b>24</b>	Shear stresses from forces/torsional moments	$E, G, b, Q_y, Q_z, ns, nsp, ityp, M_{T0}, ic$
<b>25</b>		
<b>26</b>	$\bar{B}$ -plain strain/axisym element	$imat$ $ityp, II, \rho, K, G$
<b>27</b>		
<b>28</b>		
<b>29</b>		
<b>30</b>	Surface load element	—
<b>31</b>		
<b>32</b>		
<b>33</b>	SRI plate element with stabilization	$E, \nu, h, q, \rho, r_w, r_\beta$

Element No.	Element type	Input Data
<b>34</b>		
<b>35</b>		
<b>36</b>		
<b>37</b>		
<b>38</b>		
<b>39</b>		
<b>40</b>		
<b>41</b>		
<b>42</b>		
<b>43</b>		
<b>44</b>		
<b>45</b>	3D solid shell element EAS	$matn, nlay, lin, ieas1, ibd, ibs, ieas2, h, ngp, scf$ $q_x, q_y, q_z, \Delta T, \alpha_t, \rho$ $E, \nu$ (for $matn=1$ ) for $n=1, nlay$ : $\phi_i, h_n$
<b>46</b>		
<b>47</b>		
<b>48</b>		

## 6.4 Detailed description of elements

### 6.4.1 ELMT01

$E, A, \rho$

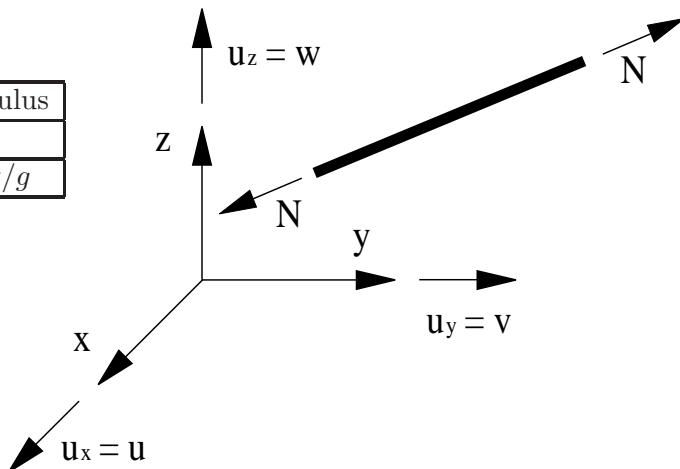
- Theory

ELMT01 is a general 2D/3D truss element.

- Material input data

$E$	$[F/L^2]$	Young's modulus
$A$	$[L^2]$	Area
$\rho$	$[F/L^3]/[L/T^2]$	density $\rho = \gamma/g$

- Displacements and normal forces



- Output of results

Normal forces, strains and stresses are calculated for each element in the local axial direction.

ELMT01.WPG

- Plot of results

Results are plotted due to following numbers

Stress number	1	7	8
Quantity plotted	$N [F]$	$\varepsilon [-]$	$\sigma [F/L^2]$

### 6.4.2 ELMT02

$E, G, A, I, \rho, \eta, \alpha_t, h, ityp$   
 $q_1, q_2, n_1, n_2$

- Theory

ELMT02 is a 2-node 2D Timoshenko–Beam Element with **linear** shape functions. Thus **more** elements are necessary for a correct solution. Geometrical nonlinear calculations are possible for different nonlinear theories. For shear correction  $\kappa = 5/6$  together with  $GA_s = \frac{\kappa \cdot GA}{(1 + \kappa \cdot \ell^2/12 \cdot GA/EI)}$  is used.

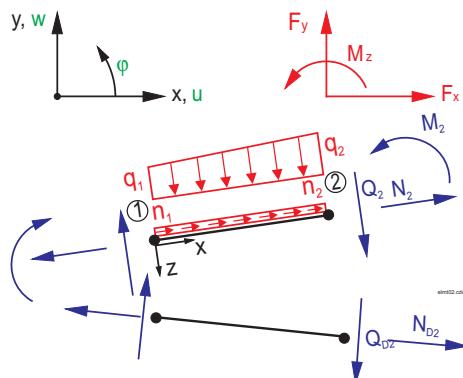
$E$	$[F/L^2]$	Elastic modulus
$G$	$[F/L^2]$	Shear modulus
$A$	$[L^2]$	Area
$I$	$[L^4]$	Moment of Inertia
$\rho$	$[F/L^3]/[L/T^2]$	density $\rho = \gamma/g$
$\eta$	$[F/L^3]/[L/T]$	viscosity
$\alpha_t$	$[1/K]$	coefficient of thermal expansion
$h$	$[L]$	thickness for temperature loads
$ityp$	$[-]$	1 = finite rotations (Green)(default) 2 = finite rotations (Reissner) 3 = moderate rotations (consistent, Green) 4 = moderate rotations (simplified, Green) 5 = linear
$q_1$	$[F/L]$	load in local z-dir. at left node
$q_2$	$[F/L]$	load in local z-dir. at right node
$n_1$	$[F/L]$	load in local x-dir. at left node
$n_2$	$[F/L]$	load in local x-dir. at right node

- Material input data

$ma$	$[-]$	Material number in Inputfile
$q_1$	$[F/L]$	load in local z-dir. at left node
$q_2$	$[F/L]$	load in local z-dir. at right node
$n_1$	$[F/L]$	load in local x-dir. at left node
$n_2$	$[F/L]$	load in local x-dir. at right node
ifol	$[-]$	follower load 0/1, only for q
$T_u$	$[\text{°}K]$	Temperature at bottom
$T_o$	$[\text{°}K]$	Temperature at top

- Loads via macro **qloa**.

- Displacements and stress resultants



- **Output of stress resultants**

Stress resultants (N–Axial Force, Q–Shear Force, M–Bending Moment) are calculated at left and right node of the element in local directions for the reference configuration (ityp=5) or the current configuration(ityp $\leq$ 4).

- **Plot of stress resultants**

The above defined stress resultants are plotted due to following numbers

Stress number	1	2	3
Stress resultant	N [F]	Q [F]	M [FL]

### 6.4.3 ELMT03

$E, A, I, h, q_1, q_2, n_1, n_2, T2O, num$   
 $\rho, c, b, \alpha_t, t_t, t_b$

- Theory

**ELMT03** is a 2-node 2D Bernoulli–Beam Element with Hermitean shape functions (cubic). Thus at least one element leads for a part of a structure in most cases to the correct solution. Symmetry with respect to local z-axis is assumed.

- Material input data

Record 1

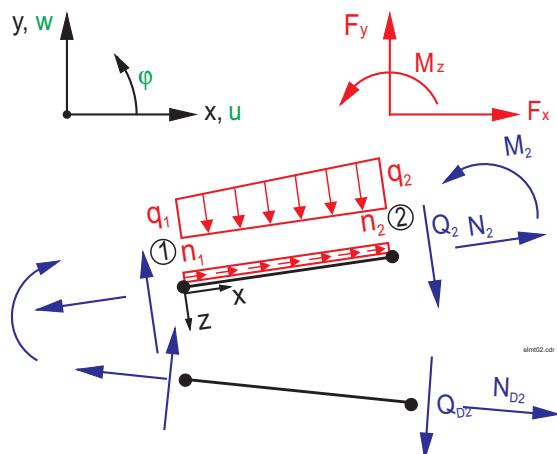
Record 2

$E[F/L^2]$	Elastic modulus	$\rho[F/L^3]/[L/T^2]$	density $\rho = \gamma/g(onlyLMAS)$
$A[L^2]$	Area	$c[F/L^3]$	el. foundation modulus
$I[L^4]$	Moment of Inertia	$b[L]$	width for foundation
$h[L]$	thickness	$\alpha_t[1/T]$	coefficient of thermal expansion
$q_1[F/L]$	load in local z-dir / left node	$t_N[T]$	temperature at $z = 0$
$q_2[F/L]$	load in local z-dir / right node	$t_M[T]$	temperature difference $T_{z-} - T_{z+}$
$n_1[F/L]$	load in local x-dir / left node		
$n_2[F/L]$	load in local x-dir / right node		
$T2O$	II.Order theory(0=F,1=T,2=T(Resi)		
$num$	Plotnumbers (0=F,1=T)		

$ma$	$[-]$	Material number in Inputfile
$q_1$	$[F/L]$	load in local z-dir. at left node
$q_2$	$[F/L]$	load in local z-dir. at right node
$n_1$	$[F/L]$	load in local x-dir. at left node
$n_2$	$[F/L]$	load in local x-dir. at right node

- Loads via macro **qloa**.

- Displacements and stress resultants



- **Output of stress resultants**

Stress resultants (N–Axial Force, Q–Shear Force, M–Bending Moment and Foundation Pressure) are calculated at left and right node of the element in local directions due to the standard sign convention.

- **Plot of stress resultants**

The above defined stress resultants are plotted due to following numbers

Stress number	1	2	3	4
Stress resultant	N [F]	Q [F]	M [F · L]	P(c) [F/L <sup>2</sup> ]

- **Remark on II. Order Theory**

#T20

= 1: Full iteration  $[\mathbf{K}_e + \mathbf{K}_g]\Delta\mathbf{v}^i = \mathbf{P} - [\mathbf{K}_e + \mathbf{K}_g]\mathbf{v}^i$

= 2: Iteration only on right hand side  $\mathbf{K}_e\Delta\mathbf{v}^i = \mathbf{P} - [\mathbf{K}_e + \mathbf{K}_g]\mathbf{v}^i$

# iteration until convergence, e.g. 10 steps:

**loop**,,10

**tang**,,1

**next**

# results: **disp**,all **stre**,all **reac**,all

- For the elastic foundation two models are implemented:

$c > 0$  el. found. always (tension and compression)

$c < 0$  el. found. only for compression (Solution only available after equilibrium iteration!)

**loop**,,10

**tang**,,1

**next**

#### 6.4.4 ELMT04

$E, G, A, I_x, I_y, I_z, q_{xL}, q_{yL}, q_{zL}, q_{xG}, q_{yG}, q_{zG}, \alpha$   
 $T2O, \rho$

- **Theory**

**ELMT04** is a 2(+1)-node 3D Bernoulli–Beam Element. The element bases on a Bernoulli-theory for the bending terms whereas for the torsional terms a St.–Venant theory is used. The local axis are chosen as follows:  $x_L$  is the axial axis through the center of gravity,  $y_L$  and  $z_L$  are the **principal axes** of the beam. The base vectors can be defined in two ways:

- Element with 3 nodes: introduce third node for each element (thus, define in first line of input: nel = 3 and use for generation of elements the macro 'el3b')

$$\mathbf{e}_{xL} \text{ in element direction} \quad \mathbf{e}_{yL} = [\mathbf{x}_3 - \mathbf{x}_1] \times \mathbf{e}_{xL} \quad \mathbf{e}_{zL} = \mathbf{e}_{xL} \times \mathbf{e}_{yL}$$

- Element with 2 nodes: Element with 2 nodes: use the global z-axis for the definition of basis, thus nel = 2 and the macro 'elem' can be used.

$$\mathbf{e}_{xL} \text{ in element direction} \quad \mathbf{e}_{yL} = \mathbf{e}_{xL} \times \mathbf{e}_{zG} \quad \text{with} \quad \mathbf{e}_{zG} = [0, 0, 1] \quad \mathbf{e}_{zL} = \mathbf{e}_{xL} \times \mathbf{e}_{yL}$$

For the special case that the element direction is  $\pm$  the global z-axis  $\mathbf{e}_{yL}$  is defined as  $\mathbf{e}_{yL} = [0, -1, 0]$ .

A rotation of the local  $y_L$  and  $z_L$ -axis can be introduced by the angle  $\alpha$ , which starts positive from the  $y_L$ -axis in direction to the  $z_L$ -axis. Input of  $\alpha$  in degree! Thus un-symmetric cross sections can be used.

- **Material input data**

Record 1

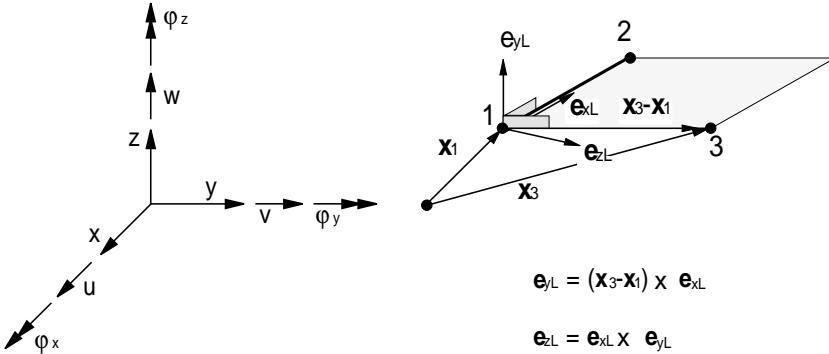
Record 2

$E[F/L^2]$	Elastic modulus	$T2O$	II.Order theory(0= F, 1= T)
$G[F/L^2]$	Shear modulus	$\rho[FT^2/L^4]$	density $\rho = \gamma[F/L^3]/g[L/T^2]$
$A[L^2]$	Area		(only for LMAS)
$I_x[L^4]$	Moment of Inertia / local x-axis		
$I_y[L^4]$	Moment of Inertia / local y-axis		
$I_z[L^4]$	Moment of Inertia / local z-axis		
$q_{xL}[F/L]$	const. load / local x-direction		
$q_{yL}[F/L]$	const. load / local y-direction		
$q_{zL}[F/L]$	const. load / local z-direction		
$q_{xG}[F/L]$	const. load / global x-direction		
$q_{yG}[F/L]$	const. load / global y-direction		
$q_{zG}[F/L]$	const. load / global z-direction		
$\alpha[DEG.]$	Rot. angle for local coor.system		

<i>ma</i>	[—]	Material number in Inputfile
$q_{xL}$	[F/L]	const. load / local x-direction
$q_{yL}$	[F/L]	const. load / local y-direction
$q_{zL}$	[F/L]	const. load / local z-direction
$q_{xG}$	[F/L]	const. load / global x-direction
$q_{yG}$	[F/L]	const. load / global y-direction
$q_{zG}$	[F/L]	const. load / global z-direction

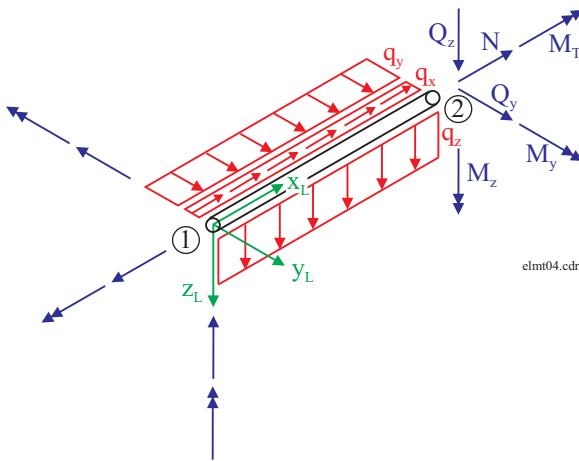
- Loads via macro **qloa**.

- Displacements and base vectors for element



### • Output of stress resultants

Stress resultants ( $N_x$ —Axial Force,  $Q_y/Q_{dy} = R_y$ —Shear Force,  $Q_z/Q_{dz} = R_z$ —Shear Force,  $M_x$ —Torsional Moment,  $M_y$ —Bending Moment,  $M_z$ —Bending Moment) are calculated at left and right node of the element in local directions.



### • Plot of stress resultants [FORC]

The above defined stress resultants are plotted due to following numbers

Force number	1	2	3	4	5	6
Stress resultant	$N_x$ [F]	$Q_y/Q_{dy}$ [F]	$Q_z/Q_{dz}$ [F]	$M_x$ [FL]	$M_y$ [FL]	$M_z$ [FL]

• **Remark:** The local base vectors can be visualized for control of input data with **Forc,7**.

• **Remark:** Stress resultants are plotted in the local 1-3 direction. The plot plane can be modified with **Forc, $\pm 12$**  or  **$\pm 13$**  (default 13).

### • Remark on II. Order Theory

see remark for **ELMT03**

### 6.4.5 ELMT05

$E, \nu, \rho, <i>, <l>, <k>$   
 $h, b_1, b_2, \alpha, T_0$  or  $T_0 - T_1$

- **Theory**

ELMT05 is a general 3–9 node plane stress, plane strain, and (torsionless) axisymmetric element. The 3–node version is a triangle, the 4–node is a quadrilateral with linear shape functions, whereas the 9–node element has quadratical shape functions. The 5–7 node elements have some midside nodes. The 8–node element is of Serendipity–type.

- **Material input data**

$E[F/L^2]$	Young's modulus
$\nu$	Poisson ratio
$\rho[F/L^3]/[L/T^2]$	density $\rho = \gamma/g$
$<i>$ (def.=1)	1=plane stress, 2=plane strain, 3=axisymmetric calculation
$<l>$ (def.=2)	number of Gauss quadrature points/direction for computing the FE arrays (e.g., stiffness)
$<k>$ (def.=1)	number of Gauss quadrature points/direction for stress outputs
$h[L]$	thickness of plane stress slice (set to 1.0 for plane strain, axisymmetric)
$b_1[F/L^3]$	body force value (constant) for 1(x)-direction
$b_2[F/L^3]$	body force value (constant) for 2(y)-direction $(1 = x_1 = x \text{ or } r, 2 = x_2 = y \text{ or } z)$
$\alpha[1/T]$	coefficient of linear thermal expansion
$T_0[T]$	base temperature for this material or
$T_0 - T_1[T]$	temperature difference for this material

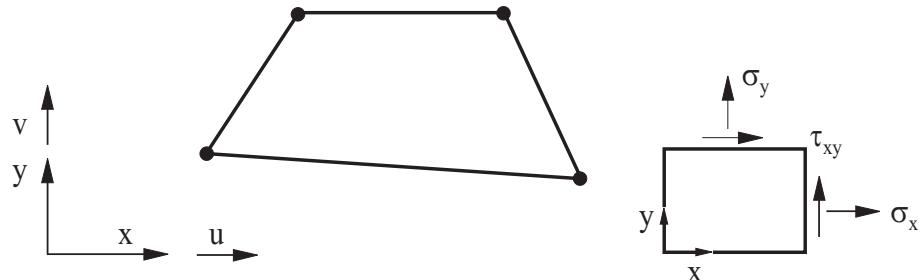
- **Loads via macro `qloa`.**

$ma$	$[ - ]$	Material number in Inputfile
$b_1$	$[F/L]$	body force value (constant) for 1(x)-direction
$b_2$	$[F/L]$	body force value (constant) for 2(y)-direction

- **Temperature**

If the input is the base temperature  $T_0$ , then macros **temp** or **btem** have to be used for input of actual temperature  $T_1$ . Thus a complicated temperature distribution may be possible. In case of an input  $T_0 - T_1$  the system is loaded by a constant temperature difference. Thus the use of **temp** or **btem** is not necessary. Note that  $T_0$  is the base temperature and  $T_1$  is the actual temperature.

- Displacements ( $u, v$ ) and stresses



- Output of stresses and strains

Stresses and strains are calculated at each quadrature point in global directions 1(x) and 2(y). The stresses  $\sigma_1$  and  $\sigma_2$  are the principal stresses with the angle  $\varphi_1$  between 1-stress and 1-coordinate direction (in degrees).

- Plot of stresses

Stresses and strains are plotted due to following numbers

Stress number	1	2	3	4	5	6	7	8	9	10	11
Stress plotted	$\sigma_{xx}$	$\sigma_{xy}$	$\sigma_{yy}$	$\sigma_{zz}$	$\sigma_1$	$\sigma_2$	$\varphi_1$	$\varepsilon_{xx}$	$\varepsilon_{xy}$	$\varepsilon_{yy}$	$\varepsilon_{zz}(i = 1)$ or $\sigma_{\varphi\varphi}(i = 3)$

For  $i=1$  (plane stress): Stress(4) =  $\sigma_v$

Element-wise constant stresses can be plotted with macro **str1**.

Plot-results for the 8-node serendipity-element could not be used due negative weighting functions.

- Remarks

Loads (in **load**, **eloa**, ...) must be multiplied by the term  $2\pi r$  in case of axisymmetric problems!

Especially for inplane bending problems in case of plane stress/strain **ELMT06** should be used which leads to better results.

The element routine will compute any combination of nodes, up to 9, provided all vertex nodes are specified. Triangles may be specified by either giving the same node number for contiguous vertices, or for a 3-node element, by specifying only the first 3 nodes (i.e., node 4 has a zero value).

### 6.4.6 ELMT06

$E, \nu, \rho, h, i$

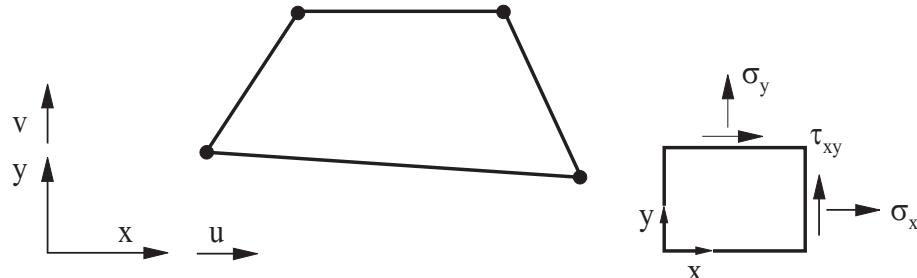
- Theory

**ELMT06** is a 4 node plane stress, plane strain element. It is called PIAN/SUMIHARA element and is based on a mixed formulation. This element is superior to **ELMT05** and should be used instead of **ELMT05**, especially for inplane bending problems.

- Material input data

$E[F/L^2]$	Young's modulus
$\nu$	Poisson ratio
$\rho[F/L^3]/[L/T^2]$	density $\rho = \gamma/g$
$h[L]$	thickness (set to 1.0 for plane strain)
$i$	1=plane stress, 2=plane strain

- Displacements ( $u, v$ ) and stresses



- Output of stresses

Stresses are calculated at the element center in global directions 1(x) and 2(y). The stresses  $\sigma_1$  and  $\sigma_2$  are the principal stresses with the angle  $\varphi_1$  between 1-stress and 1-coordinate direction (in degrees).

- Plot of stresses

Stresses are plotted due to following numbers

Stress number	1	2	3	4	5	6	7
Stress plotted	$\sigma_{xx}$	$\sigma_{xy}$	$\sigma_{yy}$	$\sigma_{zz}$	$\sigma_1$	$\sigma_2$	$\varphi_1$

Element-wise constant stresses can be plotted with macro **str1**.

### 6.4.7 ELMT07

$E, \nu, h, q, \rho, c, f_{cd}, f_{yd}, d_x, d_y$   
 $\alpha_t, \Delta T$

- **Theory**

ELMT07 is a general 4 - node thin plate bending element based on the Discrete Kirchhoff Theory.

- **Material input data**

Record 1

$E[F/L^2]$	Modulus of elasticity
$\nu$	Poisson ratio
$h[L]$	Plate thickness
$q[F/L^2]$	Uniform normal load in z-dir.
$\rho[F/L^3]/[L/T^2]$	density : $\rho = \gamma/g$
$c[F/L^3]$	elastic foundation in z-dir.
$f_{cd}[F/L^2]$	Strength concrete
$f_{yd}[F/L^2]$	Strength steel
$d_x[L]$	thickness until steel in x-dir.
$d_y[L]$	thickness until steel in y-dir.

Record 2

$\alpha_t$	coefficient of thermal expansion
$\Delta T$	temperature difference ( $T_{top} - T_{bottom}$ )

- For the elastic foundation two models are implemented:

$c > 0$  el. found. always (tension and compression)

$c < 0$  el. found. only for compression (Solution only available after equilibrium iteration!)

- **Loads via macro **qloa**.**

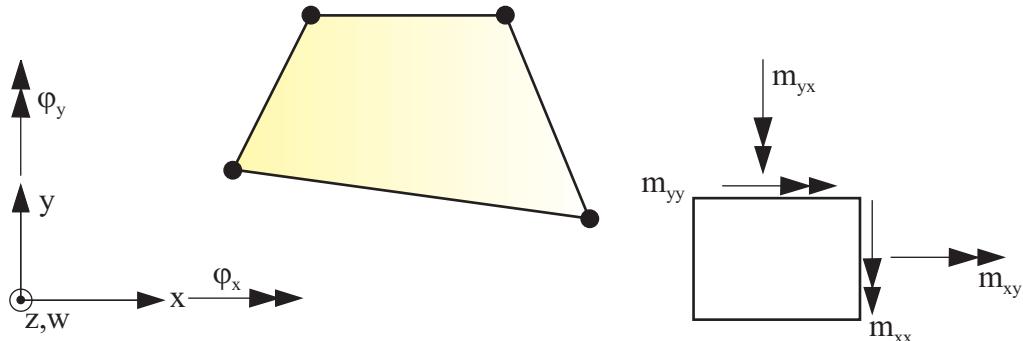
$ma$	$[ - ]$	Material number in Inputfile
$q$	$[F/L^2]$	uniform normal load in z-direction
$\Delta T$	$[^\circ K]$	Temperature difference ( $T_{top} - T_{bottom}$ )

- Be sure to have the same units for all input values!

- Definition of stress resultants

$$m_{xx} = - \int \sigma_{xx} z dz, \quad m_{yy} = - \int \sigma_{yy} z dz, \quad m_{xy} = - \int \sigma_{xy} z dz = m_{yx}$$

- Displacements ( $w, \varphi_x, \varphi_y$ ) and moments per length



ELMT07.CDR

- Output of moments

The moments per unit length are calculated at each center of element in global directions x and y. The moments  $m_1$  and  $m_2$  are the principal moments with the angle  $\varphi_1$  between 1-moment and x-coordinate direction (in degrees). Furthermore the foundation pressure is calculated at each center (Positive Pressure is in -z direction).

The reinforcements are calculated based on the general design diagram due to DIN 1045-1 (new). Note that ultimate(!) loads are input and that minimum values of reinforcements and constructive reinforcements have to be added.

- Plot of moments

The moments per unit length, the foundation pressure and the reinforcements are plotted due to following numbers

Stress number	Type	value plotted	Dimension
1	Moment	$m_{xx}$	[ $FL/L$ ]
2	Moment	$m_{xy}$	[ $FL/L$ ]
3	Moment	$m_{yy}$	[ $FL/L$ ]
4	Pressure	Press(c)	[ $F/L^2$ ]
5	Moment	$m_1$	[ $FL/L$ ]
6	Moment	$m_2$	[ $FL/L$ ]
7	Angle	$\varphi_1$	[rad]
8	Shear force	$q_x$	[ $FL/L$ ]
9	Shear force	$q_y$	[ $FL/L$ ]
10	Reinforcement	$as_x \text{ bot}$	[ $L^2/L$ ]
11	Reinforcement	$as_x \text{ top}$	[ $L^2/L$ ]
12	Reinforcement	$as_y \text{ bot}$	[ $L^2/L$ ]
13	Reinforcement	$as_y \text{ top}$	[ $L^2/L$ ]

Element-wise constant values can be plotted with macro **str1**.

### 6.4.8 ELMT08

$E, \nu, h$   
 $g, p_c, p_0, z_0, idir, s$   
 $\alpha_t, t_b, t_t$

- **Theory**

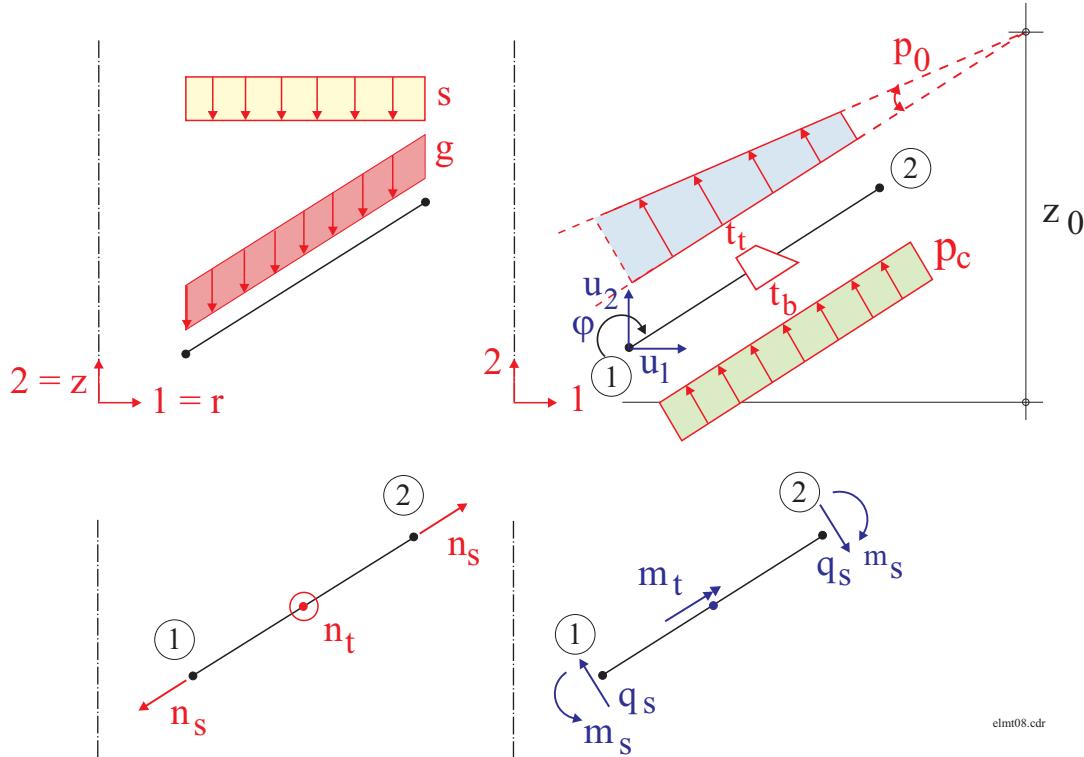
Elmt08 is a shear-elastic plane 2-node shell-element for axisymmetric shells with axisymmetric loading. Due to linear shape functions  $C^0$ -continuity is achieved. The geometry is described with a two-dimensional mesh (ndm=2).

- **Material input data**

Record 1

$E[F/L^2]$	Youngs modulus
$\nu$	Poison's ratio
$h[L]$	thickness
$\rho[F/L^3]/[L/T^2]$	density $\rho = \gamma/g$
$c_b[F/L^3]$	elastic foundation
$SCF$	shear correction factor, see Elmt18
Record 2	
$g[F/L^3]$	deadload, positive in global negative 2-direction
$p_c[F/L^2]$	const. pressure load, positive in positive local 2-direction
$p_0[F/L^3]$	slope value for linear pressure
$z_0[L]$	coordinate of zero pressure, only for linear pressure loading
$idir$	direction of linear pressure loading, 1=r, 2=z (Default=2)
$s[F/L^2]$	snowload, positive in global negative 2-direction
Record 3	
$\alpha_t[1/K]$	coefficient of thermal expansion
$t_b[K]$	temperature at shell bottom
$t_t[K]$	temperature at shell top

- Displacements and stress resultants



- Output of stress resultants

Stress resultants are calculated at the center of the element:

$n_s$	Axial Force direction s
$n_t$	Axial Force direction t
$m_s$	Bending Moment direction s
$m_t$	Bending Moment direction t
$q_s$	Shear Force direction s

- Plot of stress resultants

The above defined stress resultants are plotted due to following numbers

Force number	1	2	3	4	5
Stress resultant	$n_s$	$n_t$	$m_s$	$m_t$	$q_s$

- Remark: Input of loads is also possible with the macros **load** and **eloa**. Here, the factor  $2\pi r$  has to be added for all terms! Furthermore **eloa** can be used for loads  $q$ , if the term  $q$   $2\pi r$  fits in the schemes of load cases ( $r$  may be not constant!).

### 6.4.9 ELMT09

$E, \nu, \rho, h$   
 $b_1, b_2, p, b_x, b_y, b_z, t_c, c$   
 igeo,a1,a2,a3

- **Theory**

**ELMT09** is a 4-node general shell-Element. The element is based on a DKQ-theory for the bending terms with modifications due to the element warping whereas the membrane part is modified due to the use of the rotation around the local z-axis which is called drilling degree of freedom.

- **Input data**

# Record 1: Material input data

$E[F/L^2]$	Youngs modulus
$\nu$	Poison's ratio
$\rho[F/L^3]/[L/T^2]$	density $\rho = \gamma/g$
$h[L]$	thickness

# Record 2: Load data

$b_1[F/L^2]$	uniform loading in 1-direction
$b_2[F/L^2]$	uniform loading in 2-direction
$p[F/L^2]$	uniform loading in 3-direction
$b_x[F/L^2]$	gravity loading in x-direction
$b_y[F/L^2]$	gravity loading in y-direction
$b_z[F/L^2]$	gravity loading in z-direction
$t_c$	load type (0=lumped; 1=consistent)
$c[F/L^3]$	elastic found. constant in local 3-dir.

# Record 3: Local base vectors

igeo	1	2	3-17
a1		$\mathbf{t}_{qx}[L]$	see macro <b>base</b>
a2		$\mathbf{t}_{qy}[L]$	
a3		$\mathbf{t}_{qz}[L]$	

- **Loads via macro **qloa**.**

$ma$	$[ - ]$	Material number in Inputfile
$b_1$	$[F/L^2]$	uniform loading in 1-direction
$b_2$	$[F/L^2]$	uniform loading in 2-direction
$b_3$	$[F/L^2]$	uniform loading in 3-direction
$b_x$	$[F/L^2]$	gravity loading in x-direction
$b_y$	$[F/L^2]$	gravity loading in y-direction
$b_z$	$[F/L^2]$	gravity loading in z-direction

- **Base Systems**

The local coordinate system is assumed in the center of the element.

# **igeo = 0,1** (default option)

$\mathbf{t}_3$  is the normal vector. The local base vector  $\mathbf{t}_1$  bisects the line between node 2 and 3 of the element, the base vector  $\mathbf{t}_2$  is calculated from  $\mathbf{t}_2 = \mathbf{t}_3 \times \mathbf{t}_1$ .

# **igeo = 2**

Input is a vector  $\mathbf{t}_q = [\mathbf{t}_{qx}, \mathbf{t}_{qy}, \mathbf{t}_{qz}]^T$ . Again  $\mathbf{t}_3$  is the normal vector, defined like for igeo=1.

It follows:  $\mathbf{t}_2 = \mathbf{t}_3 \times \mathbf{t}_q$  and  $\mathbf{t}_1 = \mathbf{t}_2 \times \mathbf{t}_3$ .

# **igeo = 3-17**

Other base systems can be defined by using data from the macro **base**.

### Remarks

# For regular meshes igeo=0,1 leads to correct results, whereas for distorted meshes each element has different directions. Thus stress resultants are calculated in different directions but added for e.g. **stre,1**. This is nonsense!

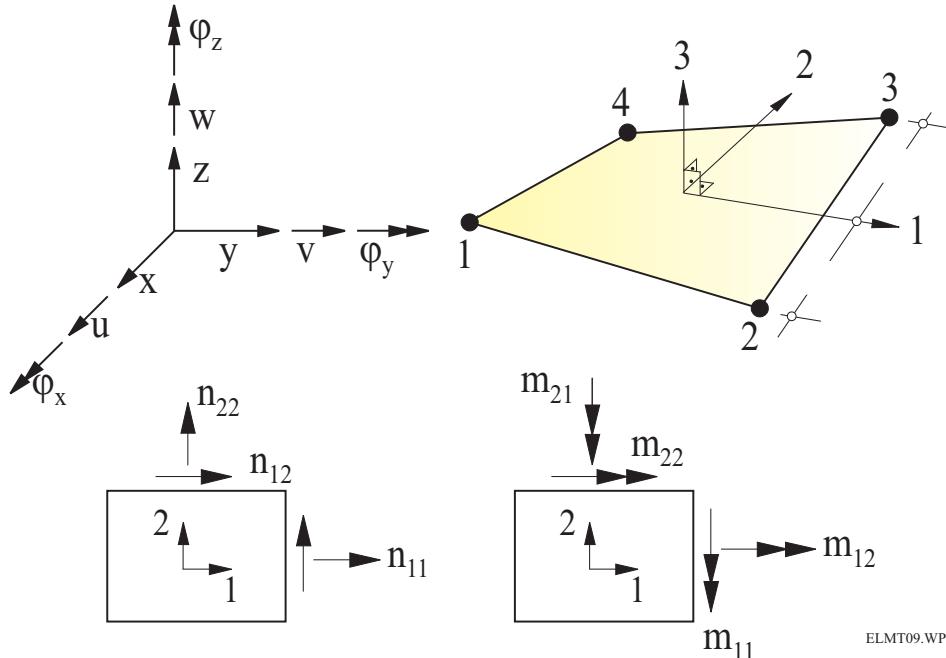
# It is strongly recommended to **control** the orientation of the local base system using the macro **forc,1**

# If the element is used in combination with adaptivity the input of  $\mathbf{t}_q$  is mandatory!!

- **Elastic foundation**

For the elastic foundation two models are implemented:  
 $c > 0$  el. found. always (tension and compression)  
 $c < 0$  el. found. only for compression (Solution only available after equilibrium iteration!)

- **Displacements and stress resultants**



ELMT09.WPG

- **Output of stress/strain resultants of shell (stre,...)** (at center/midsurface of element)

$n_{11}$	Axial Force direction 1	$m_{11}$	Bending Moment direction 1
$n_{12}$	Axial Force direction 12	$m_{12}$	Bending Moment direction 12
$n_{22}$	Axial Force direction 2	$m_{22}$	Bending Moment direction 2
$n_1$	Principal Force 1	$m_1$	Principal Moment 1
$n_2$	Principal Force 2	$m_2$	Principal Moment 2
$\alpha_N$	principal angle for forces	$\alpha_M$	principal angle for moments
$\varepsilon_{11}$	membrane strain in direction 1	$\kappa_{11}$	curvature in direction 1
$\varepsilon_{12}$	membrane strain in direction 12	$\kappa_{12}$	curvature in direction 12
$\varepsilon_{22}$	membrane strain in direction 2	$\kappa_{22}$	curvature in direction 2
p	pressure in direction 3 (+p in -3dir.)		

- **Plot of stress resultants (stre,...)**

Stress number	1	2	3	4	5	6	7	8	9	10	11
Stress resultant	$n_{11}$	$n_{12}$	$n_{22}$	$n_1$	$n_2$	$m_{11}$	$m_{12}$	$m_{22}$	$m_1$	$m_2$	p

- **Output of stresses/strains (stre,lay,1,n,-1)** (at center/top(t)+bottom(b) of element)

Values at top (t)

Values at bottom (b)

$\sigma_{11}(t)$	Stress in direction 1	$\sigma_{11}(b)$	Stress in direction 1
$\sigma_{12}(t)$	Stress in direction 12	$\sigma_{12}(b)$	Stress in direction 12
$\sigma_{22}(t)$	Stress in direction 2	$\sigma_{22}(b)$	Stress in direction 2
$\sigma_1(t)$	Principal stress 1	$\sigma_1(b)$	Principal stress 1
$\sigma_2(t)$	Principal stress 2	$\sigma_2(b)$	Principal stress 2
$\alpha_\sigma(t)$	principal angle for stresses	$\alpha_\sigma(b)$	principal angle for stresses
$\varepsilon_{11}(t)$	strain in direction 1	$\varepsilon_{11}(b)$	strain in direction 1
$\varepsilon_{12}(t)$	strain in direction 12	$\varepsilon_{12}(b)$	strain in direction 12
$\varepsilon_{22}(t)$	strain in direction 2	$\varepsilon_{22}(b)$	strain in direction 2
p	pressure in direction 3 (+p in -3dir.)		

- **Plot of stresses (stre,i,,,-1)**

Values at top (t)

Values at bottom (b)

Stress number	1	2	3	4	5	6	7	8	9	10	11
Stress	$\sigma_{11}$	$\sigma_{12}$	$\sigma_{22}$	$\sigma_1$	$\sigma_2$	$\sigma_{11}$	$\sigma_{12}$	$\sigma_{22}$	$\sigma_1$	$\sigma_2$	p

- **Plot of element-wise constant stresses/stress resultants**

Use macro **str1,i**.

### 6.4.10 ELMT10

---

$$K(i), i = 1, \text{ndf}$$

$$M(i), i = 1, \text{ndf}$$

$$D(i), i = 1, \text{ndf}$$

---

- **Theory**

**ELMT10** is a point stiffness/mass/damping element for adding point (diagonal) entries to the stiffness/mass/damping matrix. An element with one node has to be defined in the Input-file.

If **two** nodes are defined the stiffness terms are set between these two nodes for the defined dofs. Thus a coupling of elements is possible. Other dofs between these nodes may be coupled with **link**.

- **Material input data**

$K(1)[F/L]$	Stiffness value to be added to 1-dof
$K(2)[F/L]$	Stiffness value to be added to 2-dof
	... up to ndf
$M(1)[F]/[L/T^2]$	Mass value to be added to 1-dof
$M(2)[F]/[L/T^2]$	Mass value to be added to 2-dof
	... up to ndf
$D(1)[F]/[L/T]$	Damping value to be added to 1-dof
$D(2)[F]/[L/T]$	Damping value to be added to 2-dof
	... up to ndf

- **Output of internal forces/reactions**

The internal forces=reactions for each degree of freedom are calculated and could be printed via the **stre**-macro .

Reactions at nodes are calculated and printed via **reac** only in the case of two nodes.

- **Plot of internal forces/reactions**

A plot of the internal forces is not available.

Reactions at nodes are calculated and plotted via **reac** only in the case of two nodes.

### 6.4.11 ELMT11

$E, G, A, I_T, I_y, I_z, q_x, q_y, q_z, \alpha$   
 $lin, \rho, I_{yz}, y_S, z_S, y_M, z_M, y_O, z_O, scf$

- **Theory**

**ELMT11** is a geometrically nonlinear 2(+1)-node 3D Timoshenko–Beam Element. The element bases on a Timoshenko-theory for the bending terms whereas for the torsional terms a St.–Venant theory is used. The element allows an eccentric formulation. Thus, it should be used especially in combination with plate and shell elements. If the eccentricity is not used the similar Bernoulli-element (**ELMT04**) is advantageous.

The base vectors can be defined in two ways:

# Element with 3 nodes: introduce third node for each element (thus, define in first line of input: nel = 3 and use for generation of elements the macro 'el3b')

$$\mathbf{e}_{xL} \text{ in element direction} \quad \mathbf{e}_{yL} = [\mathbf{x}_3 - \mathbf{x}_1] \times \mathbf{e}_{xL} \quad \mathbf{e}_{zL} = \mathbf{e}_{xL} \times \mathbf{e}_{yL}$$

# Element with 2 nodes: Element with 2 nodes: use the global z-axis for the definition of basis, thus nel = 2 and the macro 'elem' can be used.

$$\mathbf{e}_{xL} \text{ in element direction} \quad \mathbf{e}_{yL} = \mathbf{e}_{xL} \times \mathbf{e}_{zG} \quad \text{with} \quad \mathbf{e}_{zG} = [0, 0, 1] \quad \mathbf{e}_{zL} = \mathbf{e}_{xL} \times \mathbf{e}_{yL}$$

For the special case that the element direction is  $\pm$  the global z-axis  $\mathbf{e}_{yL}$  is defined as  $\mathbf{e}_{yL} = [0, -1, 0]$ .

A rotation of the local  $y_L$  and  $z_L$ -axis can be introduced by the angle  $\alpha$ , which starts positive from the  $y_L$ -axis in direction to the  $z_L$ -axis. Input of  $\alpha$  in degree!

- **Material input data**

Record 1

$E[L^4]$	Elastic modulus
$G[L^4]$	Shear modulus
$A[L^2]$	Area
$I_x[L^4]$	Moment of inertia-x / local x-axis
$I_y[L^4]$	Moment of inertia-y / local x-axis
$I_z[L^4]$	Moment of inertia-z / local x-axis
$q_x[F/L]$	constant load in local x-direction / local x-axis
$q_y[F/L]$	constant load in local y-direction / local x-axis
$q_z[F/L]$	constant load in local z-direction / local x-axis
$\alpha[DEG.]$	Rotation angle for local coordinate system

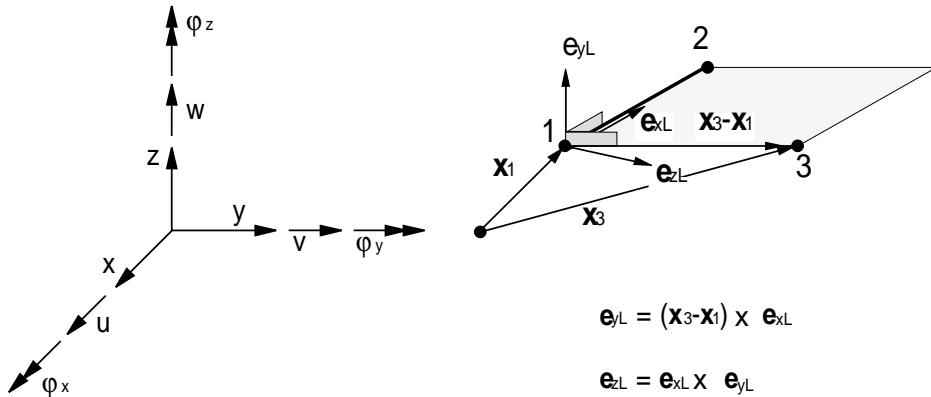
Record 2

$lin[-]$	0[default]=linear, 1=moderate rotations
$\rho[FT^2/L^4]$	density $\rho = \gamma[F/L^3]/g[L/T^2]$ (only for LMAS)
$I_{yz}[L^4]$	Moment of inertia-yz $= + \int yz da /$ local x-axis
$y_S[L]$	y-coordinate of center of gravity point
$z_S[L]$	z-coordinate of center of gravity point
$y_M[L]$	y-coordinate of center of shear point
$z_M[L]$	z-coordinate of center of shear point
$y_O[L]$	y-coordinate of center of output point
$z_O[L]$	z-coordinate of center of output point
$scf[-]$	shear correction 0(def.)=on, 1=off

- Loads via macro **qloa**.

$ma$	$[-]$	Material number in Inputfile
$q_x$	$[F/L]$	constant load in local x-direction
$q_y$	$[F/L]$	constant load in local y-direction
$q_z$	$[F/L]$	constant load in local z-direction

- Displacements and base vectors for element



ELMT04.WPG

- Output of stress resultants

Stress resultants (  $N_x$ -Axial Force,  $Q_y$ -Shear Force,  $Q_z$ -Shear Force,  $M_x$ -Torsional Moment,  $M_y$ -Bending Moment,  $M_z$ -Bending Moment) are calculated at left and right node of the element in local directions at the defined output point.

- Plot of stress resultants [FORC]

The above defined stress resultants are plotted due to following numbers

Force number	1	2	3	4	5	6
Stress resultant	$N_x$ [F]	$Q_y$ [F]	$Q_z$ [F]	$M_x$ [FL]	$M_y$ [FL]	$M_z$ [FL]

- **Remark:** The local base vectors can be visualized for control of input data with **forc**,<sup>7</sup>.
- **Remark:** Stress resultants are plotted in the local 1-3 direction. The plot plane can be modified with **forc, $\pm 12$**  or  **$\pm 13$**  (default 13).

### 6.4.12 ELMT12

---

*ityp, kat, sym*  
 $K_1, K_2, \text{angl}1 - x, \rho \cdot h, \rho \cdot c, \rho \cdot g, elev$

---

- **Theory**

**ELMT12** is a general plane and axisymmetric element for use in the analysis of problems modeled by a **Laplace equation**. In general 3–9 nodes are allowed. The element has been written with options which use parameters defined by the linear heat transfer analysis, the linear flow in porous isotropic media and the calculation of shear stresses and warping functions for cross sections.

The element requires a two dimensional mesh and uses the first two coordinate values for x and y (r and z for axisymmetry), respectively. Similarly, the element uses the **first** degree-of-freedom at each node for the temperature or head nodal values. Thus with the macro **disp** for example the nodal **temperature** values are printed.

The element routine will compute any combination of nodes, up to 9, provided all vertex nodes are specified. Triangles may be specified by either giving the same node number for contiguous vertices, or for the 3-node element, by specifying only the first 3 nodes (i.e., node 4 has a zero value).

- **Material input data** Record 1 define the type of problem:

<i>ityp</i>	Problem type, see below
<i>kat</i>	1=plane, 2=axisymmetric (not for ityp=5,6)
<i>sym</i>	Symmetry conditions (only for ityp=5,6)
	(1: to 1-axis, 2: to 2-axis, 3: to 1+2-axis)

Record 2 depends on which problem has been chosen:

- **Linear anisotropic heat transfer** (*ityp* = 1):

$K_1$	Thermal conductivity in 1-direction
$K_2$	Thermal conductivity in 2-direction
$\text{angl}1 - x$	Angle 1-axis makes with x(or r)-axis
$\rho \cdot h$	density · heat source
$\rho \cdot c$	density · specific heat
$\rho \cdot g$	not used
<i>elev</i>	not used

- Flow in a horizontal porous media ( $ityp = 2$ ):

$K_1$	Flow permeability in 1-direction
$K_2$	Flow permeability in 2-direction
$angl1 - x$	Angle 1-axis makes with x(or r)-axis
$\rho \cdot h$	density · flow source
$\rho \cdot c$	density · specific flow
$\rho \cdot g$	not used
$elev$	not used

- Flow in vertically oriented porous media ( $ityp = 3$ ):

$K_1$	Flow permeability in 1-direction
$K_2$	Flow permeability in 2-direction
$angl1 - x$	Angle 1-axis makes with x(or r)-axis
$\rho \cdot h$	density · flow source
$\rho \cdot c$	density · specific flow
$\rho \cdot g$	density · $g$
$elev$	Surface elevation of zero pressure when head = 0

- Simulated free surface in a horizontal porous media ( $ityp = 4$ ):

$K_1$	Flow permeability in 1-direction
$K_2$	Flow permeability in 2-direction
$angl1 - x$	Angle 1-axis makes with x(or r)-axis
$\rho \cdot h$	density · flow source
$\rho \cdot c$	density · specific flow
$\rho \cdot g$	not used
$elev$	not used

- Cross section properties, warping function and shear stresses from  $M_T$  ( $ityp = 5$ ):

Input data only on ksym, see above.

A boundary condition has to be set only at one (unloaded) node.

- Cross section properties, stress function and shear stresses from  $M_T$ , also center of stress function ( $y_0, z_0$ ) ( $ityp = 6$ ):

Input data only on ksym, see above.

For such problems zero boundary conditions have to be defined at the surface of the system. If more than one surface exists all nodes at a certain surface must have the same boundary values. This can be achieved by using the macro **link**.

There exist no external load case.

- warping function and shear stresses from  $Q_y, Q_z$  (*ityp* = 7):

$A$	area of cross section
$I_y$	moment of inertia
$I_z$	moment of inertia
$I_{yz}$	moment of inertia = $+\int yz \, da$
$y_s$	y-coordinate of center of gravity
$z_s$	z-coordinate of center of gravity
$Q_y$	shear force: 0 or 1
$Q_z$	shear force: 1 or 0
$\nu$	Poisson's ratio
$y_0$	y-coordinate of center of stress function(only for $\nu \neq 0$ )
$z_0$	z-coordinate of center of stress function(only for $\nu \neq 0$ ) ( $y_0$ and $z_0$ can be calculated under <i>ityp</i> = 6)

A boundary condition has to be set only at one node.

### • Print of quantities

The output quantities for each option are printed using the basic form or the **stre**, , n1, n2, n3 macro command and are reported as follows:

- Linear anisotropic heat transfer (*ityp* = 1):

$q_1$	Heat flow in 1-direction (1=x in plane, 1=r in axisym)
$q_2$	Heat flow in 1-direction (2=y in plane, 2=z in axisym)
$ q $	Maximum heat flow ( $\sqrt{(q_1^2 + q_2^2)}$ )
$\Theta$	Temperature at element output point

- Flow in a horizontal porous media (*ityp* = 2):

$q_1$	Fluid flow in 1-direction (1=x in plane, 1=r in axisym)
$q_2$	fluid flow in 1-direction (2=y in plane, 2=z in axisym)
$ q $	Maximum fluid flow ( $\sqrt{(q_1^2 + q_2^2)}$ )
$p$	Pressure at element output point

- Flow in a vertically oriented porous media ( $ityp = 3$ ):

$q_1$	Fluid flow in 1-direction (1=x in plane, 1=r in axisym)
$q_2$	fluid flow in 1-direction (2=y in plane, 2=z in axisym)
$ q $	Maximum fluid flow ( $\sqrt{(q_1^2 + q_2^2)}$ )
$p$	Pressure at element output point ( $p = head - rho^*(y - y_0)$ )

- Simulated free surface in a horizontal porous media( $ityp = 4$ ):

$q_1$	Fluid flow in 1-direction (1=x in plane, 1=r in axisym)
$q_2$	fluid flow in 1-direction (2=y in plane, 2=z in axisym)
$ q $	Maximum fluid flow ( $\sqrt{(q_1^2 + q_2^2)}$ )
$h$	Head at element output point ( $h = \sqrt{2 * u - 1}$ ) (N.B. u-1 is the dependent nodal variable used in the analysis.)

- Cross section properties, warping function and shear stresses from  $M_T$  ( $ityp = 5$ ):

The first call of the macro **stre**,all leads to the cross sectional values  $A$ ,  $I_{11}$ ,  $I_{22}$ ,  $I_{12}$  for the reference axis, for parallel axis and main axis through the center of gravity and the coordinates of the center of gravity.

The second call of the macro **stre**,all leads to the coordinates of the center of shear.

Finally the third call of the macro **stre**,all leads to the cross sectional values  $I_T$  and  $C_M$ .

After these three calls has been done (for simplicity define a procedure [ **tang**,,1 ; **loop**,,3 ; **stre**,all ; **next**] ) shear stresses are available using the standard call **stre**,,n1,n2,n3. Note that plots of all values are only available after the above mentioned three calls.

Values of the warping functions can be printed with the macro **stre**,node,n1,n2,n3.

- Cross section properties, stress function and shear stresses from  $M_T$  ( $ityp = 6$ ):

see  $ityp = 5$

- Warping function and shear stresses from  $Q_y, Q_z$  ( $ityp = 7$ ):

The macro **stre**,all has to be used 3 times to get all results.

After these three calls has been done (for simplicity define a procedure [ **tang**,,1 ; **loop**,,3 ; **stre**,all ; **next**] ) shear stresses are available using the standard call **stre**,,n1,n2,n3. Note that plots of all values are only available after the above mentioned three calls.

Values of the warping functions can be printed with the macro **stre**,node,n1,n2,n3.

- Plot of quantities

Basic quantities are plotted via the macro **disp** due to following numbers

<b>disp</b> number	1
<b>disp</b> (ityp = 1)	temperature
<b>disp</b> (ityp = 2)	pressure
<b>disp</b> (ityp = 3)	pressure
<b>disp</b> (ityp = 4)	pressure
<b>disp</b> (ityp = 5)	warping function $w$ from $M_T$ (not $w_Q, w_T$ !)
<b>disp</b> (ityp = 6)	stress function $\Phi$
<b>disp</b> (ityp = 7)	warping function $w$ from $Q_y, Q_z$

All quantities are plotted via the macro **stre** due to following numbers

'Stress' number	1	2	3	4	5
'stress' (ityp = 1)	$q_1$	$q_2$	$ q $	$\Theta$	
'stress' (ityp = 2)	$q_1$	$q_2$	$ q $	$p$	
'stress' (ityp = 3)	$q_1$	$q_2$	$ q $	$p$	
'stress' (ityp = 4)	$q_1$	$q_2$	$ q $	$h$	
'stress' (ityp = 5)	$T_{13}$	$T_{23}$	$ T $	$w_q$	$w_t$
'stress' (ityp = 6)	$T_{13}$	$T_{23}$	$ T $	$\Phi$	
'stress' (ityp = 7)	$T_{13}$	$T_{23}$	$ T $	$w_q$	

Remark: Note that for ityp=5,6,7 plots of all values are only available after calling three times **stre,all** in the macro mode.

The resultant flows (stresses) can be plotted with arrows using the macro **flux**.

### 6.4.13 ELMT13

$E, \nu, h, q, \rho, lin$

- Theory

ELMT13 is a general 4/8/9 - node thin shallow shell element based on the Reissner–Mindlin Theory with selective reduced integration (SRI). Be careful, this element may lead to hourglass-modes. The element formulation is based on the Green Lagrangian strain tensor and the 2. Piola-Kirchhoff stress tensor. No transformation to local coordinates has to be done. Thus, all results are stated in the 1-2 plane.

- Material input data      Record 1

$E[F/L^2]$	Modulus of elasticity
$\nu$	Poisson ratio
$h[L]$	Plate thickness
$q[F/L^2]$	Uniform normal load in z-dir.
$\rho[F/L^3]/[L/T^2]$	density : $\rho = \gamma/g$
$lin$	linear/nonlinear = 0/1

- Loads via macro **qloa**.

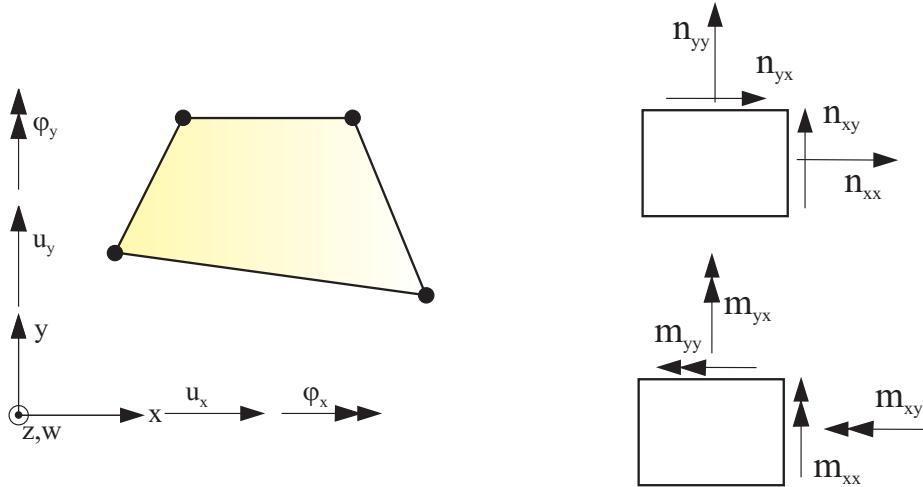
$ma$	$[-]$	Material number in Inputfile
$q$	$[F/L^2]$	uniform normal load in z-direction

- Be sure to have the same units for all input values!

- Definition of stress resultants

$$\begin{aligned} n_{xx} &= \int \sigma_{xx} dz, \quad n_{yy} = \int \sigma_{yy} dz, \quad n_{xy} = \int \sigma_{xy} dz = n_{yx} \\ m_{xx} &= \int \sigma_{xx} z dz, \quad m_{yy} = \int \sigma_{yy} z dz, \quad m_{xy} = \int \sigma_{xy} z dz = m_{yx} \\ q_{xz} &= \int \tau_{xz} dz, \quad q_{yz} = \int \tau_{yz} dz \end{aligned}$$

- Displacements ( $u_x, u_y, w, \varphi_x, \varphi_y$ ) and stress resultants per length



Shear forces show on positive side in positive direction.

- **Output of moments/shear forces**

The normal forces/moment/shear forces per unit length are calculated at each center of element in global directions x and y. The moments  $m_1$  and  $m_2$  are the principal moments with the angle  $\varphi_1$  between 1-moment and x-coordinate direction (in degrees).

- **Plot of moments/shear forces**

The normal forces/moment/shear forces per unit length are plotted due to following numbers

Stress number	Type	value plotted	Dimension
1	Force	$n_{xx}$	[F/L]
2	Force	$n_{yy}$	[F/L]
3	Force	$n_{xy}$	[F/L]
4	Moment	$m_{xx}$	[FL/L]
5	Moment	$m_{yy}$	[FL/L]
6	Moment	$m_{xy}$	[FL/L]
7	Shear force	$q_{xz}$	[F/L]
8	Shear force	$q_{yz}$	[F/L]
9	Moment	$m_1$	[FL/L]
10	Moment	$m_2$	[FL/L]
11	thickness	$h$	[L]

### 6.4.14 ELMT14

*idof, pen, tol*

- **Theory**

ELMT14 is a 2D node-to-node contact element. Contact is possible in one direction. The gap is defined by

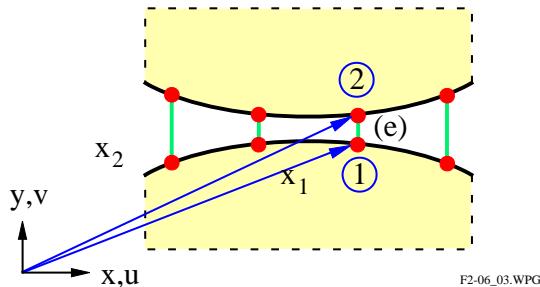
$$(x_2 + u_2) - (x_1 + u_1) > 0$$

where  $x$  is the chosen direction. A penalty and an augmented lagrange (**augm**) formulation is available.

- **Material input data**

idof	coordinate direction for contact
ipen	penalty value for imposing contact constraint
tol	tolerance on gap (def.= 0.0)

- **Displacements and normal forces**



- **Output of results**

Contact force  $N$ .

- **Plot of results**

Results are plotted due to following numbers

Stress number	1
Quantity plotted	$N [F]$

### 6.4.15 ELMT15

$E, G, A, I_y, I_z, I_{yz}, I_T, , y_s, z_s, y_m, z_m, \alpha$   
 $iGeo, iPlo, iscf, \rho$

- **Theory**

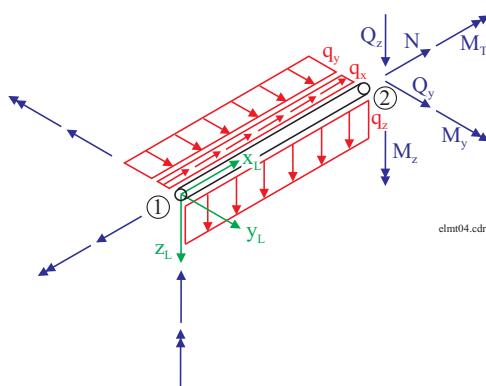
ELMT15 is a 2–5 node geometrically nonlinear 3D-Timoshenko–Beam Element with Reissner strains and finite rotations.

$E$	$[F/L^2]$	Elastic modulus
$G$	$[F/L^2]$	Shear modulus
$A$	$[L^2]$	Area
$I_y$	$[L^4]$	Moment of Inertia wrt. y-axis
$I_z$	$[L^4]$	Moment of Inertia wrt. z-axis
$I_{yz}$	$[L^4]$	Moment of Inertia wrt. yz-axis
$I_T$	$[L^4]$	Moment of Inertia wrt. x-axis
$y_s$	$[L]$	y-coordinate of center of gravity
$z_s$	$[L]$	z-coordinate of center of gravity
$y_m$	$[L]$	y-coordinate of center of shear
$z_m$	$[L]$	z-coordinate of center of shear
$\alpha$	$[DEG.]$	Rotation angle for local coordinate system
iGeo		geometrically nonlinearity 0 = linear, 1 = moderate 2 = finite, 3 = finite $> 360^\circ$
iPlo		0/1 Stress resultants related to reference(0)/current(1) configuration
iscf		0/1 FE shear correction off(0)/on(1)
$\rho$	$[F/L^3]/[L/T^2]$	density $\rho = \gamma/g$

$ma$	$[-]$	Material number in Inputfile
$y_p$	$[L]$	y-coordinate of load
$z_p$	$[L]$	z-coordinate of load
$q_X$	$[F/L]$	load in global X-direction
$q_Y$	$[F/L]$	load in global Y-direction
$q_Z$	$[F/L]$	load in global Z-direction

- **Loads via macro qloa.**

- **Displacements and stress resultants**



- **Output of stress resultants**

Stress resultants ( $N_1$ —Axial Force,  $Q_2, Q_3$ —Shear Forces,  $M_1$  Torsional Moment,  $M_2, M_3$ —Bending Moments) are calculated at left and right node of the element with respect to iPlo=0/1 related to reference(0) or current(1) configuration.

- **Plot of stress resultants**

The above defined stress resultants are plotted due to following numbers

Stress number	1	2	3	4	5	6
Stress resultant	$N_1 [F]$	$Q_2 [F]$	$Q_3 [F]$	$M_1 [FL]$	$M_2 [FL]$	$M_3 [FL]$

### 6.4.16 ELMT16

$$\begin{aligned} E, A, \rho, \alpha_t, s_0/\ell_0 \\ S_0, q_x, q_y, q_z, \Delta t \end{aligned}$$

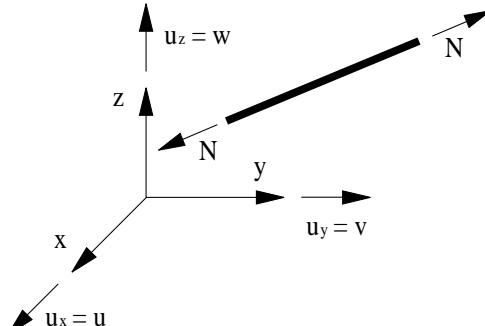
- **Theory**

ELMT16 is a general 3D cable element.

- **Material input data**

$E$	$[F/L^2]$	Young's modulus
$A$	$[L^2]$	Area
$\gamma$	$[F/L^3]$	specific weight for dead load in global -z direction
$\alpha_t$	$[1/K]$	coefficient of thermal expansion
$s_0/\ell_0$	$[-]$	cable length/chord length ( $\ell_0 = \mathbf{X}_2 - \mathbf{X}_1$ )
$S_0$	$[F]$	prestress force in chord dir.
$q_x$	$[F/L]$	load in global x-direction
$q_y$	$[F/L]$	load in global y-direction
$q_z$	$[F/L]$	load in global z-direction
$\Delta T$	$[K]$	temperature difference

- **Displacements and normal forces**



- **Output of results**

Cable forces are calculated for each element in the local axial direction at beginning and end of element as well as for the chord direction.

- **Plot of results**

Results are plotted due to following numbers

Force number	1
Cable Force	$S [F]$

Remark: Cable forces are plotted in the local 1-3 direction. The plot plane can be modified with **forc**,±12 or ±13 (default 13).

- **Remarks**

- Each cable can be modelled with one element.
- In case of large deflections more than one element should be used. The position of all nodal points is on the chord-line! 4 elements are recommended.
- The problem to find the right cable position is nonlinear. Thus an iteration is necessary!
- The deformed configuration of the cable could only be seen in case of more than one element describing the cable!

### 6.4.17 ELMT17

$E, \nu, h, q, \rho$

- **Theory**

[ELMT17](#) is a general 4 - node thin plate bending element based on the Reissner–Mindlin Theory with special assumptions for the shear terms (Bathe/Dvorkin approach).

- **Material input data**

$E[F/L^2]$	Modulus of elasticity
$\nu$	Poisson ratio
$h[L]$	Plate thickness
$q[F/L^2]$	Uniform normal load in z-dir.
$\rho[F/L^3]/[L/T^2]$	density : $\rho = \gamma/g$
$SCF$	Shear correction factor (def=5/6)

- **Loads via macro [qloa](#).**

$ma$	$[ - ]$	Material number in Inputfile
$q$	$[F/L^2]$	uniform normal load in z-direction

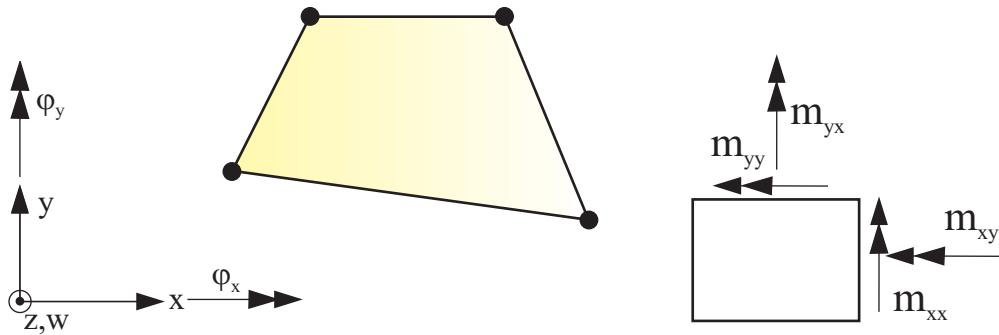
- Be sure to have the same units for all input values!

- Definition of stress resultants

$$m_{xx} = \int \sigma_{xx} z dz, \quad m_{yy} = \int \sigma_{yy} z dz, \quad m_{xy} = \int \sigma_{xy} z dz = m_{yx}$$

$$q_{xz} = \int \tau_{xz} dz, \quad q_{yz} = \int \tau_{yz} dz$$

- **Displacements ( $w, \varphi_x, \varphi_y$ ) and moments per length**



ELMT17.CDR

Shear forces show on positive side in positive direction.

- **Output of moments/shear forces**

The moments/shear forces per unit length are calculated at each center of element in global directions x and y. The moments  $m_1$  and  $m_2$  are the principal moments with the angle  $\varphi_1$  between 1-moment and x-coordinate direction (in degrees).

- **Plot of moments/shear forces**

The moments/shear forces per unit length are plotted due to following numbers

Stress number	Type	value plotted	Dimension
1	Moment	$m_{xx}$	[ $FL/L$ ]
2	Moment	$m_{xy}$	[ $FL/L$ ]
3	Moment	$m_{yy}$	[ $FL/L$ ]
5	Moment	$m_1$	[ $FL/L$ ]
6	Moment	$m_2$	[ $FL/L$ ]
7	Angle	$\varphi_1$	[rad]
8	Shear force	$q_{xz}$	[ $F/L$ ]
9	Shear force	$q_{yz}$	[ $F/L$ ]

Element-wise constant values can be plotted with macro **str1**.

- **Shear correction factor  $\kappa$**

input value SCF	$\kappa$
$SCF < 0$	$\kappa = SCF / (1 + \frac{1}{2}(\frac{1}{1+\nu})\frac{L^2}{h^2})$ with $L = \max(L_x, L_y)$ , see Tessler, Hughes
$SCF = 0$	$\kappa = \frac{5}{6}$
$SCF > 0$	$\kappa = SCF$

### 6.4.18 ELMT18

$E, \nu, h, q, \rho, nb, ns, \kappa$

- **Theory**

ELMT18 is a general 4/8/9 - node thin/thick plate bending element based on the Reissner–Mindlin Theory with selective reduced integration (SRI). Be careful, this element may lead to hourglass-modes for thin plates using a 2/1 or 3/2 integration.

- Definition: thin plate:  $L/h > 10$ , thick plate:  $L/h \leq 10$
- Integration: thin plate 2/1 or 3/2, thick plate 2/2 or 3/3

- **Material input data**

$E[F/L^2]$	Modulus of elasticity
$\nu$	Poisson ratio
$h[L]$	Plate thickness
$q[F/L^2]$	Uniform normal load in z-dir.
$\rho[F/L^3]/[L/T^2]$	density : $\rho = \gamma/g$
$nb$	No. of Gauss points bending(def=2)
$ns$	No. of Gauss points shear(def=1)
$SCF$	Shear correction factor (def=5/6)

- **Loads via macro `qloa`.**

$ma$	$[-]$	Material number in Inputfile
$q$	$[F/L^2]$	uniform normal load in z-direction

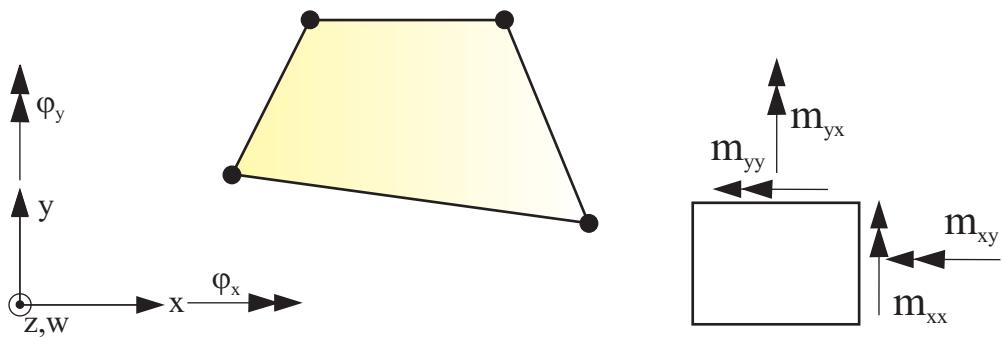
- Be sure to have the same units for all input values!

- Definition of stress resultants

$$m_{xx} = \int \sigma_{xx} z dz, \quad m_{yy} = \int \sigma_{yy} z dz, \quad m_{xy} = \int \sigma_{xy} z dz = m_{yx}$$

$$q_{xz} = \int \tau_{xz} dz, \quad q_{yz} = \int \tau_{yz} dz$$

- **Displacements ( $w, \varphi_x, \varphi_y$ ) and moments per length**



ELMT17.CDR

Shear forces show on positive side in positive direction.

- **Output of moments/shear forces**

The moments/shear forces per unit length are calculated at each center of element in global directions x and y. The moments  $m_1$  and  $m_2$  are the principal moments with the angle  $\varphi_1$  between 1-moment and x-coordinate direction (in degrees).

- **Plot of moments/shear forces**

The moments/shear forces per unit length are plotted due to following numbers

Stress number	Type	value plotted	Dimension
1	Moment	$m_{xx}$	[ $FL/L$ ]
2	Moment	$m_{xy}$	[ $FL/L$ ]
3	Moment	$m_{yy}$	[ $FL/L$ ]
5	Moment	$m_1$	[ $FL/L$ ]
6	Moment	$m_2$	[ $FL/L$ ]
7	Angle	$\varphi_1$	[rad]
8	Shear force	$q_{xz}$	[ $F/L$ ]
9	Shear force	$q_{yz}$	[ $F/L$ ]

Element-wise constant values can be plotted with macro **str1**.

- **Shear correction factor  $\kappa$**

input value SCF	$\kappa$
$SCF < 0$	$\kappa = SCF / (1 + \frac{1}{2}(\frac{1}{1+\nu})\frac{L^2}{h^2})$ with $L = \max(L_x, L_y)$ , see Tessler, Hughes
$SCF = 0$	$\kappa = \frac{5}{6}$
$SCF > 0$	$\kappa = SCF$

### 6.4.19 ELMT19

$E, \nu, h, q$

- **Theory**

ELMT19 is a general 3 - node thin plate bending element based on the Discrete Kirchhoff Theory, compare ELMT07 for a 4 - node element.

- **Material input data**

$E[F/L^2]$	Modulus of elasticity
$\nu$	Poisson ratio
$h[L]$	Plate thickness
$q[F/L^2]$	Uniform normal load in z-dir.

- **Loads via macro **qloa**.**

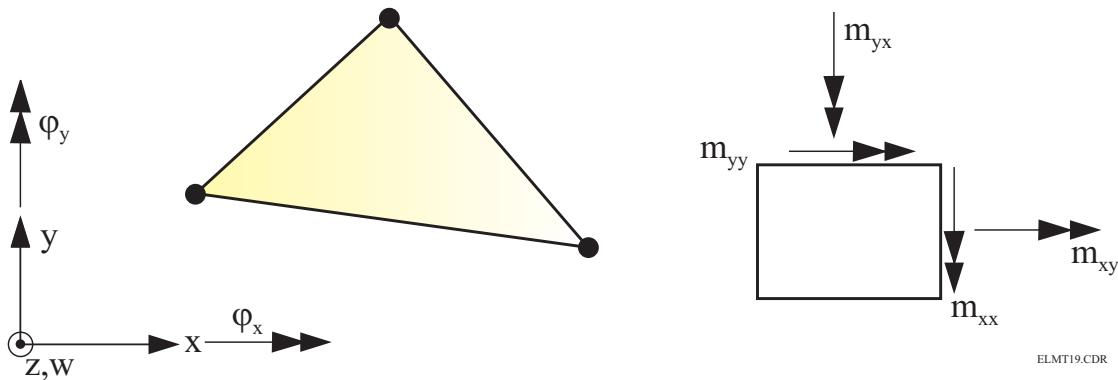
$ma$	$[-]$	Material number in Inputfile
$q$	$[F/L^2]$	uniform normal load in z-direction

- Be sure to have the same units for all input values!

- Definition of stress resultants

$$m_{xx} = - \int \sigma_{xx} z dz, \quad m_{yy} = - \int \sigma_{yy} z dz, \quad m_{xy} = - \int \sigma_{xy} z dz = m_{yx}$$

- **Displacements ( $w, \varphi_x, \varphi_y$ ) and moments per length**



- **Output of moments**

The moments per unit length are calculated at each center of element in global directions x and y. The moments  $m_1$  and  $m_2$  are the principal moments with the angle  $\varphi_1$  between 1-moment and x-coordinate direction (in degrees).

**• Plot of moments**

The moments per unit length are plotted due to following numbers

Stress number	Type	value plotted	Dimension
1	Moment	$m_{xx}$	[ $FL/L$ ]
2	Moment	$m_{xy}$	[ $FL/L$ ]
3	Moment	$m_{yy}$	[ $FL/L$ ]
5	Moment	$m_1$	[ $FL/L$ ]
6	Moment	$m_2$	[ $FL/L$ ]
7	Angle	$\varphi_1$	[rad]

### 6.4.20 ELMT20

$E, G, I_T, I_y, q_1, q_2$

- Theory

ELMT20 is a 2-node Bernoulli/St.Venant Grid–Beam Element.

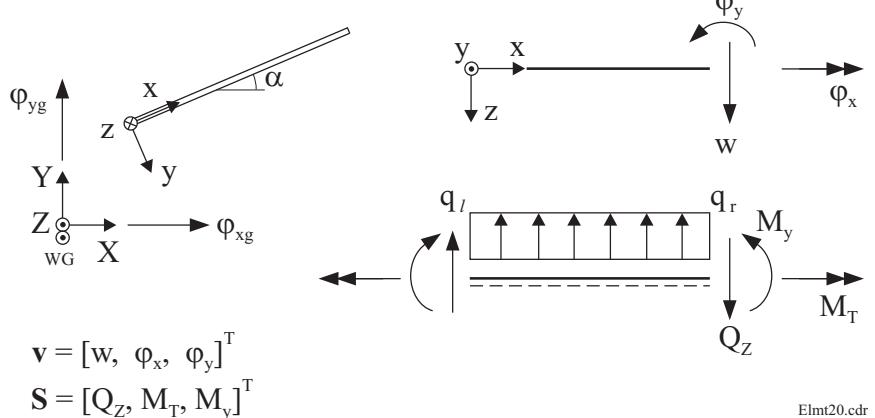
- Material input data

$E[F/L^2]$	Elastic modulus
$G[F/L^2]$	Shear modulus
$I_T[L^4]$	Torsional rigidity
$I_y[L^4]$	Moment of Inertia
$q_1[F/L]$	load in z-dir. at left node
$q_2[F/L]$	load in z-dir. at right node

- Loads via macro **qloa**.

$ma$ [–]	Material number in Inputfile
$q_1$ [ $F/L$ ]	load in z-dir. at left node
$q_2$ [ $F/L$ ]	load in z-dir. at right node

- Displacements and stress resultants



- Output of stress resultants

Stress resultants ( $Q_z$ —Shear Force,  $M_T$ —Bending Moment,  $M_y$ —Bending Moment) are calculated at left and right node of the element in local directions.

- Plot of stress resultants

The above defined stress resultants are plotted due to following numbers

Stress number	1	2	3
Stress resultant	$Q_z$ [ $F$ ]	$M_T$ [ $F \cdot L$ ]	$M_T$ [ $F \cdot L$ ]

### 6.4.21 ELMT21

$matn, <lin>, <isym>, <istr>$   
 $qx, qy, qz, \Delta T, \alpha_t, \rho$   
 $E, \nu$  (for  $matn=1$ )

- **Theory**

**ELMT21** is a 4/8/20/21/27/64-node geometrical linear/nonlinear 3D-Solid Element. The definition of the nodes is similar to the macro **bloc**. The 20 node element can be used with  $nen=20$  or  $nen=21$ . In the latter case an additional unused node is generated. Hidden line plots are only available for the 8 node element. Different material models are implemented and can be used with the **3D-Material library**.

- **Material input data**

$matn$	[ $-$ ]	material number
$lin$	[ $-$ ]	0: geometrical linear, 1: geometrical nonlinear
$isym$	[ $-$ ]	0: symmetric, 1: nonsymmetric
$istr$	[ $-$ ]	stresses( $lin=0$ ): 1<0>=linear stresses( $lin=1$ ): 1<0>=2.P.K, 2=Cauchy
$qx$	[ $F/L^3$ ]	load in global x-direction
$qy$	[ $F/L^3$ ]	load in global y-direction
$qz$	[ $F/L^3$ ]	load in global z-direction
$\Delta T$	[ $K$ ]	temperature difference
$\alpha_t$	[ $1/K$ ]	coefficient of thermal expansion
$\rho$	[ $F/L^3$ ]	specific weight
Material		input data, see the <b>3D-Material library</b>

Details on the derivation of the material models can be found in the **Theory Manual**.

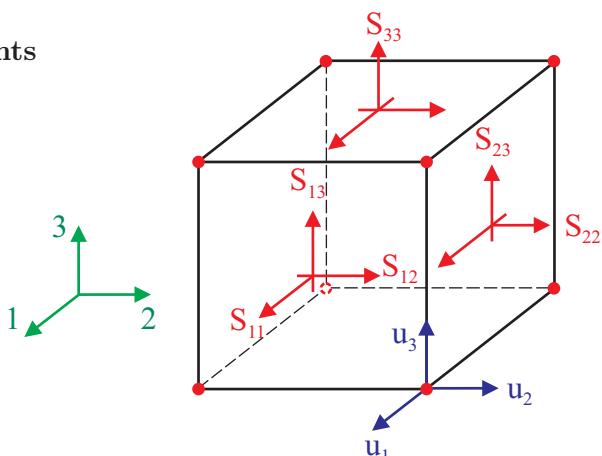
Details on the implementation of elements can be found in the **Manual for adding elements**.

- **Loads via macro **qloa**.**

$ma$	[ $-$ ]	Material number in Inputfile
$qx$	[ $F/L$ ]	load in global x-direction
$qy$	[ $F/L$ ]	load in global y-direction
$qz$	[ $F/L$ ]	load in global z-direction
$\Delta T$	[ $K$ ]	temperature difference

- **Displacements and stress/strain resultants**

$$\mathbf{S} = \begin{bmatrix} S_{11} \\ S_{22} \\ S_{33} \\ S_{12} \\ S_{13} \\ S_{23} \end{bmatrix} \quad \mathbf{E} = \begin{bmatrix} E_{11} \\ E_{22} \\ E_{33} \\ 2E_{12} \\ 2E_{13} \\ 2E_{23} \end{bmatrix} \quad \mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$$



- **Output of stresses**

Normal and shear stresses are calculated at the Gauss points of the element in global directions.

- **Plot of stresses**

The above defined stresses are plotted due to following numbers

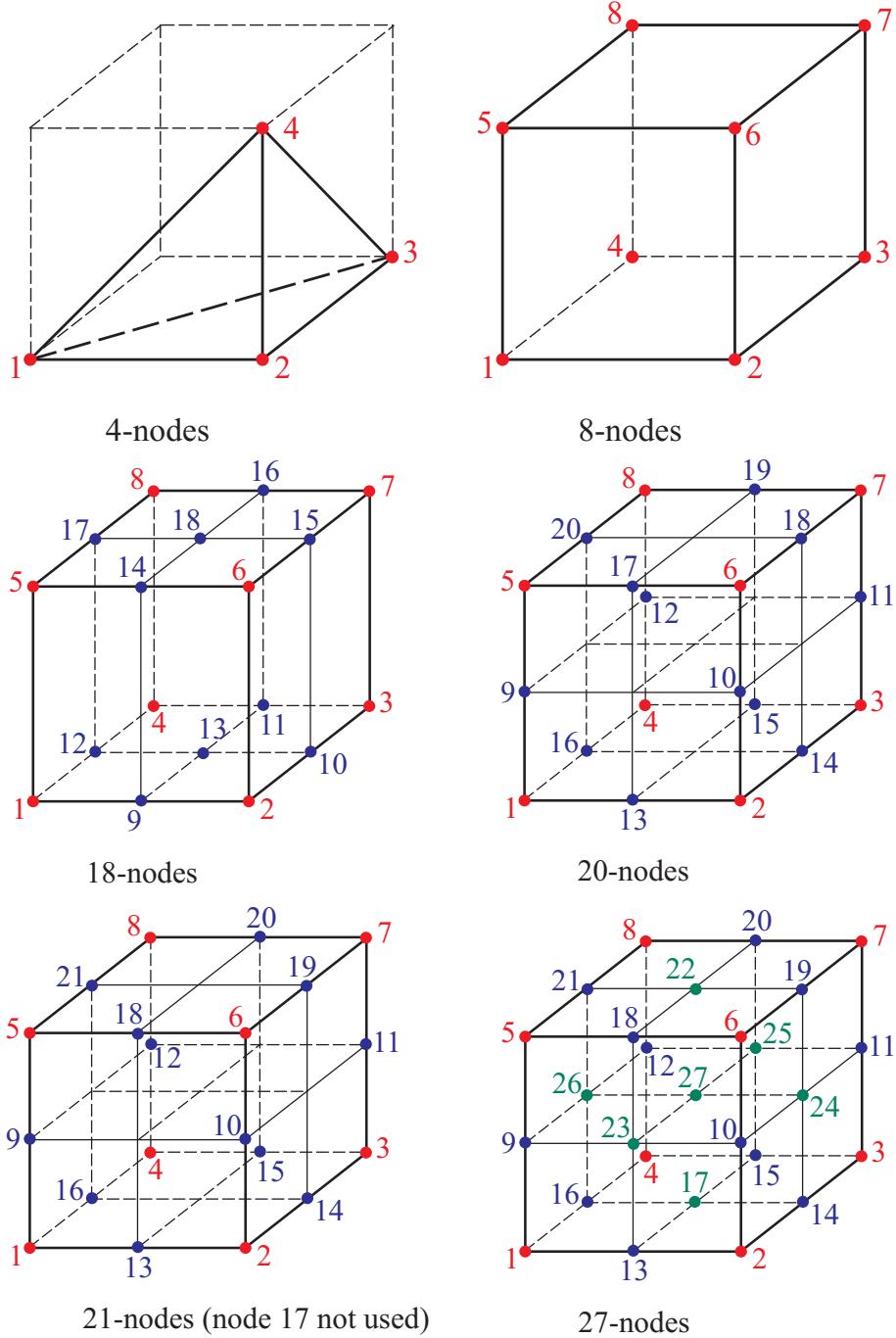
Stress number	1	2	3	4	5	6	7	8	9
Stress plotted [F/L <sup>2</sup> ]	$S_{11}$	$S_{22}$	$S_{33}$	$S_{12}$	$S_{13}$	$S_{23}$	$S_1$	$S_2$	$S_3$

- Plot problems with tetraeder

Macros `disp`, `stre` and similar macros do not work together with `hide`. Thus, use other post processing programs, e.g. TECPLOT for plotting of results. An associated interface to TECPLOT is implemented (`tec`).

- Element nodes

(see macro `bloc`)



## 6.4.22 ELMT22

$E, \nu, h, \alpha_T, \Delta T, lin, Y_0, c_p$

- **Theory**

**ELMT22** is a general 4/8/9 node plane stress element. The 4-node version is a quadrilateral with linear shape functions, whereas the 9-node element has quadratical shape functions. The 8-node element is of Serendipity-type. A geometrical nonlinear option and a material nonlinear option ( $J_2$ -plasticity with isotropic hardening) are available.

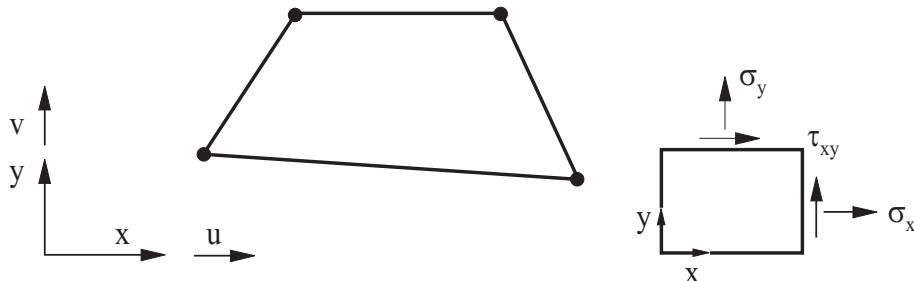
- **Material input data**

$E[F/L^2]$	Young's modulus
$\nu$	Poisson ratio
$h[L]$	thickness of plane stress slice
$\alpha_T[1/T]$	coefficient of linear thermal expansion
$\Delta T[T]$	temperature difference for this material
$lin$	0=linear, 1=nonlinear
$Y_0$	initial yield stress
$c_p$	Hardening parameter

- **Loads via macro `qloa`.**

$ma$	$[-]$	Material number in Inputfile
$b_1$	$[F/L]$	body force value (constant) for 1(x)-direction
$b_2$	$[F/L]$	body force value (constant) for 2(y)-direction

- **Displacements ( $u,v$ ) and stresses ( $S=\sigma, P$ )**



- **Output of stresses and strains**

Stresses and strains are calculated at each quadrature point in global directions 1(x) and 2(y). The stresses  $S_{\alpha\beta}$  and  $P_{\alpha\beta}$  are the linear stresses in case of  $lin = 0$ . For nonlinear problems  $S_{\alpha\beta}$  denotes the 2.Piola-Kirchhoff stresses and  $P_{\alpha\beta}$  the 1.Piola-Kirchhoff stresses.

- **Plot of stresses**

Stresses ( $S, P$ ) and plastic strains  $E_p$  are plotted due to following numbers

Stress number	1	2	3	4	5	6	7	8	9	10	11	12	13
Stress plotted	$S_{xx}$	$S_{yy}$	$S_{xy}$	$E_{pxx}$	$E_{pyy}$	$E_{pxy}$	$E_{pq}$	$Y/Y_0$	$\Delta\lambda_0$	$P_{xx}$	$P_{yy}$	$P_{xy}$	$P_{yx}$

Plot-results for the 8-node serendipity-element could not be used due negative weighting functions.

### 6.4.23 ELM023

$E, \nu, h, q$

- **Theory**

ELM023 is a general 4 - node thin plate bending element based on the Discrete Kirchhoff Theory. This is an improved version of element 7.

- **Material input data**

Record 1

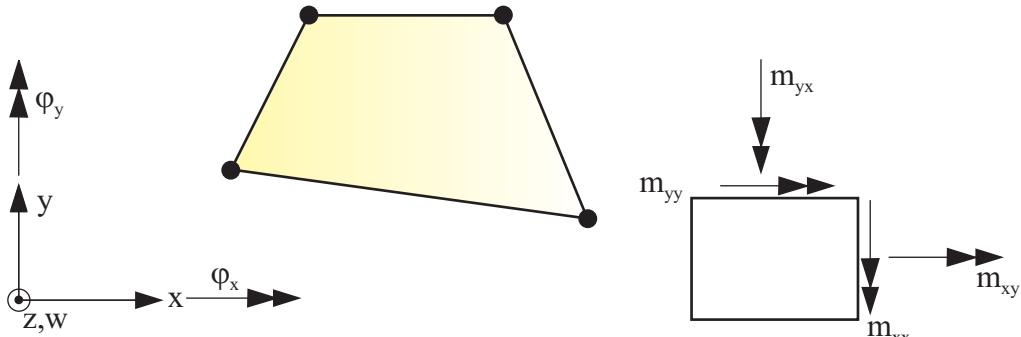
$E[F/L^2]$	Modulus of elasticity
$\nu$	Poisson ratio
$h[L]$	Plate thickness
$q[F/L^2]$	Uniform normal load in z-dir.

- Be sure to have the same units for all input values!

- Definition of stress resultants

$$m_{xx} = -\int \sigma_{xx} z dz, \quad m_{yy} = -\int \sigma_{yy} z dz, \quad m_{xy} = -\int \sigma_{xy} z dz = m_{yx}$$

- **Displacements ( $w, \varphi_x, \varphi_y$ ) and moments per length**



ELM023.CDR

- **Output of moments**

The moments per unit length are calculated at each center of element in global directions x and y. The moments  $m_1$  and  $m_2$  are the principal moments with the angle  $\varphi_1$  between 1-moment and x-coordinate direction (in degrees).

- **Plot of moments**

The moments per unit length, the foundation pressure and the reinforcements are plotted due to following numbers

Stress number	Type	value plotted	Dimension
1	Moment	$m_{xx}$	$[FL/L]$
2	Moment	$m_{xy}$	$[FL/L]$
3	Moment	$m_{yy}$	$[FL/L]$

### 6.4.24 ELMT24

---

$E, G, b, Q_y, Q_z, ns, nsp, ityp, M_{T0}, ic$

---

- Theory

**ELMT24** is a 2-node Element for shear stresses from shear forces or torsional moments in thin walled structures with different materials. All cross section values are calculated.

For  $ityp=1$ (def) shear stresses from  $Q$  are calculated. Here, the center of shear can be found with two calculations:

1 :  $y_M$  with  $Q_y = 0$  and  $Q_z = 1$

2 :  $z_M$  with  $Q_y = 1$  and  $Q_z = 0$

For  $ityp=2$  shear stresses for  $M_T$  are calculated for open or closed sections. Here the calculation of center of shear is done automatically. The total torsional moment is

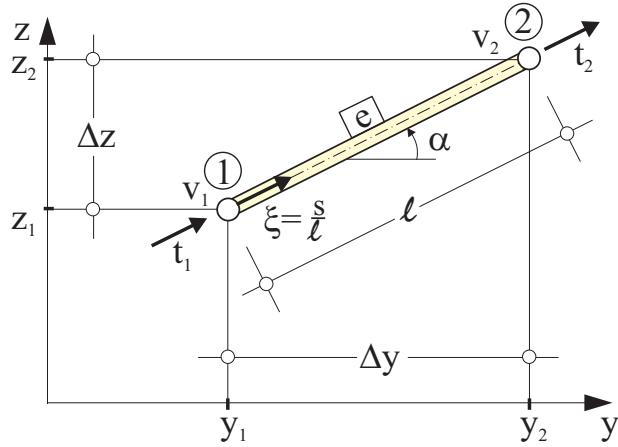
$$M_T = M_{T0} + Q_y \cdot z_{qm} - Q_z \cdot y_{qm} \quad (Q_y, Q_z \text{ in } S)$$

For  $ic=0$  only the closed part of the cross section is investigated (Bredt' shear flux). For  $ic=1$  the whole cross section is used. Thus pure open and mixed cross sections can be calculated. Note that in closed parts more than pure Bredt' shear flux occur. In this case different stresses are calculated: Values at the border for open cross sections(linear over thickness) and constant values for closed parts (constant over thickness).

- Material input data

$E_i[F/L^2]$	Elastic modulus
$G_i[F/L^2]$	Shear modulus
$b_i[L]$	thickness
$Q_y[F]$	Shear force
$Q_z[F]$	Shear force
$ns[-]$	Number of points for STRE (def.=4)
$nsp[-]$	Number of points for PLOT (def.=10)
$ityp[-]$	Typ: 1= Shear force, 2= Torsional moment
$M_{T0}[KL]$	Torsional moment
$ic[-]$	Typ: 0= only closed cross section, 1= open or mixed cross section

- Displacements and stress resultants



- Output of stress resultants

Stress resultants (shear stress  $\tau$ , shear flux  $t$  and main warping function  $\varphi_q$ ) are calculated exactly at all points.

- Plot of stress resultants

The above defined stress resultants are plotted due to following numbers

Stress number	1	2	3
Stress resultant	$\tau [F/L^2]$	$t [F/L]$	$\varphi_q [-]$

- Necessary calculation procedure

- chec** calculate cross section values
- tang,,1** solve equations
- stre,all** calculate further cross section values
- stre,all** print stresses

#### 6.4.25 ELMT25

### 6.4.26 ELMT26

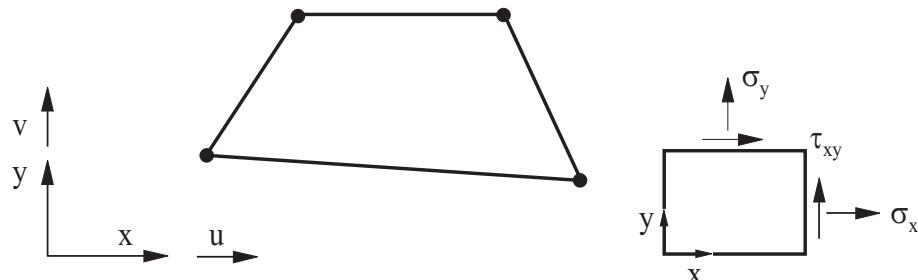
*imat*  
*ityp*, *II*,  $\rho$ , *K*, *G*

- Theory **ELMT26** is a 4 node plane strain/axisym element. It is called  $\bar{\mathbf{B}}$ -element and is based on a mixed formulation.

- Material input data**

<i>imat</i>	[ <i> </i> ]	1= isotropic linear elastic material
<i>ityp</i>	[ <i> </i> ]	0: plane strain, 1: axisymmetric
<i>II</i>	[ <i> </i> ]	0: mixed hybrid approach, 1: displacement based approach
$\rho[F/L^3]/[L/T^2]$ density $\rho = \gamma/g$		
<i>K</i> [ <i>F/L<sup>2</sup></i> ]		compression modulus $K = \frac{E}{3(1 - 2\nu)}$
<i>G</i> [ <i>F/L<sup>2</sup></i> ]		shear modulus $G = \frac{E}{2(1 + \nu)}$

- Displacements (*u,v*) and stresses**



- Output of stresses**

Stresses are calculated at the element center in global directions 1(x) and 2(y). ??????????????

- Plot of stresses**

Stresses are plotted due to following numbers

Stress number	1	2	3	4	5
Stress plotted	$\sigma_{xx}$	$\sigma_{xy}$	$\sigma_{yy}$	$\sigma_{zz}$	$\sigma_{????}$

**6.4.27 ELMT27**

**6.4.28 ELMT28**

**6.4.29 ELMT29**

### 6.4.30 ELMT30

---

---

- **Theory**

ELMT30 is a general 4/9 - node flat element to set surface loads e.g. on solid elements in arbitrary directions. Furthermore displacement depending (follower) loads are possible.

- **Material input data**

---

- **Loads via macro `qloa`.**

$ma$	$[ - ]$	Material number in Input file
$q_x$	$[F/L^2]$	uniform normal load in x-direction
$q_y$	$[F/L^2]$	uniform normal load in y-direction
$q_z$	$[F/L^2]$	uniform normal load in z-direction
$ltyp$	$[ - ]$	0=loads in global directions 1=loads in local directions 2=follower loads (only local $q_z$ )

**6.4.31 ELMT31**

**6.4.32 ELMT32**

### 6.4.33 ELMT33

$E, \nu, h, q, \rho, r_w, r_\beta$

- **Theory**

ELMT33 is a general 4 - node thin plate bending element based on the Reissner–Mindlin Theory with uniform reduced integration (URI) and stabilization matrix (Belytschko, Tsay: IJNME 19(83)405-419). Be careful, this element may lead to hourglass-modes for thin plates.

- **Material input data**

$E[F/L^2]$	Modulus of elasticity
$\nu$	Poisson ratio
$h[L]$	Plate thickness
$q[F/L^2]$	Uniform normal load in z-dir.
$\rho[F/L^3]/[L/T^2]$	density : $\rho = \gamma/g$
$r_w$	stabilization parameter for w-terms
$r_\beta$	stabilization parameter for $\beta$ -terms

- **Loads via macro `qloa`.**

$ma$	$[ - ]$	Material number in Inputfile
$q$	$[F/L^2]$	uniform normal load in z-direction

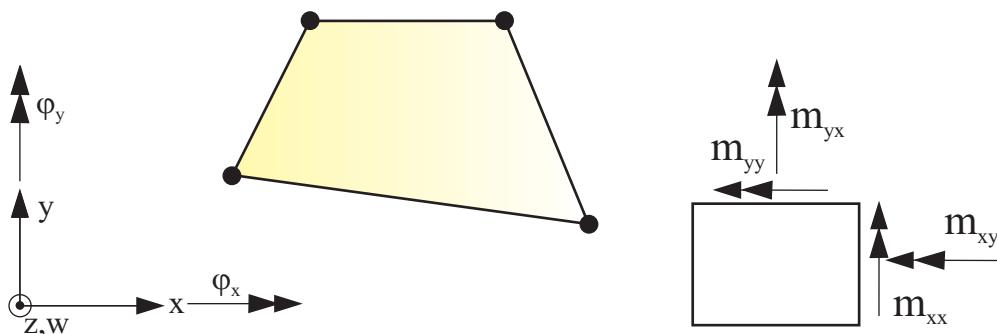
- Be sure to have the same units for all input values!

- Definition of stress resultants

$$m_{xx} = \int \sigma_{xx} z dz, \quad m_{yy} = \int \sigma_{yy} z dz, \quad m_{xy} = \int \sigma_{xy} z dz = m_{yx}$$

$$q_{xz} = \int \tau_{xz} dz, \quad q_{yz} = \int \tau_{yz} dz$$

- **Displacements ( $w, \varphi_x, \varphi_y$ ) and moments per length**



ELMT17.CDR

Shear forces show on positive side in positive direction.

- **Output of moments/shear forces**

The moments/shear forces per unit length are calculated at each center of element in global directions x and y. The moments  $m_1$  and  $m_2$  are the principal moments with the angle  $\varphi_1$  between 1-moment and x-coordinate direction (in degrees).

- **Plot of moments/shear forces**

The moments/shear forces per unit length are plotted due to following numbers

Stress number	Type	value plotted	Dimension
1	Moment	$m_{xx}$	[ $FL/L$ ]
2	Moment	$m_{xy}$	[ $FL/L$ ]
3	Moment	$m_{yy}$	[ $FL/L$ ]
5	Moment	$m_1$	[ $FL/L$ ]
6	Moment	$m_2$	[ $FL/L$ ]
7	Angle	$\varphi_1$	[rad]
8	Shear force	$q_{xz}$	[ $F/L$ ]
9	Shear force	$q_{yz}$	[ $F/L$ ]

**6.4.34 ELMT34**

**6.4.35 ELMT35**

**6.4.36 ELMT36**

**6.4.37 ELMT37**

**6.4.38 ELMT38**

**6.4.39 ELMT39**

**6.4.40 ELMT40**

**6.4.41 ELMT41**

**6.4.42 ELMT42**

**6.4.43 ELMT43**

**6.4.44 ELMT44**

### 6.4.45 ELMT45

---

```

matn, < nlay >, < lin >, ieas1, ibd, ibs, ieas2, h, ngp, scf
qx, qy, qz, ΔT, αt, ρ
E, ν (for matn=1)
for n=1,nlay
phin, hn
...

```

---

- **Theory**

ELMT45 is a 8-node geometrical linear/nonlinear 3D-Solid Shell Element. The definition of the nodes is similar to the macro **bloc**. Different material models are implemented and can be used with the **3D-Material library**.

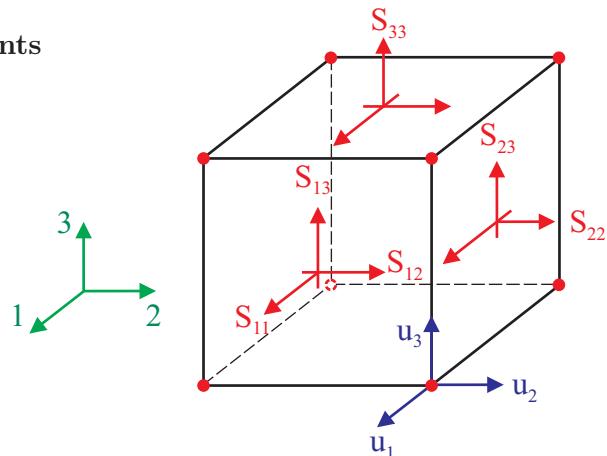
- **Material input data**

<i>matn</i>	[ <i>-</i> ]	material number
<i>nlay</i>	[ <i>-</i> ]	number of layers (def.=1)
<i>lin</i>	[ <i>-</i> ]	0: geometrical linear, 1: geometrical nonlinear
<i>ieas1</i>	[ <i>-</i> ]	Enhanced Assumed Strain terms 0/3/5/8/11/7/30
<i>ibd</i>	[ <i>-</i> ]	Bathe-Dvorkin 0/1
<i>ibs</i>	[ <i>-</i> ]	Betsch-Stein 0/1
<i>ieas2</i>	[ <i>-</i> ]	1=reg. mesh 2= distorted mesh
<i>h</i>	[ <i>-</i> ]	shell thickness=thickness of 1 Element
<i>ngp</i>	[ <i>-</i> ]	no. of Gauss Points/layer (def.=2)
<i>scf</i>	[ <i>-</i> ]	shear corr.factor: < 0 =   value   · FE-SCF, > 0=value (def.=1)
<i>q<sub>x</sub></i>	[ <i>F/L<sup>3</sup></i> ]	volumetric load in global x-direction
<i>q<sub>y</sub></i>	[ <i>F/L<sup>3</sup></i> ]	volumetric load in global y-direction
<i>q<sub>z</sub></i>	[ <i>F/L<sup>3</sup></i> ]	volumetric load in global z-direction
<i>ΔT</i>	[ <i>K</i> ]	temperature difference
<i>α<sub>t</sub></i>	[ <i>1/K</i> ]	coefficient of thermal expansion
<i>ρ</i>	[ <i>F/L<sup>3</sup></i> ]	specific weight
Material	input data, see the <b>3D-Material library</b>	
Input data for each layer i (on separate input line) beginning at z=-h/2		
<i>phi<sub>n</sub></i>	[ <i>°</i> ]	angle between global and local coordinate system
<i>h<sub>n</sub></i>	[ <i>L</i> ]	thickness of layer

Details on the derivation of the material models can be found in the **Theory Manual**.

- Displacements and stress/strain resultants

$$\mathbf{S} = \begin{bmatrix} S_{11} \\ S_{22} \\ S_{33} \\ S_{12} \\ S_{13} \\ S_{23} \end{bmatrix} \quad \mathbf{E} = \begin{bmatrix} E_{11} \\ E_{22} \\ E_{33} \\ 2E_{12} \\ 2E_{13} \\ 2E_{23} \end{bmatrix} \quad \mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}$$



- Output of stresses

Normal and shear stresses are calculated at the Gauss points of the element in global directions.

- Plot of stresses

The above defined stresses are plotted due to following numbers

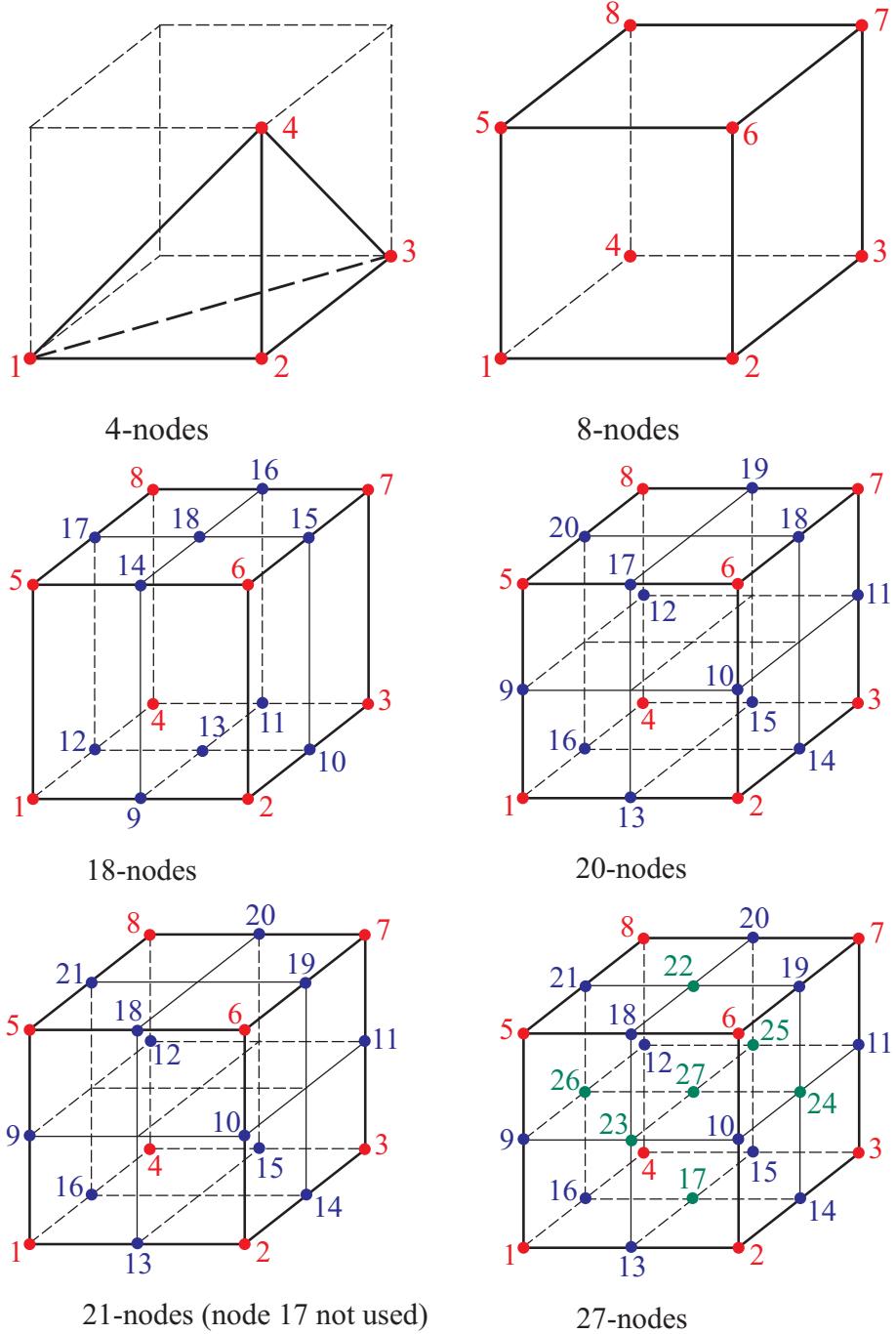
Stress number	1	2	3	4	5	6	7	8	9
Stress plotted [F/L <sup>2</sup> ]	\$S_{11}\$	\$S_{22}\$	\$S_{33}\$	\$S_{12}\$	\$S_{13}\$	\$S_{23}\$	\$S_1\$	\$S_2\$	\$S_3\$

- Plot problems with tetraeder

Macros `disp`, `stre` and similar macros do not work together with `hide`. Thus, use other post processing programs, e.g. TECPLOT for plotting of results. An associated interface to TECPLOT is implemented (`tec`).

- Element nodes

(see macro `bloc`)



**6.4.46 ELMT46**

**6.4.47 ELMT47**

**6.4.48 ELMT48**

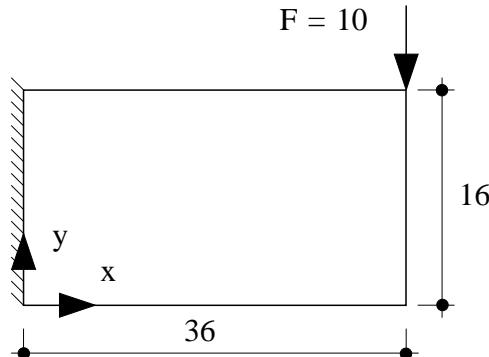
# Chapter 7

## Principal example

### A principal example with extended documentation

---

- Problem



$$E = 100\,000 \quad v = 0.3 \quad t = 1.2$$

- General Description using **FEAP**

The problem is modeled as plane stress problem using element Type 5 with 35 nodes and 24 elements. Thus 4-node quadrilaterals in an x,y coordinate system with 2-displacements at each node (u,v components in the x,y directions are used. Element Type Number 1 is used. The left end of the mesh is on vertical rollers except for the center point which is fully fixed .

The data required to analyze the problem using **FEAP** is given by the following sets.

- Input File

*firstrecord* : 1> **feap** \* \* example 1. Plane stress beam  
2> 35,24,1,2,2,4

In the above record 1 is used to describe the start of a new problem. The first four characters must be **feap**.

**second record:**

35 = numnp	number of nodes in the model
24 = numel	number of elements in the model
1 = nummat	number of material property sets in the model
2 = ndm	spatial dimension of the finite element mesh
2 = ndf	maximum number of unknowns (dof) at any node
4 = nen	maximum number of nodes connected to any element.

### 1.) Block generation of nodes and elements.

```
1> bloc
2> 4,4,6,1,1,1 (no.block nodes,NR,NS,1st node,1st elmt,matl.set )
3> 1,0.,16. (block node#, x,y-coordinates)
4> 2,0.,0.
5> 3,36.,0.
6> 4,36.,16.
```

### 2.) Boundary Condition Generation.

\* edge with constant coordinate

```
1> ebou
2> 1,0.,1,0 (direction , value, 1-dof b.c., 2-dof b.c.)
Direction interpreted as follows
    1 = 1st coordinate direction (i.e., x-dir)
    2 = 2nd coordinate direction (i.e., y-dir).
Boundary codes interpreted as follows:
    0 = unrestrained dof
    1 = restrained degree-of-freedom
```

\* node with constrained values

```
1> boun
2> 3,,1,1 (node#, , 1-dof fixed, 2-dof, fixed)
3>
```

### 3.) Forced Conditions (non-zero nodal loads/displacements)

```
1> load
2> 31,,,10. (node#, 1-dof load, 2-dof load)
3>
N.B. A negative load is in a direction opposite to coordinate direction.
```

### 4.) Material Property Sets

```
1> mate
2> 1,5
3> 100000.,0.3,0.0,1 (E, $\nu$ , $\rho$ ,1=plane stress)
4> 1.2 (Thickness)
N.B. The last item on record 3 is the problem type. A one (1) is plane stress,
a two (2) plane strain and a three (3) is the axisymmetrix case.
```

## 5.) End of Inputfile

```
1> end (end of input data)  
2> inte (execution in interactive mode)  
3> stop (end of file)
```

## 6.) Summary of Complete Input File for Mesh

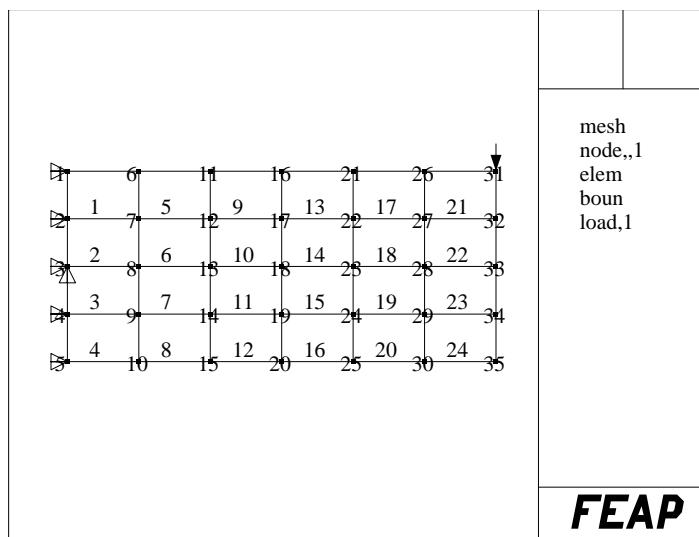
```
1> feap * * example 1. Plane stress beam  
2> 35,24,1,2,2,4  
3>  
4> bloc  
5> 4,4,6,1,1,1  
6> 1,0.,16.  
7> 2,0.,0.  
8> 3,36.,0.  
9> 4,36.,16.  
10>  
11> ebou  
12> 1,0.,1,0  
13>  
14> boun  
15> 3,,1,1  
16>  
17> load  
18> 31,,, -10.  
19>  
20> mate  
21> 1,5  
22> 100000.,0.3,0.0,1  
23> 1.2  
24>  
25> end  
26> inte  
27> stop
```

## •Macro Executions in interactive mode

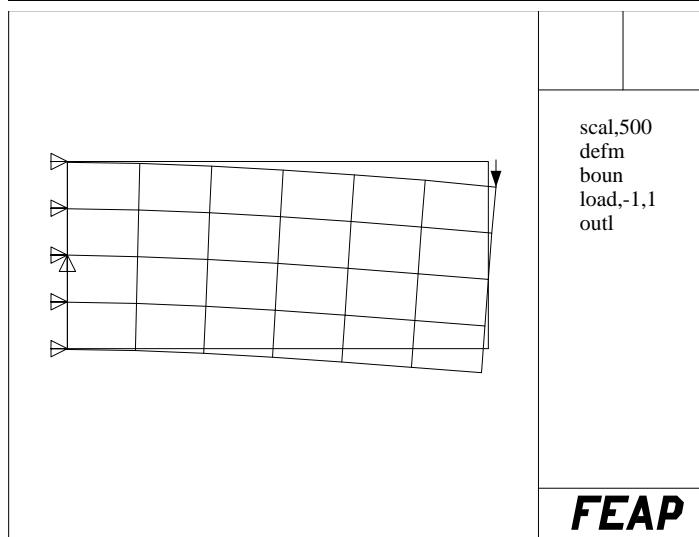
```
1> feap start program  
2> tang,,1 form tangent, residual, solve eq.  
3> disp,all output all displacements  
4> stre,node,1,35 compute and output stresses at nodes  
5> stre,all compute and output stresses in elmts  
6> reac,all compute and output all nodal reactions  
7> exit end macro execution
```

Results using PLOT-macro commands

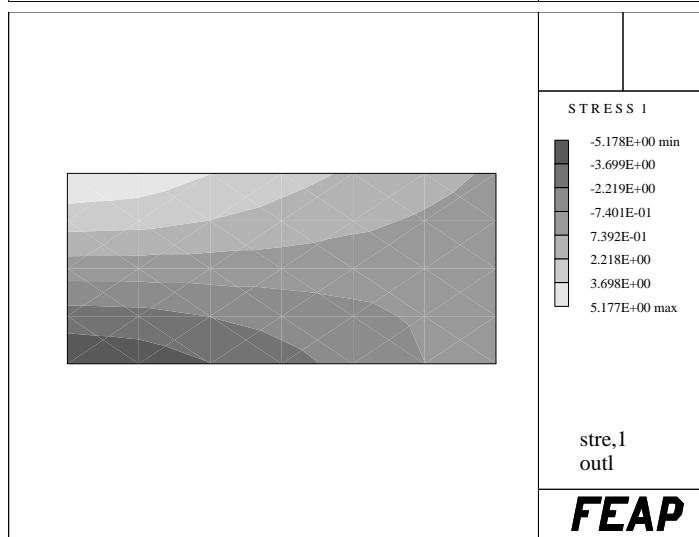
- System



- deformed mesh

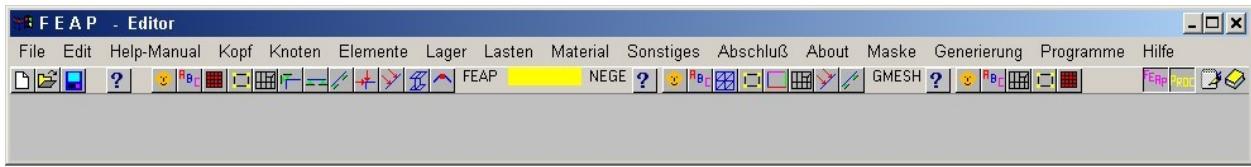


- stress  $\sigma_{11}$



## Chapter 8

### F E A P - E D Editor



Das Programm **FEAP-ED** dient dazu, die Dateneingabe für das Finite-Element-Programm **FEAP** zu vereinfachen. Das Programm kann auch als ein einfacher Editor verwendet werden. Alternativ kann die Dateneingabe mit einem beliebigen Windows-Editor erfolgen. Dies ist die Standardeinstellung und vorzuziehen.

Alle Befehle können aus der Menüliste abgerufen werden. Häufig verwendete Befehle liegen auch direkt auf der TASK-Leiste und sind schon an ihrem Symbol erkennbar. Symbole werden in dem gelben Feld des Menüs erklärt, wenn die Maus über dem jeweiligen Symbol steht.

Daten werden entweder direkt eingegeben oder mit den entsprechenden Menüs erzeugt. Der Datenaustausch zwischen FEAP-ED und dem Editor erfolgt über die Zwischenablage. CTRL+V kopiert die in der jeweiligen Maske erzeugten Daten in das (zu aktivierende) Editorfenster.

Vorhandene Daten in einer Datei können ebenfalls mit den Masken überarbeitet werden. Dazu sind die zu bearbeitenden Daten (jeweils nur 1 Makro!) mit der Maus zu markieren. Mit CTRL+C bzw. CTRL+X werden sie in die Zwischenablage kopiert/verschoben. Mit dem Befehl CTRL+D im zu aktivierenden FEAP-ED wird die zugehörige Maske geladen, sofern das FEAP-Makro existiert. Nach Eingabe der erforderlichen Daten können diese wiederum in die Datei geschrieben werden.

# Chapter 9

## F E A P - NEGE Mesher

### Vorbemerkungen

Dies ist die Kurzbeschreibung des Netzgenerierungsprogramms **NEGE** von J. Sienz, University College, Department of Civil Engineering, University of Swansea.

Das Programm wurde derart modifiziert, dass die Ein- und Ausgabe auf das FE-Programm **FEAP** abgestimmt und das Programm unter Windows9x/NT/2000/ME/XP lauffähig ist.

### Kurzbeschreibung

**NEGE** ist ein zweidimensionaler Netzgenerator, welcher Netze mit 3- oder 6-Knoten Dreieckelementen sowie 4-, 8- oder 9-Knoten Viereckselemente nach der Advancing Front Methode erzeugt. Das untersuchte Gebiet kann gekrümmte Ränder und Öffnungen besitzen. Es kann mehrfach zusammenhängen sowie aus mehreren Materialien bestehen. Weiterhin kann eine Dichtefunktion vorgegeben werden. Alle Dateneingaben sind formatfrei. Weiterhin können die Daten auch parametrisiert eingegeben werden. Damit FEAP die Netzgenerierung erkennt muss die FEAP-Eingabedatei mit dem Macro **NEGE** beginnen. Es folgen die Generierungsmacros sowie anschließend die FEAP-spezifischen Macros.

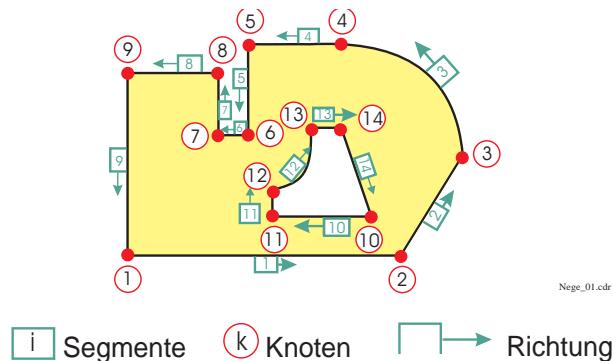


Bild 1 Beispiel einer Beschreibung eines Gebietes.

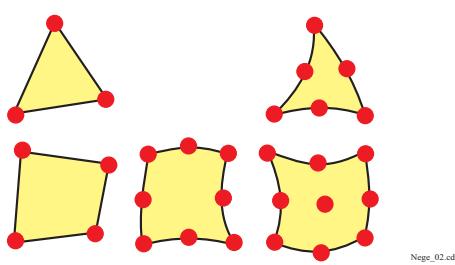
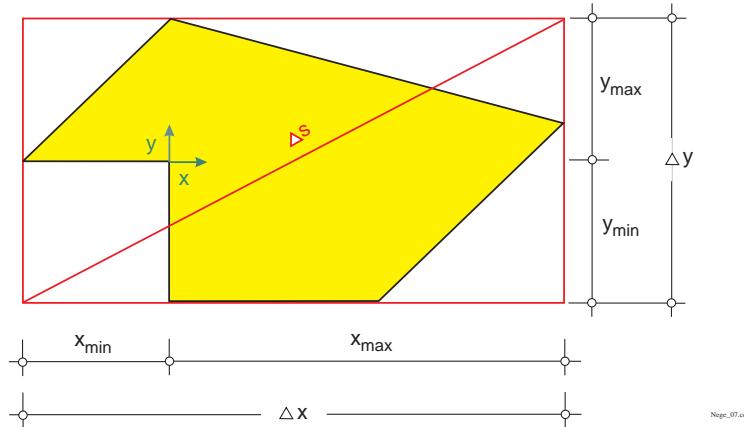


Bild 2 Generierte Elementtypen

Das Finite-Element Netz wird von **NEGE** automatisch erzeugt. Die mittlere Länge  $L_e$  des einzelnen Elementes kann mit der EingabevARIABLEN esiz vorgegeben werden. Hierzu werden vom Programm die maximalen Koordinaten  $x_{min}, x_{max}, y_{min}, y_{max}$  ermittelt.



Aus dem einschließenden Rechteck ( $\Delta x, \Delta y$ ) wird die mittlere Elementlänge  $L_e$  wie folgt ermittelt:

$$L_e = \frac{esiz}{100} \cdot \Delta s \quad \text{mit} \quad \Delta s = \sqrt{\Delta x^2 + \Delta y^2}$$

Wird kein Wert esiz eingegeben, so rechnet **NEGE** mit esiz=5. Damit ist die Elementlänge als bestimmter Prozentsatz der Diagonale des einschließenden Rechtecks zu verstehen. Somit entstehen in etwa **gleichmäßig verteilt** Netze.

Lokale Netzverdichtungen können erzeugt werden, wenn Daten für das Hintergrundnetz angegeben werden.

### Eingabebeschreibung

Die Eingabe erfolgt mit Macros. Anders als bei FEAP-Macros ist deren Reihenfolge jedoch festgelegt. Einzelne Macros sind optional ([...]). Nach jedem Macro muß mindestens eine Leerzeile eingefügt werden. Einzelne Daten sind mit Kommas abzutrennen. Die Eingabe erfolgt **formatfrei**. Anschließend werden die FEAP-Macros in der Datei eingegeben. Für eine variable Eingabe der Macros ist das Kommando **neco** zu verwenden. Die generierten Daten werden in eine temporären Ascii-Datei **ifeap.tmp** geschrieben. Diese kann ebenfalls als Eingabedatei verwendet werden. Koordinaten und Elemente sind in einer temporären Binär-Datei **bfeap.tmp**. Die eingegebenen Werte für die Generierung werden in der Datei **nege.log** protokolliert.

#### 1) **nege Kommentartext** (Kopfzeile)

**ntyp, node, ndf, ncorr, esiz**

- ntyp = 1: erzeugt nur das Hintergrundnetz
- 2: erzeugt 1 und die Segmente
- 3: erzeugt 2 und die Gebiete
- 4: erzeugt 3 sowie die Netzgenerierung und die FEAP-Eingabedatei

Hinweise: Es wird empfohlen, die Dateneingabe und Generierung schrittweise vorzunehmen (ntyp = 1 – 4). Im Plot-Modus funktionieren jeweils die Optionen zu Knoten, Elementen, Randbedingungen und Lasten.

```

node    = 3, 4, 6, 8, 9   : zu erzeugende Elementtypen
ndf     = 1, 2, 3         : Anzahl der Freiheitsgrade
                           (1 – z.B. Temperatur, 2 – z.B. Scheibe, 3 – z.B. Platte)
ncorr   = 0               : Gebiet ist geradlinig berandet
                           1               : Gebiet hat auch gekrümmte Ränder
esiz                : Vorgabe der mittleren Elementlänge in % der Diagonale
                           des einschließenden Rechtecks, etwa 10–0.5 (grob–fein)

```

[ 2) **neco** (Eingabe der Konstanten) ]

a = .....

b = .....

.....

Hier sind Konstanten zu definieren, die dann anschließend bei den weiteren Macros verwendet werden können. Die Regeln sind analog zum Macro **para**. Die Konstanten werden in den FEAP–Macros übernommen.

[ 3) **back** (Hintergrundnetz) ]

ib, lnode(ib,1), lnode(ib,2), lnode(ib,3)

.....

.....

Leerzeile

jb, corb(jb,1), corb(jb,2), d(jb,1), d(jb,2), d(jb,3), d(jb,4)

.....

.....

Es wird ein Hintergrundnetz aus Dreieckelementen definiert, mit dessen Hilfe die Netzdichte beliebig vorgegeben werden kann. Werden hier Daten eingegeben, ist der Parameter esiz (Kopfzeile) außer Kraft gesetzt.

```

ib                  : Element–Nummer
lnode(ib,i), i = 1,3 : Knoten des Elements (entgegen dem Uhrzeigersinn)
jb                  : Knoten–Nummer
corb(jb,i), i = 1,2 : Koordinaten des Knotens
d(jb,1)             : gewünschte Elementlänge
d(jb,2)             : gewünschte Dehnung
d(jb,3)             : X–Wert des Richtungskosinus der Dehnungsrichtung
d(jb,4)             : Y–Wert des Richtungskosinus der Dehnungsrichtung

```

4) **geom** (Koordinaten der Knotenpunkte)

ig, corg(ig,1), corg(ig,2)

ig : Knotennummer

corg(ig,i), i = 1,2 : Koordinaten des Knotens

Es werden die Koordinaten der Eckpunkte des Gebietes eingegeben.

5) **segm** (Eingabe der Randsegmente)

is, lnode(is,1), lnode(is,2), [mtyp, g(is,1), g(is,2), g(is,3), g(is,4)]

is : Elementnummer

lnode(is,1), lnode(is,2) : Anfangsknoten, Endknoten

### nur bei gekrümmten Rändern

mtyp	Beschreibung	g(is,1)	g(is,2)	g(is,3)	g(is,4)	
2	Kreis (M-Punkt $P_x, P_y$ , Radius R)	$P_x$	$P_y$	R		*
3	Ellipse (M-Punkt $P_x, P_y$ , Radien $R_x, R_y$ )	$P_x$	$P_y$	$R_x$	$R_y$	
6*	Kreis (Punkt $P_1$ , Punkt $P_4$ , Radius R )	$P_1$	$P_4$	R		

Erläuterung siehe nächste Seite

Es werden die Segmente zur Beschreibung des Randes definiert. Die Wahl von **Anfangs-** und **Endknoten** definiert die '**Richtung**' des Segments.

### 6) **regi** (Beschreibung des Gebietes)

ir, mr, lnode(ir,i), i = 1, mr

ir : Nummer des Gebietes  
 mr : Anzahl der Segmente, die dieses Gebiet beschreiben  
 lnode(ir,i), i=1, mr : Segmente des Gebietes in umlaufender Reihenfolge

Es werden die einzelnen Gebiete beschrieben. Jedes Gebiet hat ein eigenes Material. Wird ein Segment entgegen seiner Definition durchlaufen, so ist es negativ einzugeben. Die Segmentdefinition muss so erfolgen, dass jedes Segment **zuerst** 1-mal positiv angesprochen werden muss! Die Gebiete werden am einfachsten eingegeben, wenn sie entlang ihres Randes 'umfahren' werden. **Außenränder** sind dabei **entgegen dem Uhrzeigersinn** zu umfahren, während **Innenränder im Uhrzeigersinn** zu beschreiben sind. Die Segmente müssen **nicht** zusammenhängen. **Maximal 16 Daten je Zeile**, Fortsetzung: Folgezeilen.

### [ 7) **pres** (Eingabe von Randlasten) ]

ip, (pres(1,i), i=1,ndf), (pres(2,i), i=1,ndf)

ip : belastetes Randsegment  
 pres(1,i), i = 1,ndf : Lasten auf dem Anfangsknoten  
 pres(2,i), i = 1,ndf : Lasten auf dem Endknoten

Aus den Anfangs- und Endwerten wird eine trapezförmige Belastung des Randes ermittelt. Die Lasten wirken in Richtung der eingeführten Freiheitsgrade. Die Bezugslänge ist die wirkliche Randlänge.

### [ 8) **fixe** (Eingabe von Randbedingungen) ]

if, (bc(i), i=1,ndf)

if : Randsegment mit Randbedingungen  
 bc(1) : Randbedingung für Freiheitsgrad 1  
 bc(2) : Randbedingung für Freiheitsgrad 2  
 ...  
 bc(i)=0: frei, bc(i)=1: gelagert, nur bc(1)==2: Linie auf der Knoten liegen müssen.

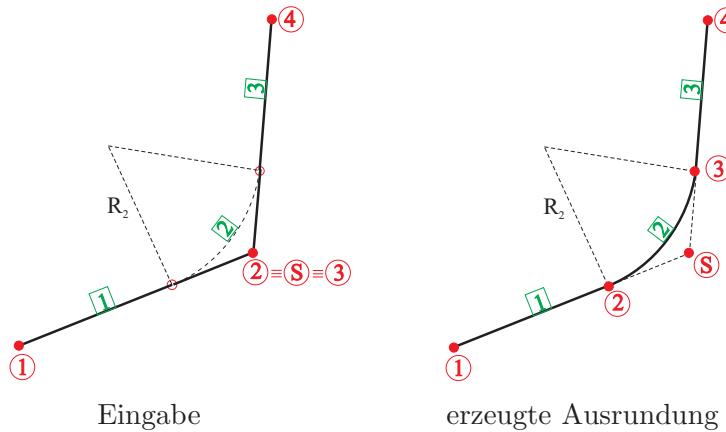
### 9) **FEAP-Eingabedaten**

Nachfolgend können beliebige FEAP-Eingabedaten folgen, welche dann in die FEAP-Eingabedatei kopiert werden. Dies ist in der Regel nur noch das **mate**-Macro (Beschreibung des verwendeten Materials bzw. Elements) sowie der Abschluß der Eingabedatei mit den Macros **end**, **inte**, **stop**. Bei Verwendung des Standardlösers **solv**,0 wird empfohlen mit **opti** eine Knotennummernoptimierung durchzuführen. Weitere variable Größen können mit dem Macro **para** eingegeben werden.

**Anmerkungen:**

- **nege** generiert Netze für ebene Scheiben- und Plattenprobleme mit  $ndf \leq 3$  Ebene Schalen mit  $ndf = 5, 6$  können generiert werden. Hier sind allerdings die zu den Freiheitsgraden  $> 3$  zugehörigen Randbedingungen und Lasten mit **FEAP**-Macros zu erzeugen.
- Mit  $mtyp = 6$  können kreisförmige Ausrundungen sehr leicht erzeugt werden, da das Programm sich die Koordinaten der zunächst unbekannten Knoten 2 und 3 selbst ermittelt. Hierzu erhalten diese Knoten zunächst in **geom** die Koordinaten des Schnittpunktes S der beiden angrenzenden Segmente. In **segm** müssen dann der Anfangsknoten des vorhergehenden Segmentes, der Endknoten des nachfolgenden Segmentes und der Radius der Ausrundung angegeben werden. Dies ist im folgenden Beispiel dokumentiert. Die zugehörige Eingabe lautet:

<b>geom</b>	<b>segm</b>
1 $x_1$ $y_1$	1 1 2
2 $x_S$ $y_S$	2 2 3 6 1 4 $R_2$
3 $x_S$ $y_S$	3 3 4
4 $x_4$ $y_4$	



## 10) Beispiele

### Beispiel 1

#### NEGE-Datensatz

```
nege Beispiel 1
    4,      4,      2,      0,      3.50
```

```
neco ns: Konstante fuer NEGE 2
a=10
b=5
```

```
geom etrie: Knoten zur Beschreibung des Randes 4 a
1,0,0
2,a,0
3,a,b
4,0,b
```

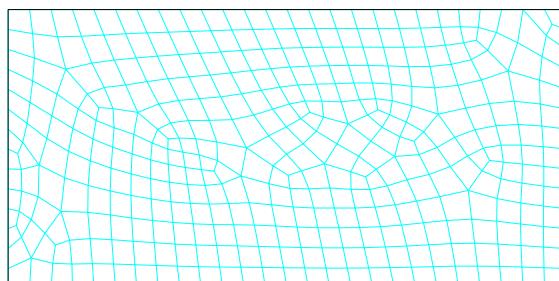
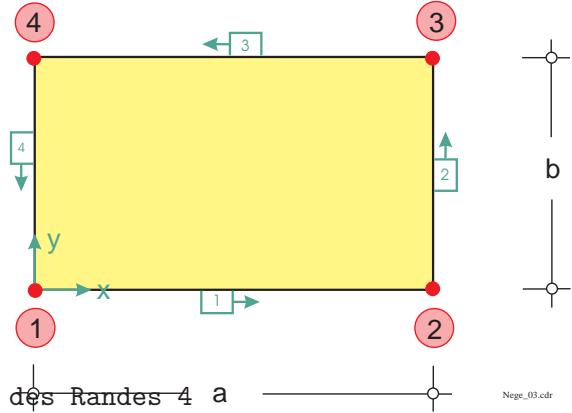
```
segm ente: Elemente zur Beschreibung des Randes 5
1,1,2
2,2,3
3,3,4
4,4,1
```

```
regi on: Beschreibung eines Gebietes 6
1,4,1,2,3,4
```

```
mate
1,5
1,,,
,,,
```

```
end
opti
```

```
inte
stop
```



4-Knoten-Netz mit 354 Elementen und 395 Knoten

## Beispiel 2

### NEGE-Datensatz

nege Beispiel 2

4, 4, 2, 0, 3.50

neco ns: Konstante fuer NEGE 2

a=10

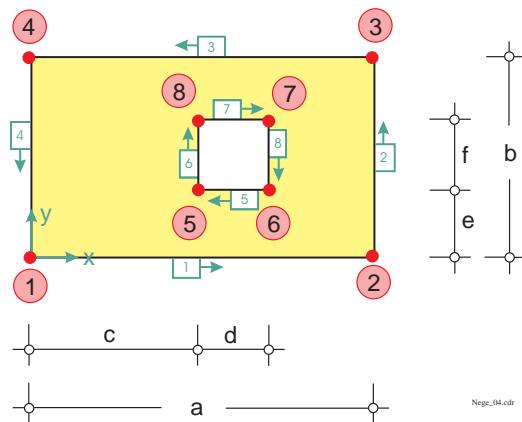
b=5

c=5

d=3

e=2

f=2



Nege\_04.cdr

geom etrie: Knoten zur Beschreibung des Randes 4

1,0,0  
2,a,0  
3,a,b  
4,0,b  
5,c,e  
6,c+d,e  
7,c+d,e+f  
8,c,e+f

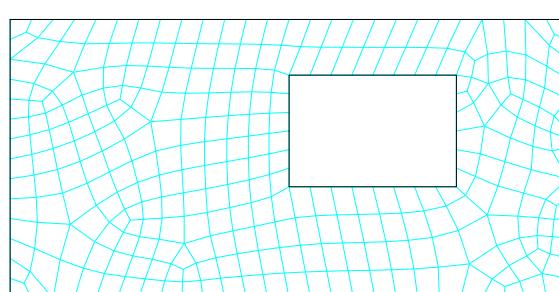
segm ente: Elemente zur Beschreibung des Randes 5

1,1,2  
2,2,3  
3,3,4  
4,4,1  
5,6,5  
6,5,8  
7,8,7  
8,7,6

regi on: Beschreibung eines Gebietes 6

1,8,1,2,3,4,5,6,7,8

mate  
1,5  
1,,,,  
,,,  
  
end  
opti  
inte  
stop



4-Knoten-Netz mit 278 Elementen und 332 Knoten

### Beispiel 3

#### NEGE-Datensatz

```
nege Beispiel 3
 4,    3,    2,    1,      3.50
```

```
neco ns: Konstante fuer NEGE 2
r=10
```

```
geom etrie: Knoten zur Beschreibung des Randes 4
1,0,0
2,r,0
3,0,r
```

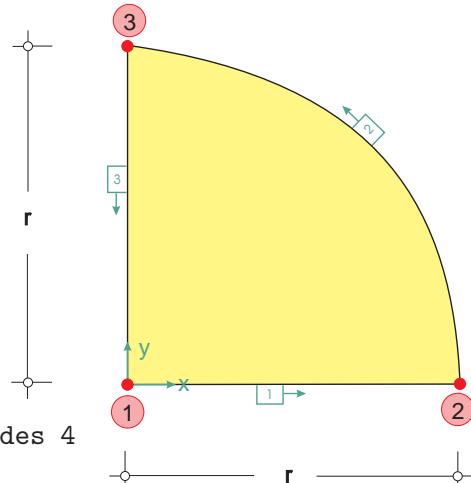
```
segm ente: Elemente zur Beschreibung des Randes 5
1,1,2
2,2,3,2,0,0,r
3,3,1
```

```
regi on: Beschreibung eines Gebietes 6
1,3,1,2,3
```

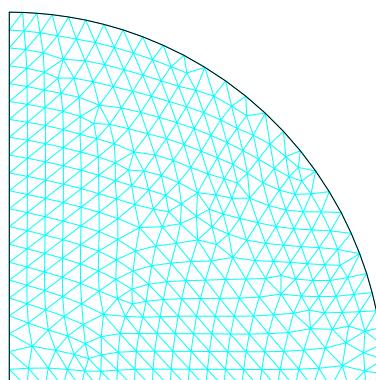
```
mate
1,5
1,,,
,,,
```

```
end
opti
```

```
inte
stop
```



Nege\_05.cdr



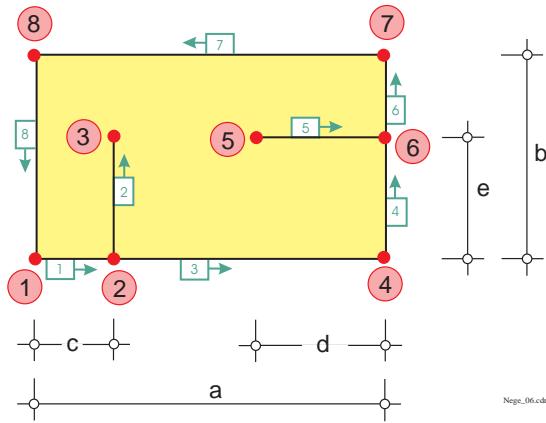
3-Knoten-Netz mit 665 Elementen und 369 Knoten

### Beispiel 4

#### NEGE-Datensatz

```
nege Beispiel 4
    4,    4,    2,    0,      3.50
```

```
neco ns: Konstante fuer NEGE 2
a=20
b=10
c=5
d=10
e=5
```



Nege\_06.cdr

```
geom etrie: Knoten zur Beschreibung des Randes 4
```

```
1,0,0
2,c,0
3,c,e
4,a,0
5,a-d,e
6,a,e
7,a,b
8,0,b
```

```
segm ente: Elemente zur Beschreibung des Randes 5
```

```
1,1,2
2,2,3
3,2,4
4,4,6
5,5,6
6,6,7
7,7,8
8,8,1
```

```
regi on: Beschreibung eines Gebietes 6
```

```
1,10,1,3,4,6,7,8,2,-2,5,-5
```

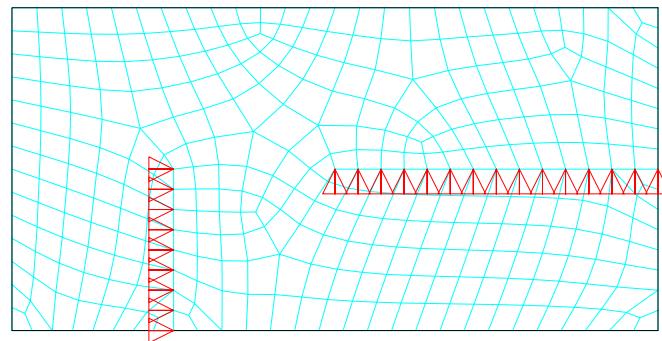
```
fixe d: Randbedingungen auf dem Rand 8
```

```
2,-2
2,1,0
5,-2
5,0,1
```

```
mate
1,5
1,,,,,
,,,,
```

```
end
opti
```

inte  
stop



4-Knoten-Netz mit 320 Elementen und 363 Knoten

### Beispiel 5

#### NEGE-Datensatz

nege Beispiel 5 (Scheibe mit Loch)  
4, 4, 2, 1, 5

neco ns: Konstante fuer NEGE 2  
 $a=5$   
 $r=1$   
 $p=10$

back ground: Hintergrundnetz 3  
1,1,2,3,  
2,2,4,3,

1,0,0,0.05,1,1,0,  
2,a,0,0.50,1,1,0,  
3,0,a,0.50,1,1,0,  
4,a,a,1.00,1,1,0,

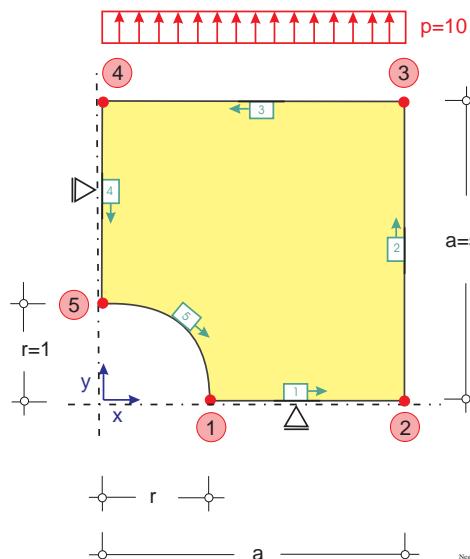
geom etrie: Knoten zur Beschreibung des Randes 4

1,r,0  
2,a,0  
3,a,a  
4,0,a  
5,0,r

segm ente: Elemente zur Beschreibung des Randes 5

1,1,2  
2,2,3  
3,3,4  
4,4,5  
5,5,1,2,0,0,r

regi on: Beschreibung eines Gebietes 6



Nege\_07.adr

1,5,1,2,3,4,5

PRES sure: Randlasten 7

3,0,p,0,p

fixe d: Randbedingungen auf dem Rand 8

1,0,1

4,1,0

mate

1,5

1000,0,0,1,1

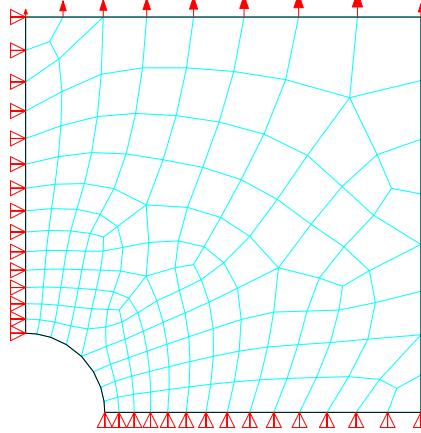
1

end

opti

inte

stop



4-Knoten–Netz mit 154 Elementen und 181 Knoten

### Beispiel 6

#### NEGE–Datensatz

nege Beispiel 6 1/4 HEA 300

4, 4, 1, 1, 2

neco ns: Konstante fuer NEGE 2

b=30

h=29

s=0.85

t=1.4

r=2.7

geom etrie: Knoten zur Beschreibung des Randes 4

1, 0,0,

2, s/2,0,

3, s/2,h/2-r-t,

4,s/2+r,h/2-t,

5, b/2,h/2-t,

6, b/2,h/2,

7, 0,h/2,

segm ente: Elemente zur Beschreibung des Randes 5

1, 1,2,0, 0, 0,0,0

2, 2,3,0, 0, 0,0,0

3, 3,4,2,s/2+r,h/2-t-r,r,0

4,4,5,0, 0, 0,0,0

5,5,6,0, 0, 0,0,0

6,6,7,0, 0, 0,0,0

7,7,1,0, 0, 0,0,0

regi on: Beschreibung eines Gebietes 6

1,7,1,2,3,4,5,6,7

fixe ed: Randbedingungen auf dem Rand 8

1,1

7,1

mate

1,12

5,,3

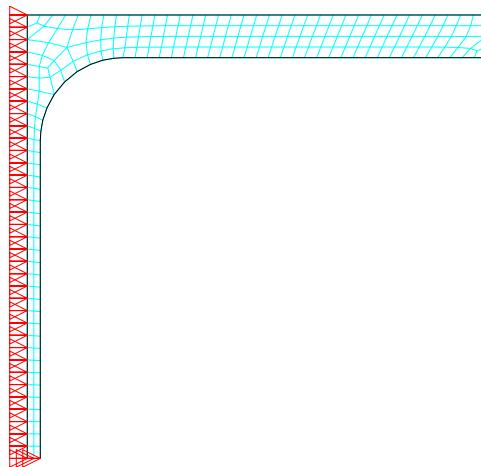
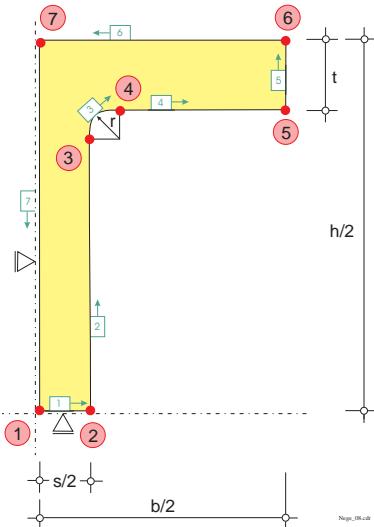
solv

2

end

inte

stop



4-Knoten–Netz mit 210 Elementen und 283 Knoten

# Chapter 10

## F E A P - GMESH Mesher

### Vorbemerkungen

**GMESH** ist ein dreidimensionaler Netzgenerator, welcher flächenhafte Netze mit 3-Knoten-Dreieckelementen sowie 4- oder 8/9-Knoten-Viereckelementen erzeugt. Die Daten werden in einer Eingabedatei zusammengestellt (analog FEAP); danach erzeugt **GMESH** schrittweise die FEAP-Eingabedatei. Alle Dateneingaben können auch parametrisiert eingegeben werden. **GMESH** generiert Netze für ebene Scheiben- und Plattenprobleme sowie auch für gekrümmte Probleme und für Faltwerke. Das zu generierende System ist so aufzuteilen, dass es durch 4–8 Knoten–Blöcke beschreibbar ist. An Zwischengrenzen der Blöcke ist darauf zu achten, dass die Elementteilung zusammen passt. Mit 4–Knoten–Blöcken können linear berandete Gebiete beschrieben werden; Seitenmittennoden führen zu parabelförmigen Rändern.

### Eingabebeschreibung

Die Eingabe erfolgt mit Macros. Anders als bei FEAP ist deren Reihenfolge jedoch festgelegt. Einzelne Macros sind optional ([...]). Nach jedem Macro muss mindestens eine Leerzeile eingefügt werden. Einzelne Daten sind mit Kommas abzutrennen. Alle Dateneingaben sind formatfrei. Damit FEAP die Netzgenerierung erkennt muss die FEAP–Eingabedatei mit dem Macro **gmesh** beginnen. Es folgen die Generierungsmacros sowie anschließend die FEAP–spezifischen Macros. Für eine variable Eingabe der Macros ist das Kommando **neco** zu verwenden. Die generierten Daten werden in eine temporäre Datei **ifeap.tmp** geschrieben. Diese kann ebenfalls als Eingabedatei verwendet werden. Die eingegebenen Werte für die Generierung werden in der Datei **gmesh.log** protokolliert.

- 1) **gmesh** Kommentartext (Kopfzeile)

**node, ndm, ndf, iopt, ityp**

node = 3, 4, 8, 9	:	zu erzeugende Elementtypen
ndm = 2,3	:	Anzahl der Dimensionen
ndf = 1-6	:	Anzahl der Freiheitsgrade
[iopt] = 0,1	:	ohne/mit Knotennummeroptimierung
[ityp] = 0,1	:	normale Generierung/Kontrolle der Eingabe
[iprin] = 0,1	:	ohne/mit Ausdruck von Zwischenergebnissen

- [ 2) **neco** (Eingabe der Konstanten) ]

a = .....

b = .....

.....

Hier sind Konstanten zu definieren, die dann anschließend bei den weiteren Macros verwendet werden können. Die Regeln sind analog zum Macro **para**. Die Konstanten werden von FEAP übernommen! Eine Neudefinition unter **para** ist nicht erforderlich!

3) **gele** (Eingabe der Eingabeblocks mit 4–8 Knoten)

is, matno(is), lnode(is,1),..., lnode(is,8)  
 is : Blocknummer  
 matno(is) : Materialnummer des Blocks  
 lnode(is,1), ..., lnode(is,8) : Knotennummern der Blöcke

Es sind 4–8 Knoten einzugeben (Eckknoten 1–4 zwingend, Seitenmittenknoten 5–8 optional) Die Reihenfolge der Knotennummern ist analog der FEAP-Eingabe, entgegen dem Uhrzeigersinn.

4) **gcor** (Koordinaten der Knotenpunkte)

ig, corg(ig,1), corg(ig,2), [corg(ig,3)], [ityp]  
 ig : Knotennummer  
 corg(ig,i), i = 1,2(3) : Koordinaten des Knotens  
 ityp : 0=cart., 1=pola(1-2), 2=pola(2-3), 3=pola(1-3), 4=sphe

Für die **eingeführten** Zwischenknoten müssen die Koordinaten eingegeben werden. Ihre Position kann im Rahmen des isoparametrischen Konzeptes (siehe Macro **bloc**) gewählt werden.

5) **ndvi** (Unterteilung der Blöcke)

je Block sind die folgenden Daten einzugeben:  
 is, ndvix, [dx(i), i = 1,ndvix]  
 is, ndviy, [dy(i), i = 1,ndviy]  
 is : Blocknummer  
 ndvix,ndviy : Anzahl der Unterteilungen in x,y-Richtung  
 dx(i), dy(i) : Unterteilungen je Richtung

Annmerkungen:

- # An Zwischengrenzen der Blöcke ist darauf zu achten, dass die Unterteilungen zusammenpassen.
- # Die Eingabe der Unterteilungen ist nur bei unterschiedlichen Werten notwendig.
- # Es können beliebige Werte eingegeben werden. Die Verteilung ergibt sich aus dem Verhältnis der Eingabewerte.
- # Je Zeile dürfen maximal 16 Werte eingegeben werden. Daüberhinaus erfolgt die Eingabe in Folgezeilen.

6) **feap**–Eingabedaten

Danach können beliebige FEAP–Eingabedaten folgen, welche dann in die FEAP–Eingabedatei kopiert werden. Dies ist in der Regel nur noch das **mate**–Macro (Beschreibung des verwendeten Materials bzw. Elements) sowie der Abschluss der Eingabedatei mit den Macros **end**, **inte**, **stop**. Bei Verwendung des Standardlösers **solv**,0 wird empfohlen mit **opti** eine Knotennummernoptimierung durchzuführen. Weitere variable Größen können mit dem Macro **para** eingegeben werden.

## 7) Beispiele

**Beispiel 1** im Raum befindliche gekrümmte Fläche

Isoparametrische Ansatzfunktion  $N_1$  auf einem Quadrat  $5 \times 5$  durch Vorgabe der Seitenmittenknoten.

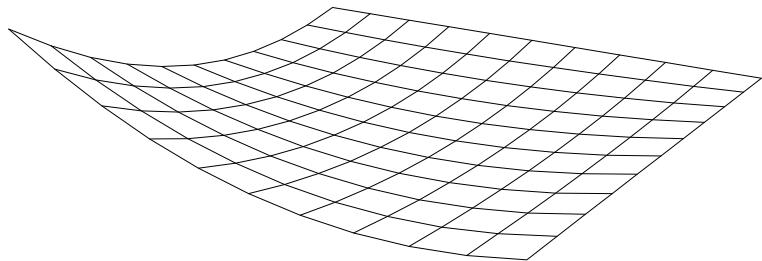
**GMESH–Datensatz**

```
gmesh Isoparametrische Ansatzfunktion N_1  
4,3,2,0,0
```

```
gele  
1,1,1,2,3,4,5,0,0,6
```

```
gcor Koordinaten, (5+6=Seitenmittenknoten)
```

```
1,0,0,h  
2,a,0,0  
3,a,a,0  
4,0,a,0  
5,a/2,0,0  
6,0,a/2,0
```



```
ndvi Teilung 10*10 Elemente  
1,10  
1,10
```

```
end
```

```
inte  
stop
```

**Beispiel 2** ebene Fläche mit Loch

**GMESH–Datensatz**

```
gmesh Scheibe mit Kreis-Öffnung Dreieckelemente  
3,2,2,0,0
```

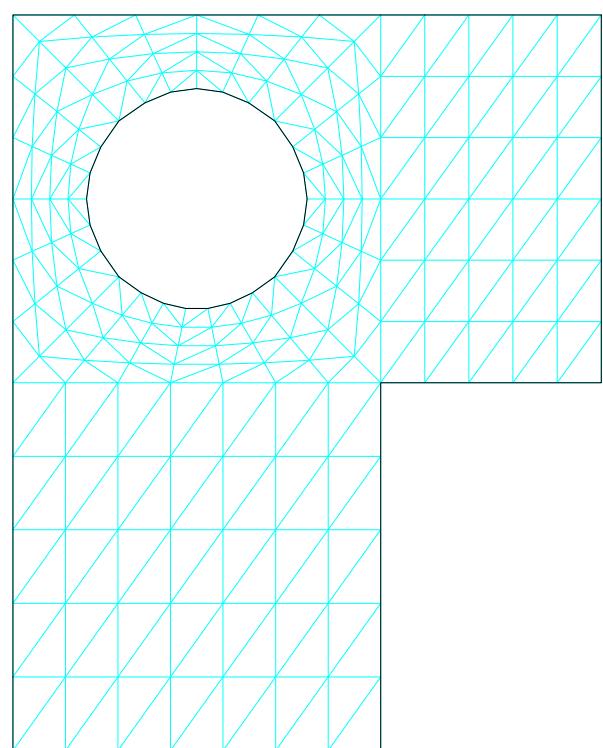
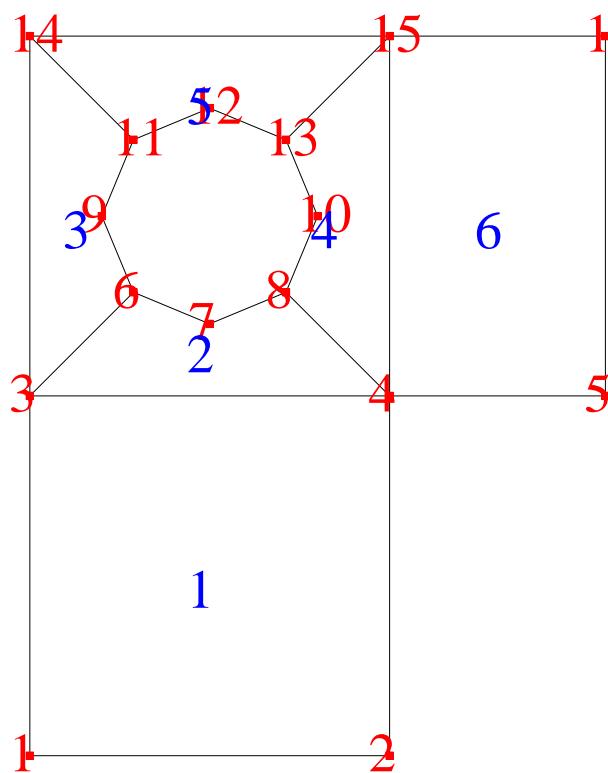
```
neco  
m=4  
k=5  
n=6  
l=7
```

```
gele
1,1,1,2,4,3
2,1,3,4,8,6,0,0,7
3,1,3,6,11,14,0,9
4,1,8,4,15,13,0,0,0,10
5,1,11,13,15,14,12
6,1,4,5,16,15
```

```
gcor
1,0,0
2,5,0
3,0,5
4,5,5
5,8,5
6,1.4393,6.4393
7,2.5,6
8,3.5607,6.4393
9,1,7.5
10,4,7.5
11,1.4393,8.5607
12,2.5,9
13,3.5607,8.5607
14,0,10
15,5,10
16,8,10
```

```
ndvi
1,l
1,k
2,l
2,m
3,m
3,n
4,n
4,m
5,n
5,m
6,k
6,n
```

```
end
inte
stop
```



**Beispiel 3 Zweifeldplatte auf Wand mit Punktstützung  
GMESH-Datensatz**

```
gmesh Zweifeld-Platte auf punktgestuetzter Wand
4,3,6
```

```
neco Konstanten
```

```
a=8
```

```
b=16
```

```
h=2
```

```
gele Elemente
```

```
1,1,7,8,5,2
```

```
2,2,1,3,6,4
```

```
gcor Koordinaten
```

```
1, -a,0, 0
```

```
2, 0,0, 0
```

```
3, a,0, 0
```

```
4, -a,b, 0
```

```
5, 0,b, 0
```

```
6, a,b, 0
```

```
7, 0,0,-h
```

```
8, 0,b,-h
```

```
ndvi Einteilung
```

```
1,12
```

```
1,5
```

```
2,12
```

```
2,12
```

```
cons Konstanten FEAP
```

```
a=8
```

```
b=16
```

```
h=2
```

```
ebou Rndlager
```

```
1,-a,1,1,1
```

```
1, a,1,1,1
```

```
poin Punktlager Wand
```

```
0,0,-h
```

```
0,0,0
```

```
1,1,1
```

```
poin Punktlager Wand
```

```
0,b,-h
```

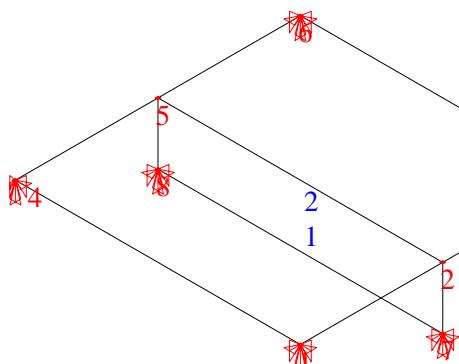
```
0,0,0
```

```
1,1,1
```

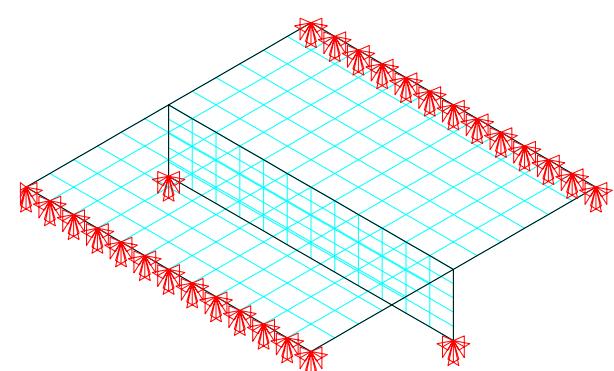
```
end
```

```
inte
```

```
stop
```



Eingabenetz



generiertes Netz

**Beispiel 4 1/2 Doppel-T-Träger mit Löchern und seitlichem Anschluß**  
**GMESH-Datensatz**

```
gmesh I-Traeger mit Loechern  
4,3,1,0,0
```

```
neco  
i=4  
k=6  
m=8  
n=12  
  
gele  
1, 1,30, 2,24,32  
2, 2, 5, 6,28,27  
3, 3, 1, 2,30,29  
4, 4, 3, 4, 9, 7, 0, 0, 8  
5, 5, 3, 7,17,25, 0,13  
6, 6, 9, 4,26,19, 0, 0, 0,14  
7, 7,17,19,26,25,18  
8, 8, 4, 5,12,10, 0, 0,11  
9, 9, 4,10,20,26, 0,15  
10,10,12, 5,27,22, 0, 0, 0,16  
11,11,20,22,27,26,21  
12,12,23,24,32,31  
  
gcor          ndvi  
1,    0,-8,-15.5      1,i  
2,    94,-8,-15.5     1,m  
3,    0, 0,-15.5      2,1  
4,    47, 0,-15.5     2,m  
5,    94, 0,-15.5     3,2*n  
6,   128.5, 0,-15.5  
7,17.964, 0,-3.536    3,i  
8,   21.5, 0,-5       4,n  
9,25.036, 0,-3.536    4,k  
10,69.964, 0,-3.536   5,k  
11,  73.5, 0,-5       5,m  
12,77.036, 0,-3.536   6,k  
13,  16.5, 0, 0        6,m*1.5  
14,  26.5, 0, 0        7,n  
15,  68.5, 0, 0        7,k  
16,  78.5, 0, 0        8,n  
17,17.964, 0, 3.536    8,k  
18,  21.5, 0, 5        9,k  
19,25.036, 0, 3.536    9,m*1.5  
20,69.964, 0, 3.536    10,k  
21,  73.5, 0, 5        10,m  
22,77.036, 0, 3.536    11,n  
23,    0,-8, 15.5      11,k
```

```

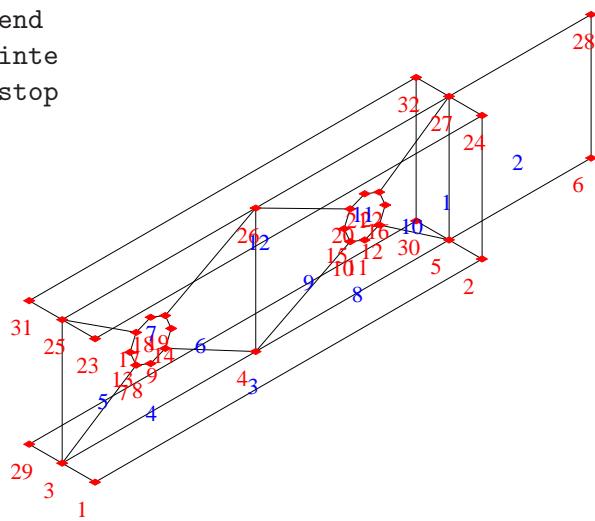
24,      94,-8, 15.5      12,2*n
25,      0, 0, 15.5
26,      47, 0, 15.5      12,i
27,      94, 0, 15.5
28, 128.5, 0, 15.5
29,      0, 8,-15.5
30,      94, 8,-15.5
31,      0, 8, 15.5
32,      94, 8, 15.5

```

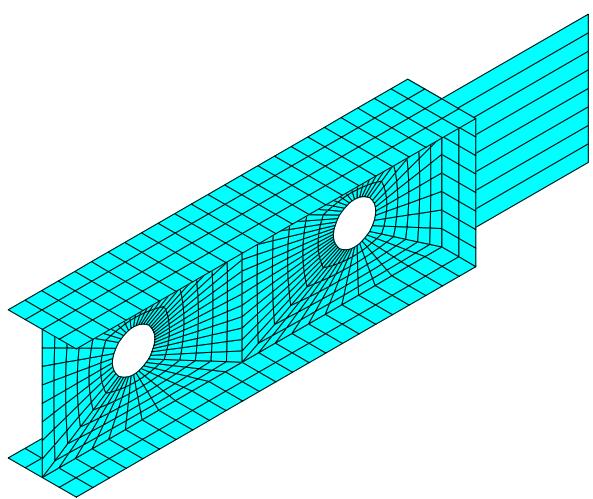
```

end
inte
stop

```



Eingabenetz



generiertes Netz

# Chapter 11

## F E A P - ISOGEO

### 11.1 Overview

Input files for isogeometric FEAP - ISOGEO calculations have the same format as input files for standard FEAP calculations. For the definition of the geometry the three additional macros **nmpq**, **k nv1** and **k nv2** are introduced. In Isogeometric Analysis the shape functions from the NURBS geometry are used. The usage of the exact NURBS geometry requires a distinct form of input. Necessary changes include a slightly different use of macros **feap** and **coor**, the definition of new macros **nmpq**, **k nv1** and **k nv2**. As the mesh is predefined by the NURBS geometry description, meshing macros like **elem** or **bloc** are neither needed nor allowed. All other feap mesh macros can be used as usual. **tie** can be used to connect patches if the control points coincide. Do not use **opti**, usage of PARDISO-solver is recommended. The input macros are explained in the **mesh**-part of this manual. Macro commands and plot commands work as in standard FEAP.

### 11.2 Terminology

NURBS surfaces are a tensor product area spanned by two knot vectors  $\Xi^1 = [\xi_1^1, \xi_2^1, \dots, \xi_{n+p+1}^1]$  and  $\Xi^2 = [\xi_1^2, \xi_2^2, \dots, \xi_{n+p+1}^2]$ . The physical position of a point with the parameters  $\xi^1 \in \Xi^1$  and  $\xi^2 \in \Xi^2$  can be computed with the basis functions and the control points **B**. The basis functions are computed from the knot vectors for every distinct set of parameters  $\xi^1$  and  $\xi^2$ . So necessary information for a NURBS patch include the two knot vectors  $\Xi^1$  and  $\Xi^2$ , the control points **B**, the order of basis functions  $p$  in direction of  $\Xi^1$ ,  $q$  in direction of  $\Xi^2$  and the number of control points  $n$  in direction of  $\Xi^1$  and  $m$  in direction of  $\Xi^2$ . NURBS geometries may consist of several patches. Every patch has its own control points and constants described above. Common control points of patches have to be given separately, as the sequence of control points may not be disturbed.

### 11.3 Known Errors

The plot macros **disp** and **base** do not work. Please report further bugs to [wolfgang.dornisch@bauing.uni-kl.de](mailto:wolfgang.dornisch@bauing.uni-kl.de).

## Chapter 12

# F E A P – CYLT Mesh Generator for Yield-Line Predictions

### 1. Preliminaries

This is a short instruction for the yield-line mesh prediction program **CYLT – Computational Yield-Line Theory** by J. Wüst, Institut für Baustatik, Universität Karlsruhe.

The program is adjusted to the finite element program **FEAP** being executable by Windows9x/NT/2000/ME/XP where further steps of yield-line calculation and optimization are performed.

### 2. Description

**CYLT** is a 2-dimensional mesh generator for yield-line prediction. Depending on the type option it shows the yield-line pattern or corresponding triangulation steps, see Section 3 or **cylt** command.

The generation is achieved by plane rotations where the edges of the considered plate define the rotation axes. Ruled by the **yedg** command the rotation angles will be defined to  $45^\circ$  in case of a supported edge or to  $90^\circ$  in case of a free edge. The intersection lines between the correspondingly deflected planes form potential yield-lines, see Fig. 1, and the intersection points of three plates yield potential branching points of yield-lines. In this way a possible failure geometry is found which can be optimized by a later optimization calculation performed by **FEAP**.

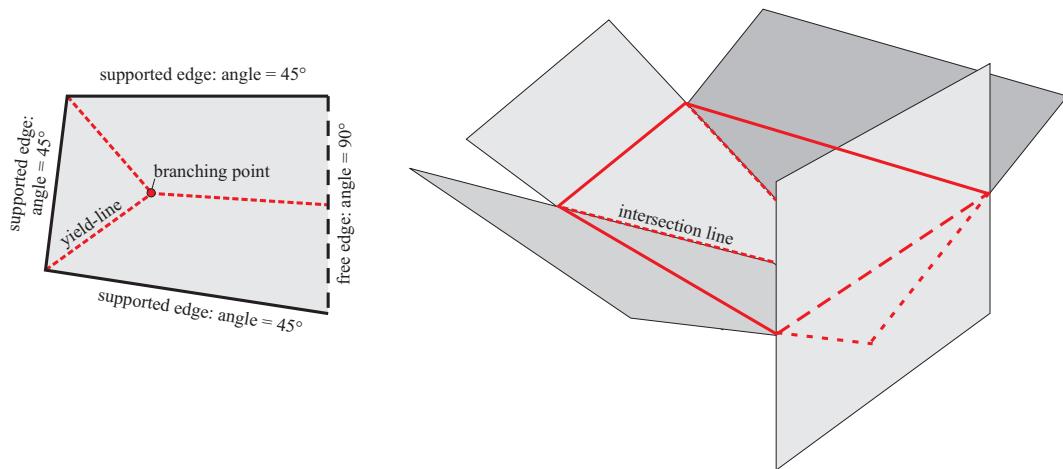


Fig. 1: Plane intersections forming yield-lines

The prediction algorithm detects all admissible configurations of yield-line patterns which will be numbered. In a further step the user can choose between one or more configurations depending on the given plate geometry. For internal reasons it may be possible that one yield-line pattern will be represented by several configuration numbers, see Section 4.

In order to perform a calculation with the triangular yield-line element in **FEAP** it is necessary to triangulate the yield-line pattern as mentioned in the beginning of this abstract. The configuration numbers will be the same for all type options.

### 3. General Input

The input file is divided into two parts. The first part rules the **CYLT** yield-line mesh generation, the second one is needed by **FEAP** to perform a calculation using the corresponding triangular yield-line element. Its general form reads as follows:

<pre> CYLT &lt;problem name&gt; &lt;type&gt;  <b>ycon</b>  <b>ynod</b> &lt;node_i&gt;, &lt;x&gt;, &lt;y&gt;  <b>yedg</b> &lt;edge_i&gt;, &lt;node_i&gt;, &lt;node_j&gt;, &lt;angle&gt; </pre>	<ul style="list-style-type: none"> <li>• header with optional problem name           <ul style="list-style-type: none"> <li>· triangulation type option</li> </ul> </li> <li>• optional parameter definition</li> <li>• nodal coordinates of the plate           <ul style="list-style-type: none"> <li>· node number, coordinates</li> </ul> </li> <li>• edge definition           <ul style="list-style-type: none"> <li>· edge number, nodes, support</li> </ul> </li> </ul>
<pre> ... <b>mate</b>  <b>edge</b>  <b>ybou</b>  ... <b>solv</b> 11 </pre>	<ul style="list-style-type: none"> <li>• material properties of YLT-element           <ul style="list-style-type: none"> <li>· including <math>m_{pl}</math> of area</li> </ul> </li> <li>• definition of b.c. along line in FEAP           <ul style="list-style-type: none"> <li>(other FEAP commands are also possible)</li> </ul> </li> <li>• boundary conditions of YLT-element           <ul style="list-style-type: none"> <li>· including <math>m_{pl}</math> of edges</li> </ul> </li> <li>• switch to simplex optimization solver</li> </ul>

#### 3.1. The CYLT Part

The header **CYLT** introduces the mesh generation which is directed by the **<type>** option in the following line.

- **<type>** = 0: prediction of yield-line pattern(s).

It is recommended to check the predicted yield-line pattern at the beginning. Note: this type of mesh generation cannot be used for a calculation.

- <type> = i (for  $i > 0$ ): triangulation step.

For a calculation with the triangular yield-line element in **FEAP** it is necessary to use <type> = 1.

Polygonal plate areas (see Fig. 1) will be divided into triangles. Higher triangulation steps are optional to other programs; the successively subdivided mesh structure is shown in Fig. 2.

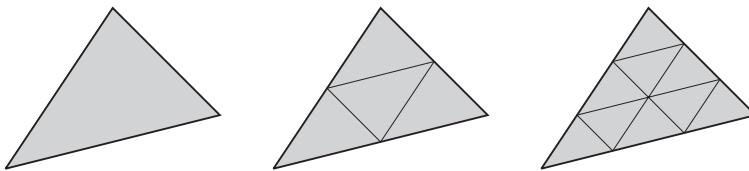


Fig. 2: Triangulation steps defined by <type> = 1, 2 and 3

With the optional **ycon** command parameters can be defined similar to **para** in **FEAP**.

The nodes of the plate border have to be input counterclockwise together with their coordinates using the **ynod** command.

In the next step the edges will be given with their numbers and nodes introduced by the **yedg** command. The last entry of each line defines the angle of the plane deflection in order to detect intersection lines, see Fig. 1. The convention is

- <angle> = 0: free edge.  
By creating the yield-line pattern the corresponding planes will be deflected with  $90^\circ$ .
- <angle> = 1: supported edge.  
The deflection angle will be  $45^\circ$ .

Note: this command rules the yield-line mesh generation with **CYLT** whereas **ybou** specify the boundary conditions of edges in **FEAP** (e.g. plastic moments, clamping, etc.). In **CYLT** only the division into supported and non-supported –i.e. free– edges is needed.

### 3.2. The FEAP Part

The **FEAP** part defines an element type with the **mate** command. In order to perform a yield-line optimization process the corresponding triangular yield-line element has to be chosen. The upper and lower plastic moments have to be given for both coordinate directions; in this way orthotropic reinforcement can be modelled.

To specify the boundary conditions of an edge –especially in order to define individual upper and lower plastic moments  $m_{pl,u}$  and  $m_{pl,l}$  – the **ybou** command has been implemented into **FEAP**. This information contributes to the internal work calculation. In the subsequent part of geometrical optimization regular **FEAP** macros defining boundary conditions like **poin** or mainly **edge** are used. In this way the search directions during the optimization process are ruled: all degrees of freedom within the plate plane have to be left free.

Note: symmetrical problems can be mapped using free edges with plastic moments.

For further details see the **mesh** manual or the examples of Section 5.

### 4. Remarks

At the end of the prediction process the user has to select a yield-line mesh being submitted to **FEAP**. The number of which has nothing in common with the optimal solution. It is strongly recommended to perform all by starting a new calculation.

Especially symmetrical geometries may produce several numbers representing identical configurations. The reason is that they are internally encoded differently, see Fig. 3. So the calculation results will be the same.

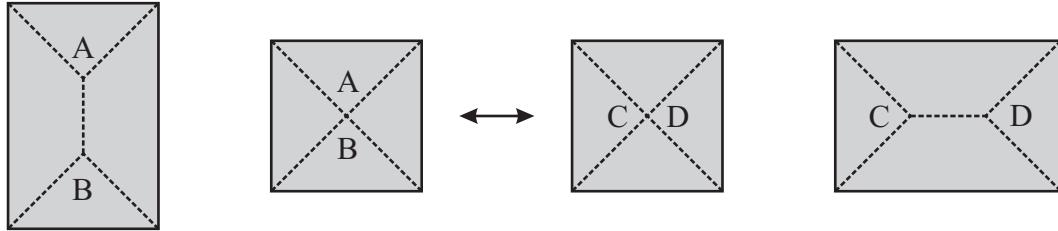


Fig. 3: A quadratic plate signify the link between two rectangular plates

## 5. Examples

The following examples show the usage of the **CYLT** mesh generator in context to the **FEAP** yield-line optimization process. The required macro commands will be found in the [macro](#) manual.

Note: Using the [ygra](#) command it may be possible that an initial step length is required which can be given by the command [dt](#).

### Example 5.1: Rectangular Plate

This simple yield-line geometry of a rectangular plate can be reproduced by hand. All edges are simply supported.

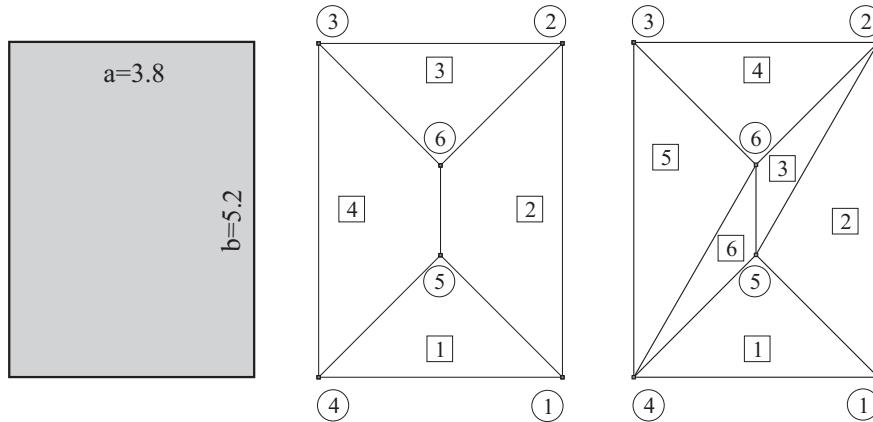


Fig. 4: Example 5.1: Geometry and mesh solution for `<type> =0` and `<type>=1`

### Notes:

- Defining different plastic moments at the boundaries ([ybou](#) command) will produce a non-symmetrical solution during the optimization process in **FEAP**. The mesh generation performed by **CYLT** does not depend on material parameters, clamping moments, etc.
- Contemporarily the number of the yield-line element is 40, the simplex optimization solver has the number 11.

```
CYLT Rectangular Plate
```

```
1
```

```
ycon
```

```
a=3.8
```

```
b=5.2
```

```
ynod
```

```
1, 0, 0
```

```
2, a, 0
```

```
3, a, b
```

```
4, 0, b
```

```
yedg
```

```
1,1,2,1
```

```
2,2,3,1
```

```
3,3,4,1
```

```
4,4,1,1
```

```
cons
```

```
a=3.8
```

```
b=5.2
```

```
m=6.558
```

```
edge
```

```
0,0,,a,0
```

```
1,1,1
```

```
a,0,,a,b
```

```
1,1,1
```

```
0,b,,a,b
```

```
1,1,1
```

```
0,0,,0,b
```

```
1,1,1
```

```
mate
```

```
1, 40 <ylt-element-number>
```

```
1,m,m,m,m
```

```
ybou
```

```
0, 0,, a, 0,,1,0,0
```

```
a, 0,, a, b,,1,0,0
```

```
0, b,, a, b,,1,0,0
```

```
0, 0,, 0, b,,1,0,0
```

```
solv
```

```
11 <simplex optimization solver>
```

```
end
```

```
inte
```

```
stop
```

**Example 5.2: Modelling of a Symmetrical Problem**

With the definition of a free edge clamped by the same plastic moment as the plate area the precedent example can be modelled symmetrically. The following input data shows the differences compared to Example 5.1. Edge 4 between the nodes 4 and 1 will be the new symmetrical axis. The corresponding entry for the **edge** command (see first line) allows that node points –which will be searching points for the optimization process– move vertically and along the edge line; the perpendicular direction is fixed.

```
CYLT Rectangular Plate (Symmetrical)
```

```
1
```

```
ycon
```

```
a=3.8
```

```
b=2.6
```

```
ynod
```

```
1, 0, 0
```

```
2, a, 0
```

```
3, a, b
```

```
4, 0, b
```

```
yedg
```

```
1,1,2,1
```

```
2,2,3,1
```

```
3,3,4,1
```

```
4,4,1,0
```

```
...
```

```
edge
```

```
0,0,,a,0
```

```
1,0,0
```

```
a,0,,a,b
```

```
1,1,1
```

```
0,b,,a,b
```

```
1,1,1
```

```
0,0,,0,b
```

```
1,1,1
```

```
...
```

```
ybou
```

```
0, 0,, a, 0,,1,m,m
```

```
a, 0,, a, b,,1,0,0
```

```
0, b,, a, b,,1,0,0
```

```
0, 0,, 0, b,,1,0,0
```

```
...
```

**Example 5.3: Different Yield-line Configurations**

An example of two different mesh solutions is shown. The following **CYLT** part of an input file produces the two yield-line configurations in Fig. 5.

```

cylt Example with two yield-line configurations
0

ycon
a=0.5
b=0.5
k=0.25

ynod
1,    0,2*b,
2,    k,   b
3,    k,   0
4,  2*a,  0
5,2*a-k,  b
6,2*a-k,2*b

yedg
1, 1, 2, 1
2, 2, 3, 1
3, 3, 4, 1
4, 4, 5, 1
5, 5, 6, 1
6, 6, 1, 1

```

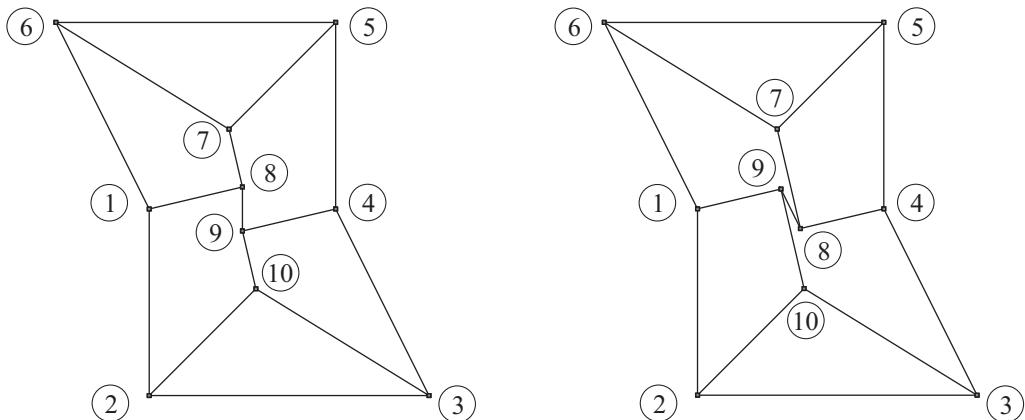


Fig. 5: Example 5.3: One plate geometry with two yield-line configurations

#### Example 5.4: Yield-line Optimization Performed by FEAP

The yield-line element implemented in FEAP uses geometrical relations containing nodal displacements and edge rotations. Since three points define a plane it is a triangular element.

Two different calculation types can be employed for the yield-line calculation: the gradient optimization method or the direct search method. Beyond the CYLT mesh generator the following arbitrary example will show the differences – Fig. 6b displays the search directions respectively degrees of freedom.

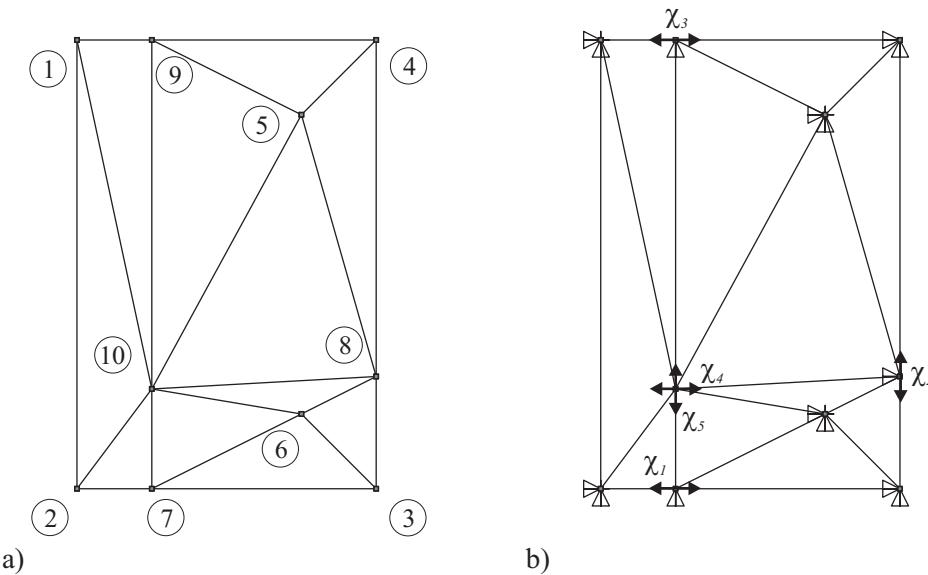


Fig. 6: Example 5.4: Two ways of yield-line calculations

Coordinates:

1 :	[ 0,00   0,00 ] ;	6 :	[ 9,00   15,00 ]
2 :	[ 0,00   18,00 ] ;	7 :	[ 3,00   18,00 ]
3 :	[ 12,00   18,00 ] ;	8 :	[ 12,00   13,50 ]
4 :	[ 12,00   0,00 ] ;	9 :	[ 3,00   0,00 ]
5 :	[ 9,00   3,00 ] ;	10 :	[ 3,00   14,00 ]

#### Direct Search Method

The direct search method is a simple approach of trial steps into the search directions. It is a very robust method but it is quite slow and takes increasing resources depending on the number of degrees of freedom. In every step the limit load has to be calculated in order to find the descend slope. The following macro commands have to be executed consecutively:

<b>ytab</b>	• calculate simplex tableau
<b>ytry</b>	• perform a searching step (trial)

## Gradient Method

The gradient method moves along the slope of the objective function and is therefore very effective. It is necessary to define a step length which will be determined by FEAP. Otherwise it can be given manually by the **dt** command. Especially in the beginning of a calculation it may be necessary to define an initial step length. The procedure is as follows:

<b>ytab</b>	• calculate simplex tableau
<b>iyang;</b>	• print yield-line rotation angles (optional)
<b>ygra</b>	• calculate gradient
<b>ymsh</b>	• determine step length and remesh

In this example FEAP calculates a zero step length within the first iteration step. So an initial value (e.g. here **dt,,1** has to be given. After the next turn of the presented procedure it should be removed by **dt,,0** – it will be calculated automatically again. The following iterations lead to the load minimum. As the implemented optimization algorithm contains a sensitivity analysis, the method is able to detect a lower value than the direct search method. The reason is that the influence between the moves of the degrees of freedom is taken into account – the direct search method stacks at a local minimum.

Fig. 7 shows the different solutions, Fig. 8 gives an idea of the optimization function depending on several degrees of freedom.

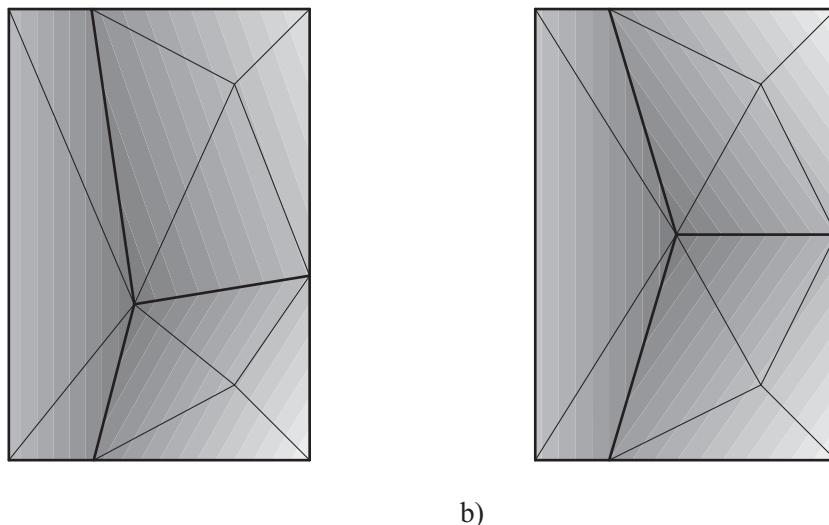


Fig. 7: a) Direct search – b) Gradient method

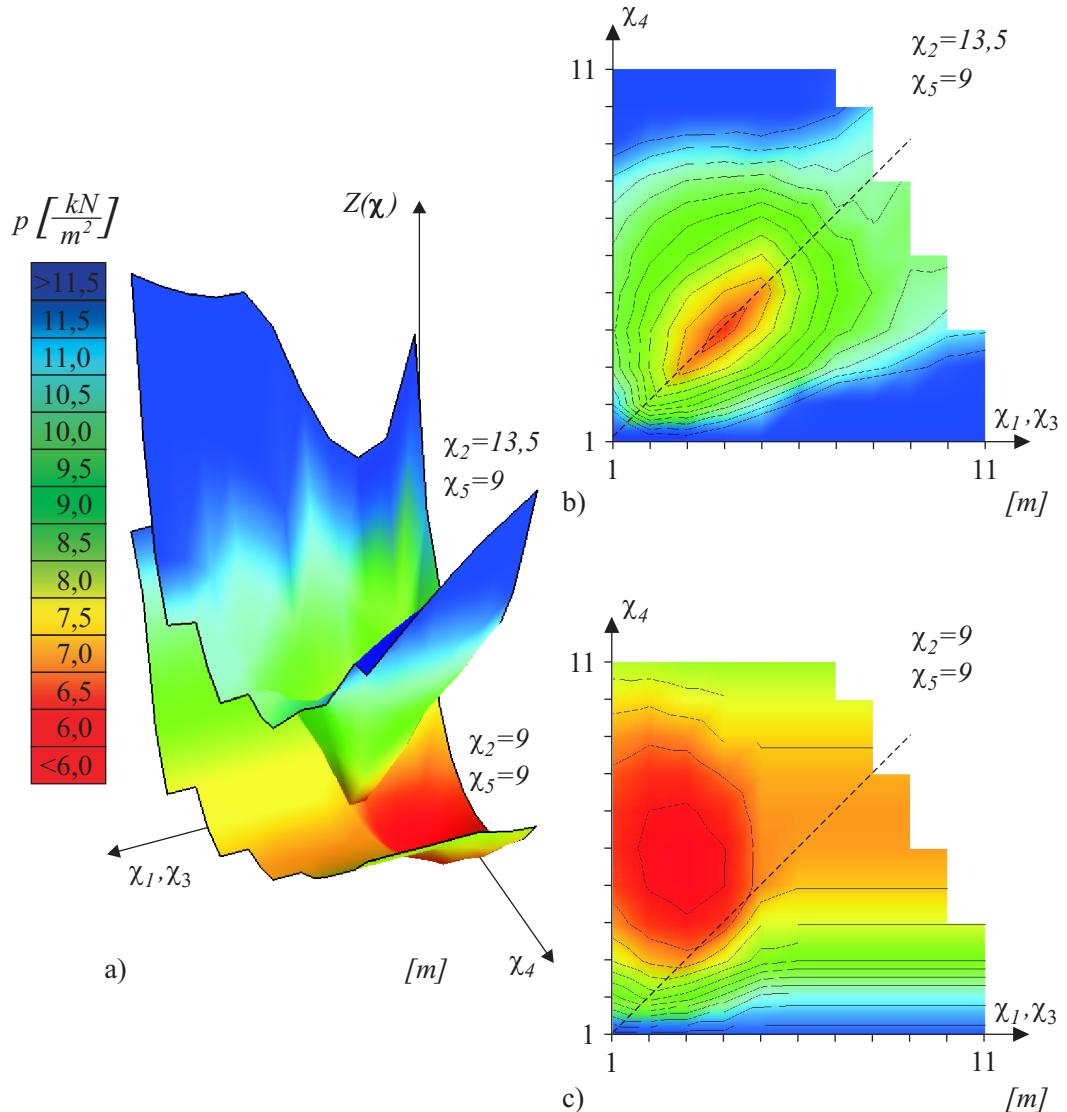


Fig. 8: Optimization function

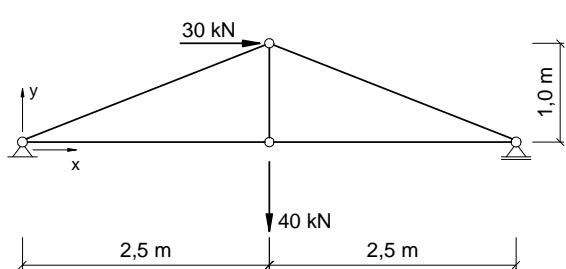
## Chapter 13

### F E A P - Beispielsammlung

## 13.1 Beispiele für 2D/3D - Fachwerkelement

### 13.1.1 Beispiel 1 - Dachbinder

#### 13.1.1.1 System und Belastung



Querschnittswerte: Nadelholz,  $10 \times 10$   
 $E = 1000 \text{ kN/cm}^2$   
 $A = 100,0 \text{ cm}^2$

gesucht: Schnittgrößen,  
Knotenverschiebungen

#### 13.1.1.2 Eingabedatensatz (file: it01)

```
feap ebenes Fachwerk, 1. Beispiel: Dachbinder
4, 5, 1, 2, 2, 2, 0, 0
```

```
coor
1, 1, 0.000, 0.000
3, 0, 5.000, 0.000
4, 0, 2.500, 1.000
```

```
elem
1, 1, 1, 2, 1
4, 1, 1, 4
5, 1, 2, 4
```

```
boun
1, 0, 1, 1
3, 0, 0, 1
```

```
load
2, , 0.00, -40.00
4, , 30.00, 0.00
```

```
mate
1, 1
1000.d+04, 100.d-04, , NH 10 x 10
```

```
end
```

```
inte
```

```
stop
```

### 13.1.1.3 Ergebnisse

Die Stabkräfte, Dehnungen und Längsspannungen ergeben sich unter der angegebenen Belastung zu:

T r u s s   E l e m e n t

elem	mate	1-coord	2-coord	3-coord	force	strain	stress
1	1	1.2500	0.0000	0.0000	0.65000E+02	0.65000E-03	0.65000E+04
2	1	3.7500	0.0000	0.0000	0.65000E+02	0.65000E-03	0.65000E+04
3	1	3.7500	0.5000	0.0000	-0.70007E+02	-0.70007E-03	-0.70007E+04
4	1	1.2500	0.5000	0.0000	-0.37696E+02	-0.37696E-03	-0.37696E+04
5	1	2.5000	0.5000	0.0000	0.40000E+02	0.40000E-03	0.40000E+04

Die Knotenverschiebungen sind:

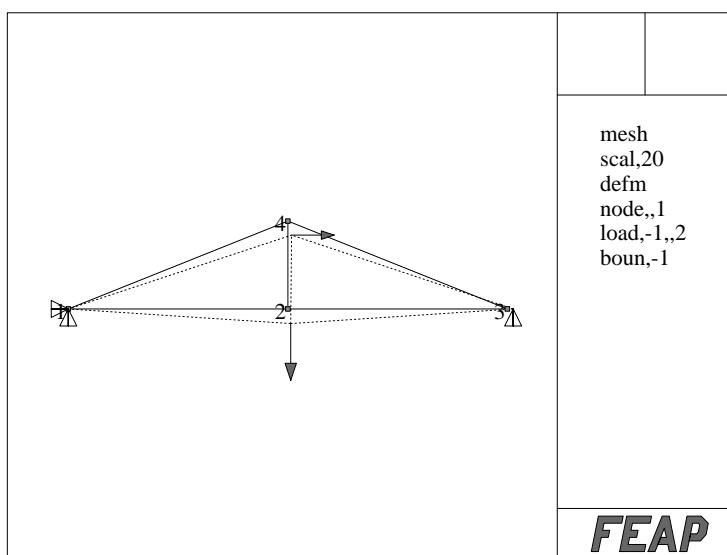
n o d a l	d i s p l a c e m e n t s	time	0.00000E+00	
node	1 coord	2 coord	1 displ	2 displ
1	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
2	2.50000E+00	0.00000E+00	1.62500E-03	-8.36674E-03
3	5.00000E+00	0.00000E+00	3.25000E-03	0.00000E+00
4	2.50000E+00	1.00000E+00	2.09351E-03	-7.96674E-03

### 13.1.1.4 Vergleichslösung

Eine Handrechnung mit dem PVK ergibt als vertikale Verschiebung für den Knoten 2:

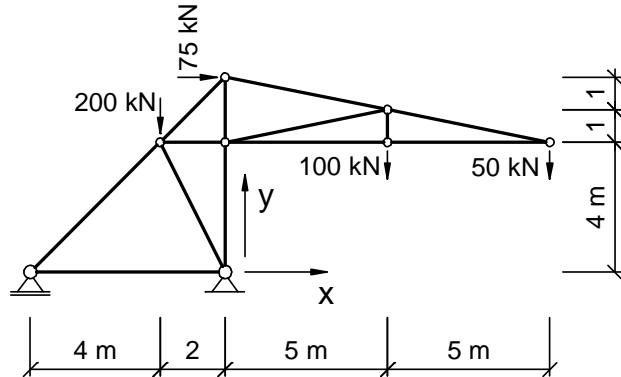
$$\delta_v = \int \frac{N\bar{N}}{EA} = 0,83667 \text{ cm}$$

### 13.1.1.5 verformte Struktur (20-fach überhöht)



### 13.1.2 Beispiel 2 - Kran

#### 13.1.2.1 System und Belastung



Querschnittswerte: St 37, I 300  
 $E = 21000 \text{ kN/cm}^2$   
 $I = 9800 \text{ cm}^4$   
 $A = 69,0 \text{ cm}^2$

Belastung: wie angegeben in Skizze

gesucht: Schnittgrößen,  
 Knotenverschiebungen

#### 13.1.2.2 Eingabedatensatz (file: it02)

```
feap ebenes Fachwerk, 2. Beispiel: Kran
8, 13, 1, 2, 2, 2, 0, 0
```

```
coor
1, 0, -6.000, 0.000
2, 0, 0.000, 0.000
3, 0, -2.000, 4.000
4, 1, 0.000, 4.000
6, 1, 10.000, 4.000
8, 0, 0.000, 6.000
```

```
elem
1, 1, 1, 2, 1
8, 1, 5, 7,
9, 1, 4, 7,
10, 1, 4, 8,
11, 1, 3, 8,
12, 1, 2, 4,
13, 1, 1, 3,
```

boun	mate
1, 0, 0, 1	1, 1
2, 0, 1, 1	21000.d+04, 69.d-04, , I 300

load	end
3, , 0.00, -200.00	
5, , 0.00, -100.00	inte
6, , 0.00, -50.00	
8, , 75.00, 0.00	stop

### 13.1.2.3 Ergebnisse

Die Stabkräfte, Dehnungen und Längsspannungen ergeben sich unter der angegebenen Belastung zu:

#### Truss Element

elem	mate	1-coord	2-coord	3-coord	force	strain	stress
1	1	-3.0000	0.0000	0.0000	-0.17500E+03	-0.12077E-03	-0.25362E+05
2	1	-1.0000	2.0000	0.0000	0.22361E+03	0.15432E-03	0.32407E+05
3	1	-1.0000	4.0000	0.0000	-0.50000E+03	-0.34507E-03	-0.72464E+05
4	1	2.5000	4.0000	0.0000	-0.25000E+03	-0.17253E-03	-0.36232E+05
5	1	7.5000	4.0000	0.0000	-0.25000E+03	-0.17253E-03	-0.36232E+05
6	1	7.5000	4.5000	0.0000	0.25495E+03	0.17595E-03	0.36949E+05
7	1	2.5000	5.5000	0.0000	0.50990E+03	0.35190E-03	0.73899E+05
8	1	5.0000	4.5000	0.0000	0.10000E+03	0.69013E-04	0.14493E+05
9	1	2.5000	4.5000	0.0000	-0.25495E+03	-0.17595E-03	-0.36949E+05
10	1	0.0000	5.0000	0.0000	-0.67500E+03	-0.46584E-03	-0.97826E+05
11	1	-1.0000	5.0000	0.0000	0.81317E+03	0.56120E-03	0.11785E+06
12	1	0.0000	2.0000	0.0000	-0.72500E+03	-0.50035E-03	-0.10507E+06
13	1	-4.0000	2.0000	0.0000	0.24749E+03	0.17080E-03	0.35868E+05

Die Knotenverschiebungen sind:

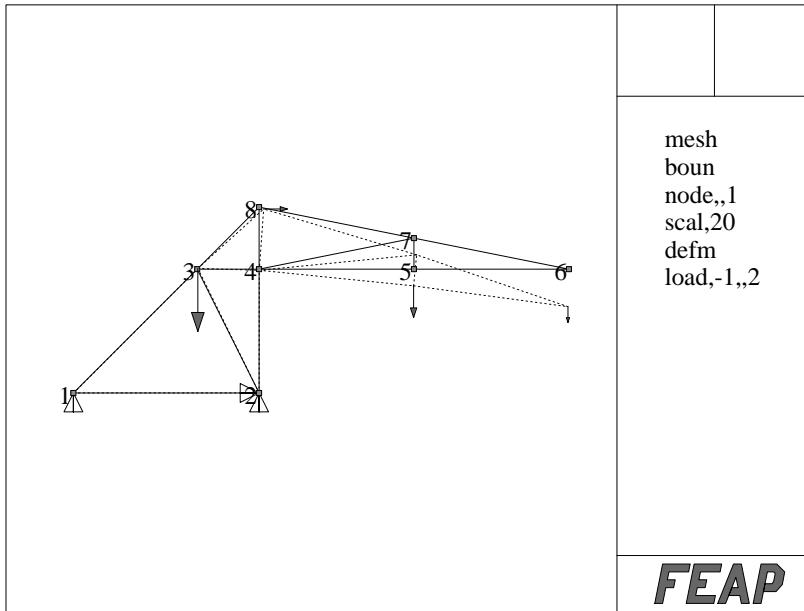
node	1 coord	2 coord	1 displ	2 displ
1	-6.00000E+00	0.00000E+00	7.24638E-04	0.00000E+00
2	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
3	-2.00000E+00	4.00000E+00	8.79625E-04	1.21140E-03
4	0.00000E+00	4.00000E+00	1.89494E-04	-2.00138E-03
5	5.00000E+00	4.00000E+00	-6.73170E-04	-2.70967E-02
6	1.00000E+01	4.00000E+00	-1.53583E-03	-6.06806E-02
7	5.00000E+00	5.00000E+00	4.27982E-03	-2.70277E-02
8	0.00000E+00	6.00000E+00	7.26887E-03	-2.93306E-03

### 13.1.2.4 Vergleichslösung

Die Stabkräfte können leicht über Knotengleichgewicht überprüft werden.

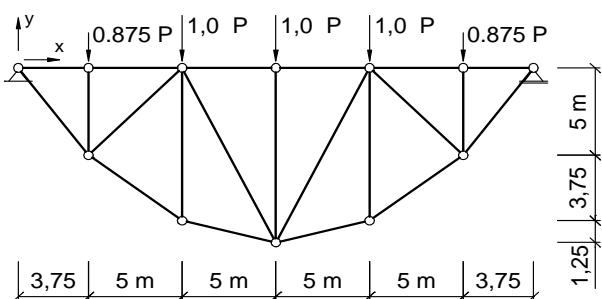
Die Verschiebungen können mit dem PVK errechnet werden.

### 13.1.2.5 verformte Struktur (20-fach überhöht)



### 13.1.3 Beispiel 3 - Brückenbogen

#### 13.1.3.1 System und Belastung



Querschnittswerte: Obergurt: HEM 300

$$I = 59200 \text{ cm}^4$$

$$A = 303,0 \text{ cm}^2$$

Untergurt: HEM 160

$$I = 5100 \text{ cm}^4$$

$$A = 97,1 \text{ cm}^2$$

Pfosten: HEM 100

$$I = 1140 \text{ cm}^4$$

$$A = 53,2 \text{ cm}^2$$

Diagonalen: HEM 140

$$I = 3290 \text{ cm}^4$$

$$A = 80,6 \text{ cm}^2$$

Belastung:  $P = 250 \text{ kN}$

wie in Skizze aufgebracht

gesucht: Schnittgrößen,

Knotenverschiebungen

#### 13.1.3.2 Eingabedatensatz (file: it03)

```
feap ebenes Fachwerk, 3. Beispiel: Bruecke
12, 21, 4, 2, 2, 2, 0, 0
```

```
c l = Laenge
c h = Hoehe
c P = Last vertikal
```

```
cons
l= 5.d0
h= 5.d0
p= 250.d0

coor
1, 0, 0.000,    0.000
2, 0, 0.750*l,  0.000
3, 0, 1.750*l,  0.000
4, 0, 2.750*l,  0.000
5, 0, 3.750*l,  0.000
6, 0, 4.750*l,  0.000
7, 0, 5.500*l,  0.000
8, 0, 0.750*l, -1.000*h
9, 0, 1.750*l, -1.750*h
10, 0, 2.750*l, -2.000*h
11, 0, 3.750*l, -1.750*h
12, 0, 4.750*l, -1.000*h
```

```
elem
 1, 1, 1, 2, 1
 7, 2, 1, 8, 1
 8, 2, 8, 9, 1
 12, 2, 12, 7, 1
 13, 3, 2, 8, 1
 18, 4, 3, 8, 1
 19, 4, 3, 10, 1
 20, 4, 5, 10, 1
 21, 4, 5, 12, 1

boun
 1, 0, 1, 1
 7, 0, 0, 1

load
 2, , 0.00, -0.875*p
 3, 1, 0.00, -1.000*p
 5, , 0.00, -1.000*p
 6, , 0.00, -0.875*p

mate
 1, 1
 21000.d+04, 303.d-04, , HEM 300
 2, 1
 21000.d+04, 97.1d-04, , HEM 160
 3, 1
 21000.d+04, 53.2d-04, , HEM 100
 4, 1
 21000.d+04, 80.6d-04, , HEM 140

end

inte

stop
```

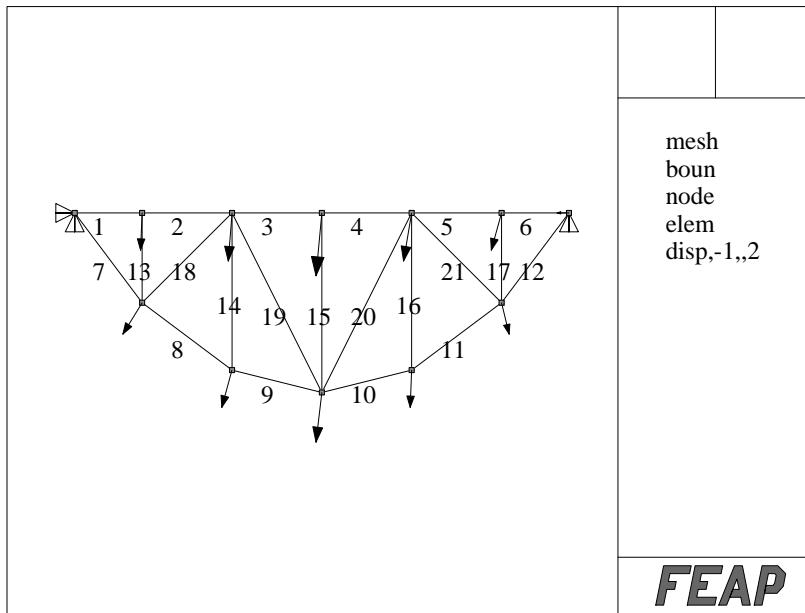
### 13.1.3.3 Ergebnisse

Die Stabkräfte können leicht über Knotengleichgewicht überprüft werden.  
Die Verschiebungen können mit dem PVK errechnet werden.

Die Knotenverschiebungen sind:

node	1 coord	2 coord	1 displ	2 displ
1	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
2	3.75000E+00	0.00000E+00	-2.62443E-04	-6.24625E-03
3	8.75000E+00	0.00000E+00	-6.12366E-04	-8.08225E-03
4	1.37500E+01	0.00000E+00	-9.83776E-04	-1.05702E-02
5	1.87500E+01	0.00000E+00	-1.35519E-03	-8.08225E-03
6	2.37500E+01	0.00000E+00	-1.70511E-03	-6.24625E-03
7	2.75000E+01	0.00000E+00	-1.96755E-03	0.00000E+00
8	3.75000E+00	-5.00000E+00	-3.23155E-03	-5.26724E-03
9	8.75000E+00	-8.75000E+00	-1.72114E-03	-6.24660E-03
10	1.37500E+01	-1.00000E+01	-9.83776E-04	-8.33246E-03
11	1.87500E+01	-8.75000E+00	-2.46412E-04	-6.24660E-03
12	2.37500E+01	-5.00000E+00	1.26400E-03	-5.26724E-03

### 13.1.3.4 Struktur und Verschiebungen



### 13.1.4 Beispiel 4 - Raumfachwerk

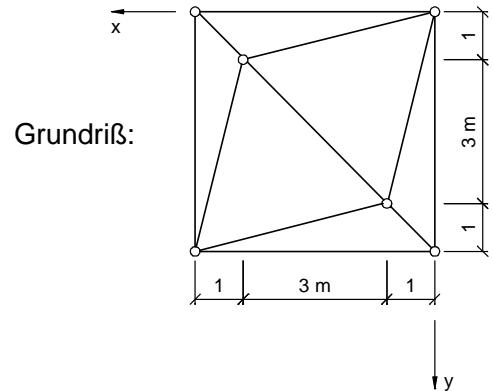
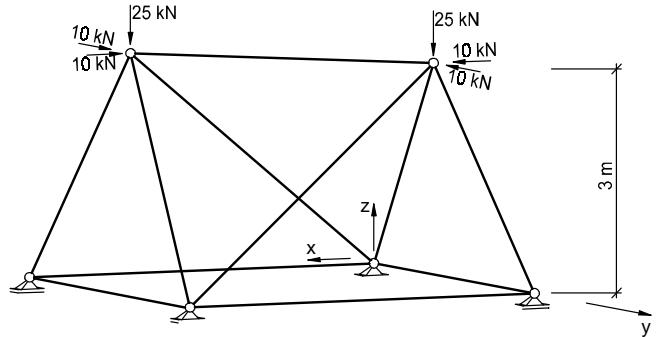
#### 13.1.4.1 System und Belastung

Querschnittswerte: St 37, Rohr  $101,6 \times 2,9$

$$\begin{aligned} E &= 21000 \text{ kN/cm}^2 \\ A &= 8,99 \text{ cm}^2 \end{aligned}$$

Belastung:  $P$  wie angegeben in Skizze

gesucht: Schnittgrößen,  
Knotenverschiebungen



#### 13.1.4.2 Eingabedatensatz (file: it04)

### 13.1.4.3 Ergebnisse

Die Stabkräfte, Dehnungen und Längsspannungen ergeben sich unter der angegebenen Belastung zu:

#### Truss Element

elem	mate	1-coord	2-coord	3-coord	force	strain	stress
1	1	2.5000	0.0000	0.0000	0.59524E+01	0.31529E-04	0.66211E+04
2	1	5.0000	2.5000	0.0000	0.59524E+01	0.31529E-04	0.66211E+04
3	1	2.5000	5.0000	0.0000	0.59524E+01	0.31529E-04	0.66211E+04
4	1	0.0000	2.5000	0.0000	0.00000E+00	0.00000E+00	0.00000E+00
5	1	2.0000	0.5000	1.5000	-0.60703E+01	-0.32154E-04	-0.67522E+04
6	1	4.5000	0.5000	1.5000	-0.19742E+02	-0.10457E-03	-0.21960E+05
7	1	4.5000	3.0000	1.5000	-0.60703E+01	-0.32154E-04	-0.67522E+04
8	1	3.0000	4.5000	1.5000	-0.60703E+01	-0.32154E-04	-0.67522E+04
9	1	0.5000	4.5000	1.5000	-0.19742E+02	-0.10457E-03	-0.21960E+05
10	1	0.5000	2.0000	1.5000	-0.60703E+01	-0.32154E-04	-0.67522E+04
11	1	2.5000	2.5000	3.0000	-0.17509E+02	-0.92745E-04	-0.19476E+05

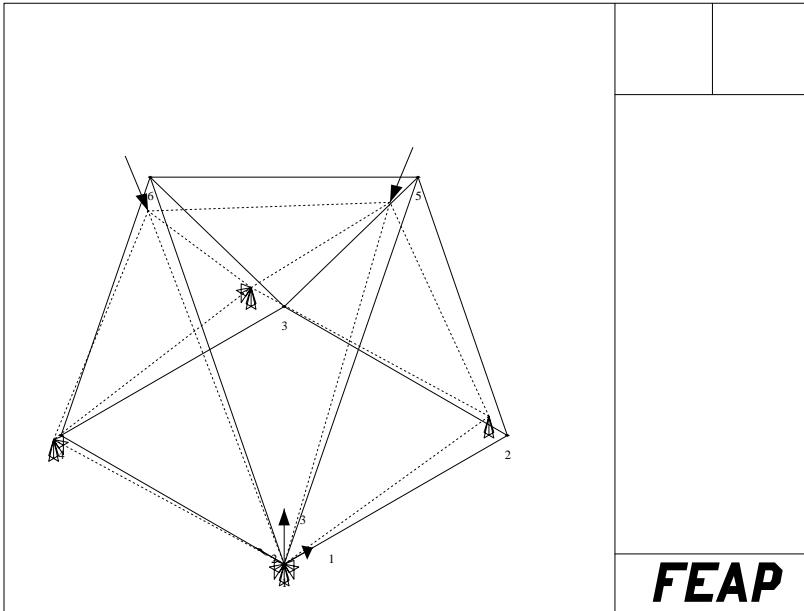
Die Knotenverschiebungen sind:

node	1 coord	2 coord	3 coord	1 displ	2 displ	3 displ
1	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00
2	5.00000E+00	0.00000E+00	0.00000E+00	1.57646E-04	5.54547E-04	0.00000E+00
3	5.00000E+00	5.00000E+00	0.00000E+00	0.00000E+00	7.12192E-04	0.00000E+00
4	0.00000E+00	5.00000E+00	0.00000E+00	-1.57646E-04	0.00000E+00	0.00000E+00
5	4.00000E+00	1.00000E+00	3.00000E+00	-1.65240E-05	5.86278E-04	-4.52058E-04
6	1.00000E+00	4.00000E+00	3.00000E+00	4.80531E-05	9.43854E-05	-4.20529E-04

Die Stabkräfte können leicht über Knotengleichgewicht überprüft werden.

Die Verschiebungen können mit dem PVK errechnet werden.

#### 13.1.4.4 Struktur und Verschiebungen

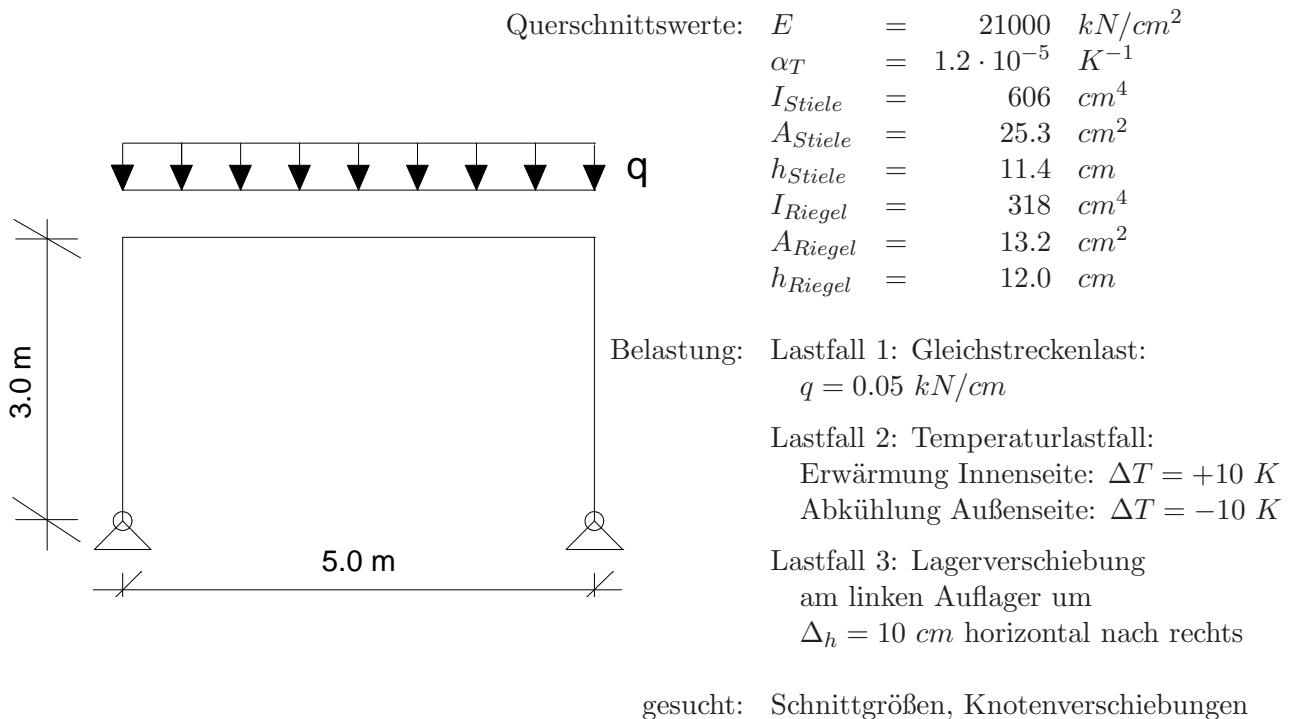


**FEAP**

## 13.2 Beispiele für 2D - Stabelement

### 13.2.1 Beispiel 1 - Zweigelenkrahmen

#### 13.2.1.1 System und Belastung



#### 13.2.1.2 Eingabedatensatz für Lastfall 1 (file: ib2d1a)

#### 13.2.1.3 Eingabedatensatz für Lastfall 2 (file: ib2d1b)

Eingabedatensatz analog zu Lastfall 1 (file: ib2d1a) mit Ausnahme folgender Änderungen:

```

mate
1,3
21000,25.3,606.,11.4,0.,0.,0.,0.,0.,0,0
1,0,0,0.000012,-10.,10.
2,3
21000,13.2,318.,12.,0.,0.,0.,0.,0.,0,0
1,0,0,0.000012,-10.,10.

```

#### 13.2.1.4 Eingabedatensatz für Lastfall 3 (file: ib2d1c)

Eingabedatensatz analog zu Lastfall 1 (file: ib2d1a) mit Ausnahme folgender Änderungen:

```
load
1.,10.,0.,0

mate
1,3
21000,25.3,606.,11.4,0.,0.,0.,0.,0,0
1,0,0,0,0,0
2,3
21000,13.2,318.,12.,0.,0.,0.,0.,0,0
1,0,0,0,0,0
```

### 13.2.1.5 Ergebnisse

Das Eckmoment und die horizontale Knotenverschiebung in der Rahmenecke betragen:

*Lastfall 1 :  $M = -860.8 \text{ kNm}$        $\Delta h = 2.58767E - 03 \text{ cm}$*   
*Lastfall 2 :  $M = -180.1 \text{ kNm}$        $\Delta h = 5.41333E - 04 \text{ cm}$*   
*Lastfall 3 :  $M = -367.9 \text{ kNm}$        $\Delta h = 5.00111E + 00 \text{ cm}$*

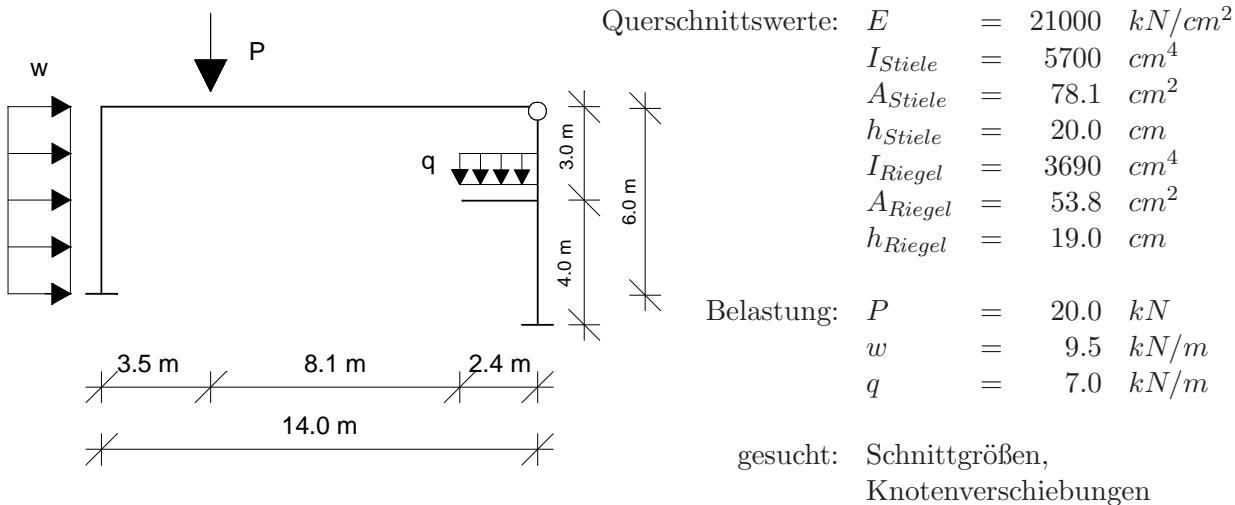
### 13.2.1.6 Vergleichslösung

Eine Handrechnung ergibt unter der Annahme von  $A \sim \infty$  für das Eckmoment im Rahmen

*Lastfall 1 :  $M = -860.95 \text{ kNm}$*   
*Lastfall 2 :  $M = -180.11 \text{ kNm}$*   
*Lastfall 3 :  $M = -367.96 \text{ kNm}$*

### 13.2.2 Beispiel 2 - Rahmentragwerk

#### 13.2.2.1 System und Belastung



#### 13.2.2.2 Eingabedatensatz (file: ib2d2)

#### 13.2.2.3 Ergebnisse

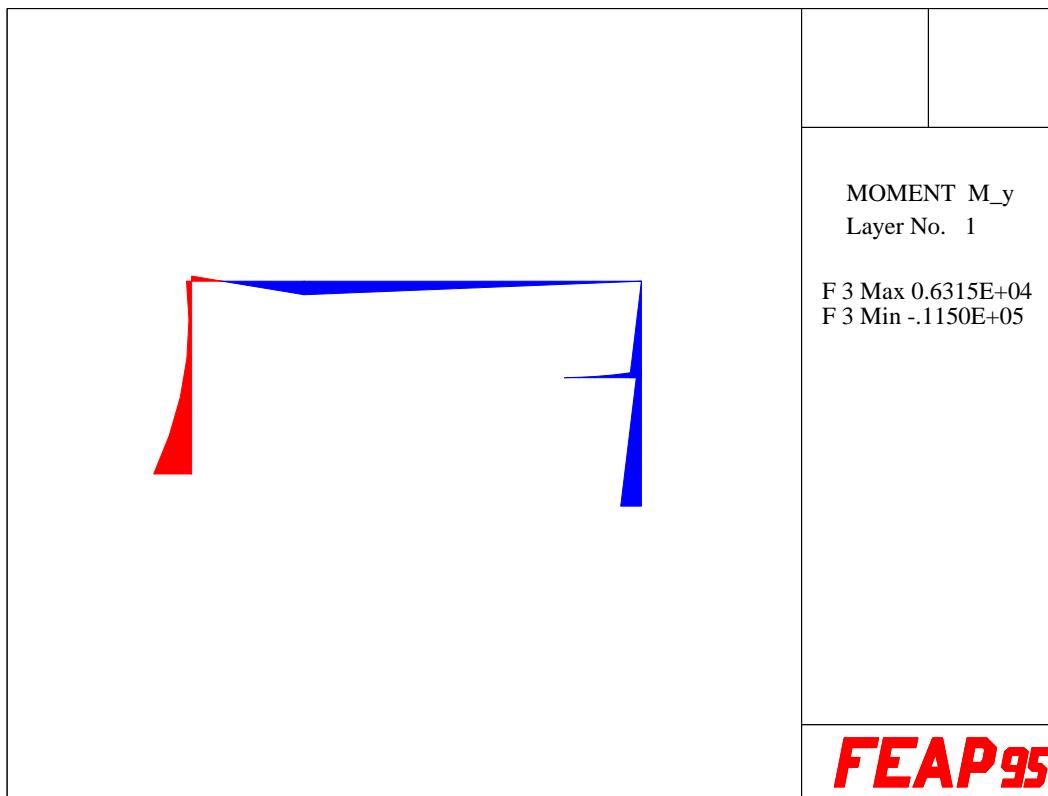
Das Eckmoment und die horizontale Knotenverschiebung in der Rahmenecke betragen:  
 $M = -15.40 \text{ kNm}$  und  $\Delta h = 8.01 \text{ cm}$ .

#### 13.2.2.4 Vergleichslösung

Eine Handrechnung ergibt unter der Annahme von  $A \rightarrow \infty$  für das Eckmoment im Rahmen  $M = -15.42 \text{ kNm}$ .

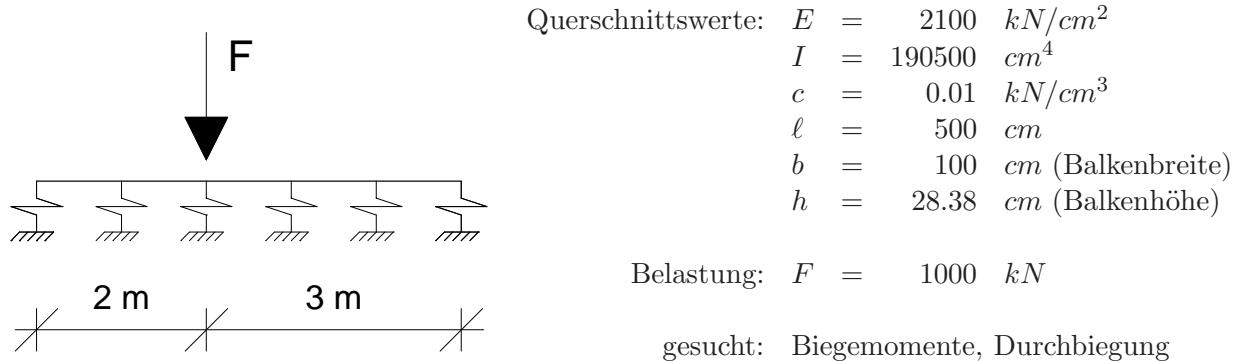
#### 13.2.2.5 Schnittkraftlinie

Angegeben ist der Verlauf der Biegemomente  $M_y$ .



### 13.2.3 Beispiel 3 - elastisch gebetteter Balken

#### 13.2.3.1 System und Belastung



#### 13.2.3.2 Eingabedatensatz (file: ib2d3)

#### 13.2.3.3 Ergebnisse

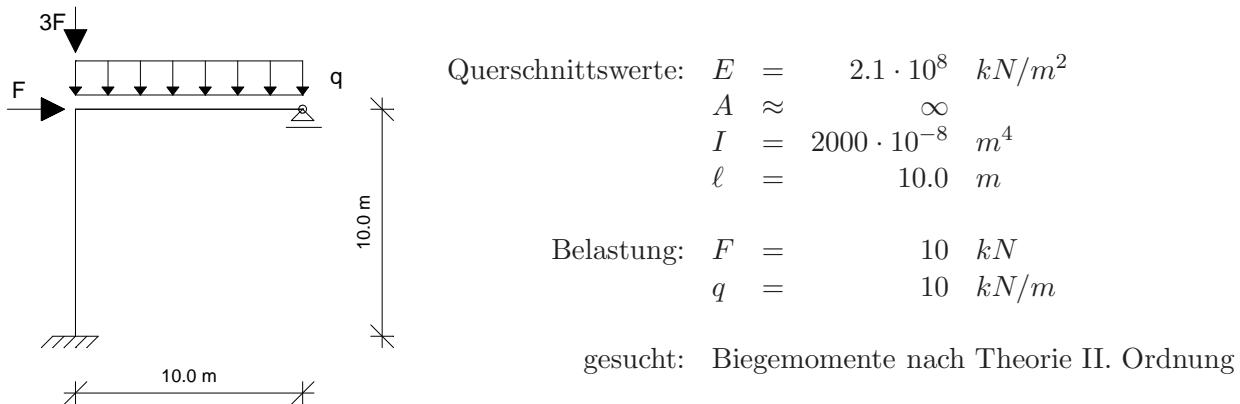
Das Biegemoment unter der Last beträgt 491.9 kNm, die Durchbiegung 2.86 cm.

#### 13.2.3.4 Vergleichslösung

Die exakte Lösung liefert für das Moment 492 kNm, für die Durchbiegung 2.86 cm.

### 13.2.4 Beispiel 4 - Halbrahmen

#### 13.2.4.1 System und Belastung



#### 13.2.4.2 Eingabedatensatz (file: ib2d4)

Makroliste:

```
DT,,1
PROP <CR>
TIME
TANG,,1      → lineare Lösung
DT,,0
TIME
TANG,,1      } jeweils eine Iteration
```

#### 13.2.4.3 Ergebnisse

Das Eckmoment und die horizontale Knotenverschiebung in der Rahmenecke betragen nach Theorie II. Ordnung nach dem ersten Iterationsschritt:

$$M = -44.0 \text{ kNm} \quad \Delta h = 1.046 \text{ cm}$$

#### 13.2.4.4 Vergleichslösung

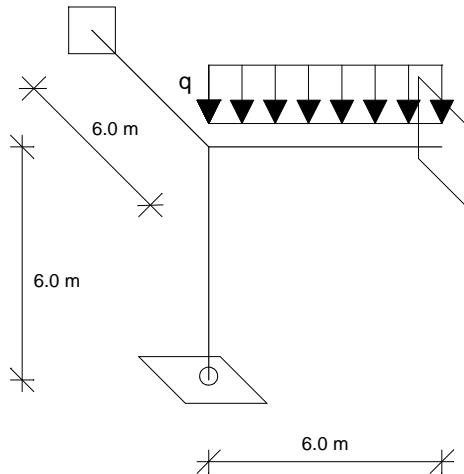
Eine Handrechnung ergibt unter der Annahme von  $A \sim \infty$  für das Eckmoment im Rahmen und die horizontale Knotenverschiebung nach dem ersten Iterationsschritt

$$M = -44.06 \text{ kNm} \quad \Delta h = 1.046 \text{ cm}$$

## 13.3 Beispiele für 3D - Stabelement

### 13.3.1 Beispiel 1 - räumliche Rahmenecke

#### 13.3.1.1 System und Belastung



Querschnittswerte:  $E = 21000 \text{ kN/cm}^2$   
 $I_y = 4000 \text{ cm}^4$   
 $I_z = 4000 \text{ cm}^4$   
 $I_T = 8000 \text{ cm}^4$   
 $G = 8100 \text{ kN/cm}^2$   
 $A \approx \infty$

Belastung:  $q = 1.00 \text{ kN/m}$

gesucht: Schnittgrößen

#### 13.3.1.2 Eingabedatensatz (file: ib3d1)

```

feap                                mate
4, 3, 2, 3, 6, 2                  1,4
                                    21000,8100,1000,8000,4000,4000,0.,0.,0.,0
coor                               0,0.
1,, 0., 0., 0.                     2,4
2,, 0.,600.,600.                  21000,8100,1000,8000,4000,4000,0.,0.,1,0
3,, 0., 0.,600.                   0,0.
4,,600., 0.,600.                 end
elem
  1,1,1,3                           inte
  2,1,3,2
  3,2,3,4                           stop

boun
  1,,1,1,1,0,0,0
  2,,1,1,1,1,1,1
  4,,1,1,1,1,1,1

```

#### 13.3.1.3 Ergebnisse

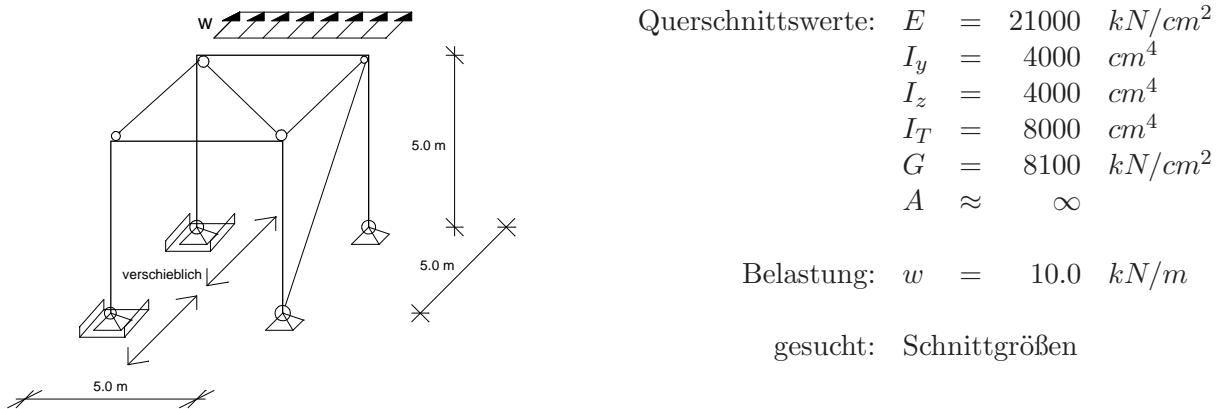
Das Eckmoment im Riegel (in der Lastebene) beträgt -14,6 kNm.

#### 13.3.1.4 Vergleichslösung

Die analytische Vergleichsrechnung ergibt als Eckmoment im Riegel (in der Lastebene) -14,6 kNm.

### 13.3.2 Beispiel 2 - räumlicher Rahmen

#### 13.3.2.1 System und Belastung



#### 13.3.2.2 Eingabedatensatz (file: ib3d2)

```

feap                                boun
12, 14, 3, 3, 6, 2                  1,,1,0,1,0,0,0
                                         4,,1,1,1,0,0,0
coor                                 5,,1,0,1,0,0,0
1, , 0., 0., 0.                      12,,1,1,1,0,0,0
2, , 0., 0.,500.
3, ,500., 0.,500.                    mate
4, ,500., 0., 0.                      1,4
5, , 0.,500., 0.                      21000,8100,10000,8000,4000,4000,0.,0.,0.,0
6,1, 0.,500.,500.                     0,0.
11, ,500.,500.,500.                  2,1
12, ,500.,500., 0.                   21000,10000,0.
                                         3,4
elem                                 21000,8100,10000,8000,4000,4000,0.,-0.1,0.,0
1,1, 1, 2,1                          0,0.
4,1, 5, 6,
5,3, 6, 7,1                          end
9,3,10,11,
10,1,11,12,
11,2, 2, 6,
12,2, 3, 6,
13,2, 3,11,
14,2, 4,11,

```

#### 13.3.2.3 Ergebnisse

Die Eckmomente in den beiden Rahmen errechnen sich zu  $\pm 62,5 \text{ kNm}$ .

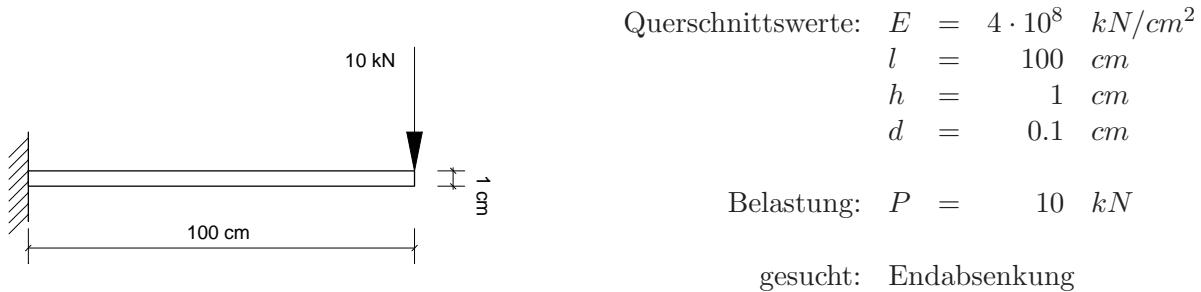
#### 13.3.2.4 Vergleichslösung

Die analytische Vergleichsrechnung unter der Annahme  $A \rightsquigarrow \infty$  ergibt für die Eckmomente in den beiden Rahmen jeweils einen Wert von  $\pm 62,5 \text{ kNm}$ .

## 13.4 Beispiele für Scheibenelemente

### 13.4.1 Beispiel 1 - Kragarm mit Einzellast

#### 13.4.1.1 System und Belastung



#### 13.4.1.2 Eingabedatensatz für Beispiel 1 (file: is1)

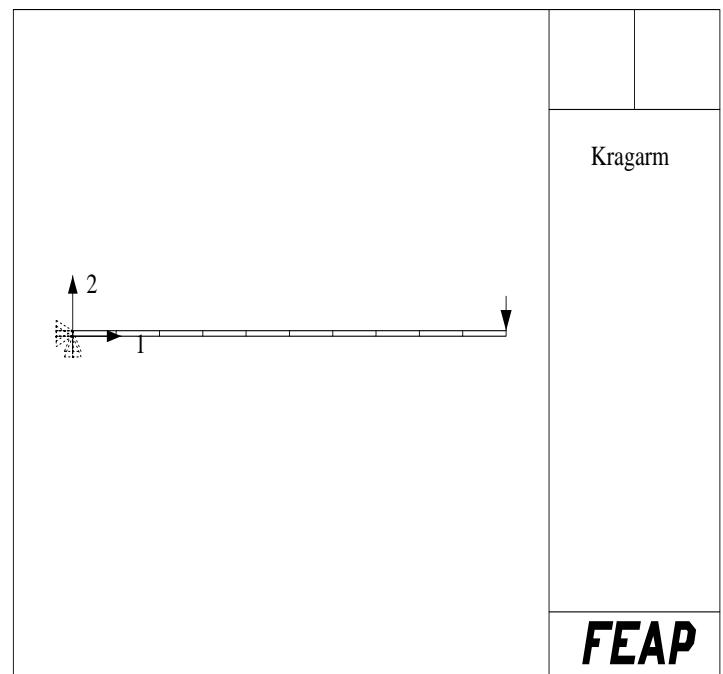
```
feap kragarm mit einzellast
22, 10, 1, 2, 2, 4
c n = Anzahl El in x
c m = Anzahl El in y
c l = Laenge
c h = Hoehe
c d = Dicke
c p = Last
```

```
cons
n=10
m=1
l=100
h=1
d=0.1
p=10
```

```
bloc
4,n,m,1,1,1,0
1,0,0
2,1,0
3,l,h
4,0,h
```

```
ebou
1,0,1,1
```

```
load
(n+1)*(m+1),0,0,-p
```



### 13.4.1.3 Ergebnisse

Die Verschiebungen der äußeren Eckknoten für die beiden Elemente betragen:

Plane Stress Linear Elastic Element (elmt05)

```
node    1 coord    2 coord    1 displ    2 displ
22 1.00000E+02 1.00000E+00 1.47059E-04-1.96088E-02
```

Plane Stress Element (elmt06)

```
node    1 coord    2 coord    1 displ    2 displ
22 1.00000E+02 1.00000E+00 7.50000E-03-9.97550E-01
```

Das Beispiel zeigt, daß Biegeprobleme nur eingeschränkt mit Scheibenelementen zu berechnen sind. Elmt05 ist nicht geeignet, Elmt06 liefert gute Ergebnisse. Bei den gegebenen Abmessungen sollte jedoch ein Stabelement verwendet werden!

Eine Verbesserung der Ergebnisse läßt sich für Elmt05 bei Netzverfeinerung erreichen (Netz 100 x 1):

Plane Stress Linear Elastic Element (elmt05)

```
node    1 coord    2 coord    1 displ    2 displ
202 1.00000E+02 1.00000E+00 5.00000E-03-6.66700E-01
```

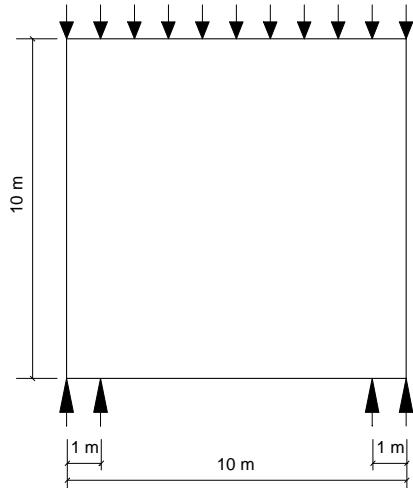
### 13.4.1.4 Vergleichslösung

Die Handrechnung für einen Bernoulli-Stab ergibt:

$$f = \frac{Pl^3}{3EI} = 1 \text{ cm}$$

### 13.4.2 Beispiel 2 - Wandscheibe mit Gleichstreckenlast auf zwei Stützen

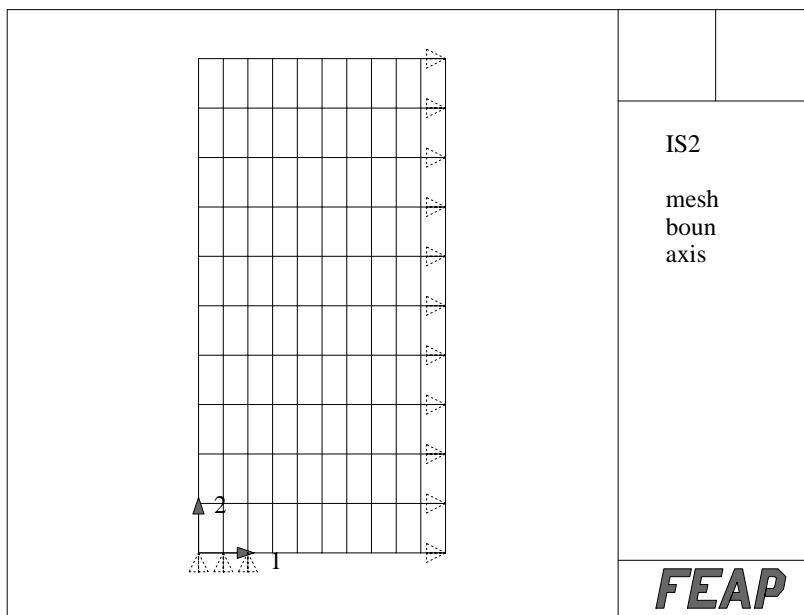
#### 13.4.2.1 System und Belastung



Systemkennwerte:  $E = 4 \cdot 10^8 \text{ kN/m}^2$   
 $l = 10 \text{ m}$   
 $h = 10 \text{ m}$   
 $d = 0.1 \text{ m}$   
 $b = 1 \text{ m}$

Belastung:  $q = 22 \text{ kN/m}$

gesucht: Spannungsverlauf  $\sigma_x$   
 entlang der Symmetriearchse



### 13.4.2.2 Eingabedatensatz (file: is2)

```
feap scheibe mit streckenlast 4 knoten/element
121, 100, 1, 2, 2, 4
```

```
c n = Anzahl El in x
c m = Anzahl El in y
c l = Laenge
c h = Hoehe
c d = Dicke
c q = Streckenlast
c e = Anfangsknoten: Last
c f = Endknoten: Last
```

```
cons
```

```
n=10
```

```
m=10
```

```
l=10
```

```
h=10
```

```
d=0.1
```

```
q=22
```

```
f=(n+1)*(m+1)
```

```
e=f-n
```

```
bloc
```

```
4,n,m,1,1,1,0
```

```
1, 0,0,0
```

```
2,1/2,0,0
```

```
3,1/2,h,0
```

```
4, 0,h,0
```

```
edge *auflager
```

```
0,0,1,0,0,0,0,0,0,0,0,0
```

```
0,1
```

```
ebou *symmetrierand
```

```
1,1/2,1,0
```

```
eloa *streckenlast auf oberen rand
```

```
0,h,1/2,h,0,0,0
```

```
1,1,0,-q
```

```
mate
```

```
1,5
```

```
4.0e+8,0.0,0.0,1
```

```
d,0.0,0.0,0.0,0.0
```

```
mate
```

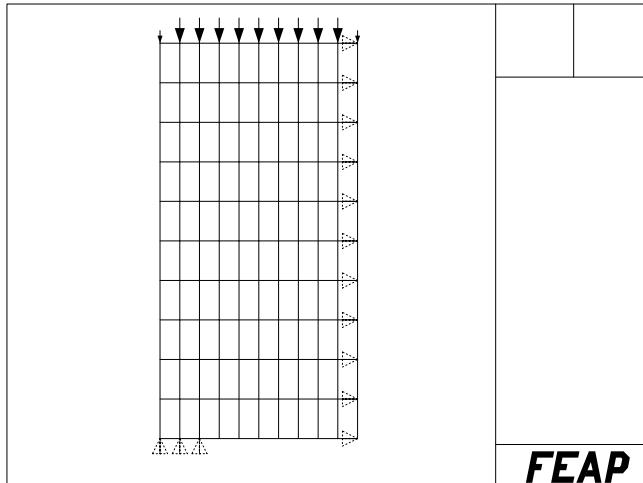
```
1,6
```

```
4.0e+8,0.0,0.0,d,1
```

```
end
```

```
inte
```

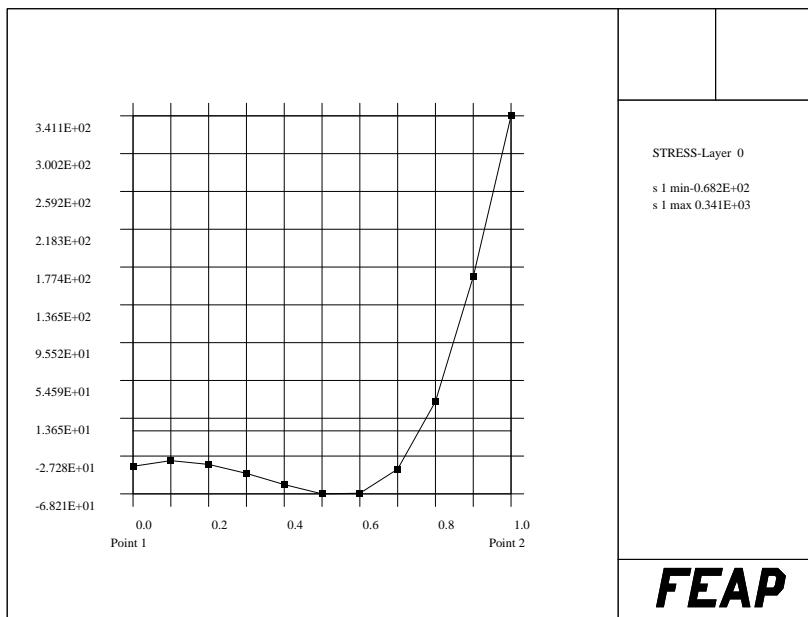
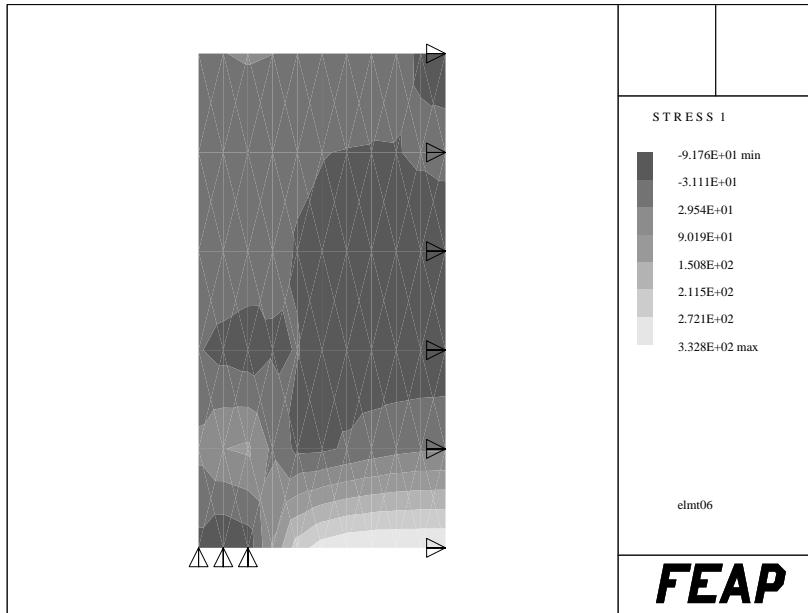
```
stop
```



**FEAP**

### 13.4.2.3 Ergebnisse

Unter Ausnutzung der Symmetrie sehen die Ergebnisse wie folgt aus:



Die Spannung der Knoten auf der Symmetrielinie (Kn.11-unten/Kn.121-oben):

```
n o d a l   s t r e s s e s   (elmt05)

node   N-FOR N_11    N-FOR N_12    N-FOR N_22    N-FOR N_33    N-FOR N_1
       N-FOR N_2    ANG Phi_1
  11  2.48136E+02  3.79780E-01  6.36894E-01  0.00000E+00  2.48137E+02
      6.36312E-01  8.79185E-02
  121 -3.58294E+01 -1.06836E+00 -2.17919E+02  0.00000E+00 -3.58231E+01
      -2.17925E+02 -3.36153E-01

n o d a l   s t r e s s e s   (elmt06)

node   N-FOR N_11    N-FOR N_12    N-FOR N_22    N-FOR N_33    N-FOR N_1
       N-FOR N_2    ANG Phi_1
  11  3.40312E+02  5.30462E-01  1.10814E+00  0.00000E+00  3.40313E+02
      1.10731E+00  8.96014E-02
  121 -3.88116E+01 -1.06232E+00 -2.17835E+02  0.00000E+00 -3.88053E+01
      -2.17842E+02 -3.39974E-01
```

#### 13.4.2.4 Vergleichslösung

Das Näherungsverfahren von Schlee führt zu folgendem Ergebnis:

$$\text{Scheibenlösung} = \text{Balkenlösung} + \text{Korrektur}$$

$$\sigma_x^{\text{Scheibe}} = \sigma_x^{\text{Balken}} + \Delta\sigma_x$$

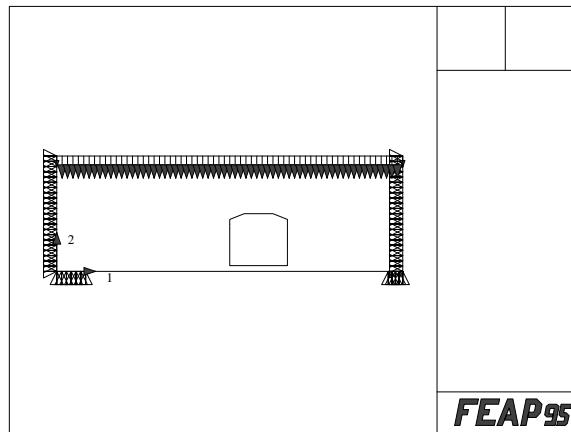
$$\sigma_y^{\text{Scheibe}} = 0 + \Delta\sigma_y$$

Damit ergeben sich entlang der Symmetrielinie die Werte:

$$\begin{array}{cccc} \frac{h}{l} & \sigma_x^{\text{oben}} & \sigma_x^{\text{mitte}} & \sigma_x^{\text{unten}} \\ \frac{10}{10} & -55,0 \text{ kN/m}^2 & -74,8 \text{ kN/m}^2 & +365,4 \text{ kN/m}^2 \end{array}$$

### 13.4.3 Beispiel 3 - Wandscheibe mit Loch

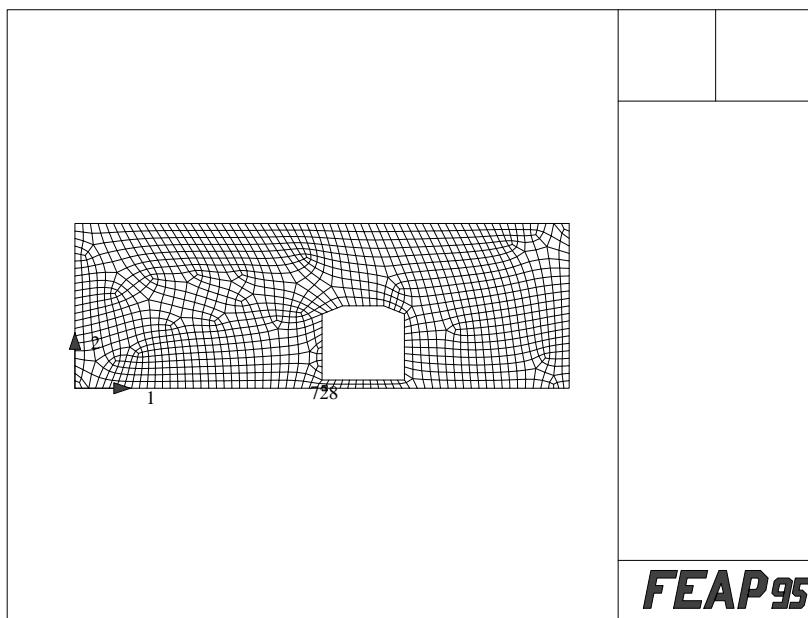
#### 13.4.3.1 System und Belastung



Querschnittswerte:  $E = 21 \cdot 10^6 \text{ kN/m}^2$   
 $l_1 = 12 \text{ m}$   
 $l_2 = 4 \text{ m}$   
 $d = 20 \text{ cm}$

Belastung:  $q = 20 \text{ kN/m}^2$

gesucht: Verschiebung Wandmitte



### 13.4.3.2 NEGE-Eingabedatensatz (file: is3.neg)

```
nege
4,4,2,0,1.5
```

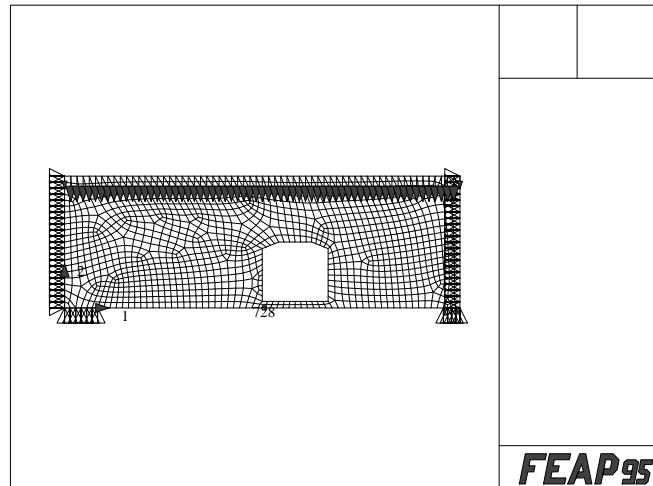
```
geom
1,0.0,0.0
2,1.0,0.0
3,11.5,0.0
4,12.0,0.0
5,12.0,4.0
6,0.0,4.0
7,6.0,0.2
8,8.0,0.2
9,8.0,1.8
10,7.5,2.0
11,6.5,2.0
12,6.0,1.8
```

```
segm
1,1,2
2,2,3
3,3,4
4,4,5
5,5,6
6,6,1
7,7,12
8,12,11
9,11,10
10,10,9
11,9,8
12,8,7
```

```
regi
1,12,1,2,3,4,5,6,7,8,9,10,11,12
```

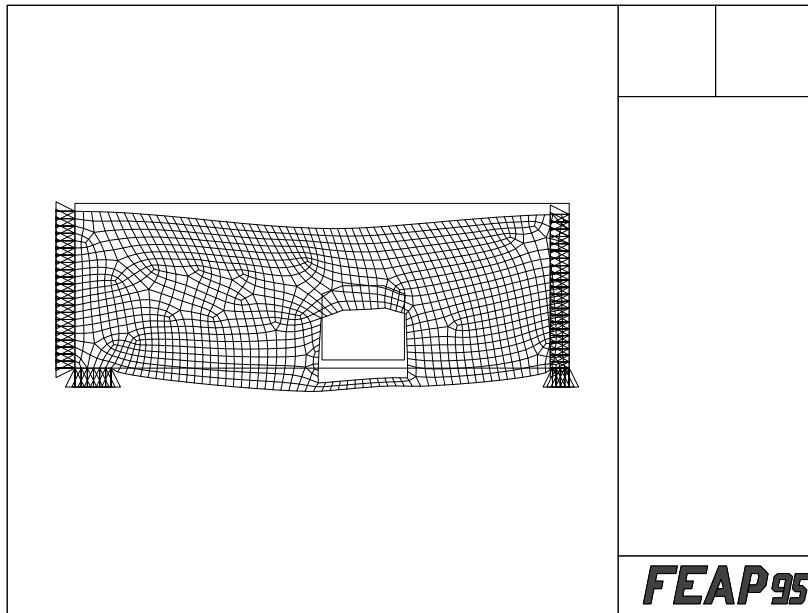
```
pres
5,0,-20,0,-20
```

```
fixe
1,01
3,01
4,10
6,10          end
               opti
mate
1,6          inte
21e+6,0,0,0.2,2 stop
```



**FEAP95**

### 13.4.3.3 Ergebnisse – verformtes Netz



Verformung des mittleren Knotens:

node	1 coord	2 coord	1 displ	2 displ
728	6.06239E+00	0.00000E+00	-3.46907E-06	-2.78284E-05

### 13.4.3.4 Vergleichslösung

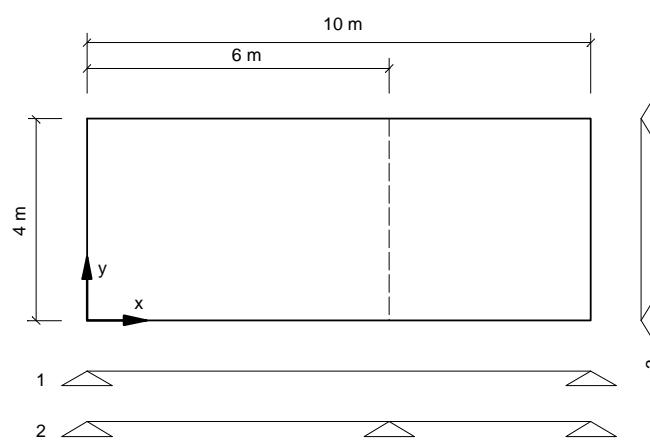
Eine Abschätzung der Durchbiegung durch Handrechnung mit Analogie zur Balkenlösung.

$$\max f = \frac{q l^4}{384 \cdot EI} = 4,8214 \cdot 10^{-5} \text{ m}$$

## 13.5 Beispiele für Plattenelement

### 13.5.1 Beispiel 1 - Platte mit verschiedenen Lagerbedingungen

#### 13.5.1.1 Systeme und Belastung



Kennwerte:  $E = 3.0 \cdot 10^7 \text{ kN/m}^2$

$$l_x^{ges} = 10.0 \text{ m}$$

$$l_x^1 = 6.0 \text{ m}$$

$$l_x^2 = 4.0 \text{ m}$$

$$l_y = 4.0 \text{ m}$$

$$d = 0.1 \text{ m}$$

Belastung:  $q = 5.0 \text{ kN/m}^2$

gesucht: Biegemomente  $m_x$  und  $m_y$

- Einfeldträger – Lagerbedingung 1
- Allseitig gelenkig gelagerte Platte – Lagerbedingungen 1 und 3
- Zweifeldplatte – Lagerbedingungen 2 und 3

### 13.5.1.2 Eingabedatensatz

```
feap  Platte mit Flaechenlast
77,60,1,3,3,4
```

```
c  n = Anzahl El in x
c  m = Anzahl El in y
c  l = Laenge
c  b = Breite
c  d = Dicke
c  q = Flaechenlast
```

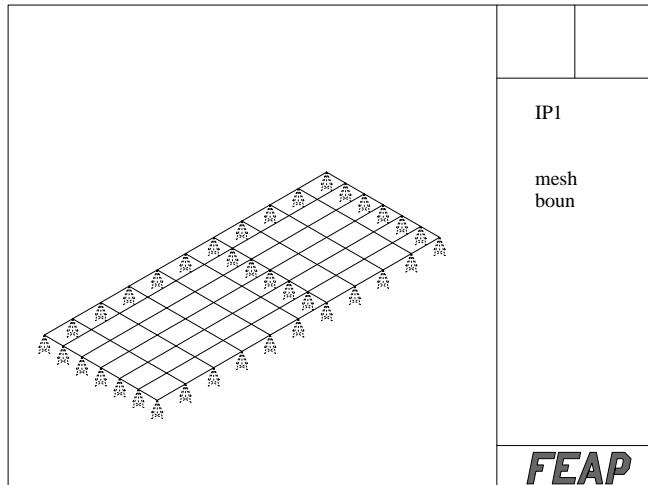
```
cons
n=10
m=6
l=10
b=4
d=0.10
q=5
```

```
bloc
4,n,m,1,1,1,0
1,0,0,0
2,1,0,0
3,l,b,0
4,0,b,0
```

```
ebou  *Einfeldtraeger
1,0,1,0,0
1,1,1,0,0
```

```
mate  *DKQ-Plate Element
1,7
3.0e+7,0,d,-q,0,0,0,0,0,0
0,0
```

```
end
inte
stop
```



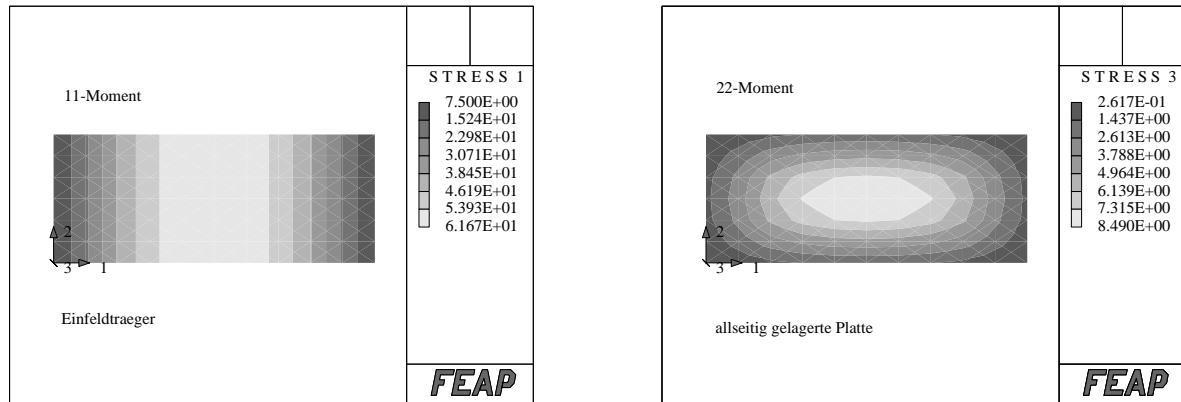
Durch den Austausch von *ebou* werden die übrigen Lagerbedingungen modelliert:

```
ebou  *allseitig gelagerte Platte
1,0,1,0,0
1,1,1,0,0
2,0,1,0,0
2,b,1,0,0
```

```
ebou  *unterstuetzte Platte
1,0,1,0,0
1,1,1,0,0
1,1*0.6,1,0,0
2,0,1,0,0
2,b,1,0,0
```

### 13.5.1.3 Ergebnisse

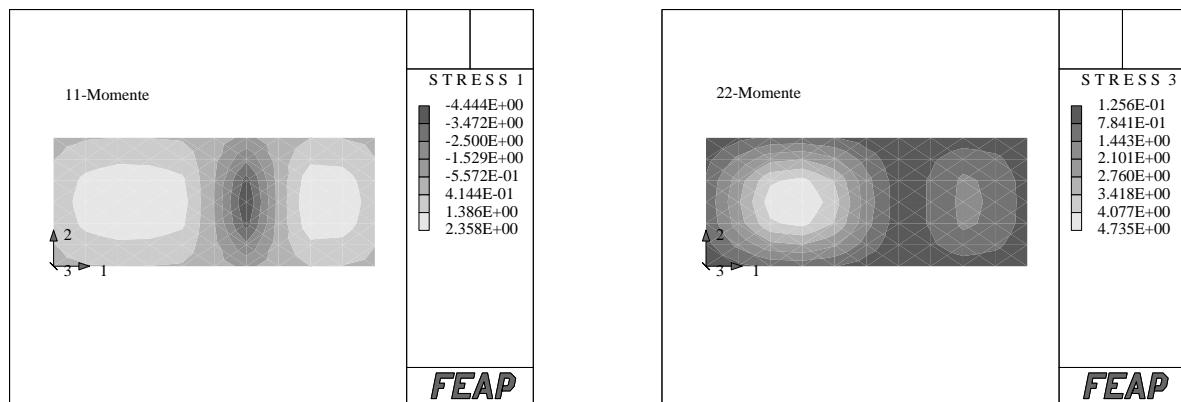
Einfeldträger / allseitig gelagerte Platte:



$$FE-Lsg.: m_{11} = 61,6667 \text{ kNm/m}$$

$$FE-Lsg.: m_{22} = 8,49015 \text{ kNm/m}$$

Unterstützte Platte:



$$\begin{aligned} FE-Lsg.: m_{11}^1 &= 2,28624 \text{ kNm/m} \\ m_{11}^2 &= 2,39101 \text{ kNm/m} \\ m_{11}^s &= -4,44301 \text{ kNm/m} \end{aligned}$$

$$\begin{aligned} FE-Lsg.: m_{22}^1 &= 4,75442 \text{ kNm/m} \\ m_{22}^2 &= 2,31684 \text{ kNm/m} \end{aligned}$$

### 13.5.1.4 Vergleichslösungen

Handrechnung bzw. Anwendung der Tafeln von Pieper/Martens führen zu folgenden Ergebnissen:

- Einfeldträger:

Feldmoment  $m_x$ :

$$\begin{aligned} m_x &= \frac{q l^2}{8} \\ &= \frac{5 \cdot 10^2}{8} = 62,5 \text{ kNm/m} \end{aligned}$$

- Allseitig gelenkig gelagerte Platte:

Feldmomente  $m_x$  und  $m_y$ :

$$\begin{aligned} l_x/l_y > 2 \Rightarrow m_x &\rightarrow 0 \\ m_y &= \frac{5 \cdot 4^2}{8} = 10,0 \text{ kNm/m} \end{aligned}$$

- Zweifeldplatte

Momente  $m_x$  und  $m_y$  in Feld 1:

$$\begin{aligned} l_x/l_y = 1,5 \Rightarrow m_x &= \frac{5 \cdot 4^2}{33,5} = 2,39 \text{ kNm/m} \\ m_y &= \frac{5 \cdot 4^2}{15,0} = 5,33 \text{ kNm/m} \end{aligned}$$

Momente  $m_x$  und  $m_y$  in Feld 2:

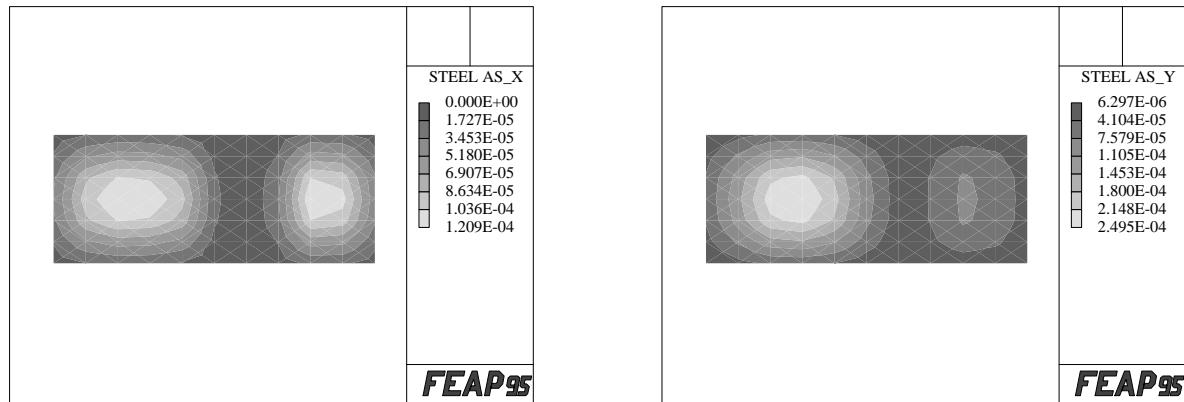
$$\begin{aligned} l_x/l_y = 1,0 \Rightarrow m_x &= \frac{5 \cdot 4^2}{29,1} = 2,75 \text{ kNm/m} \\ m_y &= \frac{5 \cdot 4^2}{32,8} = 2,44 \text{ kNm/m} \end{aligned}$$

Stützmoment  $m_x$ :

$$\begin{aligned} m_{s0} &= \frac{m_{so1} + m_{so2}}{2} \\ &= 5 \cdot 4^2 \cdot \frac{1}{2} \cdot \left( \frac{1}{-8,9} + \frac{1}{-11,9} \right) = -7,85 \text{ kNm/m} \end{aligned}$$

### 13.5.1.5 Bemessung der unterstützten Platte

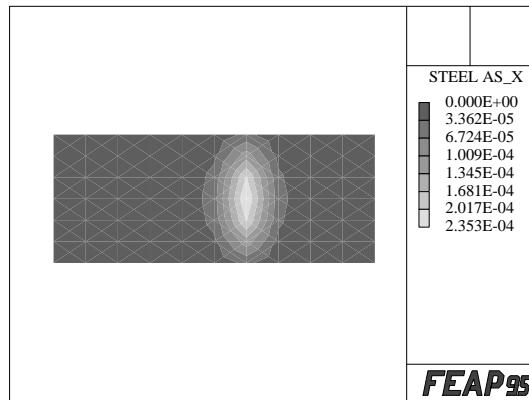
Feldbewehrung:



$$FE - Lsg. : a_{s11}^F = 1,209 \text{ } cm^2/m$$

$$FE - Lsg. : a_{s22}^F = 2,495 \text{ } cm^2/m$$

Stützbewehrung:

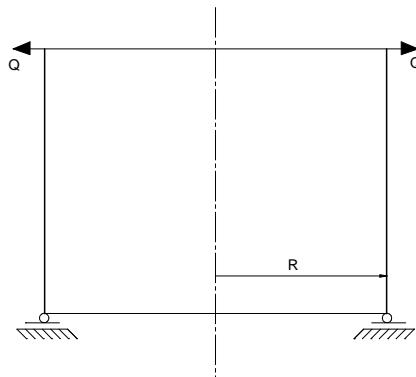


$$FE - Lsg. : a_{s11}^S = 2,353 \text{ } cm^2/m$$

## 13.6 Beispiele für Rotationsschalenelement

### 13.6.1 Beispiel 1 - Zylinder mit Randlast

#### 13.6.1.1 System und Belastung



Sytemdaten:  $E = 21000 \text{ kN/cm}^2$   
 $\mu = 0.2$   
Radius  $R = 100 \text{ cm}$   
Dicke  $t = 0.5 \text{ cm}$   
Höhe  $H = 200 \text{ cm}$

Belastung:  $Q = 1.0 \text{ kN/cm}$   
für die Eingabe als Knotenlast muß über  
den Umfang aufsummiert werden.  
 $\Rightarrow F = 2\pi R Q = 628.3185 \text{ kN}$

gesucht: Schnittgrößen, Knotenverschiebungen

#### 13.6.1.2 Eingabedatensatz (file: isr-1)

```
feap ** Zylinder mit Querlast oben (32 Elemente) **
33,32,1,2,3,2
```

```
coor
1, 1, 100, 0,
9, 1, 100, 150
33,0, 100, 200
```

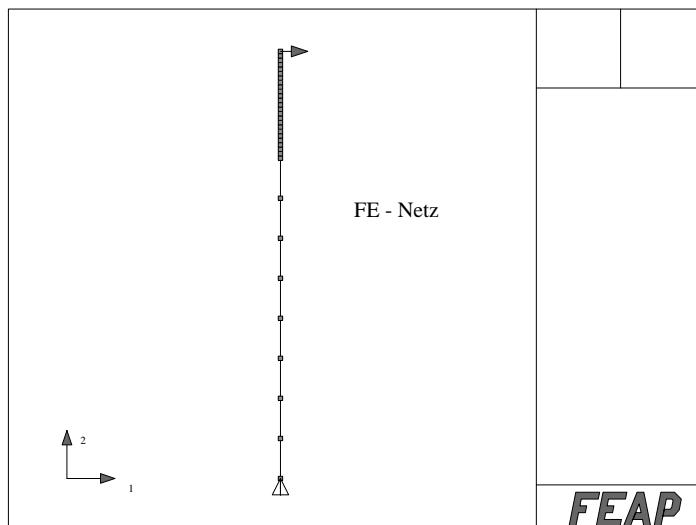
```
elem
1,1,1,2,1,
```

```
boun
1,,0,1,0
```

```
load
33,,628.3185,0,0
```

```
mate
1,8
21000,0.2,0.5
0,0,0,0,0,0
0,0,0
```

```
end
inte
stop
```



### 13.6.1.3 Ergebnisse

Analytische Lösung :

Radiale Verschiebung am oberen Rand:  $w = 0.3590917 \text{ cm}$

Verdrehung am oberen Rand:  $\beta = 0.06464978$

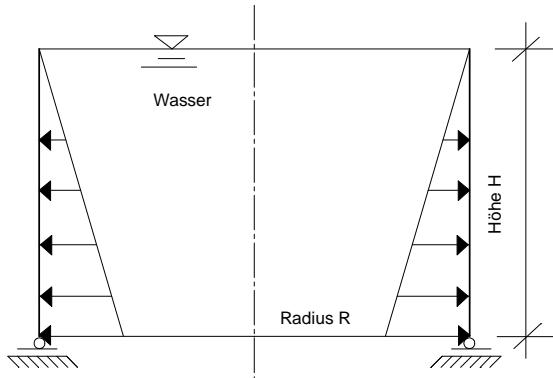
FEM Lösung :

n o d a l   d i s p l a c e m e n t s

node	1 coord	2 coord	1 displ	2 displ	3 displ
33	1.00000E+02	2.00000E+02	3.51289E-01	-1.90476E-03	6.46498E-02

## 13.6.2 Beispiel 2 - Zylinder mit Wasserfüllung

### 13.6.2.1 System und Belastung



Systemdaten:  $E = 21000 \text{ kN/cm}^2$   
 $\mu = 0.2$   
Radius  $R = 100 \text{ cm}$   
Dicke  $t = 0.5 \text{ cm}$   
Höhe  $H = 200 \text{ cm}$

Belastung: Wasser mit  $\gamma = 0.00001 \text{ kN/cm}^3$   
Die Eingabe der Wasserlast erfolgt im Materialdatensatz

gesucht: Schnittgrößen, Knotenverschiebungen

### 13.6.2.2 Eingabedatensatz (file: isr-2)

```
feap ** Zylinder mit Wasserdruck (16 Elemente) ***
17,16,1,2,3,2
```

```
coor
1, 1, 100, 0
17,0, 100, 200
```

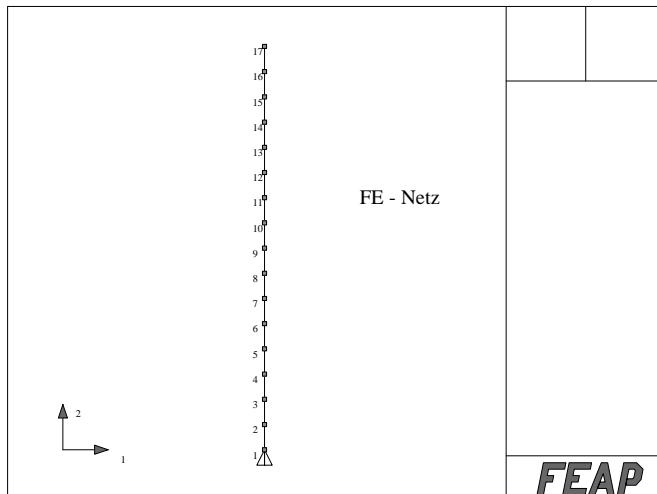
```
elem
1,1,1,2,1
```

```
boun
1,,0,1,0
```

```
mate
1,8
21000,0.2,0.5
0,0,-1E-5,200,2,0
0,0,0
```

```
end
```

```
inte
stop
```



### 13.6.2.3 Ergebnisse

Analytische Lösung am unteren Rand ( $p = \gamma H = 0.002 \text{ kN/cm}^2$ )

$$\text{Radiale Verschiebung : } w = R^2 p / Et = 0.0019 \text{ cm}$$

$$\text{Verdrehung : } \beta = R^2 p / EtH = w/H = 9.5^{-6}$$

$$\text{Umfangskraft : } n_t = Rp = 0.2 \text{ kN/cm}$$

FEM Lösung :

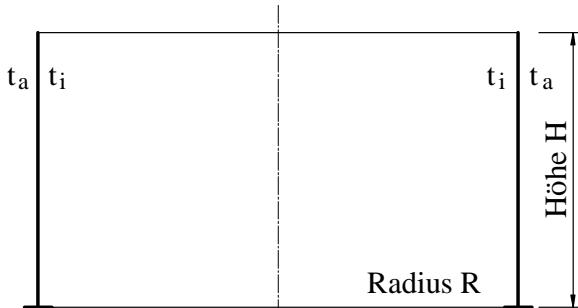
```
nodal displacements
node    1 coord    2 coord    1 displ    2 displ    3 displ
 1 1.00000E+02 0.00000E+00 1.90356E-03 0.00000E+00-9.34138E-06

stress resultants for axisymmetric shell

elmt    ns-force    nt-force    ms-moment    mt-moment    qs-force
 1 1.634E-16 1.937E-01 -2.738E-06 -5.477E-07 4.381E-07
```

### 13.6.3 Beispiel 3 - Zylinder mit gleichmäßiger Temperaturbelastung

#### 13.6.3.1 System und Belastung



Systemdaten: E = 2100 kN/cm<sup>2</sup>  
 $\mu$  = 1/6  
 Radius R = 1200 cm  
 Dicke t = 16 cm  
 Höhe H = 500 cm

Belastung:  $t_i = t_a = +20 K$

gesucht: Schnittgrößen

#### 13.6.3.2 Eingabedatensatz (file: isr-6)

```

feap
51,50,1,2,3,2

coor
1, 1, 1200, 0
21,1, 1200, 100
30,1, 1200, 400
51,0, 1200, 500

elem
1,1,1,2,1

boun
1,,1,1,1

mate
1,8
2100,1/6,16
0,0,0,0,0,0
12e-6,20,20

end
inte
stop

```

#### 13.6.3.3 Ergebnisse

Lösungen am unteren Rand:

$$\begin{aligned} N_t \text{ anal.} &= -8.06 \text{ kN} & N_t \text{ FEM} &= -8.052 \text{ kN} \\ M_s \text{ anal.} &= 38.00 \text{ kN} & M_s \text{ FEM} &= 35.680 \text{ kN} \end{aligned}$$

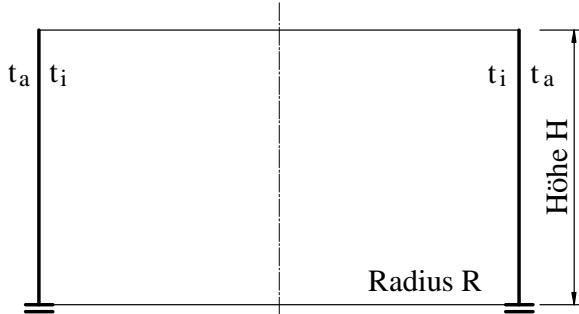
Lösungen am oberen Rand:

$$\begin{aligned} N_t \text{ anal.} &= 0.07 \text{ kN} & N_t \text{ FEM} &= 0.366 \text{ kN} \\ M_s \text{ anal.} &\approx 0.00 \text{ kN} & M_s \text{ FEM} &= 0.075 \text{ kN} \end{aligned}$$

Die analytischen Lösungen sind entnommen aus E. HAMPE; Statik rotationssymmetrischer Flächentragwerke Band 2; VEB Verlag für Bauwesen, Berlin 1964.

### 13.6.4 Beispiel 4 - Zylinder mit ungleichmäßiger Temperaturbelastung

#### 13.6.4.1 System und Belastung



Systemdaten: E = 2100 kN/cm<sup>2</sup>  
 $\mu$  = 1/6  
 Radius R = 1200 cm  
 Dicke t = 16 cm  
 Höhe H = 500 cm

Belastung:  $t_i$  = +30 K  
 $t_a$  = 0 K

gesucht: Schnittgrößen

#### 13.6.4.2 Eingabedatensatz (file: isr-7)

```

feap                                boun
51,50,1,2,3,2                      1,,0,1,1

coor                                mate
1, 1, 1200,   0                      1,8
21,1, 1200, 100                      2100,1/6,16
30,1, 1200, 400                      0,0,0,0,0,0
51,0, 1200, 500                      12e-6,0,30

elem                                end
1,1,1,2,1                            inte
                                      stop

```

#### 13.6.4.3 Ergebnisse

Lösungen am unteren Rand:

$$N_{t \text{ anal.}} \approx -0.00 \text{ kN} \quad N_{t \text{ FEM}} = -0.844 \text{ kN}$$

$$M_{s \text{ anal.}} = 19.34 \text{ kN} \quad M_{s \text{ FEM}} = 20.340 \text{ kN}$$

Lösungen am oberen Rand:

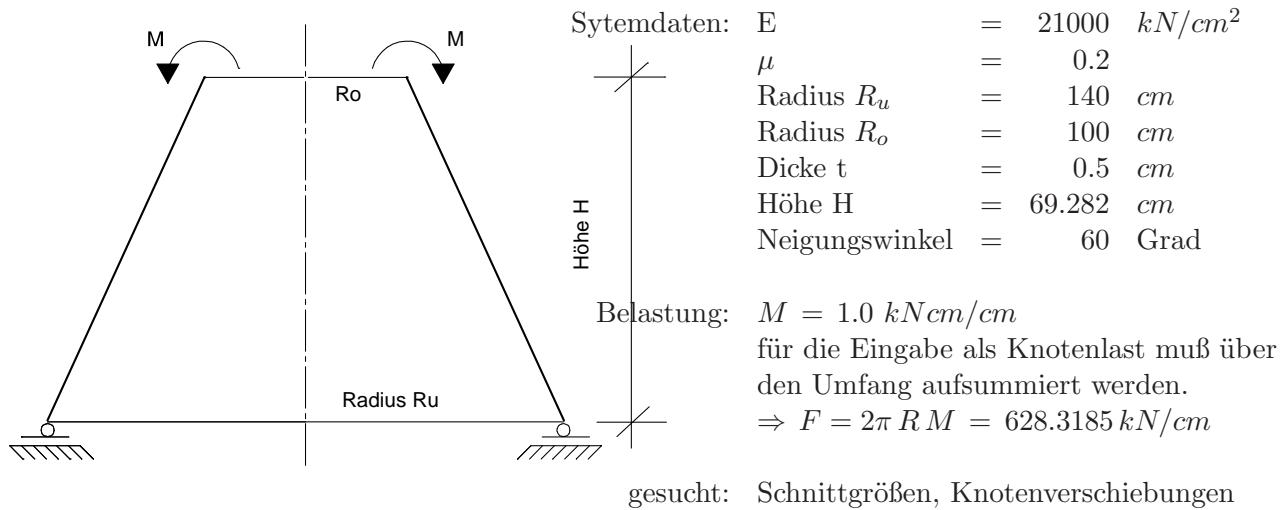
$$N_{t \text{ anal.}} = 4.13 \text{ kN} \quad N_{t \text{ FEM}} = 4.307 \text{ kN}$$

$$M_{s \text{ anal.}} \approx 0.00 \text{ kN} \quad M_{s \text{ FEM}} = 0.018 \text{ kN}$$

Die analytischen Lösungen sind entnommen aus E. HAMPE; Statik rotationssymmetrischer Flächentragwerke Band 2; VEB Verlag für Bauwesen, Berlin 1964.

### 13.6.5 Beispiel 5 - Kegel mit Randmoment

#### 13.6.5.1 System und Belastung



#### 13.6.5.2 Eingabedatensatz (file: isr-3)

```
feap ** Kegel mit Randmoment **
65,64,1,2,3,2
```

```
coor
1, 1, 140., 0.0
65, , 100., 69.282
```

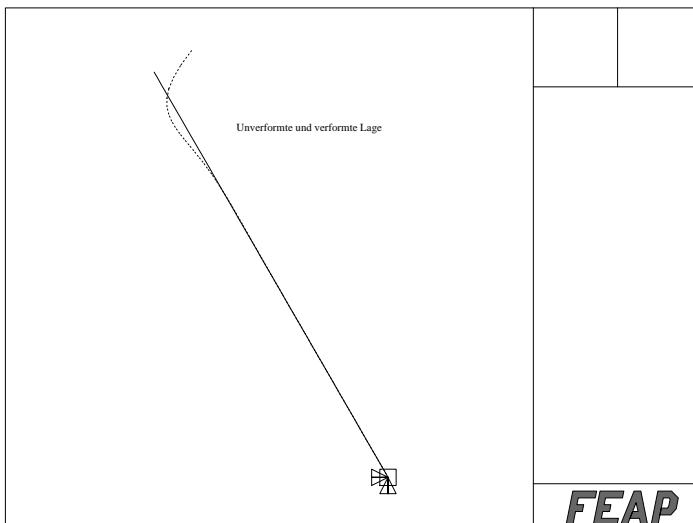
```
elem
1,1,1,2,1
```

```
boun
1,,1,1,1
```

```
load
65,0,0,0,628.3185,,
```

```
mate
1,8
21000,0.2,0.5
0,0,0,0,0,0
0,0,0
```

```
end
inte
stop
```



**FEAP**

### 13.6.5.3 Ergebnisse

Analytische Lösung :

Radiale Verschiebung am oberen Rand:  $w = 0.0646497 \text{ cm}$

Verdrehung am oberen Rand:  $\beta = 0.025597$

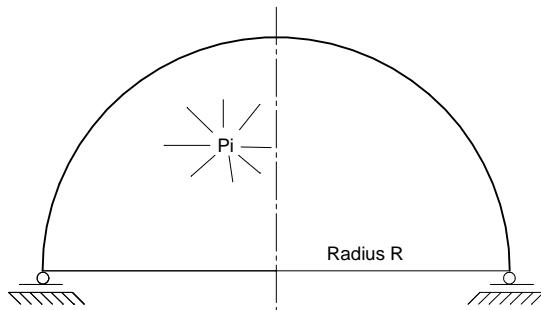
FEM Lösung :

n o d a l d i s p l a c e m e n t s

node	1 coord	2 coord	1 displ	2 displ	3 displ
65	1.00000E+02	6.92820E+01	6.45399E-02	3.73242E-02	2.55777E-02

### 13.6.6 Beispiel 6 - Kugel mit Innendruck

#### 13.6.6.1 System und Belastung



Systemdaten:  $E = 21000 \text{ kN/cm}^2$   
 $\mu = 0.2$   
 Radius  $R = 100 \text{ cm}$   
 Dicke  $t = 0.25 \text{ cm}$

Belastung: Innendruck  $p = 0.1 \text{ kN/cm}^2$   
 Die Eingabe des Druckes erfolgt im Materialdatensatz

gesucht: Schnittgrößen, Knotenverschiebungen

#### 13.6.6.2 Eingabedatensatz (file: isr-4)

```
feap ** Kugelschale mit Innendruck Membranlagerung (50 Elemente) **
51,50,1,2,3,2
```

```
coor
1,1,100, 0
51,,100,90

pola
1,51,1

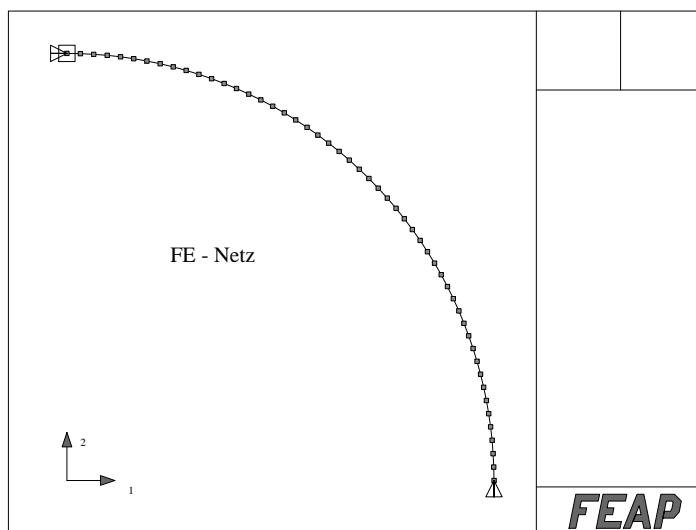
elem
1,1,1,2,1

boun
1,, 0,1,0
51,,1,0,1

mate
1,8
2.1e+04,0.2,0.25
0,-0.1,0,0,0,0
0,0,0

end
```

```
inte
stop
```



### 13.6.6.3 Ergebnisse

Analytische Lösung :

$$\begin{aligned} N_s &= N_t = pR/2 &= 5.0 \text{ kN/cm} \\ w &= ((1 - \mu)pr^2)/2Et = 76.190476 \cdot 10^{-3} \text{ cm} \end{aligned}$$

FEM Lösung :

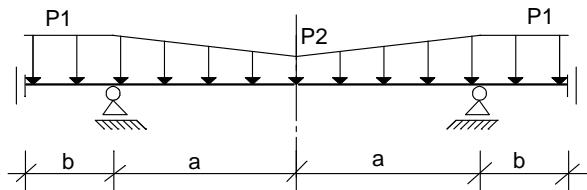
```
n o d a l   d i s p l a c e m e n t s
node      1 coord      2 coord      1 displ      2 displ      3 displ
  1 1.00000E+02 0.00000E+00 7.61733E-02 0.00000E+00-3.60341E-09

stress resultants for axisymmetric shell

elmt    ns-force    nt-force    ms-moment    mt-moment    qs-force
  1  4.999E+00  4.999E+00 -4.164E-09 -8.164E-10  2.651E-09
```

### 13.6.7 Beispiel 7 - Kreisplatte mit Flächenlast

#### 13.6.7.1 System und Belastung



Sytemdaten:  $E = 20000 \text{ MN/m}^2$   
 $\mu = 0.0 \text{ bzw. } 0.2$   
 $a = 4 \text{ m}$   
 $b = 2 \text{ m}$   
 Dicke h = 0.1 m

Belastung:  $p_1 = 12 \text{ kN/m}^2 \quad p_2 = 8 \text{ kN/m}^2$

Die Eingabe der Belastung erfolgt im Materialdatensatz

gesucht: Schnittgrößen, Verschiebungen

#### 13.6.7.2 Eingabedatensatz (file: isr-5)

```
feap ** Kreisplatte (20 Elemente) **
151,150,2,2,3,2
```

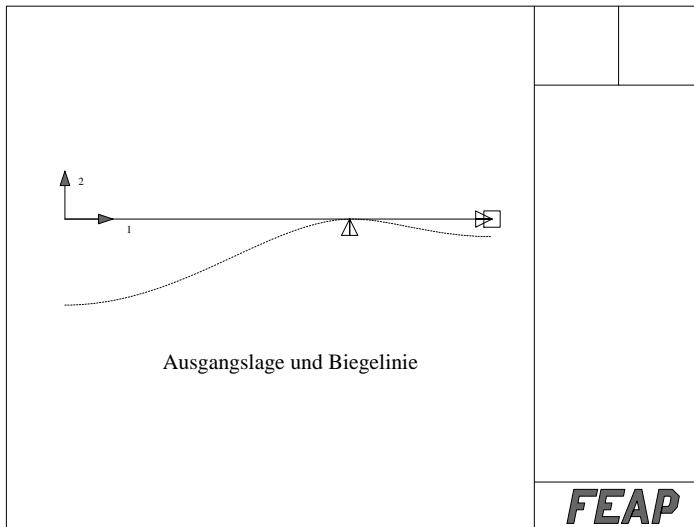
```
coor
1, 1, 0.0, 0.0
100,1, 4.0, 0.0
151,0, 6.0, 0.0
```

```
elem
1, 1, 1, 2,1
100,2,100,101,1
```

```
boun
100,0,0,1,0
151,0,1,0,1
```

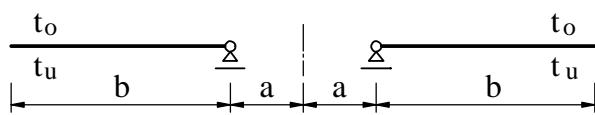
```
mate
1,8
2.0e+07,0.0,0.1
0,-12,1,4,1,0
0,0,0
2,8
2.0e+07,0.0,0.1
0,-12,0,0,0,0
0,0,0
```

```
end
inte
stop
```



### 13.6.8 Beispiel 8 - Kreisplatte mit Loch unter Temperaturbelastung

#### 13.6.8.1 System und Belastung



Sytemdaten:  $E = 2100 \text{ kN/cm}^2$   
 $\mu = 1/6$   
 $a = 100 \text{ cm}$   
 $b = 900 \text{ cm}$   
Dicke h = 16 cm

Belastung:  $t_o = +20 \text{ K}$   
 $t_u = -20 \text{ K}$

gesucht: Schnittgrößen, Verschiebungen

#### 13.6.8.2 Eingabedatensatz (file: isr-8)

```

feap
16,15,1,2,3,2

coor
1, 1, 100, 0
16,0,1000, 0

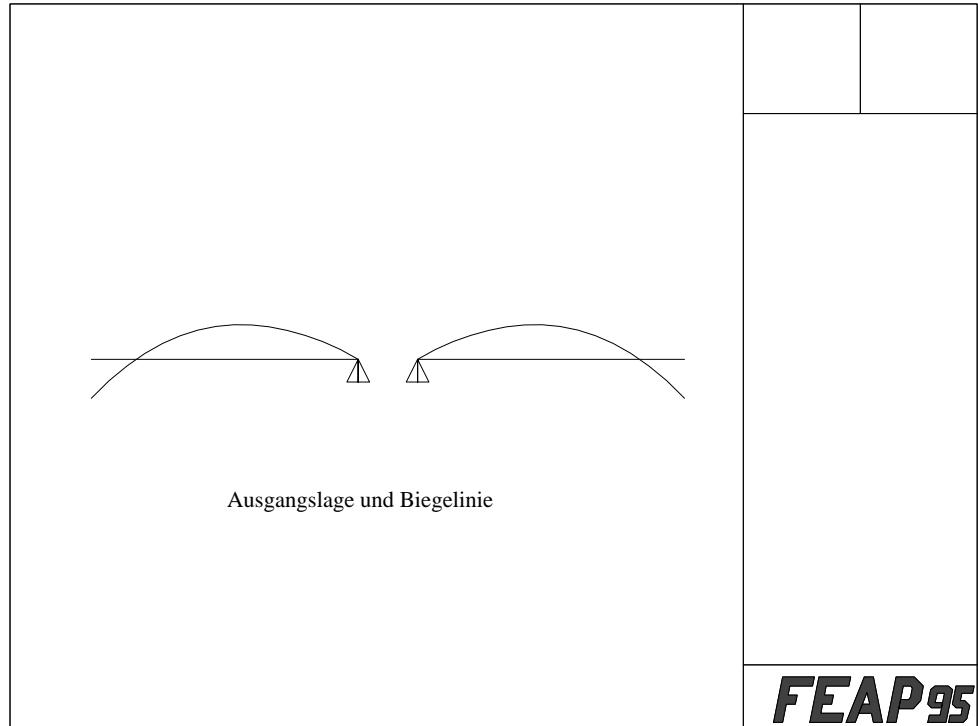
elem
1,1,1,2,1

boun
1,,0,1,0

mate
1,8
2100,1/6,16
0,0,0,0,0,0
12e-6,-20,20

end
inte
stop

```



## 13.7 Beispiele für allgemeines Schalenelement

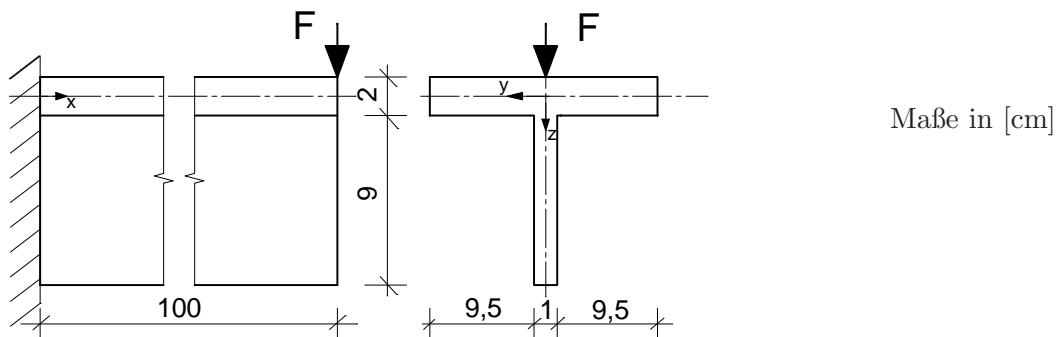
### 13.7.1 Beispiel 1 - Eingespannter T-Träger

#### 13.7.1.1 System und Belastung

Systemdaten:  $E = 21000 \text{ kN/cm}^2$   
 $\mu = 0.0$

Belastung:  $F = 10.0 \text{ kN}$  am Trägerende

gesucht: Durchbiegung, Spannungen



#### 13.7.1.2 Systemdaten für eine analytische Vergleichslösung

Fläche	$A = 49 \text{ cm}^2$
Schwerpunkt	$z_s \approx 1.0 \text{ cm}$
Trägheitsmoment	$I_y \approx 296 \text{ cm}^4$
Widerstandsmomente	$W_o \approx 148 \text{ cm}^3$
	$W_u \approx 33 \text{ cm}^3$

### 13.7.1.3 Eingabedatensatz (file: isg-1)

```
feap ** T-Traeger mit allgemeinem Schalenelement **
143,110,2,3,6,4
```

```
const                                Bemerkung zu den Konstanten :
l=100
b=20
h=10
x=10
z=5
y=6
i=(x+1)*(z+1)+1
j=x*z+1
f=(x+1)*(z+1)

l      laenge      in cm
b      breite platte  in cm
h      hoehe steg    in cm
x      anzahl elemente laengs
z      anzahl elemente hoehe
y      anzahl elemente quer (platte)
i,j   startknoten und element fuer block 2
f      knoten mit last
```

```
block 1 (Steg)
4,x,z,1,1,1
1, 0.0, 0.0, h
2, 1, 0.0, h
3, 1, 0.0, 0.0
4, 0.0, 0.0, 0.0
```

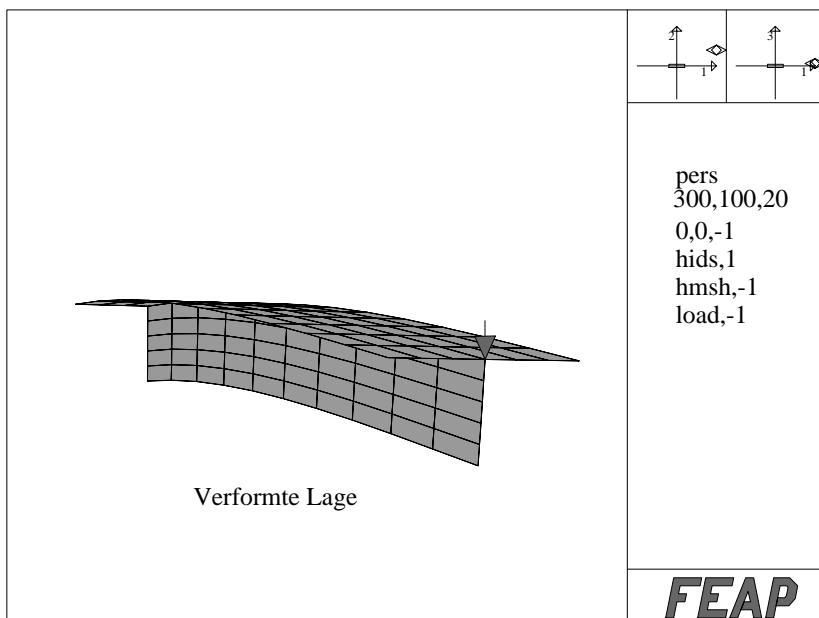
```
block 2 (Platte)
4,x,y,i,j,2
1, 0.0, b/2, 0.0,
2, 1, b/2, 0.0,
3, 1,-b/2, 0.0,
4, 0.0,-b/2, 0.0,
```

```
ebou
1,0.0, 1,1,1, 1,1,1
```

```
load
f,,0,0,10
```

```
mate
1,9
21000,0.0,0.0,1.0
0,0,0,0,0,0,0,0
0,0,0
2,9
21000,0.0,0.0,2.0
0,0,0,0,0,0,0,0
0,0,0
```

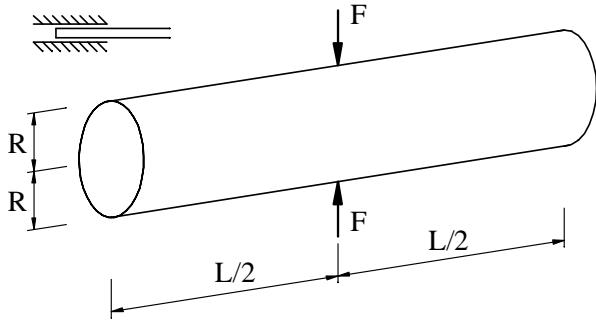
```
end
tie
inte
stop
```



### 13.7.2 Beispiel 2 - Beidseitig eingespannter Zylinder mit Einzellasten

#### 13.7.2.1 System und Belastung

Detail Auflager



Systemdaten:  $E = 3 \cdot 10^6 \text{ kN/cm}^2$

$\mu = 0.3$

Länge  $L = 600 \text{ cm}$

Radius  $R = 300 \text{ cm}$

Dicke  $t = 3 \text{ cm}$

Belastung:  $F = 1.0 \text{ kN}$

gesucht: Verschiebung der Lastknoten

#### 13.7.2.2 Eingabedatensatz (file: isg-2)

Die Berechnung erfolgt unter Ausnutzung der mehrfachen Symmetrie an einem Achtel des Zylinders.

```

feap                                mate
256,225,1,3,6,4                    1,9
                                    3.e6,0.3,0,t
cons                               0,0,0,0,0,0,0,0
i=15                               0,0,0
j=15                               end
r=300                             inte
t=3                                stop
l=300

```

```

bloc
4,i,j,1,1,1
1,r,90, 0
2,r, 0, 0
3,r, 0,-1
4,r,90,-1

```

```

pola
1,(i+1)*(j+1),1

```

```

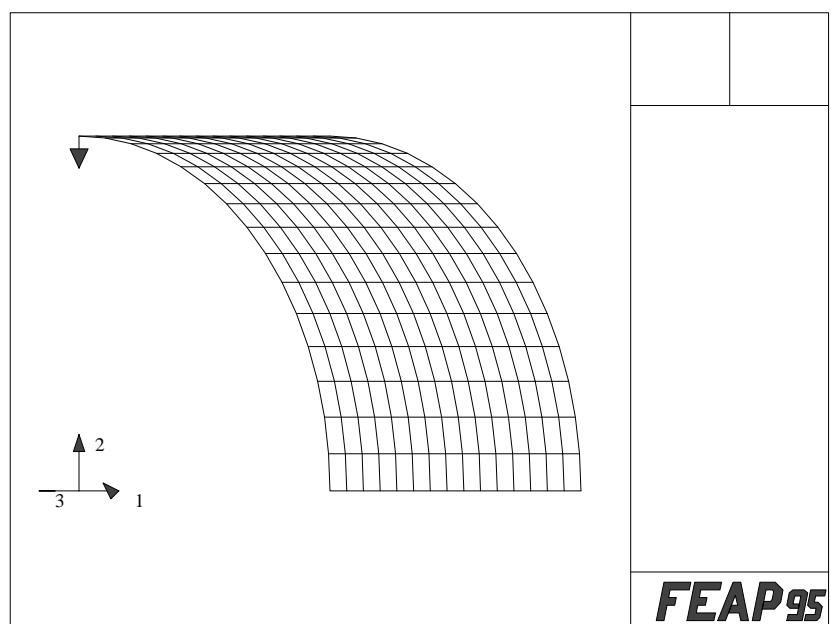
ebou
1, 0, 1,0,0, 0,0,1
2, 0, 0,1,0, 0,0,1
3, 0, 0,0,1, 1,1,0
3,-1, 1,1,0, 1,1,0

```

```

load
1,0,0,-0.25

```



**FEAP95**

### 13.7.2.3 Ergebnisse und verformtes Netz

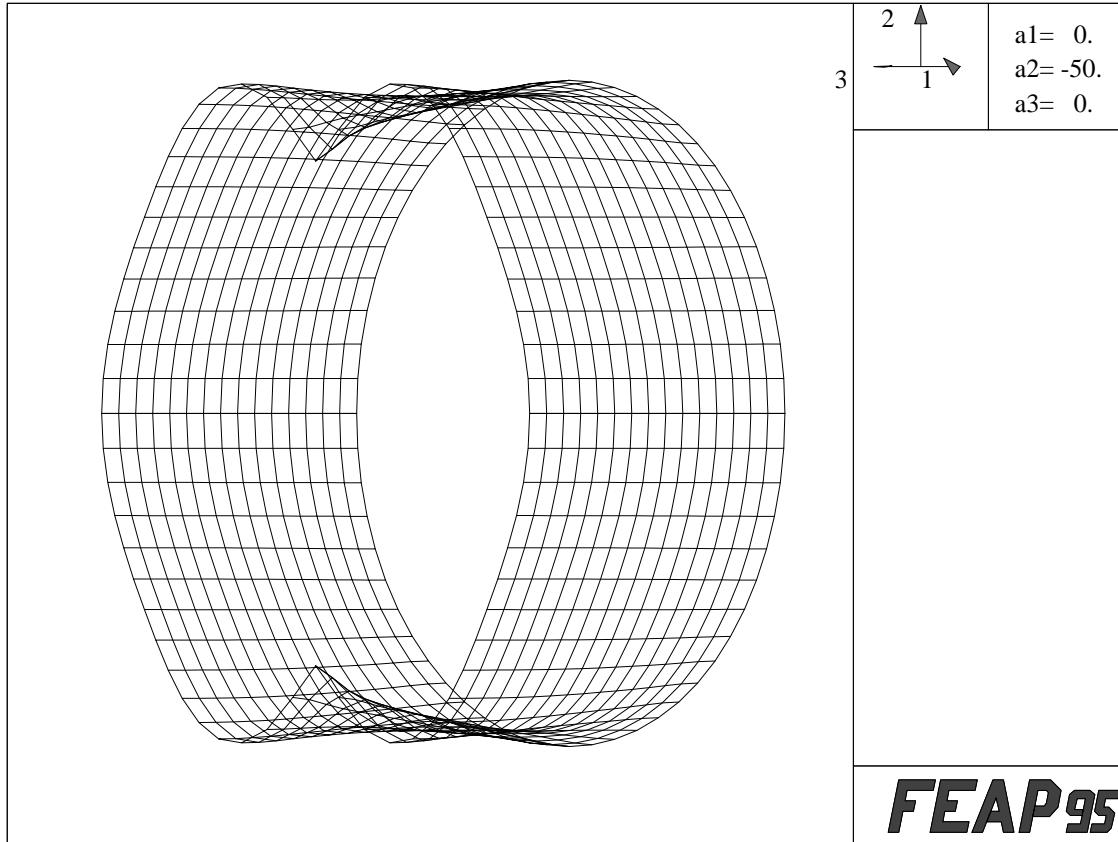
Ergebnisse der FEM-Rechnung:

```
n o d a l   d i s p l a c e m e n t s
```

node	1 displ	2 displ	3 displ	4 displ	5 displ	6 displ
1	0.00000E+00	-1.82075E-05	0.00000E+00	0.00000E+00	0.00000E+00	0.00000E+00

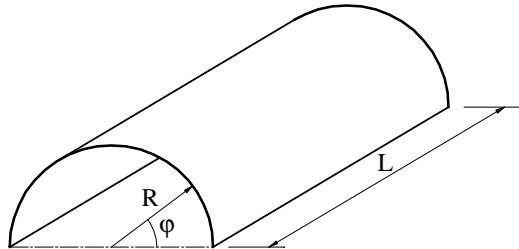
Eine analytische Berechnung ergibt eine Verschiebung des Lastknotens von:

$$u_{ref} = 1.8249 \cdot 10^{-5} \text{ cm}$$



### 13.7.3 Beispiel 3 - Membrangelagertes Tonnendach

#### 13.7.3.1 System und Belastung



Systemdaten:  $E = 3 \cdot 10^7 \text{ kN/m}^2$   
 $\mu = 0.0$   
 Länge  $L = 15.0 \text{ m}$   
 Radius  $R = 4.0 \text{ m}$   
 Dicke  $t = 0.05 \text{ m}$

Belastung: Eigengewicht  $g = 3.75 \text{ kN}$

gesucht: Schnittkraftverläufe

#### 13.7.3.2 Eingabedatensatz (file: isg-3)

```

feap                                mate
,,1,3,6,4                           1,9
                                      3.e7,0,0,t
cons                               0,0,0,0,-3.75,0,0,0
i=20                                0,0,0
j=20
r=4                                 solv
t=0.05                            2
l=15

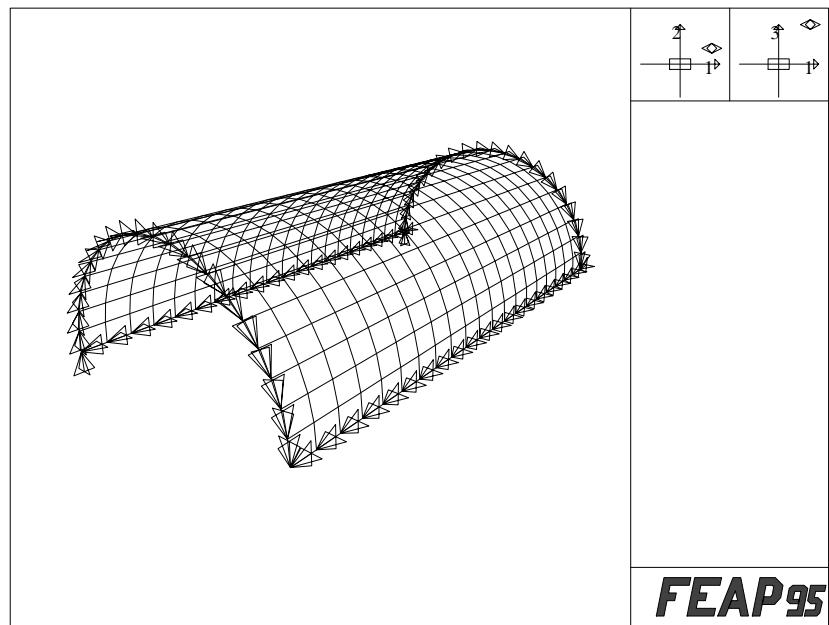
bloc
4,i,j,1,1,1
1,r,180, 0
2,r,180, 1
3,r, 0, 1
4,r, 0, 0

pola
1,(i+1)*(j+1),1

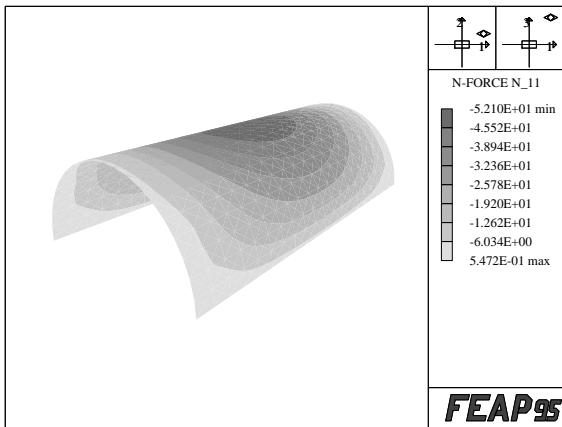
ebou
3, 0, 0,1,0, 0,0,0
3, 1, 0,1,0, 0,0,0
2, 0, 0,0,1, 0,0,0

angl
2,1
1,          (i+1), 0
(i+1)*j+1,      0,180
(i+1),        (i+1), 0
(i+1)*(j+1),    0,180

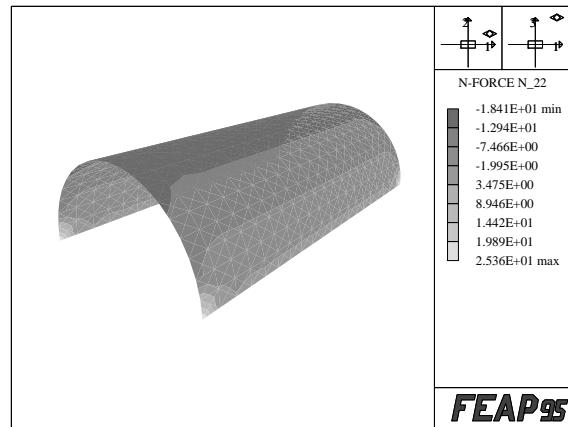
```



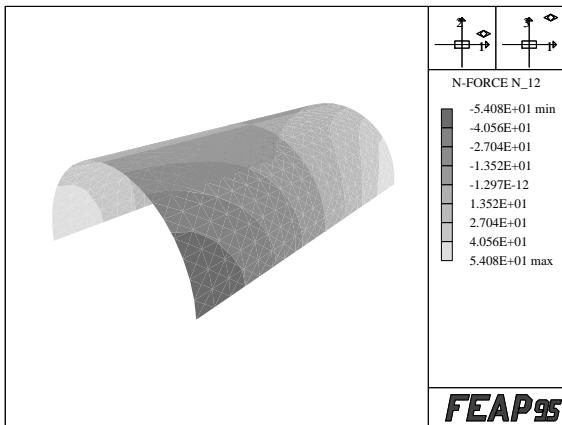
### 13.7.3.3 Schnittkraftverläufe und Ergebnisvergleich



in Längsrichtung ( $N_{xx}$ )



in Umfangsrichtung ( $N_{yy}$ )



Schubkräfte ( $N_{xy}$ )

Der Vergleich der FEM-Lösung mit einer analytischen Berechnung erfolgt exemplarisch für das Element 314 (A. PFLÜGER; Elementare Schalenstatik, Springer-Verlag, Berlin 1957).

Ergebnisse der FEM-Rechnung:

elmt	x-coord	$N_{xx}$	$N_{xy}$	$N_{yy}$	$N_1$	$N_2$	angle_N
matl	y-coord	$M_{xx}$	$M_{xy}$	$M_{yy}$	$M_1$	$M_2$	angle_M
	z-coord	$\epsilon_{ps\_xx}$	$\epsilon_{ps\_xy}$	$\epsilon_{ps\_yy}$	$\kappa_{xx}$	$\kappa_{xy}$	$\kappa_{yy}$
314	3.032	-3.014E+01	-1.461E+01	-9.829E+00	-2.193E+00	-3.777E+01	-62.40
1	2.590	1.923E-03	4.568E-03	-7.389E-02	2.197E-03	-7.416E-02	3.44
	10.125	-2.009E-05	-1.948E-05	-6.553E-06	6.152E-06	2.923E-05	-2.364E-04

Die analytische Berechnung ergibt für  $\varphi = 40.505^\circ$ :

$$N_s = -g \frac{s}{R} (L - s) \sin\varphi = -30.06 \text{ kN/m} \sim N_{xx}$$

$$N_\varphi = -g R \sin\varphi = -9.74 \text{ kN/m} \sim N_{yy}$$

$$T = -g (L - 2s) \cos\varphi = -14.61 \text{ kN/m} \sim N_{xy}$$

# Chapter 14

## Adding elements

### 14.1 General information

Elements can be added to FEAP without knowing the exact structure of the main program. Since the set of parameters is always the same in the element subroutines the following rules are of general use. The reader may also consult chapter 19 of the textbook, *The Finite Element Method* by O. C. Zienkiewicz & R. L. Taylor, McGraw Hill, 2000, to obtain additional information concerning coding elements for FEAP.

When an element subroutine is called from FEAP the main program computes all local arrays which are needed to perform different tasks on the element level. The switch to different tasks in the element subroutines depends on the value of the variable 'isw'. All variables, parameters and arrays are transferred via the list of parameters of the element subroutines and via a restricted number of common blocks which are defined below.

#### 14.1.1 Name of Subprogram, parameters

The name of an element subprogram for feap must be specified according to the following *name* and *parameter* requirements

```
subroutine elmt $nn$ (d,ul,xl,ix,tl,s,p,h1,h2,h3,ndf,ndm,nst,isw)
```

where  $nn$  defines the element number in the range between 01 and 100.

The argument list is defined as follows together with dimensioning information:

Parameter	Description
d(ndd)	material parameters
ul(ndf,1)	element displacement parameters
xl(ndm,1)	element initial coordinates
ix(1)	global/local node numbers
tl(1)	element temperatures
s(nst,nst)	element tangent arrays
p(nst)	element load vector
h1(nhmax)	element history data vector h1
h2(nhmax)	element history data vector h2
h3(nh3max)	element history data vector h3
ndf	number d.o.f. at each node (1 to 6)
ndm	coordinate dimension (1,2,3)
nst	size element array/vector
ndd	number of material data for element; default ndd = 50*
isw	control switch parameter

\* The value of ndd can be changed by the mesh macro **feap**.

The following of the above arrays and values are input quantities and will be transferred from the main program FEAP:

Parameter	Description
d(ndd)	material parameters
ul(ndf,nel)	element displacement parameters
xl(ndm,nel)	element initial coordinates
ix(1)	global/local node numbers
tl(nel)	element temperatures
ndf	number d.o.f. at each node (1 to 6)
ndm	coordinate dimension (1,2,3)
nst	size element array/vector =ndf*nel
ndd	number of material data for element; default ndd = 50
isw	control switch parameter

**Warning:** Do not change these values, since they are also used for the computation of other elements of this material group in the global f.e. problem.

The following arrays are results of the computation on element level their values depend on the task to be performed, see definition of 'isw'.

Parameter	Description
d(ndd)	material parameters (only isw = 1)
s(nst,nst)	element tangent array
p(nst)	element vector
h1(nhmax)	element history data vector
h2(nhmax)	element history data vector
h3(nh3max)	element history data vector

**Warning:** Only compute and store values within the range of these arrays. An erroneous specification of e.g. ( s(nst,nst+2) = fact ) may lead to unpredictable errors in FEAP.

### 14.1.2 Control Switch Parameter Values

The values of the control switch parameter, *isw*, which define a task to be performed on element level are defined as:

isw – Value	Operation
1	Define material parameters in d(i)
2	Check mesh for errors
3	Compute tangent array, s, and residual, p
4	Output element variables (e.g., stress)
5	Compute element mass; s = consist., p = lump.; geometrical matrix s
6	Compute element residual vector in p
7	Compute surface loading; s = surface tangent, p = surface load vector
8	Project element variables to nodes (e.g., stress)
9	Perform error analysis
10	Update variables within augmented lagrangian algorithm
11	Perform numerical differentiation within extended system
12	Compute damping matrix; s = consist., p = lump.
13	Plot element stress resultants for beam, axi-shell and – membrane elements
14	Plot element stresses at center of element without averaging
15	update $H_2 \rightarrow H_1$ on micro-level, similar to stre, until call of mateli3d
16	Compute J-integral
17	Compute stresses at nodes and layer boundaries
18	Compute director for shells and surfaces
19	Implex time step control
20	Plot stresses for sections of beam elements
21	Compute non-symmetric tangent for eigensolver ueig
22	Element load vectors

You do not have to set the vectors 'p' and 's' to zero within the element routine since FEAP zeros these arrays before transferring them to the element subroutine.

## 14.2 Common Blocks

FEAP uses several common areas to pass information to and from the main program. Each common block must have exactly the specified length and precisions in order for the program to work. The useful common blocks are as follows:

Name	Parameters	Type
/bdata/	o,head(20)	character*4
/cdata/	numnp,numel,nummat,nen,neq,ipr	integer
/eldata/	dm,n,ma,mct,iel,nel	real,integer
/evdata/	imtyp, ibuck	integer
/hdata/	nh1,nh2,nh3,nhmi,nhmf gh1(*),gh2(*),gh3(*)	integer real
/iofile/	ior,iow	integer
/pdata6/	inord(100),ipord(40,100)	integer
/pdata7/	ipb,ipma(40),ipla	integer
/pdata10/	cfp,xmaxf,xminf,scal,nfp,klay,ifor,flfp	real,integer,logical
/strnam/	np,istv,strsus(26)	integer, character*15
/prisdat/	nptyp,nprip(8)	integer (default: 1,[1,2,3,4,5,6,7,8])
/fornam/	forsus(11)	character*15
/pdam/	iprd,ipld	integer
/prlod/	prop,a(6,10),iexp(10),ik(10),npld	real,real,integer,integer, integer

The programmer of an element might want to use own common blocks within the element to transfer data to the diverse subroutines defining the element. In this case it is preferable to create names like e.g. /elnn1/ etc. where 'nn' is the element number. This avoids possible confusion with common blocks defined in FEAP.

The parameters of the common blocks listed on the previous page are defined as:

Parameter	Description
o	page output control
head	page header/title
numnp	number of nodes in problem
numel	number of elements in problem
nummat	number of material sets
nen	maximum number of nodes on element
neq	number of active equations
ipr	precision of real variables (FEAP only)
dm	no longer used
n	number of current element processed
ma	material set number of current element boundary type of surface loads (isw = 7)
mct	line counter control for output (isw = 4) surface load type (isw = 7)
iel	current element type (nn)
nel	number of nodes on current element number of nodes on current surface (isw = 7)
nh1	length of history arrays h1, h2
nh3	length of history arrays h3
np	pointer for nodal stress parameters
ior	input file unit number (if negative use *)
iow	output file unit number
istv	number of nodal stresses (default 8) a negative value prevents calc. of principal stresses $(\sigma_I = 5, \sigma_{II} = 6, \varphi_I = 7)$
strsus(26)	array for description of plotted stresses eg. $S_{11}, S_{22}$
forsus(11)	array for description of plotted forces eg. $N_{11}, N_{22}$

Parameter	Description
imtyp	1: calculate mass matrix $\mathbf{M}$ 2: calculate geom. matrix $\mathbf{K}_G$ 3: calculate geom. matrix $\mathbf{K}_U + \mathbf{K}_G$ 4: calculate lin. matrix $\mathbf{K}_L$
inord	number of nodes of element
ipord	series of node numbers to be plotted
ipb	not used in element
ipma	not used in element
ipla	0: standard displacement element ( $u_1, u_2, u_3, \varphi_1, \varphi_2, \varphi_3$ ) 1: plate element ( $w, \varphi_1, \varphi_2$ ) 2: beam, axi-shell/-membrane element ( $u_1, u_2, \varphi$ )
cfp	scaling factor for stress resultant plot
nfp	number of stress resultant to be plotted positive: calculate main stresses in 4-6 from 1-3 automatically negative: stress values 1-11 can be defined free
xmaxf	max. stress resultant value
xminf	min. stress resultant value
flfp	logical for switch between plotting stress resultants and min/max-search
klay	layer number
ifor	local plot direction 13 (default) or 12
iprd	pointer for damage print ouput (0=no,1=yes)
ipld	pointer for damage plot ouput (0=no,1=yes)
prop	actual load factor
a(6,npld)	$t_{min}, t_{max}, A_1 - A_4$ , see macro <b>prop</b>
iexp(npld)	exponent
ik(npld)	loading type (1-3)
npld	max. number of prop. cards=3

## 14.3 Treatment of history terms

FEAP provides options for each element to manage variables which must be saved during the solution. These are history variables and are separated into three groups:

- a) Variables associated with the last converged solution time  $t_n$ , stored in  $h_1$
- b) variables associated with the current solution time  $t_{n+1}$ , stored in  $h_2$ , and
- c) variables which are not associated to any particular time, stored in  $h_3$ .

Examples for a) and b) are the integration of non-linear constitutive equations over a time step ( $\Delta t = t_{n+1} - t_n$ ), examples for c) are e.g. terms used within mixed elements (e.g. based on Hu-Washizu-principles).

Before calling the element routine for each element, FEAP transfers the required history variables from global to local arrays  $h_1, h_2, h_3$ . Users may then access the history data for each element and if necessary update values and return them to FEAP. Only for specific actions the local history data will be transferred back to the appropriate global locations.

### 14.3.1 Assigning amount of storage for each element

The specification for the amount of history information to be associated with each material set is controlled in the  $isw = 1$  task of an element routine. For each material type specified within the element routine a value for the length of the 'nh1' and the 'nh3' data must be provided (the amount of 'nh2' data will be the same as for 'nh1'). This is accomplished by setting the variables 'nh1' and 'nh3' in common hdata to the required values. That is, the statements required are:

```
USE hdata
...
if(isw .eq. 1) then
...
nh1 = 24
nh3 = 10
...
```

reserves 24 words of 'nh1' and 'nh2' data and 10 words of 'nh3' data for each element with the current material number. Typically it holds  $nh1 = ngaus \cdot nh$ , with  $ngaus$  = number of integration points and  $nh$  = number of history terms at integration point. Care should be taken to minimize the number of history variables since, for very large problems, the memory requirements can become large, thus reducing the size of problem that FEAP can solve. Different materials/elements could have different amount of history data. For that FEAP calculates maximum values  $nhmax = max(nh1_i)$  and  $nh3max = max(nh3_i)$  and provide local arrays  $h1(nhmax), h2(nhmax), h3(nh3max)$ . On element level these arrays are defined in general by

```
dimension h1(*),h2(*),h3(*)
```

### 14.3.2 Accessing history data for each element

As noted above the data for each element is contained in arrays whose first word is located at local arrays  $h1(1)$ ,  $h2(1)$ . ' $nh1$ '-' $nh3$ ' are here no longer used. Note that arrays only exist if non-zero values are assigned to ' $nh1$ ' and/or ' $nh3$ ' during the  $isw = 1$  task. Any other allocated data follows immediately after each first word. It is a users responsibility to manage what is retained in each variable type; however, the order of placing the  $t_n$  and  $t_{n+1}$  data into the  $h1$ - and  $h2$ -arrays should be identical. When elements are developed for FEAP it is the programmers responsibility to define and update the necessary information in the history arrays. Normally, information is **read** from the  $h1$ -array and **written** to the  $h2$ -array. The **time**-macro command is used to advance time and also is the command which redefines the information in the  $h1$ -array from that currently stored as the  $h2$ -array. Generally, **it is not advisable to write information into the  $h1$ -array space** as this may destroy information needed for subsequent iterations or solution step.

There are no provisions to store integer history variables separately from double precision quantities. It is necessary to cast the integer data as double precision and move to the history location. For example, using the statement  $h3(6) = \text{dble}(\text{ivarbl})$  saves the value for the integer variable  $\text{ivarbl}$  in the sixth word of the  $h3$  element history array. At a subsequent iteration for this element the value of the integer would be recovered as  $\text{ivarbl} = \text{int}(h3(6))$ . While this wastes storage for integer variables, experience indicates there is little need to save many integer quantities and, thus, it was not deemed necessary to provide for integer history variables separately. Although users may define new values for any of the  $h1$ ,  $h2$ , or  $h3$  types, the new quantities will be returned to the global arrays only for  $isw$  tasks where residuals are being formed for a solution step (i.e., solution command **form**, **tang**, $,1$ , or **utan**, $,1$  and for history reinitialization during a time update (i.e., solution command, **time**). These access the task options  $isw$  equal to e.g. 3 or 6, respectively. Data will be not returned to the global arrays for  $isw$  tasks like e.g. **stre**. Otherwise each **stre**-command would lead to an update of  $h2$ . Thus, it is necessary to set the variables  $hflgu$  and  $h3flgu$  to true if an update is required, if no update is wanted the variables should be set to false. These variables are typically set in subroutine PMACR. Nothing has to be done on element level.

The mentioned parameters are located in

```
Module  
USE hdatam
```

```
nhmax,nh3max,hflgu,h3flgu
```

### 14.3.3 Typical 2D-element framework for history data

```
subroutine elmtnn(d,ul,xl,ix,tl,s,p,h1,h2,h3,ndf,ndm,nst,isw)
USE hdata
...
if(isw.eq.1) then
c...   input material (examples)
nbs = 2 ! no. of Gauss-points
nlay= 4 ! no. of layers
nhv = 3 ! no. of history terms at layer and Gauss point
nh1 = nbs*nbs*nlay*nhv
return
else if (isw.eq.3) then
c.... stiffness matrix and residual
nbs = 2
nlay = 4
nhv = 3
nn = 0 ! counter history-arrays

do 30 igs = 1,nbs*nbs ! loop Gauss points
    do 31 ilay = 1,nlay ! loop layer
c....     read history data
        s1 = h1(nn+1)
        s2 = h1(nn+2)
        s3 = h1(nn+3)
c....     modify history data (example)
        s1 = s1+1
        s2 = s1+2
        s3 = s1+3
c....     store history data
        h2(nn+1) = s1
        h2(nn+2) = s2
        h2(nn+3) = s3

do 32 ino =1,nel ! loop over node i
    p = p + p(s1,s2,s3) ! residual

    do 33 jno = ino,nel ! loop over node j
        s = s + s(s1,s2,s3) ! tangent
33    continue
32    continue
        nn = nn + nhv ! update counter history-arrays
31    continue
30    continue
        symmetry for s
        return
else if(isw.eq....) then
...
end if
end
```

## 14.4 Typical element framework

```
subroutine elmtnn(d,ul,xl,ix,tl,s,p,h1,h2,h3,ndf,ndm,nst,isw)
implicit double precision (a-h,o-z)
USE bdata
USE cdata
USE eldata
USE hdata
USE iofile
USE pdata6
USE strnam
if(isw.eq.1) then
    Input/output of property data after command: 'mate'
    d(ndd) stores information for each material set
    Return: nh1 = number of nh1/nh2 words/element
    Return: nh3 = number of nh3 words/element
else if(isw.eq.2) then
    Check element for errors. Negative jacobian, etc.
else if(isw.eq.3) then
    Return: Element coefficient matrix and residual
    s(nst,nst) element coefficient matrix
    p(nst) element residual
    h1/h2 history data base: previous/current time step
    h3 history data base: time independent
else if(isw.eq.4) then
    Output element quantities (e.g. stresses)
    n is the current element number, mct is a line counter
else if(isw.eq.5) then
    Return: Element mass matrix
    s(nst,nst) consistent matrix
    p(nst) diagonal lumped matrix
else if(isw.eq.6) then
    Compute residual only
    p(nst) element residual
else if(isw.eq.7) then
    Return: Surface loading for element
    s(nst,nst) coefficient matrix
    p(nst) nodal forces
else if(isw.eq.8) then
    Compute stress projections to nodes (diagonal)
    call plotnn (ix,strea,strea(1+numnp)....)
    strea(1+numnp) projection weight: dt
    strea          projection values: st
else if(isw.eq....) then
    ....
else if(isw.eq.22) then
    Compute element loads
end if
end
```

## 14.5 Use of Parameters and Expression Inputs in New Elements

This is part of programming instructions for adding new elements or subprograms into FEAP. The routine DINPUT must always be used if expressions are to be included as part of the data.

The subroutine DINPUT may be used to input data in any new program module. This routine inputs the data using parameters and expressions as described in the introductory part of the FEAP manual. DINPUT returns them to the subprogram in a double precision array `td(nn)`, see parameters of DINPUT:

```
subroutine dinput (td, nn).
```

The following statements may be included as part of the routine performing the input.

```
USE errchk
USE iofile

real*8 td(5)

1   if(ior.lt.0) write(*,3000)
    call dinput(td, 5)
    if(errck) go to 1
```

The parameters defined in the common blocks are:

```
ior   - input file unit number (if negative, input from terminal).
iow   - output file unit number.
errck - logical variable, if true and error occurred in dinput.
td    - a double precision variable to store values of input.
```

If any `td(i)` is to be used as an integer or `real*4` quantity, it must be assigned to the correct variable. That is, the following operations must be performed.

```
real t
integer j

call dinput (td, 5)

t = td(2)
j = td(1)
```

will perform the reassignments. DINPUT may be used to input up to 16 individual expressions on one input record (each input record is, however, limited to 80 characters).

## 14.6 Error analysis

Stress and energy error indicators are calculated on element level under  $\text{isw} = 9$ . The indicators base on norms which are calculated from the differences between stresses at gauss points and stresses which have been projected to the nodes (based on  $\text{stre},\text{node}$  see  $\text{isw} = 8$ ). The indicator is known in the literature as  $Z^2$ -indicator (Zienkiewicz-Zhu).

Implemented forms can be found for the macro **erro**.

## 14.7 Logical Flags

Flag	Initial value	Description	Macro
fl(1)	False	True=consistent mass	<b>CMAS</b>
fl(2)	F	T=lumped mass	<b>LMAS</b>
fl(3)	T	F=memory for AL	<b>UTAN</b>
fl(4)	T	F=memory for AD,AU	<b>TANG</b>
fl(5)	T	F=memory for lump.M	<b>LMAS</b>
fl(6)	T	F=memory for cons.M	<b>CMAS</b>
fl(7)	T	T=no solution computed	<b>REST</b>
fl(8)	F	T=residual computed	<b>FORM</b> <b>TANG</b> <b>UTANG</b>
fl(9)	F	T=transient calculation	<b>TRANS</b>
fl(10)	F	T=new time step	<b>TIME</b>
fl(11)	F	T=stress projection computed	<b>STRE,NODE</b>
fl(12)	F	T=memory for BFGS	<b>BFGS</b>
arcf	F	T=arc length calculation	<b>ARCL</b>
refl	F	T=reactions computed	<b>REAC</b>
ctfl	F	T=contact solution	<b>CONT</b>
hadd	F	T=macro commands added to history	
hflgu	T	F=history terms are not updated	<b>STRE</b> (ex.)
h3flgu			
pfl	F		
pfr	T	F=supress diagnostic prints	<b>PRIN</b> <b>NOPR</b>
pltfl	T		
zflg	T	F=complex solution	

## 14.8 3D-Material library

It is possible to use every 3D-material implemented in the material library via an user interface, which is described in detail in the following.

The interface is adressed by

call matelib3d

(h1,h2,nh,d(i),ndd,EPS,SIG,CMAT,nsig,ntyp,plout,xgp,tgp,dvp,detf,skfy,skfz,ngp,lgp,lay1gp,lay2gp,imat,isw)

Input of the material data is performed for isw=1. Within this procedure the number of history terms used at each Gauss-point is defined with the parameter nh. The total length of the history-arrays have to be defined in the element: e.g.  $nh_1 = n_{gaus} \cdot nh$ . As output within this phase ndd defines the number of used input parameters, which can be used to check the length of d-array.

Further calls of matelib3d are for other values of isw. These have to be done within an element loop over the Gauss-Points. Based on the calculated strains EPS at this point with respect to a local cartesian base system the subroutine matelib3d returns the associated material matrix CMAT together with the stresses SIG.

Parameters are described in detail in the following table.

Parameter	I/O	Description
h1(nh)	input	element history data vector h1 at Gauss-Point
h2(nh)	output	element history data vector h2 at Gauss-Point
nh	output	number of history terms at Gauss-Point
d(i)	in/output	array of material parameters, where i describes the position of material parameters in d-array
md	output	number of material data
EPS(*)	input	strain vector <b>E</b>
SIG(*)	output	stress vector <b>S</b>
CMAT(*,*)	output	tangent material matrix <b>C</b>
nsig	input	number of strain/stress components, typically=6
ntyp	input	type of element for A-matrix: 1=3D, 2=shell, 3=beam
plout(10)	output	internal parameter for plot
xgp(3)	input	coordinates of Gauss-Point
tgp	input	temperature load at Gauss-Point
dvp	input	det <b>J</b> × weighting factor at Gauss-Point
detf	input	det <b>F</b> at Gauss-Point
skfy	input	$\sqrt{\kappa}$ [for beams $\sqrt{\kappa_y}$ ]: sqrt. of shear correction factor
skfz	input	not used [for beams $\sqrt{\kappa_z}$ ]: sqrt. of shear correction factor
ngp	input	element number
lgp	input	Gauss-Point number
lay1gp	input	layer number
lay2gp	input	Gauss-Point number at layer
imat	input	number of material model to be used
isw	input	solution option from element

## 14.9 Adding elements for FE<sup>2</sup>

- **Update of History Parameter on micro-level**

After convergence of the global load step an update of the history variables on micro-level is necessary. This is done using the macro **updh**. For this purpose code has to be added in the macro-element. For isw=15 a loop over the integration points similar to isw=4 (**stre,print**) or isw=8 (**stre,plot**) is necessary. This loop could end after call **matelib3d(h1....)**.

- **Material number of element on macro-level**

Up to 10 different micro-problems are possible. Thus it is necessary to know the material type number of the actual element. This needs

a) USE fe2mat

b) isw=3,4,6,8,...

**matfe2=ma**

    before call **matelib3d(h1....)**

- **Storage of restart files in FE<sup>2</sup>**

The restart files of the micro-problem could be stored separately (irtyp=0) or in on global file (irtyp=1). In the latter case the local restart-file have to be extracted from the global file.

a) USE fe2tran

b) isw=3,4,6,8,... before Gauss-Point loop: extract all restart files of element

**if(matn.eq.8.and.irtyp.eq.1) call matt3d08(1,n,numel,lint)**

c) isw=3,6,... after integration-point loop: store all restart files of element

**if(matn.eq.8.and.irtyp.eq.1) then**

**if(hflgu.and.h3flgu) call matt3d08(2,n,numel,lint)**

**end if**

## 14.10 Plot of user defined data

Two user defined values 'valuse1' and 'valuse2' can be plotted and stored with the **tplo**-macro. Plots are possible with respect to displacement or time. Values are set to zero within initializing FEAP.

To provide the data some additional code is necessary on element level.

**Example 1:** Energy of system

```
USE plodfu
```

```
e.g. for isw = 4
```

```
C.... calculate values, e.g.  
energy\_pot=....  
energy\_kin=....
```

```
valuse1 = valuse1 + energy\_pot  
valuse2 = valuse2 + energy\_kin
```

**Example 2:** stress in el.x gp. y

```
USE plodfu  
loop gp  
    stress(x,y)=...  
    if(n.eq.x.and(gp.eq.y) valuse1=stress(x,y)  
end loop gp
```

The 'Run'-PCD-File has to be modified slightly to store and show the data.

e.g for one time step

```
time  
loop,,20  
tang,,1  
next  
stre,all (see above isw=4, for ex. 1)  
stre,x (see above isw=4, for ex. 2)  
tplo,us1d or  
tplo,us2d or  
tplo,us1t or  
tplo,us2t
```

The user defined values 'valuse1' and 'valuse2' can be found also in the LDF-File after the macro **tplo,save** has been used.

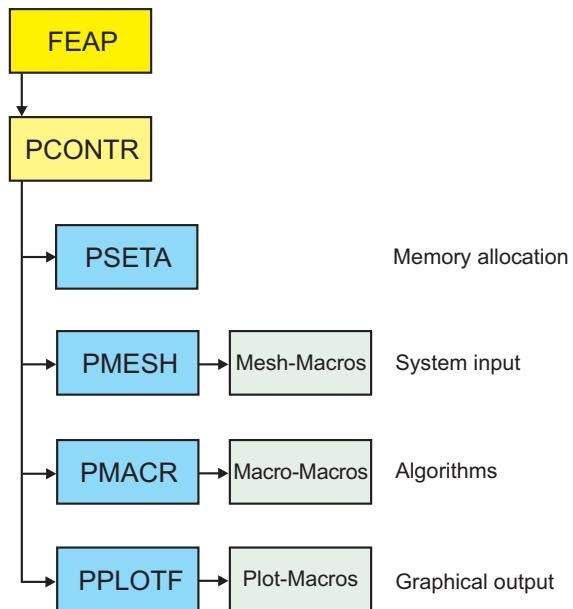
# Chapter 15

## Theory Manual

### 15.1 Programming structure of FEAP

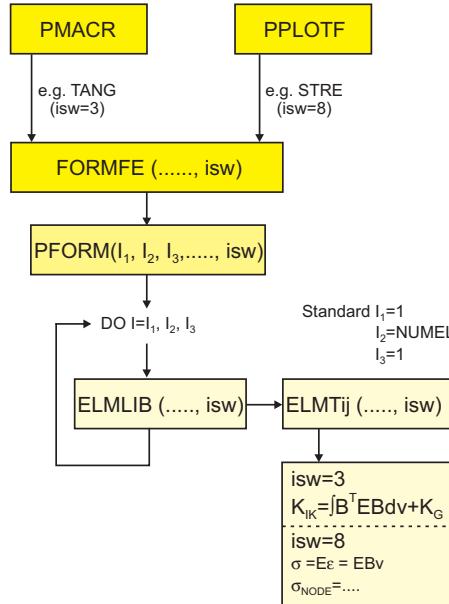
#### 15.1.1 General structure

The general structure with the associated subroutines is depicted in the following figure.

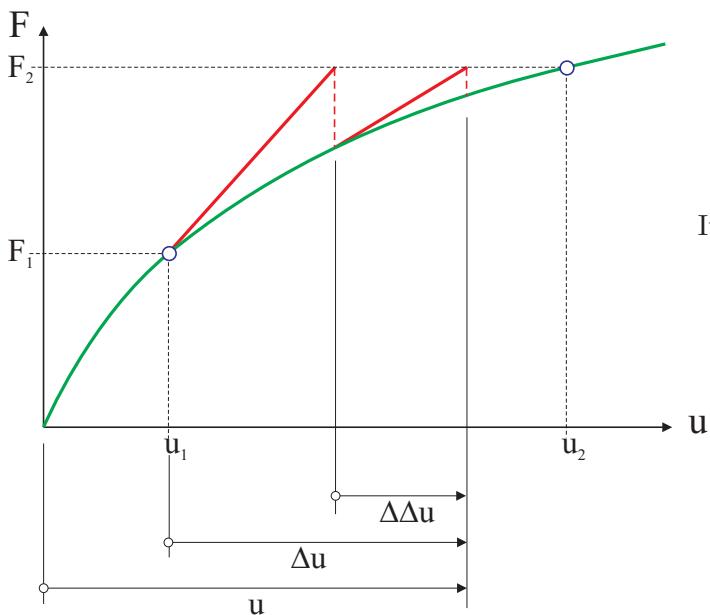


### 15.1.2 Calculating arrays on element level

Within algorithms it is necessary to calculate arrays on element level. Examples are the tangent stiffness matrix (**tang**), print or plot of stress values (**stre**, **stre**) or the mass matrix (**cmas**). Thus, these values have to be calculated from macro or plot-level. The general procedure is described in principle in the following figure. Available data are governed by the parameter **isw**. Details can be found in the part 'Adding elements' (**Control Switch Parameter Values**).



### 15.1.3 Displacement arrays

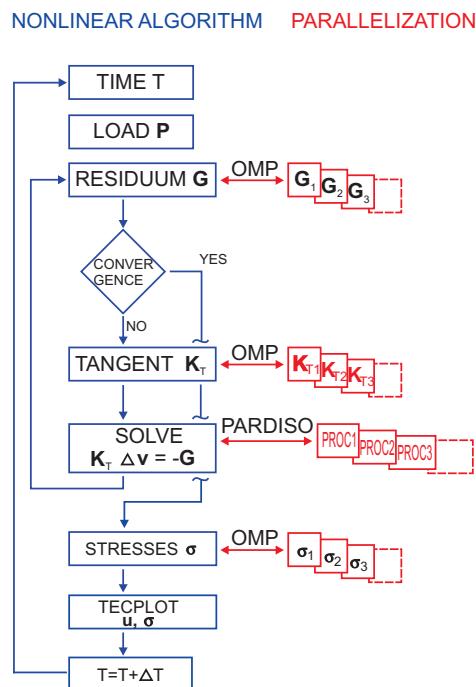


It holds for node  $i$  and degree of freedom  $idf$

$$\begin{aligned} u &\triangleq u_{idf,i} \\ \Delta u &\triangleq u_{idf,nen+i} \\ \Delta\Delta u &\triangleq u_{idf,2nen+i} \end{aligned}$$

### 15.1.4 Making FEAP faster

Two strategies have been followed. These are (i) the use of Compiler options 'Optimize' and (ii)the parallelization of the code. Within the parallelization the element loop, defined in SR PFORM, has been parallelized using OpenMP-techniques. Thus all tasks on element level, like element stiffness matrix, residual vector and stress calculation among others are calculated with respect to the number of processors. The main time consuming part of a linear/non-linear analysis is the solution process. Thus the use of a parallel-solver is necessary. For this purpose an interface to the direct parallel solver package PAR-DISO has been implemented, see the Input-macro **solv**. A schematic scheme of parallelization within the nonlinear time-/load-stepping algorithm is shown in the following figure.



## 15.2 Stability analysis

In a stability analysis critical loads and associated buckling modes (eigenvectors) are calculated. Different versions are possible.

- Classical stability analysis at  $t = 0$  ( $\mathbf{v} = \mathbf{0}$ ,  $\mathbf{P} = \mathbf{0}$ )

# 
$$[\mathbf{K}_L + \Lambda_0 \mathbf{K}_G] \Psi = \mathbf{0} \quad \mathbf{P}_{crit} = \Lambda_0 \mathbf{P}_0$$
 (estimation, correct in case of lin. prebuckling behavior)

$\mathbf{K}_L$  linear stiffness matrix  
 $\mathbf{K}_G$  geometrical stiffness matrix  
 $\Psi$  eigenvector  
 $\Lambda_0$  amplification factor  
 $\mathbf{P}$  actual load  
 $\mathbf{P}_0$  incremental load

# Realisation in FEAP

**tang,,1**  
**geom**  
**subs,,n** n: number of eigenpairs  
Note that **geom** destroys  $\mathbf{K}_T$  !

- Classical stability analysis-alternative version at  $t = 0$  ( $\mathbf{v} = \mathbf{0}$ ,  $\mathbf{P} = \mathbf{0}$ )

# 
$$[\mathbf{K}_T - \omega \mathbf{1}] \varphi = \mathbf{0} \quad \mathbf{P}_{crit} = \Lambda_0 \mathbf{P}_0$$
 (estimation, correct in case of lin. prebuckling behavior)

$\mathbf{K}_T$  tangent stiffness matrix  
 $\mathbf{1}$  unity matrix  
 $\varphi$  eigenvector  
 $\omega$  eigenvalue  
 $\Lambda_0$  critical load with  $\Lambda_0 = \frac{\varphi_1^T \mathbf{K}_L \varphi_1}{\varphi_1^T \mathbf{K}_L \varphi_1 - \omega_1 \varphi_1^T \varphi_1}$

# Realisation in FEAP

**tang,,1**  
**tang,,1**  
**iden**  
**subs,,n**  
**lamb,-1**

- Linear stability analysis  $t = \bar{t}$  ( $\mathbf{v} \neq \mathbf{0}$ ,  $\mathbf{P} = \lambda \mathbf{P}_0 \neq \mathbf{0}$ )

$$\# [\mathbf{K}_L + \Lambda \mathbf{K}_{NL}] \varphi = \mathbf{0}$$

$\mathbf{P}_{crit} = \Lambda(\lambda \mathbf{P}_0)$  (estimation, correct for  $\Lambda \rightarrow 1$ )

$\mathbf{K}_{NL}$  nonlinear part of stiffness matrix, if available  $\mathbf{K}_{NL} = \mathbf{K}_U + \mathbf{K}_G$

$\mathbf{K}_U$  nonlinear part of material matrix

$\mathbf{K}_G$  geometrical stiffness matrix

$\varphi$  eigenvector

$\lambda$  actual load factor, see macro **prop**     $\rightarrow \mathbf{P} = \lambda \mathbf{P}_0$

$\Lambda$  amplification factor

#### # Realisation in FEAP

**tang,,1**

**geom**

**subs,,n**

Note that **geom** destroys  $\mathbf{K}_T$  !

- Nonlinear stability analysis  $t = \bar{t}$  ( $\mathbf{v} \neq \mathbf{0}$ ,  $\mathbf{P} = \lambda \mathbf{P}_0 \neq \mathbf{0}$ )

$$\# [\mathbf{K}_T - \omega \mathbf{1}] \varphi = \mathbf{0}$$

$\mathbf{P}_{crit} = \lambda \mathbf{P}_0$  (estimation, correct for  $\omega \rightarrow 0$ )

$\mathbf{K}_T$  tangent stiffness matrix

$\mathbf{1}$  unity matrix

$\varphi$  eigenvector

$\omega$  eigenvalue

#### # Realisation in FEAP

**tang,,1**

**iden**

**subs,,n**

< **lamb,,,-1** >

#### # Estimation of the critical load $\lambda = f(\omega)$

The eigenvalue problem  $[\mathbf{K}_T - \omega \mathbf{1}] \varphi = \mathbf{0}$  leads to a critical load for  $\omega \rightarrow 0$ . This presents no 'engineering' information on the 'distance' to the critical load. Based on a comparison of both eigenvalue problems it holds in the limiting case:

$$\Lambda_i = \frac{\varphi_i^T \mathbf{K}_L \varphi_i}{\varphi_i^T \mathbf{K}_L \varphi_i - \omega_i \varphi_i^T \varphi_i}$$

The critical load is achieved for  $\Lambda \rightarrow 1$ .

The associated calculation in FEAP is possible via **lamb,,,-1**.

- Calculation of the Determinant of  $\mathbf{K}$  (after factorization)

$$\begin{aligned}
 \det \mathbf{K} &= \det \mathbf{L} \cdot \det \mathbf{U} \quad \text{with } \mathbf{U} = \mathbf{D}\mathbf{L}^T \text{ and } L_{ii} = 1 \\
 \det \mathbf{K} &= \det \mathbf{U} = \prod_{i=1}^{n_{eq}} U_{ii} = U_{11} \cdot U_{22} \cdot U_{33} \cdot \dots \\
 \ln : \quad \ln \det \mathbf{K} &= \sum_{i=1}^{n_{eq}} \ln U_{ii} = \ln U_{11} + \ln U_{22} + \ln U_{33} + \dots \\
 \text{with } \ln U_{ii} &= p_i \ln |U_{ii}| \\
 \text{and } p_i &= +1 \text{ for } U_{ii} > 0 \\
 p_i &= -1 \text{ for } U_{ii} < 0 \\
 e^{\dots} : \quad \det \mathbf{K} &= e^{\sum_{i=1}^{n_{eq}} p_i \ln |U_{ii}|} \\
 &= e^{p_1 \ln |U_{11}|} \cdot e^{p_2 \ln |U_{22}|} \cdot e^{p_3 \ln |U_{33}|} \cdot \dots \\
 &= p_1 \cdot p_2 \cdot p_3 \cdot e^{\ln |U_{11}| + \ln |U_{22}| + \ln |U_{33}| + \dots} \\
 \det \mathbf{K} &= \left( \prod_{i=1}^{n_{eq}} p_i \right) \cdot e^{\sum_{i=1}^{n_{eq}} \ln |U_{ii}|} \\
 \det \mathbf{K} &= (-1)^{n_{neg}} \cdot e^{\sum_{i=1}^{n_{eq}} \ln |U_{ii}|} \quad \text{with } n_{neg} = \text{number of negative diagonals} \\
 \det \bar{\mathbf{K}} &= \frac{\det \mathbf{K}}{\det \mathbf{K}_1} \quad \text{scaled with the first determinant, thus calculations start with } \det \bar{\mathbf{K}} = 1 \\
 &= e^{\ln \det \mathbf{K} - \ln \det \mathbf{K}_1}
 \end{aligned}$$

The scaled determinant  $\det \bar{\mathbf{K}}$  is calculated via **detk**.

## 15.3 Time integration procedures

### 15.3.1 Newmark–method

- Approach acceleration

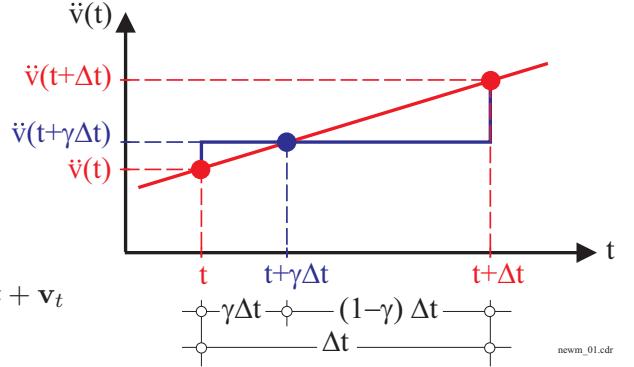
$$\begin{aligned}\ddot{\mathbf{v}}_{t+\gamma\Delta t} &= (1 - \gamma)\ddot{\mathbf{v}}_t + \gamma\ddot{\mathbf{v}}_{t+\Delta t} \quad \text{for } \dot{\mathbf{v}} \\ \ddot{\mathbf{v}}_{t+2\beta\Delta t} &= (1 - 2\beta)\ddot{\mathbf{v}}_t + 2\beta\ddot{\mathbf{v}}_{t+\Delta t} \quad \text{for } \mathbf{v}\end{aligned}$$

- Integration of accelerations

$$\begin{aligned}\dot{\mathbf{v}}_{t+\Delta t} &= (1 - \gamma)\dot{\mathbf{v}}_t\Delta t + \gamma\ddot{\mathbf{v}}_{t+\Delta t}\Delta t + \dot{\mathbf{v}}_t \\ \dot{\mathbf{v}}_{t+\Delta t} &= (1 - 2\beta)\dot{\mathbf{v}}_t\Delta t + 2\beta\ddot{\mathbf{v}}_{t+\Delta t}\Delta t + \dot{\mathbf{v}}_t \\ \mathbf{v}_{t+\Delta t} &= \left(\frac{1}{2} - \beta\right)\ddot{\mathbf{v}}_t\Delta t^2 + \beta\ddot{\mathbf{v}}_{t+\Delta t}\Delta t^2 + \dot{\mathbf{v}}_t\Delta t + \mathbf{v}_t\end{aligned}$$

- Update

$$\begin{aligned}\ddot{\mathbf{v}}_{t+\Delta t} &= \frac{1}{\beta\Delta t^2} \underbrace{(\mathbf{v}_{t+\Delta t} - \mathbf{v}_t)}_{\Delta \mathbf{v}_t} + \underbrace{\frac{-1}{\beta\Delta t}\dot{\mathbf{v}}_t + \left(1 - \frac{1}{2\beta}\right)\ddot{\mathbf{v}}_t}_{\ddot{\mathbf{v}}_{t+\Delta t}^0} \\ \dot{\mathbf{v}}_{t+\Delta t} &= \frac{\gamma}{\beta\Delta t} \underbrace{(\mathbf{v}_{t+\Delta t} - \mathbf{v}_t)}_{\Delta \mathbf{v}_t} + \underbrace{\left(1 - \frac{\gamma}{\beta}\right)\dot{\mathbf{v}}_t + \left(1 - \frac{\gamma}{2\beta}\right)\ddot{\mathbf{v}}_t\Delta t}_{\dot{\mathbf{v}}_{t+\Delta t}^0}\end{aligned}$$



- Differential equations at  $t + \Delta t$

$$\mathbf{M}\ddot{\mathbf{v}}_{t+\Delta t} + \mathbf{C}\dot{\mathbf{v}}_{t+\Delta t} + \mathbf{F}_{t+\Delta t}^{int} = \mathbf{P}_{t+\Delta t}$$

- Residual

$$\mathbf{G} = \mathbf{M} \left[ \frac{1}{\beta\Delta t^2} \Delta \mathbf{v}_t + \ddot{\mathbf{v}}_{t+\Delta t}^0 \right] + \mathbf{C} \left[ \frac{\gamma}{\beta\Delta t} \Delta \mathbf{v}_t + \dot{\mathbf{v}}_{t+\Delta t}^0 \right] + \mathbf{F}_{t+\Delta t}^{int} - \mathbf{P}_{t+\Delta t}$$

- Linearization for geometrical/material nonlinear problem  $\rightsquigarrow \mathbf{F}^{int} = \mathbf{F}^{int}(\mathbf{v})$

$$\mathbf{G}^{i+1}(\mathbf{v}_{t+\Delta t}^i + \Delta \mathbf{v}_{t+\Delta t}^i) = \underbrace{\frac{\partial \mathbf{G}^i}{\partial \mathbf{v}_{t+\Delta t}^i} \Delta \mathbf{v}_{t+\Delta t}^i}_{\mathbf{K}_{eff}^i} + \mathbf{G}^i(\mathbf{v}_{t+\Delta t}^i) \approx 0$$

$$\underbrace{\left[ \frac{1}{\beta\Delta t^2} \mathbf{M} + \frac{\gamma}{\beta\Delta t} \mathbf{C} + \mathbf{K}_T^i \right]}_{\mathbf{K}_{eff}^i} \Delta \mathbf{v}_{t+\Delta t}^i = -\mathbf{G}^i$$

- Newton algorithm

1)  $\mathbf{G}^i = \dots$

2)  $\|\mathbf{G}^i\| < \varepsilon \rightsquigarrow \text{stop}$

3)  $\mathbf{K}_{eff}^i \Delta \mathbf{v}_{t+\Delta t}^i = -\mathbf{G}^i$

4)  $\mathbf{v}_{t+\Delta t}^{i+1} = \mathbf{v}_{t+\Delta t}^i + \Delta \mathbf{v}_{t+\Delta t}^i$

$$\dot{\mathbf{v}}_{t+\Delta t}^{i+1} = \dot{\mathbf{v}}_{t+\Delta t}^i + \frac{\gamma}{\beta\Delta t} \Delta \mathbf{v}_{t+\Delta t}^i$$

$$\ddot{\mathbf{v}}_{t+\Delta t}^{i+1} = \ddot{\mathbf{v}}_{t+\Delta t}^i + \frac{1}{\beta\Delta t^2} \Delta \mathbf{v}_{t+\Delta t}^i$$

5) go to 1

- associated FEAP- macros (1 time step)

**time**

**loop,,n**

**tang,,1**

**next**

### 15.3.2 Overview Implicit Schemes

- Newmark

#  $\mathbf{G} = \mathbf{M}\ddot{\mathbf{v}}_{t+\Delta t} + \mathbf{C}\dot{\mathbf{v}}_{t+\Delta t} + \mathbf{F}_{t+\Delta t}^{int} - \mathbf{P}_{t+\Delta t}$   
 #  $\left[ \frac{1}{\beta\Delta t^2} \mathbf{M} + \frac{\gamma}{\beta\Delta t} \mathbf{C} + \mathbf{K}_T \right] \Delta\mathbf{v}_{t+\Delta t} = -\mathbf{G}$   
 # Remarks: unconditionally stable:  $\beta \geq \frac{1}{4}$ ,  $\gamma \geq \frac{1}{2}$   
 dissipation:  $\gamma = \frac{1}{2}$  no numerical dissipation,  $\gamma > \frac{1}{2}$  numerical dissipation  
 stability:  $\beta \geq \frac{1}{4}$  unconditionally stable,  $\beta = 0$  explicit

- HHT (Hilber-Hughes-Taylor)

#  $\bar{\mathbf{G}} = \mathbf{M}\ddot{\mathbf{v}}_{t+\Delta t} + \mathbf{C}[(1 - \alpha_f)\dot{\mathbf{v}}_{t+\Delta t} + \alpha_f\dot{\mathbf{v}}_t] + (1 - \alpha_f)\mathbf{F}_{t+\Delta t}^{int} - (1 - \alpha_f)\mathbf{P}_{t+\Delta t}$   
 #  $\left[ \frac{1}{1 - \alpha_f} \frac{1}{\beta\Delta t^2} \mathbf{M} + \frac{\gamma}{\beta\Delta t} \mathbf{C} + \mathbf{K}_T \right] \Delta\mathbf{v}_{t+\Delta t} = -\mathbf{G}$   
 # Remarks:  $\mathbf{G} = \frac{\bar{G}}{1 - \alpha_f}$ ,  $\beta = \frac{1}{4}(1 + \alpha_f)^2$ ,  $\gamma = \frac{1}{2} + \alpha_f$ ,  $0 \leq \alpha \leq 1/3$

- Generalized alpha

#  $\bar{\mathbf{G}} = \mathbf{M}[(1 - \alpha_m)\ddot{\mathbf{v}}_{t+\Delta t} + \alpha_m\ddot{\mathbf{v}}_t] + \mathbf{C}[(1 - \alpha_f)\dot{\mathbf{v}}_{t+\Delta t} + \alpha_f\dot{\mathbf{v}}_t] + (1 - \alpha_f)\mathbf{F}_{t+\Delta t}^{int} - (1 - \alpha_f)\mathbf{P}_{t+\Delta t}$   
 #  $\left[ \frac{1 - \alpha_m}{1 - \alpha_f} \frac{1}{\beta\Delta t^2} \mathbf{M} + \frac{\gamma}{\beta\Delta t} \mathbf{C} + \mathbf{K}_T \right] \Delta\mathbf{v}_{t+\Delta t} = -\mathbf{G}$   
 # Remarks:  $\mathbf{G} = \frac{\bar{G}}{1 - \alpha_f}$ ,  $\beta = \frac{1}{4}(1 - \alpha_m + \alpha_f)^2$ ,  $\gamma = \frac{1}{2} - \alpha_m + \alpha_f$   
 numerical damping for high and low frequencies  
 - against oscillations  
 - stabilize nonlinear calculations

### 15.3.3 Explicit Schemes

- Explicit algorithm 1

# Approach velocity, displacement  
 $\dot{\mathbf{v}}_{t+\Delta t} = \dot{\mathbf{v}}_t + \frac{\Delta t}{2}(\ddot{\mathbf{v}}_t + \ddot{\mathbf{v}}_{t+\Delta t}) = \dot{\mathbf{v}}_t + \underbrace{\frac{\Delta t}{2}\ddot{\mathbf{v}}_t}_{\mathbf{v}_{t+\Delta t}^0} + \frac{\Delta t}{2}\ddot{\mathbf{v}}_{t+\Delta t}$   
 $\mathbf{v}_{t+\Delta t} = \mathbf{v}_t + \Delta t\dot{\mathbf{v}}_t + \frac{\Delta t^2}{2}\ddot{\mathbf{v}}_t$   
# Residual  
 $\mathbf{G} = \mathbf{M}\ddot{\mathbf{v}}_{t+\Delta t} + \mathbf{C}\dot{\mathbf{v}}_{t+\Delta t} + \mathbf{F}_{t+\Delta t}^{int} - \mathbf{P}_{t+\Delta t}$   
 $\mathbf{G} = \mathbf{M}\ddot{\mathbf{v}}_{t+\Delta t} + \mathbf{C}(\mathbf{v}_{t+\Delta t}^0 + \frac{\Delta t}{2}\ddot{\mathbf{v}}_{t+\Delta t}) - \mathbf{F}_{t+\Delta t}^{int} - \mathbf{P}_{t+\Delta t}$   
# Iteration equation  
 $(\mathbf{M} + \frac{\Delta t}{2}\mathbf{C})\ddot{\mathbf{v}}_{t+\Delta t} = \mathbf{F}_{t+\Delta t}^{int} - \mathbf{P}_{t+\Delta t} - \mathbf{C}\dot{\mathbf{v}}_{t+\Delta t}^0$   
For  $\mathbf{M} = [M^{ii}]$ ,  $\mathbf{C} = [C^{ii}]$  (lumped matrices)  
 $\ddot{v}_{t+\Delta t}^{(i)} = \frac{1}{(M^{(ii)} + \frac{\Delta t}{2}C^{(ii)})} \left[ F_{t+\Delta t}^{int(i)} - P_{t+\Delta t}^{(i)} - C^{(ii)}v_{t+\Delta t}^{(i)} \right]$

### 15.3.4 Energy-conserving algorithm

- Energy conserving:  $\mathbf{C} = 0$
- Approach velocity, acceleration for  $\alpha = 1/2$

$$\dot{\mathbf{v}}_{t+\alpha\Delta t} = \frac{\dot{\mathbf{v}}_{t+\Delta t} + \dot{\mathbf{v}}_t}{2} = \frac{\mathbf{v}_{t+\Delta t} - \mathbf{v}_t}{\Delta t} \quad \ddot{\mathbf{v}}_{t+\alpha\Delta t} = \frac{\ddot{\mathbf{v}}_{t+\Delta t} - \ddot{\mathbf{v}}_t}{2} = \frac{\dot{\mathbf{v}}_{t+\Delta t} - \dot{\mathbf{v}}_t}{\Delta t}$$

- Approach velocity, acceleration at end of time step

$$\begin{aligned}\dot{\mathbf{v}}_{t+\Delta t} &= \frac{2}{\Delta t} \underbrace{(\mathbf{v}_{t+\Delta t} - \mathbf{v}_t)}_{\Delta \mathbf{v}_t} \underbrace{- \dot{\mathbf{v}}_t}_{\dot{\mathbf{v}}_{t+\Delta t}^0} \\ \ddot{\mathbf{v}}_{t+\Delta t} &= \frac{4}{\Delta t^2} \underbrace{(\mathbf{v}_{t+\Delta t} - \mathbf{v}_t)}_{\Delta \mathbf{v}_t} \underbrace{- \frac{4}{\Delta t} \dot{\mathbf{v}}_t - \ddot{\mathbf{v}}_t}_{\ddot{\mathbf{v}}_{t+\Delta t}^0} \\ \ddot{\mathbf{v}}_{t+\alpha\Delta t} &= \frac{2}{\Delta t^2} \underbrace{(\mathbf{v}_{t+\Delta t} - \mathbf{v}_t)}_{\Delta \mathbf{v}_t} \underbrace{- \frac{2}{\Delta t} \dot{\mathbf{v}}_t}_{\dot{\mathbf{v}}_{t+\alpha\Delta t}^0}\end{aligned}$$

- Residual at  $t + \alpha\Delta t$

$$\begin{aligned}\mathbf{G}_{t+\alpha\Delta t} &= \mathbf{M}\ddot{\mathbf{v}}_{t+\alpha\Delta t} + \mathbf{F}_{t+\alpha\Delta t}^{int} - \mathbf{P}_{t+\alpha\Delta t} \\ \mathbf{G}_{t+\alpha\Delta t} &= \frac{2}{\Delta t^2} \mathbf{M} \underbrace{(\mathbf{v}_{t+\Delta t} - \mathbf{v}_t)}_{\Delta \mathbf{v}} - \frac{2}{\Delta t} \mathbf{M}\dot{\mathbf{v}}_t + \mathbf{F}_{t+\alpha\Delta t}^{int} - \mathbf{P}_{t+\alpha\Delta t}\end{aligned}$$

- Iteration at  $t + \alpha\Delta t$

$$\mathbf{G}_{t+\alpha\Delta t}^{i+1} = \mathbf{G}_{t+\alpha\Delta t}^i + \left( \frac{2}{\Delta t^2} \mathbf{M} + \mathbf{K}_T^i \right) \Delta \mathbf{v}_{t+\Delta t}^i$$

- Update

$$\begin{aligned}\mathbf{v}_{t+\Delta t}^{i+1} &= \mathbf{v}_{t+\Delta t}^i + \Delta \mathbf{v}_{t+\Delta t}^i \\ \dot{\mathbf{v}}_{t+\Delta t}^{i+1} &= \dot{\mathbf{v}}_{t+\Delta t}^i + \frac{2}{\Delta t} \Delta \mathbf{v}_{t+\Delta t}^i \\ \ddot{\mathbf{v}}_{t+\Delta t}^{i+1} &= \ddot{\mathbf{v}}_{t+\Delta t}^i + \frac{4}{\Delta t^2} \Delta \mathbf{v}_{t+\Delta t}^i \\ \ddot{\mathbf{v}}_{t+\alpha\Delta t}^{i+1} &= \ddot{\mathbf{v}}_{t+\alpha\Delta t}^i + \frac{2}{\Delta t^2} \Delta \mathbf{v}_{t+\Delta t}^i\end{aligned}$$

- External forces at  $t + \alpha\Delta t$

$$\mathbf{P}_{t+\alpha\Delta t} = \frac{1}{2} (\mathbf{P}_{t+\Delta t} + \mathbf{P}_t)$$

- Internal forces at  $t + \alpha\Delta t$

$$\mathbf{F}_{t+\alpha\Delta t}^{int} = \bigcup_{e=1}^{numel} \int_{\Omega} \mathbf{B}_{t+\alpha\Delta t}^T \boldsymbol{\sigma}_{t+\alpha\Delta t} d\Omega, \quad \boldsymbol{\sigma}_{t+\alpha\Delta t} = \mathbf{C} \boldsymbol{\varepsilon}_{t+\alpha\Delta t}, \quad \boldsymbol{\varepsilon}_{t+\alpha\Delta t} = \frac{1}{2} (\boldsymbol{\varepsilon}_{t+\Delta t} + \boldsymbol{\varepsilon}_t)$$

- Tangent stiffness matrix at  $t + \alpha\Delta t$

$$\mathbf{K}_T = \bigcup_{e=1}^{numel} \left[ \int_{\Omega} \mathbf{B}_{t+\alpha\Delta t}^T \mathbf{C} \frac{1}{2} \mathbf{B}_{t+\Delta t} d\Omega + \mathbf{K}_G^e (\boldsymbol{\sigma}_{t+\alpha\Delta t}) \right] \text{ not symmetric! , thus } \texttt{utan} \text{ must be used.}$$

- Programming on element-level for  $\text{isw} = 3$  ( $\mathbf{K}_T$ ) and  $6$  ( $\mathbf{F}^{int}$ ) necessary!

include 'ddata.h' with

common /ddata/ theta(4),nrk,nrc,nrm,nrt,nop, , nop = 5  $\leadsto$  energy-conserving algorithm

### 15.3.5 Implicit composite scheme - Bathe

- First substep: see Newmark–method with  $2 \cdot \beta = \gamma = \frac{1}{2}$
- Second substep based on backward difference scheme

# Velocity and acceleration

$$\begin{aligned}\dot{\mathbf{v}}_{t+\Delta t} &= a_1 \cdot \mathbf{v}_t + a_2 \cdot \mathbf{v}_{t+\alpha\Delta t} + a_3 \cdot \mathbf{v}_{t+\Delta t} \\ \ddot{\mathbf{v}}_{t+\Delta t} &= a_1 \cdot \dot{\mathbf{v}}_t + a_2 \cdot \dot{\mathbf{v}}_{t+\alpha\Delta t} + a_3 \cdot \dot{\mathbf{v}}_{t+\Delta t} \\ a_1 &= \frac{1-\alpha}{\alpha \cdot \Delta t}, \quad a_2 = \frac{-1}{(1-\alpha) \cdot \alpha \cdot \Delta t}, \quad a_3 = \frac{2-\alpha}{(1-\alpha) \cdot \Delta t}\end{aligned}$$

# Update

$$\begin{aligned}\mathbf{v}_{t+\Delta t} &= \underbrace{\mathbf{v}_{t+\alpha\Delta t}}_{\mathbf{v}_{t+\Delta t}^0} + \underbrace{(\mathbf{v}_{t+\Delta t} - \mathbf{v}_{t+\alpha\Delta t})}_{\Delta \mathbf{v}_{t+\alpha\Delta t}} \\ \dot{\mathbf{v}}_{t+\Delta t} &= \underbrace{a_1 \cdot \mathbf{v}_t + (a_2 + a_3) \cdot \mathbf{v}_{t+\alpha\Delta t}}_{\mathbf{v}_{t+\Delta t}^0} + a_3^2 \cdot \Delta \mathbf{v}_{t+\alpha\Delta t} \\ \ddot{\mathbf{v}}_{t+\Delta t} &= \underbrace{a_1 \cdot \dot{\mathbf{v}}_t + a_2 \cdot \dot{\mathbf{v}}_{t+\alpha\Delta t} + a_3 \cdot \dot{\mathbf{v}}_{t+\Delta t}^0}_{\dot{\mathbf{v}}_{t+\Delta t}^0} + a_3 \cdot \Delta \mathbf{v}_{t+\alpha\Delta t}\end{aligned}$$

### 15.3.6 Automatic time stepping procedure

In dynamic analysis when implicit integration is used, the automatic time stepping is based on the concept of half-step residuals (Hibbit and Karlsson, 1979). The basic idea is that the time stepping operator defines the velocities and accelerations at the end of the step  $t + \Delta t$  in terms of displacement at the end of the step and conditions at the beginning of the step. Equilibrium is then established at  $t + \Delta t$  which ensures an equilibrium solution at the end of each time step and, thus, at the beginning and end of any individual time step. However, these equilibrium solutions do not guarantee equilibrium throughout the step. The time step control is based on measuring the equilibrium error (the force residuals) at some point during the time step, by using the integration operator, together with the solution obtained at  $t + \Delta t$ , to interpolate within the time step. The evaluation is performed at the half step  $t + \Delta t/2$ . If the maximum entry in this residual vector—the maximum ‘half-step residual’ — is greater than a user-specified tolerance, the time step is considered to be too big and is reduced by an appropriate factor. If the maximum half-step residual is sufficiently below the user-specified tolerance, the time step can be increased by an appropriate factor for the next increment. Otherwise, the time step is deemed adequate.

The algorithm is purely empirical, but experience shows that it works quite well, most especially in initially excited problems with high dissipation.

The half-step residual is based on the assumption that the accelerations vary linearly over the time interval (this is the basis of Newmark’s formulae)

$$\ddot{\mathbf{v}}_{t+\tau\Delta t} = (1-\tau)\ddot{\mathbf{v}}_t + \tau\ddot{\mathbf{v}}_{t+\Delta t} \quad \text{for } 0 \leq \tau \leq 1$$

Having already solved for the state at  $t + \Delta t$ , this equation, together with Newmark’s formulae now written for the time interval from  $t$  to  $t + \Delta t$ , requires that

$$\begin{aligned}
 \Delta \mathbf{v}_{t+\tau\Delta t} &= \tau^3 \Delta \mathbf{v}_t + \tau(1-\tau^2)\Delta t \dot{\mathbf{v}}_t + \tau^2(1-\tau)\frac{\Delta t^2}{2} \ddot{\mathbf{v}}_t \\
 \dot{\mathbf{v}}_{t+\tau\Delta t} &= \frac{\gamma}{\beta\tau\Delta t} \Delta \mathbf{v}_{t+\tau\Delta t} + (1-\frac{\gamma}{\beta})\dot{\mathbf{v}}_t + (1-\frac{\gamma}{2\beta})\tau\Delta t \ddot{\mathbf{v}}_t \\
 \ddot{\mathbf{v}}_{t+\tau\Delta t} &= \frac{1}{\beta\tau^2\Delta t^2} \Delta \mathbf{v}_{t+\tau\Delta t} - \frac{1}{\beta\tau\Delta t} \dot{\mathbf{v}}_t + (1-\frac{1}{2\beta})\ddot{\mathbf{v}}_t
 \end{aligned}$$

with  $\Delta \mathbf{v}_t = \Delta \mathbf{v}_{t+\Delta t} - \Delta \mathbf{v}_t$ . At the mid interval it holds  $\tau = 1/2$ , which leads to

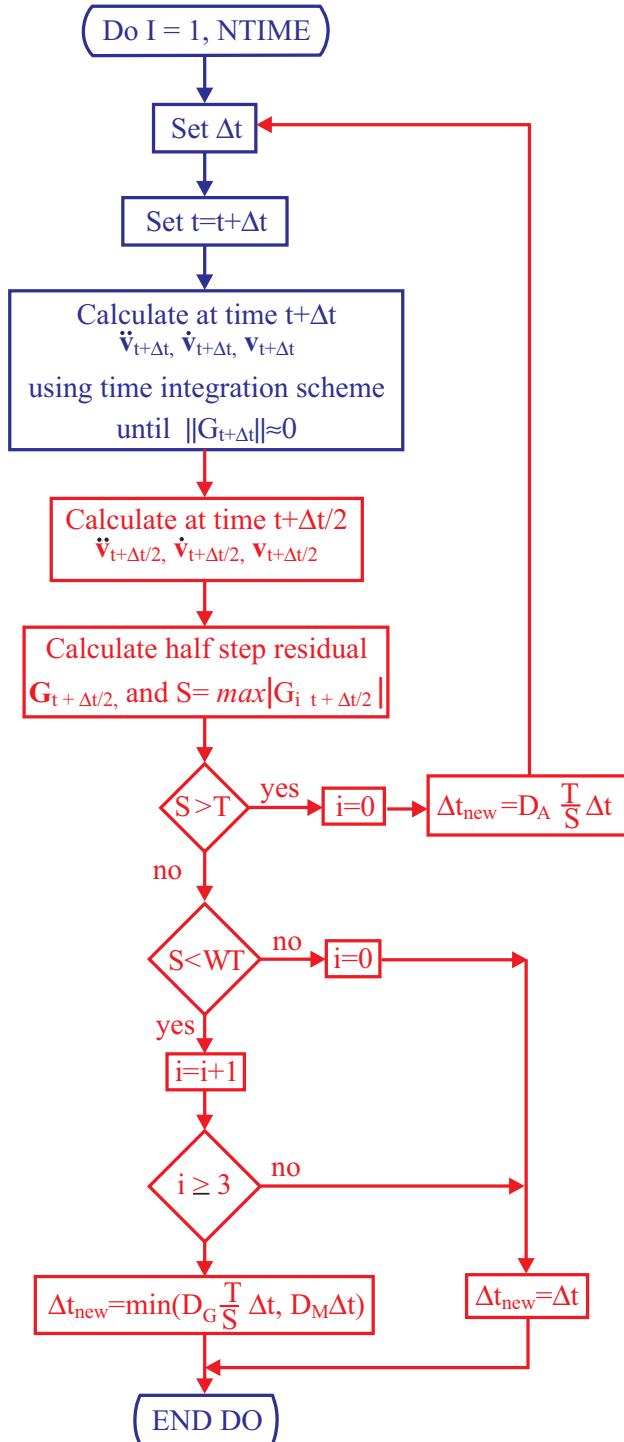
$$\begin{aligned}
 \Delta \mathbf{v}_{t+\Delta t/2} &= \frac{1}{8} \mathbf{v}_t + \frac{3}{8} \Delta t \dot{\mathbf{v}}_t + \frac{1}{16} \Delta t^2 \ddot{\mathbf{v}}_t \\
 \dot{\mathbf{v}}_{t+\Delta t/2} &= \frac{2\gamma}{\beta\Delta t} \Delta \mathbf{v}_{t+\Delta t/2} + (1-\frac{\gamma}{\beta})\dot{\mathbf{v}}_t + \frac{1}{2}(1-\frac{\gamma}{2\beta})\Delta t \ddot{\mathbf{v}}_t \\
 \ddot{\mathbf{v}}_{t+\Delta t/2} &= \frac{4}{\beta\Delta t^2} \Delta \mathbf{v}_{t+\Delta t/2} - \frac{2}{\beta\Delta t} \dot{\mathbf{v}}_t + (1-\frac{1}{2\beta})\ddot{\mathbf{v}}_t
 \end{aligned}$$

The algorithm could be summarized as follows

Suggest that, if  $P$  is a typical magnitude of real forces in an undamped elastic system (for which the high frequency response must be modeled reasonably accurately), then if  $G_{t+\Delta t/2} \approx 0.1P$  consistently, the time stepping solution has high accuracy; if  $G_{t+\Delta t/2} \approx P$  consistently, the time stepping solution has moderately good accuracy; if  $G_{t+\Delta t/2} \approx 10P$  consistently, the time stepping solution is rather coarse.

Problems with large amounts of natural dissipation of energy, such as elastic-plastic systems, are usually less sensitive to time step choice than purely elastic problems, because the energy that appears in higher frequency modes is quickly dissipated. In such cases maximum half-step residuals in the range of 1–10 times typical forces indicate quite acceptable accuracy for most studies, and even values of 10–100 times typical forces can give useful results for primary effects, such as overall deformation. Thus, the method can offer relatively cost-effective solutions for highly dissipative systems for which we require only moderately accurate prediction of the overall response.

The use of the algorithm is described in detail in the following.



Input:  $D_A(=0.85)$ ,  $D_G(=0.80)$ ,  $D_M(=1.25)$ ,  $WT(=0.75)$ ,  $S$ ,  $T$

## 15.4 PCG-Methods

Preconditioned Conjugate Gradient methods are used to solve the problem:

$$\mathbf{K} \cdot \mathbf{u} = \mathbf{f}$$

The simplest algorithm is based on the idea of minimizing the function

$$f(\mathbf{x}) = \frac{1}{2} \mathbf{u} \cdot \mathbf{K} \cdot \mathbf{u} - \mathbf{f} \cdot \mathbf{u}$$

The function is minimized when its gradient

$$\nabla f = \mathbf{K} \cdot \mathbf{u} = \mathbf{f}$$

is zero. The minimization is carried out by generating a succession of search directions  $\mathbf{p}_k$  and improved minimizers  $\mathbf{u}_k$ . At each stage a quantity  $\alpha_k$  is found that minimizes

$$f(\mathbf{u}_k + \alpha \mathbf{p}_k)$$

The efficiency of the algorithm is based on an appropriate preconditioning.

A generalization is the bi-conjugate gradient method which is valid for non positive definite matrices  $\mathbf{K}$  including preconditioning. One possible algorithm is:

1. Initial guess solution vector  $\mathbf{u}_0$

2. Initial residual vectors

$$\mathbf{r}_0 = \mathbf{f} - \mathbf{K} \cdot \mathbf{u}_0$$

$$\bar{\mathbf{r}}_0 = \mathbf{r}_0$$

Initial preconditioning

$$\mathbf{M} \cdot \mathbf{z}_0 = \mathbf{r}_0$$

$$\mathbf{M}^T \cdot \bar{\mathbf{z}}_0 = \bar{\mathbf{r}}_0$$

Initial direction vectors

$$\mathbf{p}_0 = \bar{\mathbf{z}}_0$$

$$\bar{\mathbf{p}}_0 = \bar{\mathbf{z}}_0$$

$k = 0$

3. if  $\mathbf{p}_k = 0$  or  $\mathbf{r}_k = 0$  stop.  $\mathbf{u}_k$  is the solution of  $\mathbf{K} \cdot \mathbf{u} = \mathbf{f}$ .

4. if not, then

new solution vector

$$\mathbf{u}_{k+1} = \mathbf{u}_k + \alpha_k \mathbf{p}_k \quad \text{with } \alpha_k = \frac{\bar{\mathbf{r}}_k \cdot \mathbf{z}_k}{\bar{\mathbf{p}}_k \cdot \mathbf{K} \cdot \mathbf{p}_k}$$

new residual vectors

$$\begin{aligned} \mathbf{r}_{k+1} &= \mathbf{r}_k - \alpha_k \mathbf{z}_k \\ \bar{\mathbf{r}}_{k+1} &= \bar{\mathbf{r}}_k - \alpha_k \bar{\mathbf{z}}_k \end{aligned}$$

new preconditioned vectors

$$\begin{aligned} \mathbf{K} \cdot \mathbf{p}_k &= \mathbf{z}_{k+1} \quad (\text{calculated in } \alpha_k) \\ \mathbf{K}^T \cdot \bar{\mathbf{p}}_k &= \bar{\mathbf{z}}_{k+1} \end{aligned}$$

new direction vectors

$$\begin{aligned} \mathbf{p}_{k+1} &= \mathbf{z}_{k+1} + \beta_k \mathbf{p}_k \\ \bar{\mathbf{p}}_{k+1} &= \bar{\mathbf{z}}_{k+1} + \beta_k \bar{\mathbf{p}}_k \end{aligned} \quad \text{with } \beta_k = \frac{\bar{\mathbf{r}}_{k+1} \cdot \mathbf{z}_{k+1}}{\bar{\mathbf{r}}_k \cdot \mathbf{z}_k}$$

5. Set  $k \rightarrow k + 1$  and go to 3.

Tolerances:

$$\text{itol}=1 \quad \frac{\|\mathbf{K} \cdot \mathbf{x} - \mathbf{f}\|}{\|\mathbf{f}\|} < tol$$

$$\text{itol}=2 \quad \frac{\|\mathbf{M}^{-1}(\mathbf{K} \cdot \mathbf{x} - \mathbf{f})\|}{\|\mathbf{M}^{-1}\mathbf{f}\|} < tol$$

$$\text{itol}=3 \quad \approx \|\mathbf{u}\| < tol$$

$$\text{itol}=4 \quad \approx u_i \max < tol$$

## 15.5 PGMRES-Methods

An excellent iterative equation solver for large linear equation systems with a general unsymmetric and not necessarily positive definite coefficient matrix is the Preconditioned General Minimum RESidual algorithm, which has been proposed in the current form by SAAD (1981) and SAAD (1993). This algorithm is slightly slower than the CG algorithm for well conditioned symmetric positive definite matrices, and it also requires a lot more memory which depends on the choice of the dimension  $m$  of the KRYLOV subspace, but it is far more stable and also works for badly conditioned coefficient matrices much better than the CG algorithm. The preconditioned PGMRES algorithm with modified GRAM-SCHMIDT orthogonalisation and the KRYLOV subspace (dimension  $m$ ) reads for solving  $\mathbf{K} \cdot \mathbf{u} = \mathbf{f}$

1. Initial guess  $\mathbf{u}_0$  for the solution

2.  $r_0 = \mathbf{f} - \mathbf{K} \cdot \mathbf{u}_0$

$$\mathbf{c} = \mathbf{M}^{-1} \cdot \mathbf{r}_0 \quad \mathbf{M}: \text{preconditioner}$$

$$\beta = \|\mathbf{c}\|_2$$

$$\mathbf{v}_1 = \frac{\mathbf{c}}{\beta}$$

For  $j = 1, \dots, m$

$$\mathbf{w}_j = \mathbf{M}^{-1} \cdot \mathbf{K} \cdot \mathbf{v}_j$$

For  $i = 1, \dots, j$

$$h_{ij} = \mathbf{w}_j \cdot \mathbf{v}_i$$

$$\mathbf{w}_j = \mathbf{w}_j - h_{ij} \mathbf{v}_i$$

end

$$h_{j+1,j} = \|\mathbf{w}_j\|_2$$

If  $h_{j+1,j} = 0$  then set  $m = j$  and go to 3

$$\mathbf{v}_{j+1} = \frac{\mathbf{w}_j}{h_{j+1,j}}$$

end

3. Define HESSENBERG matrix

$$\mathbf{H}_m = \{h_{ij}\}_{1 \leq i \leq m+1, 1 \leq j \leq m}$$

4. Compute  $\mathbf{y}_m$  which minimizes  $\|\beta \mathbf{e}_1 - \mathbf{H}_m \cdot \mathbf{y}\|_2$  by applying GIVENS rotations to transform the HESSENBERG matrix into an upper triangular matrix. Here,  $\mathbf{e}_1$  is the  $m + 1$  dimensional unit normal vector pointing in the 1-direction.

5.  $\mathbf{u}_m = \mathbf{u}_0 + \mathbf{V}_m \cdot \mathbf{y}_m$  with  $\mathbf{V}_m = (\mathbf{v}_1, \dots, \mathbf{v}_m)$

6. If  $\|\mathbf{f} - \mathbf{K} \cdot \mathbf{u}_m\|_2 > \text{tol}$  then set  $\mathbf{u}_0 = \mathbf{u}_m$  and go to 2

There are many different variations of the PGMRES algorithm which focus on different aspects. More detailed information about solving large linear equation systems and especially about the PGMRES algorithm can be found in SAAD (2003).

## 15.6 Error Analysis

The theoretical background may be found in Zienkiewicz/Taylor: The Finite Element Method I, 4.th ed., chapter 14.

The basic idea of error analysis is to have the same amount of error in all elements of the FE-mesh.

Thus, the element error is compared to a certain percentage of  $\frac{1}{numel} \times$  a 'system energy'.

To do that we calculate stresses at Gauss-Point  $\mathbf{S}_h$ , averaged stresses at nodes  $\mathbf{S}^*$  and furthermore stress differences  $\Delta\mathbf{S} = (\mathbf{S}^* - \mathbf{S}_h)$

At present three indicators are implemented using **erro** on macro or , **erro** on plot level.

1. Energy-norm
2. L<sub>2</sub>-norm
3. Equivalent stress  $\sigma_v$

Norms 1 and 2 are used in the following way:

$$\begin{aligned}
 & \text{'System energy values'} \quad \|u_1\|_{\Omega} = \left( \sum_{e=1}^{numel} \int_{\Omega_e} (\mathbf{S}_h^T \mathbf{D}^{-1} \mathbf{S}_h) d\Omega_e \right)^{1/2} \\
 & \quad \|u_2\|_{\Omega} = \left( \sum_{e=1}^{numel} \int_{\Omega_e} (\mathbf{S}_h^T \mathbf{S}_h) d\Omega_e \right)^{1/2} \\
 & \text{'Averaged element energy'} \quad \|u\|_{\Omega_e} = \frac{\|u_n\|_{\Omega}}{\sqrt{numel}} \\
 & \text{'relative element energy'} \quad \|e_1\|_{\Omega_e} = \left( \int_{\Omega_e} [\Delta\mathbf{S}^T \mathbf{D}^{-1} \Delta\mathbf{S}] d\Omega_e \right)^{1/2} \\
 & \quad \|e_2\|_{\Omega_e} = \left( \int_{\Omega_e} [\Delta\mathbf{S}^T \Delta\mathbf{S}] d\Omega_e \right)^{1/2} \\
 & \text{'accepted error in element'} \quad \bar{e} = \frac{n1}{100} \cdot \|u\|_{\Omega_e} \quad \text{e.g. } n1=5 \text{ which means an accepted error of 5\%}
 \end{aligned}$$

**element error measure**

$$\boxed{\xi_{\Omega_e} = \frac{\|e_n\|_{\Omega_e}}{\bar{e}}}$$

If all values  $\xi_{\Omega_e}$  are less than 1 no refinement is necessary. A value of  $\xi_{\Omega_e} = 3$  indicates that the permissible error is violated in this element by a factor 3.

Norms 3 is used to compare  $\sigma_v$  on element level with  $Y_0$  as indicator for plasticity:

'System energy values'       $\|u_3\|_{\Omega} = \left( \sum_{e=1}^{numel} Y_0 \cdot Y_0 \right)^{1/2} = Y_0 \sqrt{numel}$

'Averaged element energy'     $\|u\|_{\Omega_e} = \frac{Y_0 \sqrt{numel}}{\sqrt{numel}} = Y_0$

'relative element energy'     $\|e_3\|_{\Omega_e} = (\sigma_{ve}^2)^{1/2} = \sigma_{ve}$

'accepted error in element'     $\bar{e} = \frac{n1}{100} \cdot \|u\|_{\Omega_e}$     e.g. n1=5 which means an accepted error of 5% of  $Y_0$

Based on these errors different further options are possable

- choose a mesh refinement with **reme**,adap
- use different elements in a FE-mesh, e.g. **Elmt05** and **Elmt06** with **mate**,new
- use different material models in one element in a FE-mesh, e.g. elasticity and plasticity with **mate**,new

## 15.7 Eigenvalue Computations

### 15.7.1 Subspace Iteration

### 15.7.2 Lanczos Iteration

#### Lanczos method with complete reorthogonalization

Starting vector  $\mathbf{x}$

$$\text{normalize } \mathbf{x}_1 = \frac{1}{\gamma} \mathbf{x} \quad \text{with } \gamma = \mathbf{x}^T \mathbf{M} \mathbf{x}$$

$$\text{start value } \beta_0 = 0$$

for  $i = 1, \dots, n$

$$\begin{aligned} \mathbf{K}\tilde{\mathbf{x}}_i &= \mathbf{M}\tilde{\mathbf{x}}_i \\ \alpha_i &= \tilde{\mathbf{x}}_i^T \mathbf{M} \mathbf{x}_i \\ \tilde{\mathbf{x}}'_i &= \tilde{\mathbf{x}}_i - \alpha_i \mathbf{x}_i - \beta_{i-1} \mathbf{x}_{i-1} \\ \tilde{\mathbf{x}}_i &= \tilde{\mathbf{x}}'_i - \sum_{k=1}^i (\tilde{\mathbf{x}}_i'^T \mathbf{M} \mathbf{x}_k) \mathbf{x}_k \\ \beta_i &= (\tilde{\mathbf{x}}_i^T \mathbf{M} \mathbf{x}_i)^{\frac{1}{2}} \\ x_{i+1} &= \frac{\tilde{\mathbf{x}}_i}{\beta_i} \\ \mathbf{T}_{(i)} \tilde{\boldsymbol{\varphi}}_{(i)} &= \tilde{\lambda}_{(i)} \tilde{\boldsymbol{\varphi}}_{(i)} \end{aligned}$$

if  $\lambda_{(i)} - \lambda_{(i-1)} > \varepsilon$  go to i

end

$$\varphi = \mathbf{X}\tilde{\varphi} \quad \lambda = \frac{1}{\tilde{\lambda}}$$

Some theory:

it holds (theoretically)  $\mathbf{x}_i^T \mathbf{M} \mathbf{x}_j = \delta_{ij}$

define tridiagonal matrix  $\mathbf{X}_n^T (\mathbf{M} \mathbf{K}^{-1} \mathbf{M}) \mathbf{X}_n = \mathbf{T}_n$  with  $\mathbf{T}_n = \begin{bmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \ddots \\ & & & \alpha_{n-1} & \beta_{n-1} \\ & & & \beta_{n-1} & \alpha_n \end{bmatrix}$

and  $\mathbf{X}_n = [\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n]$

eigenvalue problem  $\mathbf{K}\varphi = \lambda \mathbf{M}\varphi \rightsquigarrow \frac{1}{\lambda} \mathbf{M}\varphi = \mathbf{M} \mathbf{K}^{-1} \mathbf{M}\varphi$

transformation  $\varphi = \mathbf{X}_n \tilde{\varphi}$

leads to  $\mathbf{T}_n \tilde{\varphi} = \frac{1}{\lambda} \tilde{\varphi} = \tilde{\lambda} \tilde{\varphi} \quad \lambda \stackrel{!}{=} \frac{1}{\tilde{\lambda}}$

application in practice  $m \ll n \quad \mathbf{T}_m \tilde{\varphi} = \tilde{\lambda}_m \tilde{\varphi} \rightsquigarrow \text{lowest values of } \lambda_n \approx \frac{1}{\tilde{\lambda}_m}$

## 15.8 Augmented Lagrange Formulation

In a calculation - e.g. including contact - an iteration is necessary. The iteration behavior can be improved with the Augmented Lagrange Formulation, which reduces the dependency on the 'correct' choice of the penalty parameters. This will be achieved by an update of the contact forces with the macro **augm**.

The following algorithms are possible.

### Classical version

Here an update is performed at the end of iteration. This is correct from mathematical point of view and leads within the iterations to quadratic convergence behaviour. To have this behaviour an additional loop is necessary!

```
loop,,n    time step  
          time  
          loop,,m    augm  
            loop,,k    iteration  
              tang,,1  
              next  
              augm  
            next  
          next
```

### Alternative version

Here an update is performed within the iteration. This may be in general faster and is practical from an engineering point of view. The additional loop is not necessary but the convergence behaviour is only linear.

```
loop,,n    time step  
          time  
          loop,,m    iteration  
            tang,,1  
            augm  
          next  
        next
```

## 15.9 Theory of element formulations

### 15.9.1 3D-Material library

The 3D-material library can be used e.g. with **ELMT21**, which is a 8/20/27-node geometrical linear/nonlinear 3D-Solid Element.

- standard stresses and strains and material model:

$$\begin{bmatrix} S_{11} \\ S_{22} \\ S_{33} \\ S_{12} \\ S_{13} \\ S_{23} \end{bmatrix} = \mathbf{C} \begin{bmatrix} E_{11} \\ E_{22} \\ E_{33} \\ 2E_{12} \\ 2E_{13} \\ 2E_{23} \end{bmatrix}$$

The elasticity matrix  $\mathbf{C}$  is described in detail for the implemented models:

#### 15.9.1.1 Material model 1:

linear elastic isotropic, ([Input data](#))

$$\mathbf{C}^{-1} = \begin{bmatrix} \frac{1}{E} & -\frac{\nu}{E} & -\frac{\nu}{E} & 0 & 0 & 0 \\ -\frac{\nu}{E} & \frac{1}{E} & -\frac{\nu}{E} & 0 & 0 & 0 \\ -\frac{\nu}{E} & -\frac{\nu}{E} & \frac{1}{E} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{G} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{G} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{G} \end{bmatrix} \quad \mathbf{C} = \begin{bmatrix} (1-\nu)E^* & \nu E^* & \nu E^* & 0 & 0 & 0 \\ \nu E^* & (1-\nu)E^* & \nu E^* & 0 & 0 & 0 \\ \nu E^* & \nu E^* & (1-\nu)E^* & 0 & 0 & 0 \\ 0 & 0 & 0 & G & 0 & 0 \\ 0 & 0 & 0 & 0 & G & 0 \\ 0 & 0 & 0 & 0 & 0 & G \end{bmatrix}$$

$$\text{with } E^* = \frac{E}{(1+\nu)(1-2\nu)}, G = \frac{E}{2(1+\nu)}$$

Plane stress condition:  $S_{33} = 0$

$$\mathbf{C} = \begin{bmatrix} \hat{E} & \nu \hat{E} & 0 & 0 & 0 \\ \nu \hat{E} & \hat{E} & 0 & 0 & 0 \\ 0 & 0 & G & 0 & 0 \\ 0 & 0 & 0 & G & 0 \\ 0 & 0 & 0 & 0 & G \end{bmatrix}$$

$$\hat{E} = \frac{E}{(1-\nu^2)}, G = \frac{E}{2(1+\nu)}$$

### 15.9.1.2 Material model 2:

linear elastic orthotropic, (Input data)

$$\mathbf{C}^{-1} = \begin{bmatrix} \frac{1}{E_1} & -\frac{\nu_{12}}{E_1} & -\frac{\nu_{13}}{E_1} & 0 & 0 & 0 \\ -\frac{\nu_{12}}{E_1} & \frac{1}{E_2} & -\frac{\nu_{23}}{E_2} & 0 & 0 & 0 \\ -\frac{\nu_{13}}{E_1} & -\frac{\nu_{23}}{E_2} & \frac{1}{E_3} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{G_{12}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{G_{13}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{G_{23}} \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} (E_2 - \nu_{23}^2 E_3) E_1^2 / \Delta & (\nu_{12} E_2 + \nu_{13} \nu_{23} E_3) E_1 E_2 / \Delta & (\nu_{12} \nu_{23} + \nu_{13}) E_1 E_2 E_3 / \Delta & 0 & 0 & 0 \\ (\nu_{12} E_2 + \nu_{13} \nu_{23} E_3) E_1 E_2 / \Delta & (E_1 - \nu_{13}^2 E_3) E_2^2 / \Delta & (\nu_{23} E_1 + \nu_{12} \nu_{13} E_2) E_2 E_3 / \Delta & 0 & 0 & 0 \\ (\nu_{12} \nu_{23} + \nu_{13}) E_1 E_2 E_3 / \Delta & (\nu_{23} E_1 + \nu_{12} \nu_{13} E_2) E_2 E_3 / \Delta & (E_1 - \nu_{12}^2 E_2) E_2 E_3 / \Delta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & G_{12} & 0 \\ 0 & 0 & 0 & 0 & 0 & G_{13} \\ 0 & 0 & 0 & 0 & 0 & G_{23} \end{bmatrix}$$

$$\text{with } \Delta = (E_1 E_2 - \nu_{13}^2 E_2 E_3 - \nu_{23}^2 E_1 E_3 - \nu_{12}^2 E_2^2 - 2\nu_{12}\nu_{13}\nu_{23}E_2 E_3)$$

An associated model with damage is described in material model 15.

### 15.9.1.3 Material model 3:

linear elastic transversal isotropic, (Input data)

$$\mathbf{C}^{-1} = \begin{bmatrix} \frac{1}{E_1} & -\frac{\nu_{12}}{E_1} & -\frac{\nu_{12}}{E_1} & 0 & 0 & 0 \\ -\frac{\nu_{12}}{E_1} & \frac{1}{E_2} & -\frac{\nu_{23}}{E_2} & 0 & 0 & 0 \\ -\frac{\nu_{12}}{E_1} & -\frac{\nu_{23}}{E_2} & \frac{1}{E_2} & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{G_{12}} & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{G_{12}} & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{G_{23}} \end{bmatrix}$$

$$\text{with } E_2 = E_3, G_{12} = G_{13} \text{ and } \nu_{12} = \nu_{13}, \nu_{23} = E_2 / (2G_{23}) - 1$$

$$\mathbf{C} = \begin{bmatrix} E_1(1 - \nu_{23}^2) / \Delta & E_2 \nu_{12}(1 + \nu_{23}) / \Delta & E_2 \nu_{12}(1 + \nu_{23}) / \Delta & 0 & 0 & 0 \\ E_2 \nu_{12}(1 + \nu_{23}) / \Delta & E_2(1 - \nu_{12}^2 \frac{E_2}{E_1}) / \Delta & E_2(\nu_{23} + \nu_{12}^2 \frac{E_2}{E_1}) / \Delta & 0 & 0 & 0 \\ E_2 \nu_{12}(1 + \nu_{23}) / \Delta & E_2(\nu_{23} + \nu_{12}^2 \frac{E_2}{E_1}) / \Delta & E_2(1 - \nu_{12}^2 \frac{E_2}{E_1}) / \Delta & 0 & 0 & 0 \\ 0 & 0 & 0 & G_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & G_{12} & 0 \\ 0 & 0 & 0 & 0 & 0 & G_{23} \end{bmatrix}$$

$$\text{with } \Delta = (1 + \nu_{23})(1 - \nu_{23} - 2\nu_{12}^2 \frac{E_2}{E_1})$$

- **Material model 3a:** linear elastic transversal isotropic and  $\sigma_{33} = 0$

Based on material model 3 the elasticity matrix is

$$\mathbf{C} = \begin{bmatrix} C_{11} & C_{12} & C_{13} & 0 & 0 & 0 \\ C_{12} & C_{22} & C_{23} & 0 & 0 & 0 \\ C_{13} & C_{23} & C_{33} & 0 & 0 & 0 \\ 0 & 0 & 0 & G_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & G_{12} & 0 \\ 0 & 0 & 0 & 0 & 0 & G_{23} \end{bmatrix}$$

Now the assumption  $\sigma_{33} = 0$  is added and  $\varepsilon_{33}$  is eliminated via  $\varepsilon_{33} = -\frac{1}{C_{33}}(C_{13}\varepsilon_{11} - C_{23}C_{22})$ . Finally it holds for the elasticity matrix

$$\mathbf{C} = \begin{bmatrix} E_1/\Delta & \nu_{12}E_2/\Delta & 0 & 0 & 0 & 0 \\ \nu_{12}E_2/\Delta & E_2/\Delta & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & G_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & G_{12} & 0 \\ 0 & 0 & 0 & 0 & 0 & G_{23} \end{bmatrix}$$

with  $\Delta = (1 - \nu_{12}^2 \frac{E_2}{E_1})$

#### 15.9.1.4 Material model 4:

elastic plastic isotropic small strains  $\mathbf{E} = \mathbf{E}_e + \mathbf{E}_p$ , ([Input data](#))

##### References:

- Mises, von R.: Mechanik der plastischen Formänderung von Metallen; ZAMM 8(3)(1928)161-185
- Schütt, J.: Ein inelastisches 3D-Versagensmodell für Beton und seine Finite-Element-Implementierung, Bericht 9(2005), Institut für Baustatik, Karlsruhe
- Perzyna, P.: Fundamental Problems in Viscoplasticity, Adv.Appl.Mech. 243-373 (1966)

#### 15.9.1.5 Material model 5:

elastic plastic isotropic finite strains  $\mathbf{F} = \mathbf{F}_e \cdot \mathbf{F}_p$ , ([Input data](#))

##### References:

- Simo, J.C.: Algorithm for static and dynamic multiplicative plasticity that preserve the classical return mapping schemes of infinitesimal theory, Comp. Meth. Appl. Mech. Eng. 99(1992),61-112
- Klinkel, S.: Theorie und Numerik eines Volumen-Schalen-Elementes bei finiten elastischen und plastischen Verzerrungen, Bericht 7(2000), Institut für Baustatik, Karlsruhe

### 15.9.1.6 Material model 6:

elastic isotropic finite strains Ogden, ([Input data](#))

**Strain energy function:**

$$W_{OG} = \sum_{r=1}^m \left( \frac{\mu_r}{\alpha_r} (\lambda_1^{\alpha_r} + \lambda_2^{\alpha_r} + \lambda_3^{\alpha_r} - 3) - \frac{\mu_r}{2} \ln(I_3) \right) + \frac{\Lambda}{4} (I_3 - 1 - \ln(I_3))$$

$$I_3 = \lambda_1^2 \lambda_2^2 \lambda_3^2$$

**Eigenvalue problem:**  $(\mathbf{C} - \lambda_A^2 \mathbf{G}) \mathbf{N}^A = 0 \quad \mathbf{C} = 2\mathbf{E} + 1, \quad \mathbf{G} = \mathbf{1}$

**Stress vector:**

$$S_A = \frac{1}{\lambda_A^2} \left[ \sum_{r=1}^m (\mu_r (\lambda_A^{\alpha_r} - 1)) + \frac{\Lambda}{2} (I_3 - 1) \right]$$

$$\hat{\mathbf{S}} = (S_1, S_2, S_3)^T \quad \underline{\mathbf{S}} = (S^{11}, S^{22}, S^{33}, S^{12}, S^{13}, S^{23})^T$$

$$\mathbf{t}^{AB} = \begin{pmatrix} (\mathbf{N}^A \cdot \mathbf{G}^1) (\mathbf{N}^B \cdot \mathbf{G}^1) \\ (\mathbf{N}^A \cdot \mathbf{G}^2) (\mathbf{N}^B \cdot \mathbf{G}^2) \\ (\mathbf{N}^A \cdot \mathbf{G}^3) (\mathbf{N}^B \cdot \mathbf{G}^3) \\ (\mathbf{N}^A \cdot \mathbf{G}^1) (\mathbf{N}^B \cdot \mathbf{G}^2) \\ (\mathbf{N}^A \cdot \mathbf{G}^1) (\mathbf{N}^B \cdot \mathbf{G}^3) \\ (\mathbf{N}^A \cdot \mathbf{G}^2) (\mathbf{N}^B \cdot \mathbf{G}^3) \end{pmatrix} \quad \mathbf{T}_1 = \begin{pmatrix} \mathbf{t}^{11T} \\ \mathbf{t}^{22T} \\ \mathbf{t}^{33T} \end{pmatrix}^T \quad \mathbf{T}_2 = \begin{pmatrix} \mathbf{t}^{12T} + \mathbf{t}^{21T} \\ \mathbf{t}^{13T} + \mathbf{t}^{31T} \\ \mathbf{t}^{23T} + \mathbf{t}^{32T} \end{pmatrix}^T$$

$$\underline{\mathbf{S}} = \mathbf{T}_1^T \dot{\mathbf{S}}$$

**Material matrix:**

$$\mathbf{C}_1 = \begin{pmatrix} C_{11} & \Lambda \frac{I_3}{\lambda_1^2 \lambda_2^2} & \Lambda \frac{I_3}{\lambda_1^2 \lambda_3^2} \\ \Lambda \frac{I_3}{\lambda_2^2 \lambda_1^2} & C_{22} & \Lambda \frac{I_3}{\lambda_2^2 \lambda_3^2} \\ \Lambda \frac{I_3}{\lambda_3^2 \lambda_1^2} & \Lambda \frac{I_3}{\lambda_3^2 \lambda_2^2} & C_{33} \end{pmatrix} \quad C_{11} = \frac{1}{\lambda_1^4} \sum_{r=1}^m (\mu_r ((\alpha_r - 2) \lambda_1^{\alpha_r} + 2)) + \frac{\Lambda}{\lambda_1^4}$$

$$C_{22} = \frac{1}{\lambda_2^4} \sum_{r=1}^m (\mu_r ((\alpha_r - 2) \lambda_2^{\alpha_r} + 2)) + \frac{\Lambda}{\lambda_2^4}$$

$$C_{33} = \frac{1}{\lambda_3^4} \sum_{r=1}^m (\mu_r ((\alpha_r - 2) \lambda_3^{\alpha_r} + 2)) + \frac{\Lambda}{\lambda_3^4}$$

$$\mathbf{C}_2 = \begin{pmatrix} \frac{S_1 - S_2}{\lambda_1^2 - \lambda_2^2} & 0 & 0 \\ 0 & \frac{S_1 - S_3}{\lambda_1^2 - \lambda_3^2} & 0 \\ 0 & 0 & \frac{S_2 - S_3}{\lambda_2^2 - \lambda_3^2} \end{pmatrix}$$

$$\underline{\mathbf{C}} = \mathbf{T}_1^T C_1 \mathbf{T}_1 + \mathbf{T}_2^T \mathbf{C}_2 \mathbf{T}_2$$

**References:**

- Ogden, R.W.: Large deformation isotropic Elasticity - On the Correlation of Theory and Experiment for compressible rubberlike solids, Proc. Royal Soc. London A328(1972)565-584
- Klinkel, S.: Theorie und Numerik eines Volumen-Schalen-Elementes bei finiten elastischen und plastischen Verzerrungen, Bericht 7(2000), Institut für Baustatik, Karlsruhe

### 15.9.1.7 Material model 7:

Method of Cells (MOC), Aboudi, ([Input data](#))

#### References:

- Gardner, J.P.: Micromechanical Modeling of Composite Materials in Finite Element Analysis Using an Embedded Cell Approach MSc-Thesis, MIT Cambridge 1994
- Aboudi, J.: Mechanics of Composite Materials: A Unified Micromechanical Approach, Vol 29 Studies in Applied Mechanics. Elsevier, Amsterdam, 1991

### 15.9.1.8 Material model 8:

FE<sup>2</sup>, ([Input data](#))

FEAP runs typically with an input-file called e.g. **ifile**

During e.g. **tang,,1** a loop through all elements and within that a loop through all Gauss-Points occur. On Gauss-point-level the so called micro-problem is executed. For that purpose FEAP is started again (FE<sup>2</sup>!).

Using the INTEL-Compiler the element loop is treated in parallel. The calculation on elements is running on one THREAD separately. Within that the calculation on Gauss Points is than sequential. Here and on the local RVE every form of parallelization is not allowed. The Micro-version of FEAP is compiled sequentially, see the Install-Manual. Furthermore PARDISO (**solv,4**) should not be used!

Using the SALFORD-Compiler FEAP runs sequentially in all loops.

Micro problems are called for **ifile\_i** (i=1,...,nproc identical files. Here i/nproc is the actual/maximum number of threads(parallel) or the actual/maximum number of Gauss-Points(sequential). After one macro step the associated ofiles contain results:

- parallel: **ofile\_i** contains results for last element associated with thread i at last Gauss-point
- sequential: **ofile\_i** contains results for last element at Gauss-point i

Each micro problem has the length values  $l_x, l_y, l_z$ . All boundarys are fixed using **eboo**. Further input are the mesh, e.g. with **bloc** and the description of the **material**. Loads are not defined. Here the macro problem transfer strains **E** which lead to prescribed displacements  $\mathbf{V}_b = \mathbf{AE}$  as 'loads'. These displacements are calculated the micro problem via the macro **epsq**. For the transfer files '**ffile\_i**' are used. The displacements are defined for the actual loading state. Thus the **prop,,1** macro has to describe  $F(t) = const$ . The actual time and time increment are provided during the strain transfer from the macro problem. After solving the local micro problem **S** and **C** are calculated with macro **sigq**. These data are send back using files '**bfile\_i**'. This completes the calculation on micro level.

The input data of Ifile.i are

nopr no output!

.....

bloc

.....

ebou

.....

mate

end

batc,tang calculate one step on RVE from MACRO-Problem

nopr no output on screen!

prop,,1 set load function

rest,,0

epsq read  $\mathbf{E}$ , calculate  $\mathbf{V} = \mathbf{A} \mathbf{E}$ , set actual time

loop,,n

tang,,1

next

sigq calculate  $\mathbf{S}$ ,  $\mathbf{C}$  and send to macro problem

end,,0

1,1,0,t\_max,1  $F(t) = \text{const.}$

stop,tang

batc,updH update H-Array on RVE within MICRO-Problem

nopr no output on screen!

prop,,1 set load function

rest,,0

updH,,2 update  $H_2 \rightarrow H_1$  in case of convergence

end,,0

1,1,0,t\_max,1  $F(t) = \text{const.}$

stop,updH

batc,micr show results on RVE for one GP of element n

nopr no output on screen!

prop,,1 set load function

rest,,0

plot,pers perspective plot

plot,hide,1 hidden line plot

plot,mesh plot mesh

plot,axis plot coordinate system

end,,0

1,1,0,t\_max,1  $F(t) = \text{const.}$

10,10,5 view point

0,0,1 vertical axis

inte switch to interactive mode

stop,micr

A procedure step.pcd for one time step on the macro scale should be

```
n1,0,0
time      set new time
loop,,n   begin iteration for solution
tang,,1   solve problem
next      end iteration at convergence
updh,,1   activate update on local level: done by batch,updh on micro-level
```

One has to prepare or to modify only **one** input file Ifile\_01. Necessary copies of this file, e.g. Ifile\_02, Ifile\_03, ... are produced using the below defined batch file **fbatch**. A minimum of nproc=number of threads(parallel) or nproc=number of Gauss points(sequential) is necessary. More copies are allowed and do not influence the calculations. Furthermore all no longer used files in the micro-directory are deleted running this file.

#### Example for file fbatch

```
cd d:\w\feap\exe\rve
copy irve_01 irve_02
copy irve_01 irve_03
copy irve_01 irve_04
copy irve_01 irve_05
copy irve_01 irve_06
copy irve_01 irve_07
copy irve_01 irve_08
....
until nproc=no. of threads(parallel) or nproc=no. of Gauss points(sequential)
del frve*
del brve*
del hrve*
del rrve*
del orve*
del *.lnk
```

**Macro-Problem**

Read Macro-mesh from ifile

LOOP **A**-Time

Set 'TIME'  $T = T + \Delta T$

'LOAD'

LOOP,,N **B**-Tang

TANG,,1

LOOP **C**-Elmt

LOOP **D**-GP

Send **E** and  $\Delta T$  to micro-problem  $\rightarrow$  to (TGP)

Start FEAP and Read micro-mesh ifile\_iproc  
(iproc=actual processor)

(Feap.exe ...

-Ifile\_iproc -Oofile\_iproc -Rrfile\_GP.EL -Srfle\_GP.EL )

Read History data of GP ( $H_{1L}$ ,  $H_{2L}$ )

(REST,rfile\_GP.EL)

PROP,,1 with parameters 1,1,0,tmax,1 ( $F(t) = \text{const!}$ )

Read **E** and actual macro-'TIME' from (TGP)

EPSQ, Set 'loads' **V=AE**

Loop,,M **A**-Tang

TANG,,1

Next **A**-Tang

SIGQ

END

(includes save of hist.data of GP to rfile\_GP.EL) ( $H_{1L}$ ,  $H_{2L}$ )

(TGP)  $\leftarrow$  Send **C**, **S** to macro-problem

Read **C**, **S** from (TGP)

NEXT **D**-GP

Calculate **K** and **G** = **R** -  $\lambda P$

NEXT **C**-Elmt

NEXT **B**-Tang

'UPDATE' History Data of MICRO-Problem ( $H_{2L} \rightarrow H_{1L}$ )

NEXT **A**-Time

END

## RVE-Boundary Conditions

- Dimensions of RVE

$$\begin{aligned} xm &\leq x \leq xp \\ ym &\leq y \leq yp \\ zm &\leq z \leq zp \end{aligned}$$

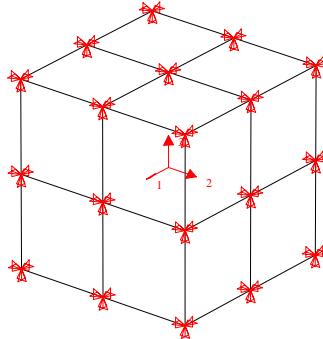
typically with  $xm=ym=zm=-0.5$  and  $xp=yp=zp=+0.5$

- Displacement boundary conditions

To do/Theory: Fix nodes on all faces. Then boundary conditions are set via  $\mathbf{v}_{bc} = \mathbf{A}\boldsymbol{\varepsilon}$ .

FEAP-Input:

```
eboun
1,xm,1,1,1
1,xp,1,1,1
2,ym,1,1,1
2,yp,1,1,1
3,zm,1,1,1
3,zp,1,1,1
```



- Stress boundary conditions

Boundary conditions are set via  $\mathbf{t}_{bc} = \boldsymbol{\sigma} \mathbf{n}_{bc}$ .

This type of boundary condition is currently not implemented.

- Periodic boundary conditions

To do/Theory:

# 1 corner nodes: nodes have to be fixed with standard boundary conditions.

These are set via  $\mathbf{v}_{bc} = \mathbf{A}\boldsymbol{\varepsilon}$

# 2 all other nodes on edges and outer faces have to be linked.

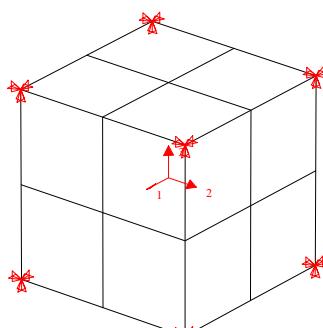
The linking condition is  $\mathbf{v}_{slave} = \mathbf{v}_{master} + \mathbf{A}(x_{slave} - x_{master})\boldsymbol{\varepsilon}$

Typically one can use  $x_{slave} = xp$  and  $x_{master} = xm$ .

FEAP-Input:

Step 1: Fix all corner nodes with

```
poin
xm,ym,zm
0,
1,1,1
xp,ym,zm
0,
1,1,1
xp,yp,zm
0,
1,1,1
xm,yp,zm
0,
1,1,1
xm,ym,zp
```

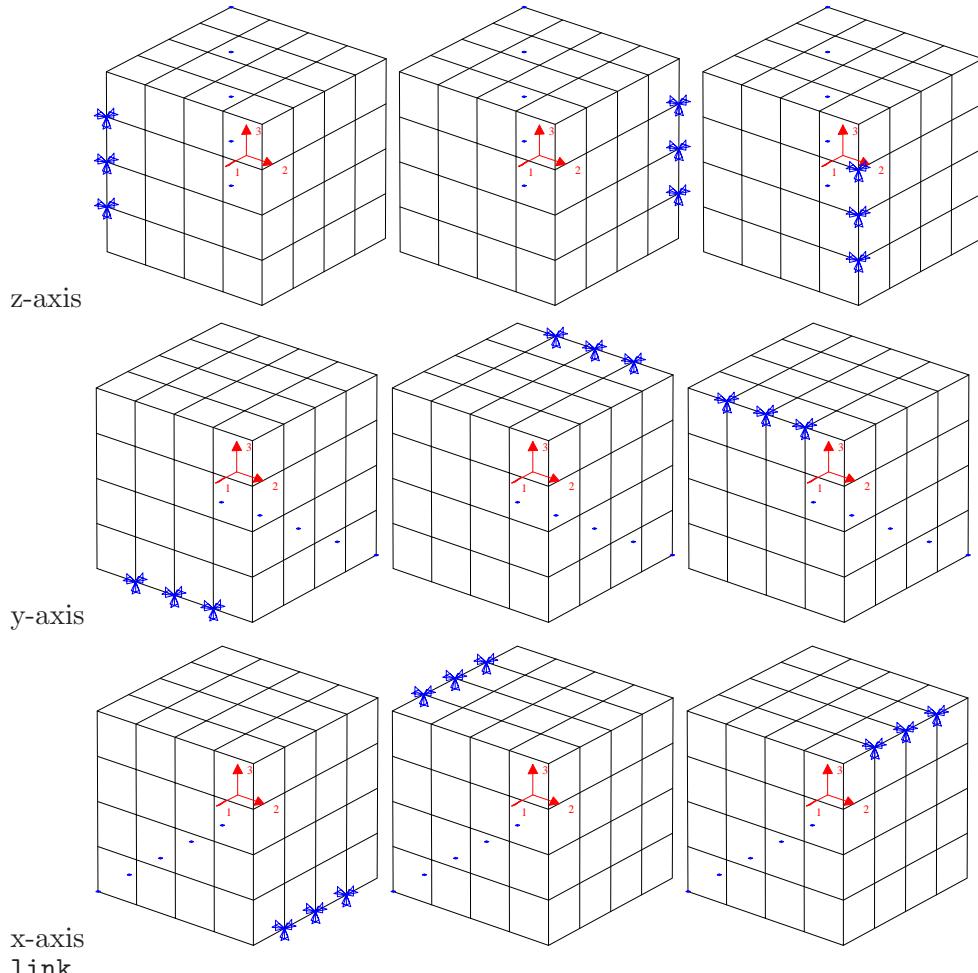


```

0,
1,1,1
xp,ym,zp
0,
1,1,1
xp,yp,zp
0,
1,1,1
xm,yp,zp
0,
1,1,1

```

Step 2a: Set first link conditions on edges. (Note that corner nodes have displacement b.c.s.)

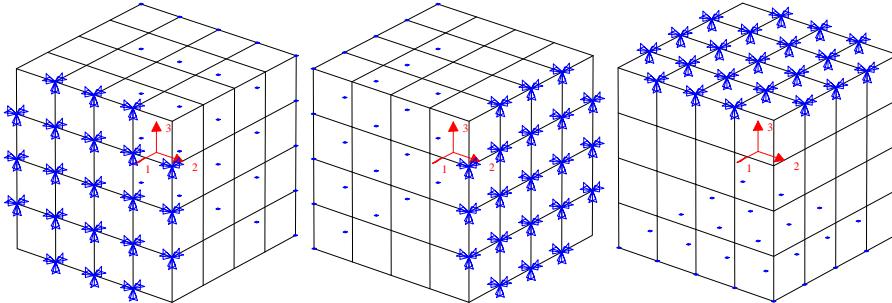


```

5,1,xm,xp,2,ym,ym,0,0,0
5,1,xm,xm,2,ym,yp,0,0,0
5,1,xm,xp,2,ym,yp,0,0,0
5,1,xm,xp,3,zm,zm,0,0,0
5,1,xm,xm,3,zm,zp,0,0,0
5,1,xm,xp,3,zm,zp,0,0,0
5,2,ym,yp,3,zm,zm,0,0,0
5,2,ym,ym,3,zm,zp,0,0,0
5,2,ym,yp,3,zm,zp,0,0,0

```

Step 2b: Set then link conditions on all faces



link

```
4,1,xm,xp,0,0,0
4,2,ym,yp,0,0,0
4,3,zm,zp,0,0,0
```

Set both link commands together!

link

```
5,1,xm,xp,2,ym,ym,0,0,0
5,1,xm,xm,2,ym,yp,0,0,0
5,1,xm,xp,2,ym,yp,0,0,0
5,1,xm,xp,3,zm,zm,0,0,0
5,1,xm,xm,3,zm,zp,0,0,0
5,1,xm,xp,3,zm,zp,0,0,0
5,2,ym,yp,3,zm,zm,0,0,0
5,2,ym,ym,3,zm,zp,0,0,0
5,2,ym,yp,3,zm,zp,0,0,0
4,1,xm,xp,0,0,0
4,2,ym,yp,0,0,0
4,3,zm,zp,0,0,0
```

- Periodic boundary conditions for RVEs of shell structures

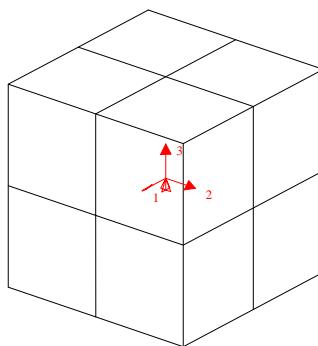
To do/Theory:

```
# 1 Fix center node vertical (to suppress rigid body rotations-other nodes are possible!).
# 2 Fix all corner nodes horizontal. Boundary conditions are set for fixed nodes via  $\mathbf{v}_{bc} = \mathbf{A}\boldsymbol{\varepsilon}$ .
# 3 Set link conditions on faces vertical with  $\mathbf{v}_{slave} = \mathbf{v}_{master} + \mathbf{A}(x_{slave} - x_{master})\boldsymbol{\varepsilon}$ 
# 4 Top and bottom faces are free from boundary conditions.
```

FEAP-Input:

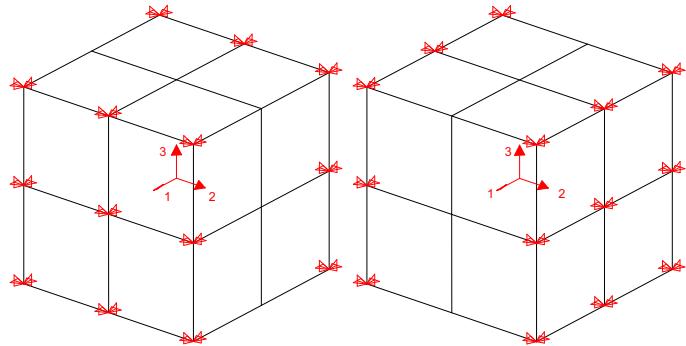
Step 1: Fix center node vertical (to suppress rigid body rotations-other nodes are possible!)

```
poin
0,0,0,f
0
0,0,1
0
0
0
```



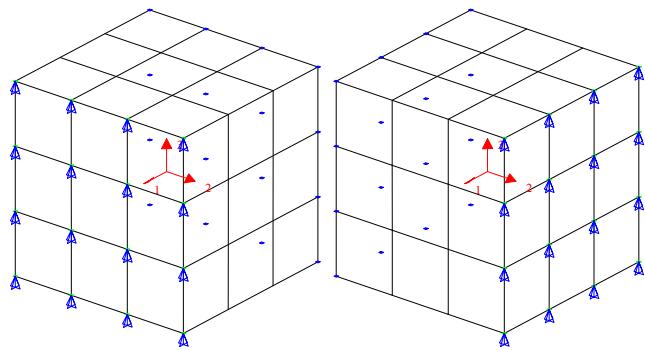
Step 2: Fix all corner nodes horizontal

```
eboun
1,xm,1,1,0
1,xp,1,1,0
2,ym,1,1,0
2,yp,1,1,0
```



Step 3: Set link conditions on faces vertical

```
link
6,1,xm,xp,1,1,0
6,2,ym,yp,1,1,0
```



## Results on MICRO-problem (RVE)

It is possible to check results on the MICRO-problem (RVE) during the calculation. For that purpose the Macro-Problem is running in interactive mode.

Results on the associated MICRO-problem RVE can be checked, if a second Version of FEAP is started. Here, the **input file** and the **restart file** have to be chosen with respect to the Gauss-point number **i** and the element number **n**.

Example: irve\_i and rrve\_i.n

## Summary

- 1 start MICRO-problem FEAP with input file Ifile\_i and restart file RFile\_i.n and the macros within batc,micr... stop,micr (see above)
- 2 view results on element n, Gauss-point i using macros
- 3 close MICRO-problem with **QUIT**
- 4 goto 1 for other element n or other Gauss-point i

### 15.9.1.9 Material model 9:

small strain isotropic damage, ([Input data](#))

**Reference:**

- Simo, J.C., Ju, J.W., Strain- and stress-based continuum damage models. Parts I and II. Int. J. Solids Structures 23(1987), 821-869

### 15.9.1.10 Material model 10:

concrete model, Schütt, ([Input data](#))

**Reference:**

- Schütt, J.: Ein inelastisches 3D-Versagensmodell für Beton und seine Finite-Element-Implementierung, Bericht 9(2005), Institut für Baustatik, Karlsruhe

### 15.9.1.11 Material model 11:

small strain visco-elastic, ([Input data](#))

The material model can be used on the deviatoric part of the strains or on all strain components.

It holds

$$\boldsymbol{\sigma} = \mathbf{s} + \mathbf{m}p \quad \text{with Cauchy stress } \boldsymbol{\sigma}, \text{ stress deviator } \mathbf{s}, \text{ and mean (pressure) stress } p = \frac{1}{3}\mathbf{m}^T \boldsymbol{\sigma}$$

Furthermore it holds

$$\boldsymbol{\varepsilon} = \mathbf{e} + \frac{1}{3}\mathbf{m}\theta \quad \text{with strain } \boldsymbol{\varepsilon}, \text{ strain deviator } \mathbf{e}, \text{ and volume change } \theta = \mathbf{m}^T \boldsymbol{\varepsilon}, \mathbf{m} = [1, 1, 1, 0, 0, 0]$$

The pressure-volume parts are governed by a linear elastic model  $p = K\theta$  with  $K = \frac{E}{3(1-2\nu)}$  the bulk elastic modulus.

The deviatoric parts are assumed to satisfy a linear viscoelastic model.

This could be

$$\mathbf{s} = 2G(\mu_0 \mathbf{e} + \sum_{i=1}^N \mu_i \mathbf{q}^i) \quad \dot{\mathbf{q}}^i + \frac{1}{\tau_i} \mathbf{q}^i = \dot{\mathbf{e}}$$

$\mu_i$  are the viscoelastic shear parameters and  $\tau_i$  are the viscoelastic relaxation times

This form of the representation is equivalent to a set of Maxwell models in parallel.

**Reference:**

- Simo, J.C., Hughes, T.J.R.: Computational Inelasticity, Springer, 1988.

### 15.9.1.12 Material model 12:

linear piezoelectric material model for extended 'stresses' and 'strains', ([Input data](#))

$$\begin{bmatrix} \mathbf{S} \\ -\vec{\mathbf{D}} \end{bmatrix} = \begin{bmatrix} \mathbf{C} & -\mathbf{e}^T \\ -\mathbf{e} & -\epsilon \end{bmatrix} \begin{bmatrix} \mathbf{E} \\ \vec{\mathbf{E}} \end{bmatrix}$$

$$\left[ \begin{array}{c|ccccc|ccc|c} S_{11} & (1-\nu)E^* & \nu E^* & \nu E^* & 0 & 0 & 0 & 0 & 0 & E_{11} \\ S_{22} & \nu E^* & (1-\nu)E^* & \nu E^* & 0 & 0 & 0 & 0 & 0 & E_{22} \\ S_{33} & \nu E^* & \nu E^* & (1-\nu)E^* & 0 & 0 & 0 & 0 & 0 & E_{33} \\ S_{12} & 0 & 0 & 0 & G & 0 & 0 & 0 & 0 & 2E_{12} \\ S_{13} & 0 & 0 & 0 & 0 & G & 0 & -\mathbf{e}_{15} & 0 & 2E_{13} \\ S_{23} & 0 & 0 & 0 & 0 & 0 & G & 0 & -\mathbf{e}_{15} & 2E_{23} \\ \hline -\vec{D}_1 & 0 & 0 & 0 & 0 & -\mathbf{e}_{15} & 0 & -\epsilon & 0 & \vec{E}_1 \\ -\vec{D}_2 & 0 & 0 & 0 & 0 & 0 & -\mathbf{e}_{15} & 0 & -\epsilon & \vec{E}_2 \\ -\vec{D}_3 & -\mathbf{e}_{13} & -\mathbf{e}_{13} & -\mathbf{e}_{33} & 0 & 0 & 0 & 0 & 0 & \vec{E}_3 \end{array} \right]$$

with  $E^* = \frac{E}{(1+\nu)(1-2\nu)}$ ,  $G = \frac{E}{2(1+\nu)}$  and  $\mathbf{C}$  as in material model 1.

### 15.9.1.13 Material model 13:

dielectric elastomers, ([Input data](#))

### 15.9.1.14 Material model 14:

finite elastic strains, Blatz-Ko, ([Input data](#))

#### References:

- Klinkel, S.: Theorie und Numerik eines Volumen-Schalen-Elementes bei finiten elastischen und plastischen Verzerrungen, Bericht 7(2000), Institut für Baustatik, Karlsruhe
- Holzapfel,G.A.; Nonlinear solid mechanics - A continuum approach for engineering, John Wiley & Sons Ltd.

### 15.9.1.15 Material model 15:

Transversal isotropic with damage, (Input data)

The general form of an orthotropic material is described for material model 2

With the assumptions  $E_2 = E_3$  and  $G_{12} = G_{13}$  follows

$$\mathbf{C} = \begin{bmatrix} (1 - \nu_{23}^2)E_1^2E_2/\Delta & (\nu_{12} + \nu_{13}\nu_{23})E_1E_2^2/\Delta & (\nu_{12}\nu_{23} + \nu_{13})E_1E_2^2/\Delta & 0 & 0 & 0 \\ (\nu_{12} + \nu_{13}\nu_{23})E_1E_2^2/\Delta & (E_1 - \nu_{13}^2E_2)E_2^2/\Delta & (\nu_{23} + \nu_{12}\nu_{13})E_1E_2^2/\Delta & 0 & 0 & 0 \\ (\nu_{12}\nu_{23} + \nu_{13})E_1E_2^2/\Delta & (\nu_{23} + \nu_{12}\nu_{13})E_1E_2^2/\Delta & (1 - \nu_{12}^2)E_1E_2^2/\Delta & 0 & 0 & 0 \\ 0 & 0 & 0 & G_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & G_{12} & 0 \\ 0 & 0 & 0 & 0 & 0 & G_{23} \end{bmatrix}$$

$$\text{with } \Delta = (E_1E_2 - \nu_{13}^2E_2^2 - \nu_{23}^2E_1E_2 - \nu_{12}^2E_2^2 - 2\nu_{12}\nu_{13}\nu_{23}E_2^2)$$

The further assumptions  $\nu_{13} = \nu_{23} = 0$  (!?) lead to:

$$\mathbf{C} = \begin{bmatrix} E_1/\Delta & \nu_{12}E_2/\Delta & 0 & 0 & 0 & 0 \\ \nu_{12}E_2/\Delta & E_2/\Delta & 0 & 0 & 0 & 0 \\ 0 & 0 & E_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & G_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & G_{12} & 0 \\ 0 & 0 & 0 & 0 & 0 & G_{23} \end{bmatrix}$$

$$\text{with } \Delta = (1 - \nu_{12}^2)\frac{E_2}{E_1}$$

This is similar to material model 3a, except  $C_{33} \neq 0$ !

**References:**

- Krawiec M.: Modellierung des 3D-Schädigungsverhaltens von Faserverbundwerkstoffen in Solid-schalen, Diplomarbeit [2013], Institut für Baustatik, Karlsruhe.

Tsai-Wu criteria:

- Liu, K.-S. and Tsai, S. W., A progressive quadratic failure criterion for a laminate. Composites Science Technology, (58)7, 1998, 1023–1032.

Hashin (extended) criteria:

- Goyal Vinay K., Jaunky Navin R., Johnson Eric R. Ambur Damodar R.: Intralaminar and interlam-inar progressive failure analyses of composite panels with circular cutouts. Composite Structures, (64)1, 2004, 91–105.

Puck criteria:

- Puck, A.: Festigkeitsanalyse von Faser-Matrix-Laminaten, Modelle für die Praxis. Carl Hanser Verlag, 1996.

Puck A., Schürmann H.: Failure Analysis of FRP Laminates by Means of Physically Based Phenomenological Models Composites Science Technology (58)7, 1998, 1045–1067

Cuntze criteria:

- Cuntze, R.G., Freund, A. : The predictive capability of failure mode concept-based strength criteria for multidirectional laminates. Composites Science Technology, (64)3–4, 2004, 343–377.

# Chapter 16

## Applications and FAQs

### 16.1 FAQs

#### 16.1.1 Crash within start procedure

If there occurs a crash of the program within the start procedure, it may happens that the file `feapname`, which contains the names of input- output- and restart file will be empty. This results in the following error message



which then stops the program.

The problem is solved by deleting the file `feapname`!

#### 16.1.2 Mixing elements

A mixing of elements in a finite element mesh is in general possible within FEAP. The following aspects have to be noted.

Here we will discuss - as an example - the mixing of a solid element(3D) ( $\text{nel}_{3D}=8$ ,  $\text{ndf}_{3D}=3$ ) with a shell element(sh) ( $\text{nel}_{sh}=4$ ,  $\text{ndf}_{sh}=6$ ).

- Within the first input card `feap` one has to define `ndf`, which is the maximum number of degrees-of-freedom on any node. In the example it holds  $\text{ndf}=\text{ndf}_{max}=\max(\text{ndf}_{3D}, \text{ndf}_{sh})=6$ .  
A similar definition is for `nen`, which is maximum number of nodes on any element. In the example it holds  $\text{nen}=\text{nel}_{max}=\max(\text{nel}_{3D}, \text{nel}_{sh})=8$ .
- The boundary conditions have to be set carefully. This means that degrees-of-freedom, where no stiffness is applied by an element have to be fixed. In the example it holds that for any node  $i$  which is not connected with a shell element a boundary card like e.g.

**boun**

i,0,0,0,0,1,1,1

has to be set.

- Within the element input via the macro **elem** only the necessary nodes are input. This means in the example

**elem**

1(3D),mat1,node1,node2,node3,node4,node5,node6,node7,node8

2(sh),mat2,node1,node2,node3,node4

- The used elements have to be prepared for the situation of larger numbers of nodes or degrees-of-freedom. For that purpose the storage of data into the element stiffness matrix **K** and the element load vector or residual vector **P** have to be correct.

**K** is stored into s(nst,nst) and **P** is stored into p(nst). Here, nst is defined as nen×ndf. Be aware of the difference nen=nel<sub>max</sub>=maximum number of nodes on elements and nel=number of nodes on actual element. In the example it holds nen=8 and nel=8 for volume elements and nel=4 for shell elements. Displacements are stored in ul(ndf,nen). Thus matrices s, p, ul are defined always for the maximal possible situation. Storage of data is then done using the nel<sub>local</sub> nodes and ndf<sub>local</sub> degrees of freedom. Jumps are necessary if nel<sub>local</sub> < nel<sub>max</sub>=nen or ndf<sub>local</sub> < ndf<sub>max</sub>=ndf.

- The introduced degrees of freedom must fit together. In the example it holds  $\mathbf{u}_{3D} = [u_x, u_y, u_z]^T$  and  $\mathbf{u}_{sh} = [u_x, u_y, u_z, \varphi_x, \varphi_y, \varphi_z]^T$ , which can be used without problems.

If problems occur it is possible to change the position of degrees of freedoms within the **mate**-macro defining an idfg-array with the global dofnumbers for the used element.

**mate**

ma,iel,&lt;idfg1,idfg2,...,idfgn&gt;

...

- The plot macro **disp** should act without any problem.
- The plot macro **stre** should be used carefully. Typically stress<sub>i</sub> is plotted for all elements which makes no sense. In the example it holds stress<sub>1(3D)</sub> =  $\sigma_{xx}$  and stress<sub>1(sh)</sub> =  $N_x$ . For this purpose one can plot only values for each material i via **matn,-1** and then **matn,i**. Note that the element code have to be changed with respect to **matn**.

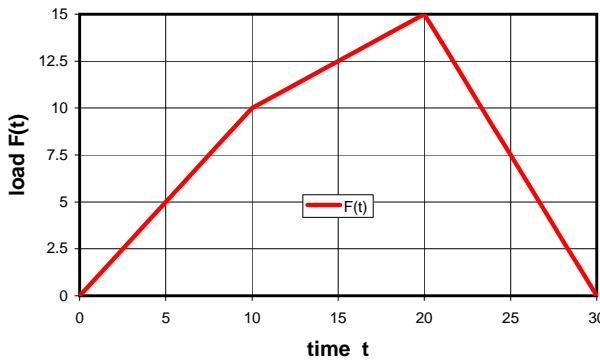
The title of the stresses in the legend are set to the first element type of **mate** in the Input-file. Thus a simple change is possible.

- The macro **sigq** calculates  $\mathbf{C} = \mathbf{A}^T \mathbf{K} \mathbf{A} - \mathbf{G}^T \mathbf{K}_a a^{-1} \mathbf{G}$  and **S** (on micro level). Using the INTEL-Compiler **C** will only be calculated, if OPEN-MP is off!

## 16.2 Some details using FEAP

### 16.2.1 Use of macro prop

The loading of a structure could be described using the macro **prop** in combination with the **time** command. Two possibilities exist, which are described in the following within an example.



Use for example a time step of 1 via `dt,,1`. The presented behaviour could be described with load type 1:  
 $\text{prop}(t) = a(1) + a(2) \cdot (t - t_{\min}) + a(3) \cdot (\sin[a(4) \cdot (t - t_{\min})]^k)$ .

Functions are valid in the range  $t_1 \leq t \leq t_2$ . In case of any problem at interval limits (e.g. no load or double load) a small 'imperfection' could be helpful, thus change e.g.  $t$  to  $t+e$  with e.g.  $e=1.e-5$ .

- Option 1: definition in advance

`prop,,3`

`1,0 0,10, 0, 1.0`

`1,0,e+10,20,10, 0.5`

`1,0,e+20,30,15,-1.0`

then

`step,,30` (`step` may be a user defined definition making one step)

- Option 2: re-definition at the current step of loading

`prop,,1`

`1,0 0,10, 0, 1.0`

`step,,10`

`prop,,1`

`1,0,e+10,20,10, 0.5`

`step,,10`

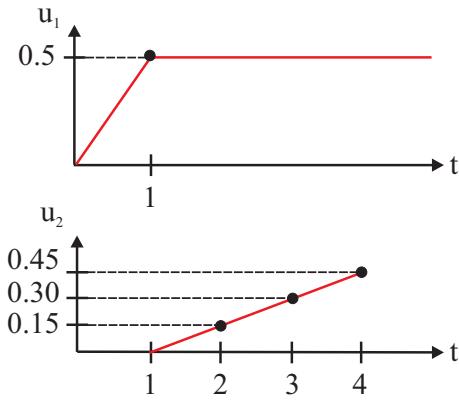
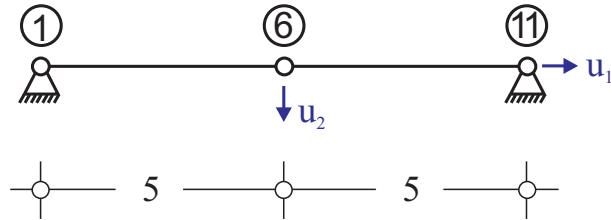
`prop,,1`

`1,0,e+20,30,15,-1.0`

`step,,10`

### 16.2.2 Use of macro newf

Problem: Structure with two prescribed displacements and different behavior over time:



Input-file:

```
...
disp
6,0,0.0,0.0,0.0
11,0,0.5,0.0,0.0
...
```

Macro-sequence:

```
dt,,1
prop,,1
```

**step** →  $u_1 = 0.5$  (**step** may be a user defined definition making one step)

**newf** → set  $F_0$

**mesh** → switch to mesh-modus and reset prescribed displacements

**disp**

```
6,0,0.0,0.15,0.0
11,0,0.0,0.00,0.0
```

**end** → switch to macro-modus

```
prop,,1
1,0,1,4,0,1
```

**step** →  $u_1 = 0.5 \quad u_2 = 0.15$

**step** →  $u_1 = 0.5 \quad u_2 = 0.30$

**step** →  $u_1 = 0.5 \quad u_2 = 0.45$