CALFEM - A Finite Element Toolbox

Release 0.1

...

May 25, 2025

CONTENTS:

Welcome to the documentation page for CALFEM for Python and CALFEM for MATLAB. On this page you will find examples of how to use CALFEM as well as reference documentation for the different modules in the CALFEM toolbox.

CHAPTER

ONE

INTRODUCTION

The computer program CALFEM is a MATLAB toolbox for finite element applications. This manual concerns mainly the finite element functions, but it also contains descriptions of some often-used MATLAB functions.

The finite element analysis can be carried out either interactively or in a batch-oriented fashion. In the interactive mode, the functions are evaluated one by one in the MATLAB command window. In the batch-oriented mode, a sequence of functions is written in a file named *.m* file and evaluated by writing the file name in the command window. The batch-oriented mode is a more flexible way of performing finite element analysis because the *.m* file can be written in an ordinary editor. This way of using CALFEM is recommended because it gives a structured organization of the functions. Changes and reruns are also easily executed in the batch-oriented mode.

A command line typically consists of functions for vector and matrix operations, calls to functions in the CALFEM finite element library, or commands for workspace operations. An example of a command line for a matrix operation is:

$$C = A + B'$$

where two matrices A and B' are added together, and the result is stored in matrix C. The matrix B' is the transpose of B.

An example of a call to the element library is:

$$Ke = spring1e(k)$$

where the two-by-two element stiffness matrix K^e is computed for a spring element with spring stiffness k, and is stored in the variable *Ke*. The input argument is given

within parentheses *( )* after the name of the function. Some functions have multiple input arguments and/or multiple output arguments. For example:

$$[lambda, X] = eigen(K, M)$$

computes the eigenvalues and eigenvectors of a pair of matrices K and M. The output variables - the eigenvalues stored in the vector λ and the corresponding eigenvectors stored in the matrix X - are surrounded by brackets *[ ]* and separated by commas. The input arguments are given inside the parentheses and also separated by commas.

The statement:

```
help function
```

provides information about the purpose and syntax for the specified function.

The available functions are organized in groups as follows. Each group is described in a separate chapter.

CHAPTER

TWO

GENERAL PURPOSE FUNCTIONS

CHAPTER

THREE

MATRIX FUNCTIONS

CHAPTER

FOUR

MATERIAL FUNCTIONS

The group of material functions comprises functions for constitutive models. The available models can treat linear elastic and isotropic hardening von Mises material.

4.1 hooke

Purpose

Compute material matrix for a linear elastic and isotropic material.

Syntax

```
D = hooke(ptype, E, v)
```

Description

The function `hooke` computes the material matrix $\mathbf{D}$ for a linear elastic and isotropic material.

The variable ptype is used to define the type of analysis:

$$\mathrm{ptype} = \begin{cases} 1 & \text{plane stress} \\ 2 & \text{plane strain} \\ 3 & \text{axisymmetry} \\ 4 & \text{three dimensional analysis} \end{cases}$$

The material parameters E and ν define the modulus of elasticity and the Poisson's ratio, respectively.

Theory

For plane stress (ptype $= 1$), $\mathbf{D}$ is formed as

$$\mathbf{D} = \frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{1-\nu}{2} \end{bmatrix}$$

For plane strain (ptype $= 2$) and axisymmetry (ptype $= 3$), $\mathbf{D}$ is formed as

$$\mathbf{D} = \frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix} 1-\nu & \nu & \nu & 0 \\ \nu & 1-\nu & \nu & 0 \\ \nu & \nu & 1-\nu & 0 \\ 0 & 0 & 0 & \frac{1}{2}(1-2\nu) \end{bmatrix}$$

For the three dimensional case (ptype $= 4$), $\mathbf{D}$ is formed as

$$\mathbf{D} = \frac{E}{(1+\nu)(1-2\nu)}\begin{bmatrix} 1-\nu & \nu & \nu & 0 & 0 & 0 \\ \nu & 1-\nu & \nu & 0 & 0 & 0 \\ \nu & \nu & 1-\nu & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2}(1-2\nu) & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{1}{2}(1-2\nu) & 0 \\ 0 & 0 & 0 & 0 & 0 & \frac{1}{2}(1- \end{bmatrix}$$

4.2 mises

Purpose

Compute stresses and plastic strains for an elasto-plastic isotropic hardening von Mises material.

Syntax

```
[es, deps, st] = mises(ptype, mp, est, st)
```

Description

The `mises` function computes updated stresses (`es`), plastic strain increments (`deps`), and state variables (`st`) for an elasto-plastic isotropic hardening von Mises material.

The input variable `ptype` defines the type of analysis, see also `hooke`. The vector `mp` contains the material constants:

$\mathbf{mp} = [\, E \;\; \nu \;\; h \,]$

where E is the modulus of elasticity, ν is the Poisson's ratio, and h is the plastic modulus.

The input matrix `est` contains trial stresses obtained by using the elastic material matrix `D` in `plants` or a similar `s`-function. The input vector `st` contains the state parameters:

$\mathbf{st} = [\, yi \;\sigma_y \;\varepsilon^p_{eff} \,]$

at the beginning of the step. The scalar yi indicates whether the material behaviour is elasto-plastic ($yi = 1$) or elastic ($yi = 0$). The current yield stress is denoted by σ_y and the effective plastic strain by ε^p_{eff}.

The output variables `es` and `st` contain updated values obtained by integration of the constitutive equations over the actual displacement step. The increments of the plastic strains are stored in the vector `deps`.

If `es` and `st` contain more than one row, then every row will be treated by the command.

Note

It is not necessary to check whether the material behaviour is elastic or elasto-plastic; this test is performed by the function. The computation is based on an Euler-Backward method, i.e., the radial return method.

Only the cases `ptype` $= 2, 3, 4$ are implemented.

4.3 dmises

Purpose

Form the elasto-plastic continuum matrix for an isotropic hardening von Mises material.

Syntax

```
D = dmises(ptype, mp, es, st)
```

Description

dmises forms the elasto-plastic continuum matrix for an isotropic hardening von Mises material.

The input variable ptype is used to define the type of analysis, cf. hooke.

The vector $\mathbf{mp}$ contains the material constants:

$$\mathbf{mp} = [\, E \;\nu \;h \,]$$

where E is the modulus of elasticity, ν is the Poisson's ratio, and h is the plastic modulus.

The matrix es contains current stresses obtained from plants or some similar s-function, and the vector st contains the current state parameters:

$$\mathbf{st} = [\, yi \;\sigma_y \;\varepsilon^p_{eff} \,]$$

where $yi = 1$ if the material behaviour is elasto-plastic, and $yi = 0$ if the material behaviour is elastic. The current yield stress is denoted by σ_y, and the current effective plastic strain by ε^p_{eff}.

Note

Only the case ptype $= 2$ is implemented.

CHAPTER

FIVE

ELEMENT FUNCTIONS

The group of element functions contains functions for computation of element matrices and element forces for different element types. The element functions have been divided into the following groups:

- Spring elements
- Bar elements
- Heat flow elements
- Solid elements
- Beam elements
- Plate elements

For each element type, there is a function for computation of the element stiffness matrix K^e. For most of the elements, an element load vector f^e can also be computed. These functions are identified by their last letter *-e*.

Using the function *assem()*, the element stiffness matrices and element load vectors are assembled into a global stiffness matrix K and a load vector f. Unknown nodal values of temperatures or displacements a are computed by solving the system of equations $Ka = f$ using the function *solveq()*. A vector of nodal values of temperatures or displacements for a specific element is formed by the function *extract()*.

When the element nodal values have been computed, the element flux or element stresses can be calculated using functions specific to the element type concerned. These functions are identified by their last letter *-s*.

For some elements, a function for computing the internal force vector is also available. These functions are identified by their last letter *-f*.

CHAPTER

SIX

SPRING ELEMENT FUNCTIONS

The spring element, shown below, can be used for the analysis of one-dimensional spring systems and for a variety of analogous physical problems.

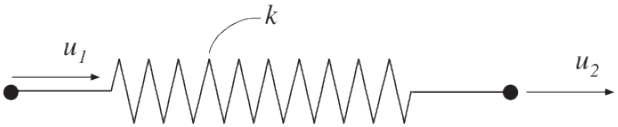


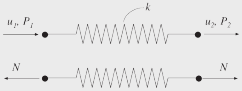
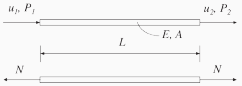
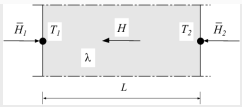
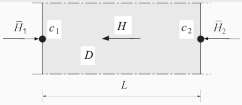
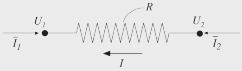
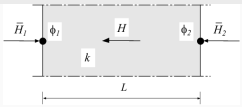
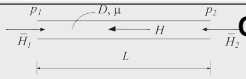
Fig. 1: Spring element

Quantities corresponding to the variables of the spring are listed in Table 1.

Table 1: Analogous quantities

Problem type	Spring stiffness	Nodal displacement	Element force	Spring force
Spring	k	u	P	N
Bar	$\frac{EA}{L}$	u	P	N
Thermal conduction	$\frac{\lambda A}{L}$	T	$\bar{H}$	H
Diffusion	$\frac{DA}{L}$	c	$\bar{H}$	H
Electrical circuit	$\frac{1}{R}$	U	$\bar{I}$	I
Groundwater flow	$\frac{kA}{L}$	ϕ	$\bar{H}$	H
Pipe network	$\frac{\pi D^4}{128\mu L}$	p	$\bar{H}$	H

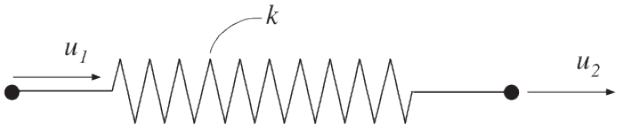
Table 2: Quantities used in different types of problems

Problem type	Quantities	Designations	Description
Spring		k, u, P, N	spring stiffness, displacement, element force, spring force
Bar		L, E, A, u, P, N	length, modulus of elasticity, area of cross section, displacement, element force, normal force
Thermal conduction		L, λ, T, $\bar{H}$, H	length, thermal conductivity, temperature, element heat flow, internal heat flow
Diffusion		L, D, c, $\bar{H}$, H	length, diffusivity, nodal concentration, nodal mass flow, element mass flow
Electrical circuit		R, U, $\bar{I}$, I	resistance, potential, element current, internal current
Groundwater flow		L, k, ϕ, $\bar{H}$, H	length, permeability, piezometric head, element water flow, internal water flow
Pipe network (laminar		L, D,	length, pipe diameter, viscosity, pressure, element fluid

6.1 spring1e

Purpose

Compute element stiffness matrix for a spring element.



Syntax

```
Ke = spring1e(ep)
```

Description

`spring1e` provides the element stiffness matrix `Ke` for a spring element.

The input variable

`ep` $= [\,k\,]$

supplies the spring stiffness k or the analog quantity defined in Table *Analogous quantities*.

Theory

The element stiffness matrix $\mathbf{K}^e$, stored in `Ke`, is computed according to

$$\mathbf{K}^e = \begin{bmatrix} k & -k \\ -k & k \end{bmatrix}$$

where k is defined by `ep`.

6.2 spring1s

Purpose

Compute spring force in a spring element.

Syntax

```
es = spring1s(ep, ed)
```

Description

spring1s computes the spring force es in a spring element.

The input variable ep is defined in spring1e and the element nodal displacements ed are obtained by the function extract.

The output variable

$$\mathrm{es} = [\, N \,]$$

contains the spring force N, or the analog quantity.

Theory

The spring force N, or analog quantity, is computed according to

$$N = k\,(u_2 - u_1)$$

CHAPTER

SEVEN

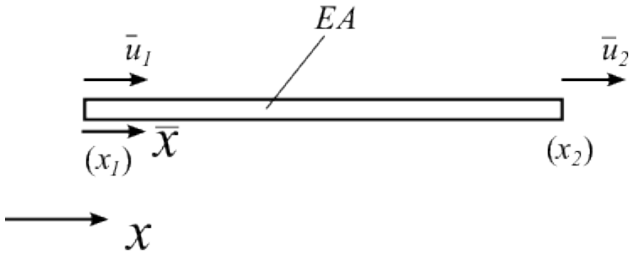
BAR ELEMENT FUNCTIONS

7.1 1D Bar elements

7.1.1 bar1e

Purpose

Compute element stiffness matrix for a one dimensional bar element.



Syntax

```
Ke = bar1e(ex, ep)
[Ke, fe] = bar1e(ex, ep, eq)
```

Description

`bar1e` provides the element stiffness matrix `Ke` for a one dimensional bar element. The input variables

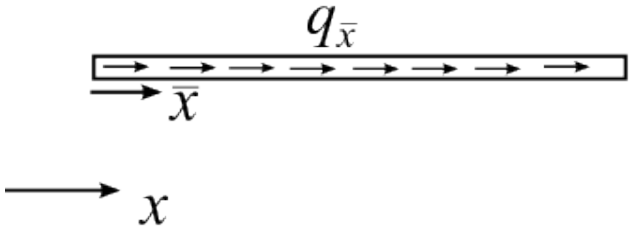
$$\text{ex} = [x_1 \;\; x_2] \qquad \text{ep} = [E \;\; A]$$

supply the element nodal coordinates x_1 and x_2, the modulus of elasticity E, and the cross section area A.

The element load vector `fe` can also be computed if a uniformly distributed load is applied to the element. The optional input variable

$$\text{eq} = [q_{\bar{x}}]$$

contains the distributed load per unit length, $q_{\bar{x}}$.



Theory

The element stiffness matrix $\bar{\mathbf{K}}^e$, stored in `Ke`, is computed according to

$$\bar{\mathbf{K}}^e = \frac{D_{EA}}{L}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

where the axial stiffness D_{EA} and the length L are given by

$$D_{EA} = EA; \quad L = x_2 - x_1$$

The element load vector $\bar{\mathbf{f}}_l^e$, stored in `fe`, is computed according to

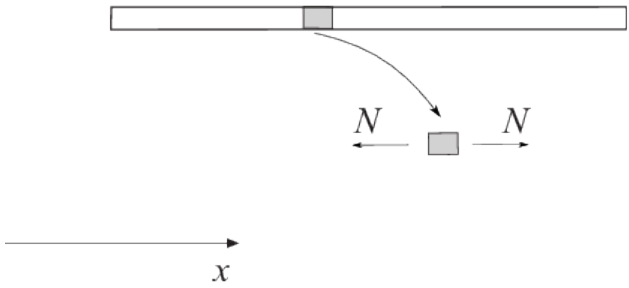
$$\bar{\mathbf{f}}_l^e = \frac{q_{\bar{x}} L}{2}\begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

7.1.2 bar1s

Purpose

Compute normal force in a one dimensional bar element.

Syntax




```
es = bar1s(ex, ep, ed)
es = bar1s(ex, ep, ed, eq)
[es, edi] = bar1s(ex, ep, ed, eq, n)
[es, edi, eci] = bar1s(ex, ep, ed, eq, n)
```

Description

`bar1s` computes the normal force in the one dimensional bar element `bar1e`.

The input variables `ex` and `ep` are defined in `bar1e` and the element nodal displacements, stored in `ed`, are obtained by the function `extract`. If distributed load is applied to the element, the variable `eq` must be included.

The number of evaluation points for normal force and displacement are determined by `n`. If `n` is omitted, only the ends of the bar are evaluated.

The output variables

$$\text{es} = \begin{bmatrix} N(0) \\ N(\bar{x}_2) \\ \vdots \\ N(\bar{x}_{n-1}) \\ N(L) \end{bmatrix} \qquad \text{edi} = \begin{bmatrix} u(0) \\ u(\bar{x}_2) \\ \vdots \\ u(\bar{x}_{n-1}) \\ u(L) \end{bmatrix} \qquad \text{eci} = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

contain the normal force, the displacement, and the evaluation points on the local $\bar{x}$-axis. L is the length of the bar element.

Theory

The nodal displacements in local coordinates are given by

$$\bar{\mathbf{a}}^e = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \end{bmatrix}$$

The transpose of $\bar{\mathbf{a}}^e$ is stored in `ed`.

The displacement $u(\bar{x})$ and the normal force $N(\bar{x})$ are computed from

$$u(\bar{x}) = \mathbf{N}\bar{\mathbf{a}}^e + u_p(\bar{x})$$

$$N(\bar{x}) = D_{EA}\mathbf{B}\bar{\mathbf{a}}^e + N_p(\bar{x})$$

where

$$\mathbf{N} = \begin{bmatrix} 1 & \bar{x} \end{bmatrix} \mathbf{C}^{-1} = \begin{bmatrix} 1 - \frac{\bar{x}}{L} & \frac{\bar{x}}{L} \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 1 \end{bmatrix} \mathbf{C}^{-1} = \frac{1}{L} \begin{bmatrix} -1 & 1 \end{bmatrix}$$

$$u_p(\bar{x}) = -\frac{q_{\bar{x}}}{D_{EA}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} \right)$$

$$N_p(\bar{x}) = -q_{\bar{x}} \left(\bar{x} - \frac{L}{2} \right)$$

in which D_{EA}, L, and $q_{\bar{x}}$ are defined in `bar1e` and

$$\mathbf{C}^{-1} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix}$$

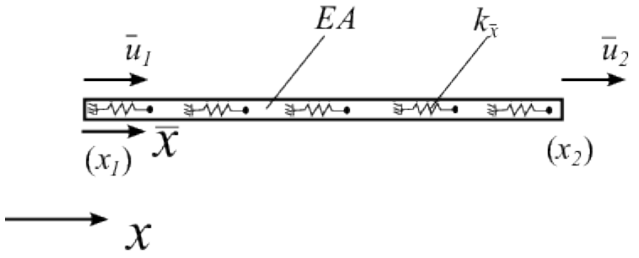
7.1.3 bar1we

Purpose

Compute element stiffness matrix for a one dimensional bar element with elastic support.

Syntax

```
Ke = bar1we(ex, ep)
[Ke, fe] = bar1we(ex, ep, eq)
```



Description

`bar1we` provides the element stiffness matrix `Ke` for a one dimensional bar element with elastic support.

The input variables

$$\text{ex} = [x_1 \;\; x_2] \qquad \text{ep} = [E \; A \; k_{\bar{x}}]$$

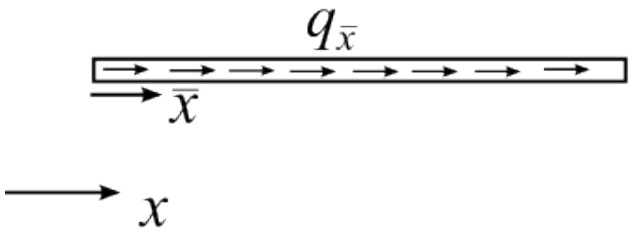
supply the element nodal coordinates x_1 and x_2, the modulus of elasticity E, the cross section area A and the stiffness of the axial springs $k_{\bar{x}}$.

The element load vector `fe` can also be computed if a uniformly distributed load is applied to the element.

The optional input variable

$$\text{eq} = [q_{\bar{x}}]$$

then contains the distributed load per unit length, $q_{\bar{x}}$.



Bar element with distributed load

Theory

The element stiffness matrix $\bar{\mathbf{K}}^e$, stored in `Ke`, is computed accord-

ing to

$$\bar{\mathbf{K}}^e = \bar{\mathbf{K}}_0^e + \bar{\mathbf{K}}_s^e$$

where

$$\bar{\mathbf{K}}_0^e = \frac{D_{EA}}{L}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

$$\bar{\mathbf{K}}_s^e = k_{\bar{x}} L \begin{bmatrix} \frac{1}{3} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{3} \end{bmatrix}$$

where the axial stiffness D_{EA} and the length L are given by

$$D_{EA} = EA; \qquad L = x_2 - x_1$$

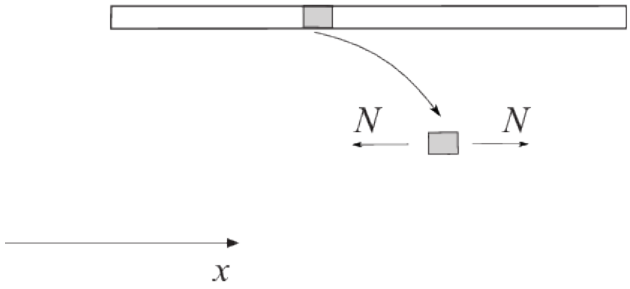
The element load vector $\bar{\mathbf{f}}_l^e$, stored in `fe`, is computed according to

$$\bar{\mathbf{f}}_l^e = \frac{q_{\bar{x}} L}{2}\begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

7.1.4 bar1ws

Purpose

Compute normal force in a one dimensional bar element with elastic support.



Syntax

```
es = bar1ws(ex, ep, ed)
es = bar1ws(ex, ep, ed, eq)
[es, edi] = bar1ws(ex, ep, ed, eq, n)
[es, edi, eci] = bar1ws(ex, ep, ed, eq, n)
```

Description

`bar1ws` computes the normal force in the one dimensional bar element `bar1we`.

The input variables `ex` and `ep` are defined in `bar1we` and the element nodal displacements, stored in `ed`, are obtained by the function `extract`. If distributed load is applied to the element, the variable `eq` must be included.

The number of evaluation points for normal force and displacement are determined by `n`. If `n` is omitted, only the ends of the bar are evaluated.

The output variables are:

$$\text{es} = \begin{bmatrix} N(0) \\ N(\bar{x}_2) \\ \vdots \\ N(\bar{x}_{n-1}) \\ N(L) \end{bmatrix} \qquad \text{edi} = \begin{bmatrix} u(0) \\ u(\bar{x}_2) \\ \vdots \\ u(\bar{x}_{n-1}) \\ u(L) \end{bmatrix} \qquad \text{eci} = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

These contain the normal force, the displacement, and the evaluation points on the local $\bar{x}$-axis. L is the length of the bar element.

Theory

The nodal displacements in local coordinates are given by

$$\bar{\mathbf{a}}^e = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \end{bmatrix}$$

The transpose of $\bar{\mathbf{a}}^e$ is stored in `ed`.

The displacement $u(\bar{x})$ and the normal force $N(\bar{x})$ are computed from

$$u(\bar{x}) = \mathbf{N}\bar{\mathbf{a}}^e + u_p(\bar{x})$$

$$N(\bar{x}) = D_{EA}\mathbf{B}\bar{\mathbf{a}}^e + N_p(\bar{x})$$

where

$$\mathbf{N} = \begin{bmatrix} 1 & \bar{x} \end{bmatrix} \mathbf{C}^{-1} = \begin{bmatrix} 1 - \frac{\bar{x}}{L} & \frac{\bar{x}}{L} \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 1 \end{bmatrix} \mathbf{C}^{-1} = \frac{1}{L} \begin{bmatrix} -1 & 1 \end{bmatrix}$$

$$u_p(\bar{x}) = \frac{k_{\bar{x}}}{D_{EA}} \begin{bmatrix} \frac{\bar{x}^2 - L\bar{x}}{2} & \frac{\bar{x}^3 - L^2\bar{x}}{6} \end{bmatrix} \mathbf{C}^{-1}\bar{\mathbf{a}}^e - \frac{q_{\bar{x}}}{D_{EA}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} \right)$$

$$N_p(\bar{x}) = k_{\bar{x}} \begin{bmatrix} \frac{2\bar{x} - L}{2} & \frac{3\bar{x}^2 - L^2}{6} \end{bmatrix} \mathbf{C}^{-1}\bar{\mathbf{a}}^e - q_{\bar{x}} \left(\bar{x} - \frac{L}{2} \right)$$

in which D_{EA}, L, $k_{\bar{x}}$ and $q_{\bar{x}}$ are defined in `bar1we` and

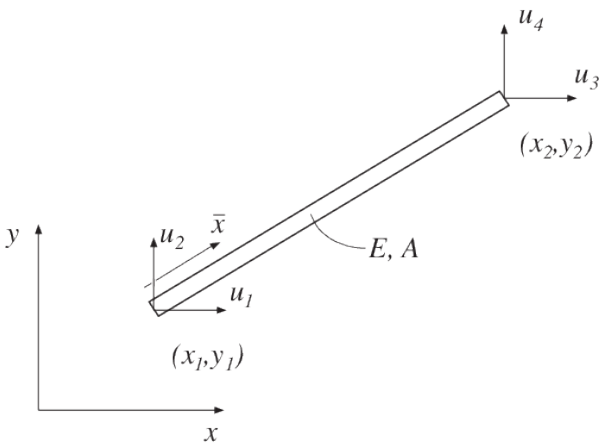
$$\mathbf{C}^{-1} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix}$$

7.2 2D bar elements

7.2.1 bar2e

Purpose

Compute element stiffness matrix for a two dimensional bar element.



Syntax

```
Ke = bar2e(ex, ey, ep)
[Ke, fe] = bar2e(ex, ey, ep, eq)
```

Description

`bar2e` provides the global element stiffness matrix `Ke` for a two dimensional bar element.

The input variables

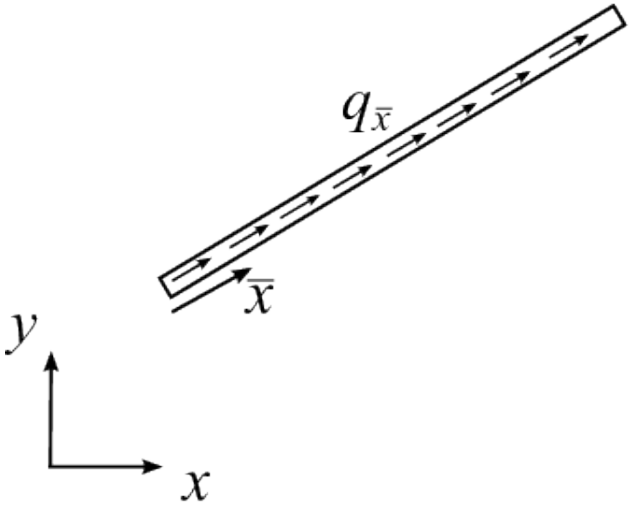
$$\begin{matrix} \text{ex} = [\, x_1 \;\; x_2 \,] \\ \text{ey} = [\, y_1 \;\; y_2 \,] \end{matrix} \qquad \text{ep} = [\, E \;\; A \,]$$

supply the element nodal coordinates x_1, y_1, x_2, and y_2, the modulus of elasticity E, and the cross section area A.

The element load vector `fe` can also be computed if a uniformly distributed axial load is applied to the element. The optional input variable

$$\text{eq} = [\, q_{\bar{x}} \,]$$

then contains the distributed load per unit length, $q_{\bar{x}}$.



Theory

The element stiffness matrix $\mathbf{K}^e$, stored in `Ke`, is computed according to

$$\mathbf{K}^e = \mathbf{G}^T \, \bar{\mathbf{K}}^e \, \mathbf{G}$$

where

$$\bar{\mathbf{K}}^e = \frac{D_{EA}}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \qquad \mathbf{G} = \begin{bmatrix} n_{x\bar{x}} & n_{y\bar{x}} & 0 & 0 \\ 0 & 0 & n_{x\bar{x}} & n_{y\bar{x}} \end{bmatrix}$$

where the axial stiffness D_{EA} and the length L are given by

$$D_{EA} = EA; \qquad L = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

and the transformation matrix $\mathbf{G}$ contains the direction cosines

$$n_{x\bar{x}} = \frac{x_2 - x_1}{L} \qquad n_{y\bar{x}} = \frac{y_2 - y_1}{L}$$

The element load vector $\mathbf{f}_l^e$, stored in `fe`, is computed according to

$$\mathbf{f}_l^e = \mathbf{G}^T \, \bar{\mathbf{f}}_l^e$$

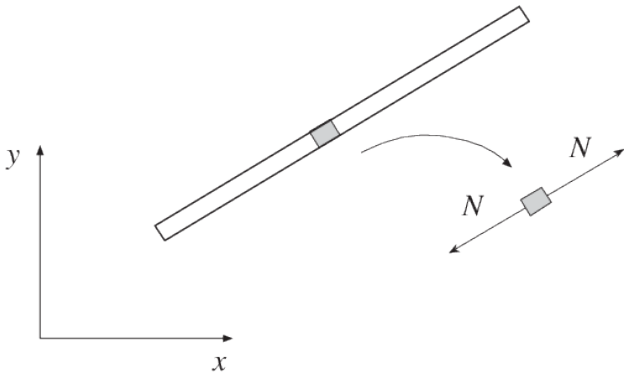
where

$$\bar{\mathbf{f}}_l^e = \frac{q_{\bar{x}} L}{2} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

7.2.2 bar2s

Purpose

Compute normal force in a two dimensional bar element.



Syntax

```
es = bar2s(ex, ey, ep, ed)
es = bar2s(ex, ey, ep, ed, eq)
[es, edi] = bar2s(ex, ey, ep, ed, eq, n)
[es, edi, eci] = bar2s(ex, ey, ep, ed, eq, n)
```

Description

`bar2s` computes the normal force in the two dimensional bar element `bar2e`.

The input variables `ex`, `ey`, and `ep` are defined in `bar2e` and the element nodal displacements, stored in `ed`, are obtained by the function `extract`. If distributed loads are applied to the element, the variable `eq` must be included. The number of evaluation points for section forces and displacements are determined by `n`. If `n` is omitted, only the ends of the bar are evaluated.

The output variables

$$\text{es} = \begin{bmatrix} N(0) \\ N(\bar{x}_2) \\ \vdots \\ N(\bar{x}_{n-1}) \\ N(L) \end{bmatrix} \qquad \text{edi} = \begin{bmatrix} u(0) \\ u(\bar{x}_2) \\ \vdots \\ u(\bar{x}_{n-1}) \\ u(L) \end{bmatrix} \qquad \text{eci} = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

contain the normal force, the displacement, and the evaluation points on the local $\bar{x}$-axis. L is the length of the bar element.

Theory

The nodal displacements in global coordinates

$$\mathbf{a}^e = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 \end{bmatrix}^T$$

are also shown in `bar2e`. The transpose of $\mathbf{a}^e$ is stored in `ed`.

The nodal displacements in local coordinates are given by

$$\bar{\mathbf{a}}^e = \mathbf{G}\mathbf{a}^e$$

where the transformation matrix $\mathbf{G}$ is defined in `bar2e`.

The displacement $u(\bar{x})$ and the normal force $N(\bar{x})$ are computed from

$$u(\bar{x}) = \mathbf{N}\bar{\mathbf{a}}^e + u_p(\bar{x})$$

$$N(\bar{x}) = D_{EA}\mathbf{B}\bar{\mathbf{a}}^e + N_p(\bar{x})$$

where

$$\mathbf{N} = \begin{bmatrix} 1 & \bar{x} \end{bmatrix} \mathbf{C}^{-1} = \begin{bmatrix} 1 - \frac{\bar{x}}{L} & \frac{\bar{x}}{L} \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 1 \end{bmatrix} \mathbf{C}^{-1} = \frac{1}{L} \begin{bmatrix} -1 & 1 \end{bmatrix}$$

$$u_p(\bar{x}) = -\frac{q_{\bar{x}}}{D_{EA}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} \right)$$

$$N_p(\bar{x}) = -q_{\bar{x}} \left(\bar{x} - \frac{L}{2} \right)$$

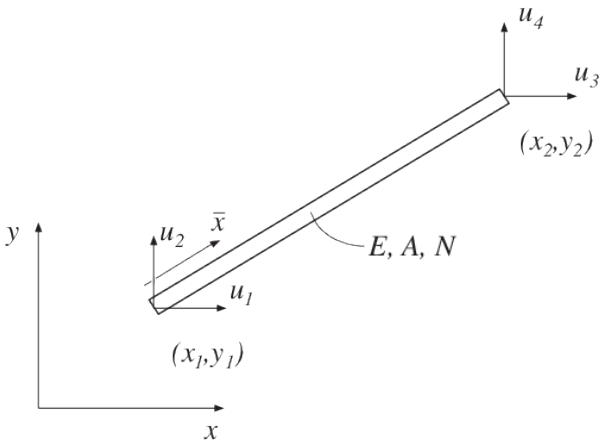
where D_{EA}, L, $q_{\bar{x}}$ are defined in `bar2e` and

$$\mathbf{C}^{-1} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix}$$

7.2.3 bar2ge

Purpose

Compute element stiffness matrix for a two dimensional geometric nonlinear bar.



Syntax

```
Ke = bar2ge(ex, ey, ep, Qx)
```

Description

`bar2ge` provides the element stiffness matrix `Ke` for a two dimensional geometric nonlinear bar element.

The input variables:

```
ex = [x1, x2]
ey = [y1, y2]
ep = [E, A]
```

supply the element nodal coordinates x_1, y_1, x_2, and y_2, the modulus of elasticity E, and the cross section area A.

The input variable:

```
Qx = [Q_{\bar{x}}]
```

contains the value of the axial force, which is positive in tension.

Theory

The global element stiffness matrix $\mathbf{K}^e$, stored in `Ke`, is computed according to

$$\mathbf{K}^e = \mathbf{G}^T \bar{\mathbf{K}}^e \mathbf{G}$$

where

$$\bar{\mathbf{K}}^e = \frac{D_{EA}}{L} \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} + \frac{Q_{\bar{x}}}{L} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} n_{x\bar{x}} & n_{y\bar{x}} & 0 & 0 \\ n_{x\bar{y}} & n_{y\bar{y}} & 0 & 0 \\ 0 & 0 & n_{x\bar{x}} & n_{y\bar{x}} \\ 0 & 0 & n_{x\bar{y}} & n_{y\bar{y}} \end{bmatrix}$$

where the axial stiffness D_{EA} and the length L are given by

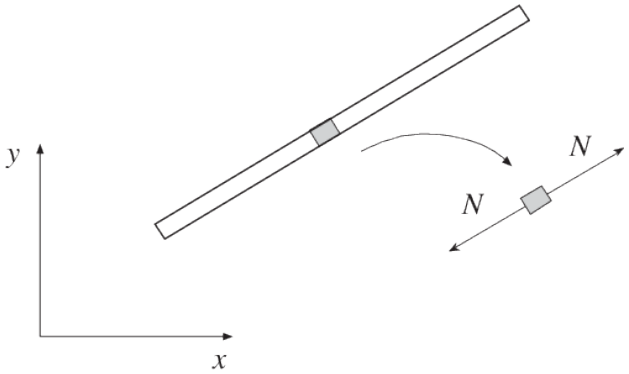
$$D_{EA} = EA \qquad L = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

and the transformation matrix $\mathbf{G}$ contains the direction cosines

$$n_{x\bar{x}} = n_{y\bar{y}} = \frac{x_2 - x_1}{L} \qquad n_{y\bar{x}} = -n_{x\bar{y}} = \frac{y_2 - y_1}{L}$$

7.2.4 bar2gs

Purpose



Compute axial force and normal force in a two dimensional bar element.

Syntax

```
[es, Qx] = bar2gs(ex, ey, ep, ed)
[es, Qx] = bar2gs(ex, ey, ep, ed, eq)
[es, Qx, edi] = bar2gs(ex, ey, ep, ed, eq, n)
[es, Qx, edi, eci] = bar2gs(ex, ey, ep, ed, eq, n)
```

Description

`bar2gs` computes the normal force in the two dimensional bar elements `bar2g`.

The input variables `ex`, `ey`, and `ep` are defined in `bar2ge` and the element nodal displacements, stored in `ed`, are obtained by the function `extract`. The number of evaluation points for section forces and displacements are determined by `n`. If `n` is omitted, only the ends of the bar are evaluated.

The output variable `Qx` contains the axial force $Q_{\bar{x}}$ and the output variables

$$\text{es} = \begin{bmatrix} N(0) \\ N(\bar{x}_2) \\ \vdots \\ N(\bar{x}_{n-1}) \\ N(L) \end{bmatrix} \qquad \text{edi} = \begin{bmatrix} u(0) \\ u(\bar{x}_2) \\ \vdots \\ u(\bar{x}_{n-1}) \\ u(L) \end{bmatrix} \qquad \text{eci} = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

contain the normal force, the displacement, and the evaluation points on the local $\bar{x}$-axis. L is the length of the bar element.

Theory

The nodal displacements in global coordinates are given by

$$\mathbf{a}^e = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 \end{bmatrix}^T$$

The transpose of $\mathbf{a}^e$ is stored in `ed`. The nodal displacements in local coordinates are given by

$$\bar{\mathbf{a}}^e = \mathbf{G}\mathbf{a}^e$$

where the transformation matrix $\mathbf{G}$ is defined in `bar2ge`. The displacements associated with bar action are determined as

$$\bar{\mathbf{a}}^e_{\text{bar}} = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_3 \end{bmatrix}$$

The displacement $u(\bar{x})$ and the normal force $N(\bar{x})$ are computed from

$$u(\bar{x}) = \mathbf{N}\bar{\mathbf{a}}^e_{\text{bar}}$$

$$N(\bar{x}) = D_{EA}\mathbf{B}\bar{\mathbf{a}}^e_{\text{bar}}$$

where

$$\mathbf{N} = \begin{bmatrix} 1 & \bar{x} \end{bmatrix}\mathbf{C}^{-1} = \begin{bmatrix} 1 - \frac{\bar{x}}{L} & \frac{\bar{x}}{L} \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 1 \end{bmatrix}\mathbf{C}^{-1} = \frac{1}{L}\begin{bmatrix} -1 & 1 \end{bmatrix}$$

where D_{EA} and L are defined in `bar2ge` and

$$\mathbf{C}^{-1} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix}$$

An updated value of the axial force is computed as

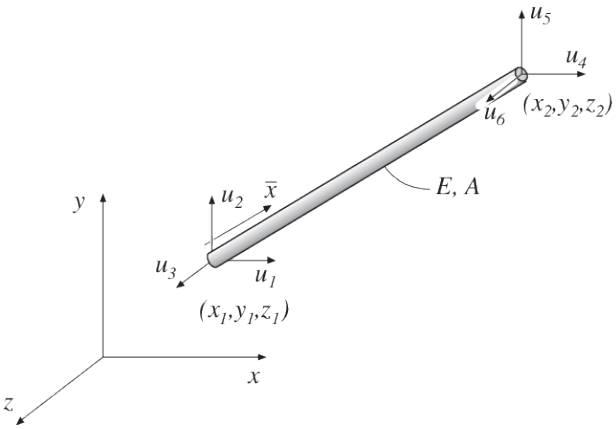
$$Q_{\bar{x}} = N(0)$$

7.3 3D bar elements

7.3.1 bar3e

Purpose

Compute element stiffness matrix for a three dimensional bar element.



Syntax

```
Ke = bar3e(ex, ey, ez, ep)
[Ke, fe] = bar3e(ex, ey, ez, ep, eq)
```

Description

`bar3e` provides the global element stiffness matrix `Ke` for a three dimensional bar element.

The input variables

$$\begin{aligned} \text{ex} &= [x_1 \;\; x_2] \\ \text{ey} &= [y_1 \;\; y_2] \\ \text{ez} &= [z_1 \;\; z_2] \end{aligned} \qquad \text{ep} = [E \;\; A]$$

supply the element nodal coordinates x_1, y_1, z_1, x_2, y_2, and z_2, the modulus of elasticity E, and the cross section area A.

The element load vector `fe` can also be computed if a uniformly distributed axial load is applied to the element. The optional input

variable

$$\text{eq} = [q_{\bar{x}}]$$

contains the distributed load per unit length, $q_{\bar{x}}$.

Theory

The element stiffness matrix $\mathbf{K}^e$, stored in `Ke`, is computed according to

$$\mathbf{K}^e = \mathbf{G}^T \, \bar{\mathbf{K}}^e \, \mathbf{G}$$

where

$$\bar{\mathbf{K}}^e = \frac{D_{EA}}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \qquad \mathbf{G} = \begin{bmatrix} n_{x\bar{x}} & n_{y\bar{x}} & n_{z\bar{x}} & 0 & 0 & 0 \\ 0 & 0 & 0 & n_{x\bar{x}} & n_{y\bar{x}} & n_{z\bar{x}} \end{bmatrix}$$

where the axial stiffness D_{EA} and the length L are given by

$$D_{EA} = EA \qquad L = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

and the transformation matrix $\mathbf{G}$ contains the direction cosines

$$n_{x\bar{x}} = \frac{x_2 - x_1}{L} \qquad n_{y\bar{x}} = \frac{y_2 - y_1}{L} \qquad n_{z\bar{x}} = \frac{z_2 - z_1}{L}$$

The element load vector $\mathbf{f}_l^e$, stored in `fe`, is computed according to

$$\mathbf{f}_l^e = \mathbf{G}^T \, \bar{\mathbf{f}}_l^e$$

where

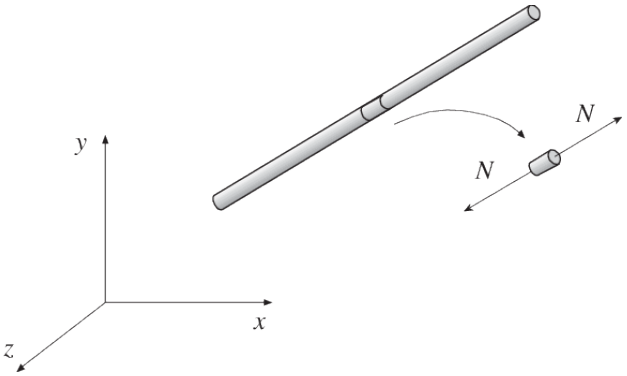
$$\bar{\mathbf{f}}_l^e = \frac{q_{\bar{x}} L}{2} \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

7.3.2 bar3s

Purpose

Compute normal force in a three dimensional bar element.

Syntax




```
es = bar3s(ex, ey, ez, ep, ed)
es = bar3s(ex, ey, ez, ep, ed, eq)
[es, edi] = bar3s(ex, ey, ez, ep, ed, eq, n)
[es, edi, eci] = bar3s(ex, ey, ez, ep, ed, eq, n)
```

Description

`bar3s` computes the normal force in a three dimensional bar element (see `bar3e`).

The input variables `ex`, `ey`, and `ep` are defined in `bar3e` and the element nodal displacements, stored in `ed`, are obtained by the function `extract`. The number of evaluation points for section forces and displacements are determined by `n`. If `n` is omitted, only the ends of the bar are evaluated.

The output variables:

$$\text{es} = \begin{bmatrix} N(0) \\ N(\bar{x}_2) \\ \vdots \\ N(\bar{x}_{n-1}) \\ N(L) \end{bmatrix} \qquad \text{edi} = \begin{bmatrix} u(0) \\ u(\bar{x}_2) \\ \vdots \\ u(\bar{x}_{n-1}) \\ u(L) \end{bmatrix} \qquad \text{eci} = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

contain the normal force, the displacement, and the evaluation points on the local $\bar{x}$-axis. L is the length of the bar element.

Theory

The nodal displacements in global coordinates are given by

$$\mathbf{a}^e = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \end{bmatrix}^T$$

The transpose of $\mathbf{a}^e$ is stored in `ed`.

The nodal displacements in local coordinates are given by

$$\bar{\mathbf{a}}^e = \mathbf{G}\mathbf{a}^e$$

where the transformation matrix $\mathbf{G}$ is defined in `bar3e`.

The displacement $u(\bar{x})$ and the normal force $N(\bar{x})$ are computed from

$$u(\bar{x}) = \mathbf{N}\bar{\mathbf{a}}^e + u_p(\bar{x})$$

$$N(\bar{x}) = D_{EA}\mathbf{B}\bar{\mathbf{a}}^e + N_p(\bar{x})$$

where

$$\mathbf{N} = \begin{bmatrix} 1 & \bar{x} \end{bmatrix} \mathbf{C}^{-1} = \begin{bmatrix} 1 - \frac{\bar{x}}{L} & \frac{\bar{x}}{L} \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 1 \end{bmatrix} \mathbf{C}^{-1} = \frac{1}{L} \begin{bmatrix} -1 & 1 \end{bmatrix}$$

$$u_p(\bar{x}) = -\frac{q_{\bar{x}}}{D_{EA}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} \right)$$

$$N_p(\bar{x}) = -q_{\bar{x}} \left(\bar{x} - \frac{L}{2} \right)$$

where D_{EA}, L, $q_{\bar{x}}$ are defined in `bar3e` and

$$\mathbf{C}^{-1} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix}$$

CHAPTER

EIGHT

HEAT FLOW ELEMENTS

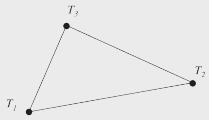
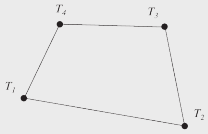
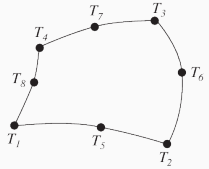
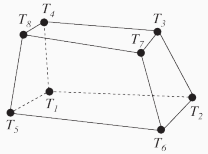
Heat flow elements are available for one, two, and three dimensional analysis. For one dimensional heat flow the spring element `spring1` is used.

A variety of important physical phenomena are described by the same differential equation as the heat flow problem. The heat flow element is thus applicable in modelling different physical applications. Table below shows the relation between the primary variable **a**, the constitutive matrix **D**, and the load vector **f_l** for a chosen set of two dimensional physical problems.

Table 1: Problem dependent parameters

Problem type	**a**	**D**	**f_l**	Designation
Heat flow	T	λ_x, λ_y	Q	T **= temperature** λ_x, λ_y = thermal conductivity Q = heat supply
Groundwater flow	ϕ	k_x, k_y	Q	ϕ **= piezometric head** k_x, k_y = permeabilities Q = fluid supply
St. Venant torsion	ϕ	$1/G_{zy}$, $1/G_{zx}$	2Θ	ϕ **= stress function** G_{zy}, G_{zx} = shear moduli Θ = angle of torsion per unit length

8.1 Element types

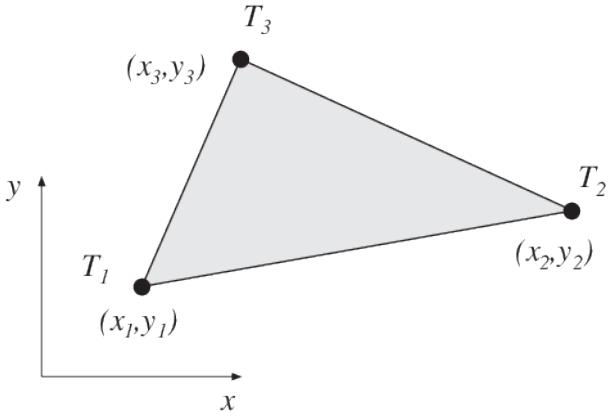
	
`flw2te`	`flw2qe`, `flw2i4e`
	
`flw2i8e`	`flw3i8e`

8.2 2D Heat Flow Functions

8.2.1 flw2te

Purpose

Compute element stiffness matrix for a triangular heat flow element.



Syntax

```
Ke = flw2te(ex, ey, ep, D)
[Ke, fe] = flw2te(ex, ey, ep, D, eq)
```

Description

`flw2te` provides the element stiffness (conductivity) matrix `Ke` and the element load vector `fe` for a triangular heat flow element.

The element nodal coordinates x_1, y_1, x_2 etc, are supplied to the function by `ex` and `ey`, the element thickness t is supplied by `ep` and the thermal conductivities (or corresponding quantities) k_{xx}, k_{xy} etc are supplied by `D`.

$$\mathbf{ex} = [\, x_1 \;\; x_2 \;\; x_3 \,] \qquad \mathbf{ep} = [\, t \,] \qquad \mathbf{D} = \begin{bmatrix} k_{xx} & k_{xy} \\ k_{yx} & k_{yy} \end{bmatrix}$$
$$\mathbf{ey} = [\, y_1 \;\; y_2 \;\; y_3 \,]$$

If the scalar variable `eq` is given in the function, the element load vector **fe** is computed, using

$$\mathbf{eq} = [\, Q \,]$$

where Q is the heat supply per unit volume.

Theory

The element stiffness matrix $\mathbf{K}^e$ and the element load vector $\mathbf{f}_l^e$, stored in `Ke` and `fe`, respectively, are computed according to

$$\mathbf{K}^e = (\mathbf{C}^{-1})^T \int_A \bar{\mathbf{B}}^T \mathbf{D} \bar{\mathbf{B}} \, t \, dA \, \mathbf{C}^{-1}$$

$$\mathbf{f}_l^e = (\mathbf{C}^{-1})^T \int_A \bar{\mathbf{N}}^T Q \, t \, dA$$

with the constitutive matrix $\mathbf{D}$ defined by `D`.

The evaluation of the integrals for the triangular element is based on the linear temperature approximation $T(x, y)$ and is expressed in terms of the nodal variables T_1, T_2 and T_3 as

$$T(x, y) = \mathbf{N}^e \mathbf{a}^e = \bar{\mathbf{N}} \, \mathbf{C}^{-1} \mathbf{a}^e$$

where

$$\bar{\mathbf{N}} = [\, 1 \;\; x \;\; y \,] \qquad \mathbf{C} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} \qquad \mathbf{a}^e = \begin{bmatrix} T_1 \\ T_2 \\ T_3 \end{bmatrix}$$

and hence it follows that

$$\bar{\mathbf{B}} = \nabla\bar{\mathbf{N}} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \nabla = \begin{bmatrix} \dfrac{\partial}{\partial x} \\ \dfrac{\partial}{\partial y} \end{bmatrix}$$

Evaluation of the integrals for the triangular element yields

$$\mathbf{K}^e = (\mathbf{C}^{-1})^T \bar{\mathbf{B}}^T \mathbf{D} \bar{\mathbf{B}} \mathbf{C}^{-1} t A$$

$$\mathbf{f}_l^e = \frac{QAt}{3} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

where the element area A is determined as

$$A = \frac{1}{2} \det \mathbf{C}$$

8.2.2 flw2ts

Purpose

Compute heat flux and temperature gradients in a triangular heat flow element.

Syntax

```
[es, et] = flw2ts(ex, ey, D, ed)
```

Description

`flw2ts` computes the heat flux vector `es` and the temperature gradient `et` (or corresponding quantities) in a triangular heat flow element.

The input variables `ex`, `ey` and the matrix `D` are defined in `flw2te`. The vector `ed` contains the nodal temperatures $\mathbf{a}^e$ of the element and is obtained by the function `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\ T_1 \ \ T_2 \ \ T_3\ ]$$

The output variables

$$\mathbf{es} = \mathbf{q}^T = [\ q_x \ q_y\ ]$$

$$\mathbf{et} = (\nabla T)^T = \begin{bmatrix} \frac{\partial T}{\partial x} & \frac{\partial T}{\partial y} \end{bmatrix}$$

contain the components of the heat flux and the temperature gradient computed in the directions of the coordinate axis.

Theory

The temperature gradient and the heat flux are computed according to

$$\nabla T = \bar{\mathbf{B}}\ \mathbf{C}^{-1}\ \mathbf{a}^e$$

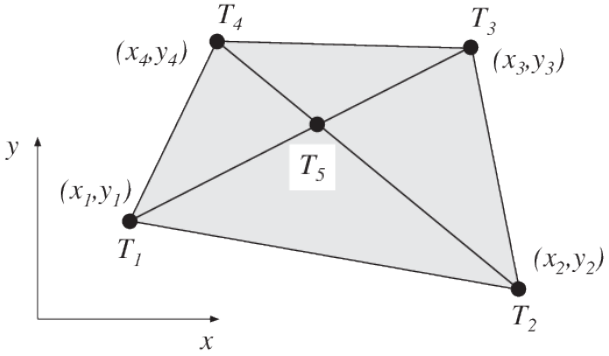
$$\mathbf{q} = -\mathbf{D}\nabla T$$

where the matrices $\mathbf{D}$, $\bar{\mathbf{B}}$, and $\mathbf{C}$ are described in `flw2te`. Note that both the temperature gradient and the heat flux are constant in the element.

8.2.3 flw2qe

Purpose

Compute element stiffness matrix for a quadrilateral heat flow element.



Syntax

```
Ke = flw2qe(ex, ey, ep, D)
[Ke, fe] = flw2qe(ex, ey, ep, D, eq)
```

Description

`flw2qe` provides the element stiffness (conductivity) matrix `Ke` and the element load vector `fe` for a quadrilateral heat flow element.

The element nodal coordinates x_1, y_1, x_2 etc, are supplied to the function by `ex` and `ey`, the element thickness t is supplied by `ep` and the thermal conductivities (or corresponding quantities) k_{xx}, k_{xy} etc are supplied by `D`.

$$\mathbf{ex} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix} \qquad \mathbf{ep} = \begin{bmatrix} t \end{bmatrix} \qquad \mathbf{D} = \begin{bmatrix} k_{xx} & k_{xy} \\ k_{yx} & k_{yy} \end{bmatrix}$$
$$\mathbf{ey} = \begin{bmatrix} y_1 & y_2 & y_3 & y_4 \end{bmatrix}$$

If the scalar variable `eq` is given in the function, the element load vector `fe` is computed, using

$$\mathbf{eq} = \begin{bmatrix} Q \end{bmatrix}$$

where Q is the heat supply per unit volume.

Theory

In computing the element matrices, a fifth degree of freedom is introduced. The location of this extra degree of freedom is defined by the mean value of the coordinates in the corner points. Four sets of element matrices are calculated using `flw2te`. These matrices are then assembled and the fifth degree of freedom is eliminated by static condensation.

8.2.4 flw2qs

Purpose

Compute heat flux and temperature gradients in a quadrilateral heat flow element.

Syntax

```
[es, et] = flw2qs(ex, ey, ep, D, ed)
[es, et] = flw2qs(ex, ey, ep, D, ed, eq)
```

Description

`flw2qs` computes the heat flux vector `es` and the temperature gradient `et` (or corresponding quantities) in a quadrilateral heat flow element.

The input variables `ex`, `ey`, `eq` and the matrix `D` are defined in `flw2qe`. The vector `ed` contains the nodal temperatures $\mathbf{a}^e$ of the element and is obtained by the function `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\ T_1 \ \ T_2 \ \ T_3 \ \ T_4\ ]$$

The output variables

$$\mathbf{es} = \mathbf{q}^T = [\ q_x \ q_y\ ]$$

$$\mathbf{et} = (\nabla T)^T = \begin{bmatrix} \frac{\partial T}{\partial x} & \frac{\partial T}{\partial y} \end{bmatrix}$$

contain the components of the heat flux and the temperature gradient computed in the directions of the coordinate axis.

Theory

By assembling four triangular elements as described in `flw2te` a system of equations containing 5 degrees of freedom is obtained. From this system of equations the unknown temperature at the center of the element is computed. Then according to the description in `flw2ts` the temperature gradient and the heat flux in each of the four triangular elements are produced. Finally the temperature gradient and the heat flux of the quadrilateral element are computed as area weighted mean values from the values of the four triangular elements. If heat is supplied to the element, the element load vector `eq` is needed for the calculations.

Note

If the input variables are given for a number of identical (`nie`) elements, i.e. `Ex`, `Ey`, and `Ed` are matrices, then the output variables are defined as

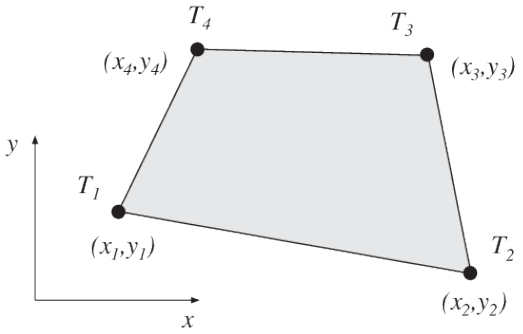
$$\mathrm{Es} = \begin{bmatrix} q_x^1 & q_y^1 \\ q_x^2 & q_y^2 \\ \vdots & \vdots \\ q_x^{nie} & q_y^{nie} \end{bmatrix} \qquad \mathrm{Et} = \begin{bmatrix} \frac{\partial T}{\partial x}^1 & \frac{\partial T}{\partial y}^1 \\ \frac{\partial T}{\partial x}^2 & \frac{\partial T}{\partial y}^2 \\ \vdots & \vdots \\ \frac{\partial T}{\partial x}^{nie} & \frac{\partial T}{\partial y}^{nie} \end{bmatrix}$$

where $\mathbf{q}^i$ and ∇T^i are computed from the nodal values located in column `i` of `Ed`.

8.2.5 flw2i4e

Purpose

Compute element stiffness matrix for a 4 node isoparametric heat flow element.



Syntax

```
Ke = flw2i4e(ex, ey, ep, D)
[Ke, fe] = flw2i4e(ex, ey, ep, D, eq)
```

Description

`flw2i4e` provides the element stiffness (conductivity) matrix `Ke` and the element load vector `fe` for a 4 node isoparametric heat flow element.

The element nodal coordinates x_1, y_1, x_2 etc, are supplied to the function by `ex` and `ey`. The element thickness t and the number of Gauss points n ($n \times n$ integration points, $n = 1, 2, 3$) are supplied to the function by `ep` and the thermal conductivities (or corresponding quantities) k_{xx}, k_{xy} etc are supplied by `D`.

$$\mathbf{ex} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix} \qquad \mathbf{ep} = \begin{bmatrix} t & n \end{bmatrix} \qquad \mathbf{D} = \begin{bmatrix} k_{xx} & k_{xy} \\ k_{yx} & k_{yy} \end{bmatrix}$$
$$\mathbf{ey} = \begin{bmatrix} y_1 & y_2 & y_3 & y_4 \end{bmatrix}$$

If the scalar variable `eq` is given in the function, the element load vector fe is computed, using

$$\mathbf{eq} = \begin{bmatrix} Q \end{bmatrix}$$

where Q is the heat supply per unit volume.

Theory

The element stiffness matrix $\mathbf{K}^e$ and the element load vector $\mathbf{f}_l^e$, stored in `Ke` and `fe`, respectively, are computed according to

$$\mathbf{K}^e = \int_A \mathbf{B}^{eT}\mathbf{D}\mathbf{B}^e t\, dA$$

$$\mathbf{f}_l^e = \int_A \mathbf{N}^{eT} Q t\, dA$$

with the constitutive matrix $\mathbf{D}$ defined by `D`.

The evaluation of the integrals for the isoparametric 4 node element is based on a temperature approximation $T(\xi, \eta)$, expressed in a local coordinate system in terms of the nodal variables T_1, T_2, T_3 and T_4 as

$$T(\xi, \eta) = \mathbf{N}^e \mathbf{a}^e$$

where

$$\mathbf{N}^e = [\, N_1^e \;\; N_2^e \;\; N_3^e \;\; N_4^e \,] \qquad \mathbf{a}^e = [\, T_1 \;\; T_2 \;\; T_3 \;\; T_4 \,]^T$$

The element shape functions are given by

$$N_1^e = \frac{1}{4}(1-\xi)(1-\eta) \qquad N_2^e = \frac{1}{4}(1+\xi)(1-\eta)$$
$$N_3^e = \frac{1}{4}(1+\xi)(1+\eta) \qquad N_4^e = \frac{1}{4}(1-\xi)(1+\eta)$$

The $\mathbf{B}^e$-matrix is given by

$$\mathbf{B}^e = \nabla \mathbf{N}^e = \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix} \mathbf{N}^e = (\mathbf{J}^T)^{-1} \begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{bmatrix} \mathbf{N}^e$$

where $\mathbf{J}$ is the Jacobian matrix

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix}$$

Evaluation of the integrals is done by Gauss integration.

8.2.6 flw2i4s

Purpose

Compute heat flux and temperature gradients in a 4 node isoparametric heat flow element.

Syntax

```
[es, et, eci] = flw2i4s(ex, ey, ep, D, ed)
```

Description

`flw2i4s` computes the heat flux vector `es` and the temperature gradient `et` (or corresponding quantities) in a 4 node isoparametric heat flow element.

The input variables $\mathbf{ex}$, $\mathbf{ey}$, $\mathbf{ep}$ and the matrix $\mathbf{D}$ are defined in `flw2i4e`. The vector $\mathbf{ed}$ contains the nodal temperatures $\mathbf{a}^e$ of the element and is obtained by `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\, T_1 \;\; T_2 \;\; T_3 \;\; T_4 \,]$$

The output variables

$$\mathbf{es} = \bar{\mathbf{q}}^T = \begin{bmatrix} q_x^1 & q_y^1 \\ q_x^2 & q_y^2 \\ \vdots & \vdots \\ q_x^{n^2} & q_y^{n^2} \end{bmatrix}$$

$$\mathbf{et} = (\bar{\nabla} T)^T = \begin{bmatrix} \frac{\partial T}{\partial x}^1 & \frac{\partial T}{\partial y}^1 \\ \frac{\partial T}{\partial x}^2 & \frac{\partial T}{\partial y}^2 \\ \vdots & \vdots \\ \frac{\partial T}{\partial x}^{n^2} & \frac{\partial T}{\partial y}^{n^2} \end{bmatrix}$$

$$\mathbf{eci} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_{n^2} & y_{n^2} \end{bmatrix}$$

contain the heat flux, the temperature gradient, and the coordinates of the integration points. The index n denotes the number of integration points used within the element, cf. `flw2i4e`.

Theory

The temperature gradient and the heat flux are computed according to

$$\nabla T = \mathbf{B}^e \, \mathbf{a}^e$$

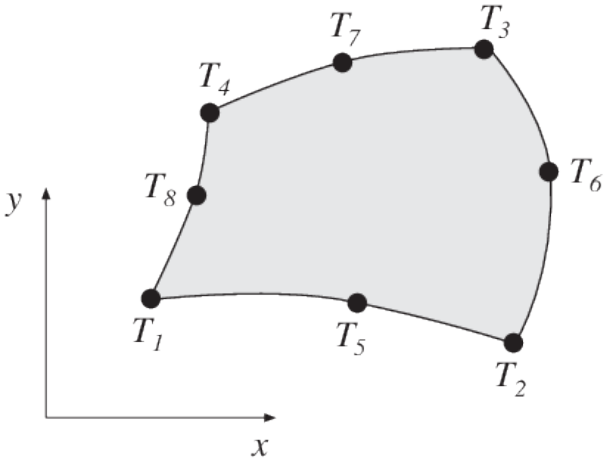
$$\mathbf{q} = -\mathbf{D} \nabla T$$

where the matrices $\mathbf{D}$, $\mathbf{B}^e$, and $\mathbf{a}^e$ are described in `flw2i4e`, and where the integration points are chosen as evaluation points.

8.2.7 flw2i8e

Purpose

Compute element stiffness matrix for an 8 node isoparametric heat flow element.



Syntax

```
Ke = flw2i8e(ex, ey, ep, D)
[Ke, fe] = flw2i8e(ex, ey, ep, D, eq)
```

Description

`flw2i8e` provides the element stiffness (conductivity) matrix `Ke`

and the element load vector `fe` for an 8 node isoparametric heat flow element.

The element nodal coordinates x_1, y_1, x_2 etc, are supplied to the function by $\mathbf{ex}$ and $\mathbf{ey}$. The element thickness t and the number of Gauss points n ($n \times n$ integration points, $n = 1, 2, 3$) are supplied to the function by $\mathbf{ep}$ and the thermal conductivities (or corresponding quantities) k_{xx}, k_{xy} etc are supplied by `D`.

$$\mathbf{ex} = [\, x_1 \;\; x_2 \;\; x_3 \;\; \ldots \;\; x_8 \,] \qquad \mathbf{ep} = [\, t \;\; n \,] \qquad \mathbf{D} = \begin{bmatrix} k_{xx} & k_{xy} \\ k_{yx} & k_{yy} \end{bmatrix}$$
$$\mathbf{ey} = [\, y_1 \;\; y_2 \;\; y_3 \;\; \ldots \;\; y_8 \,]$$

If the scalar variable `eq` is given in the function, the vector $\mathbf{fe}$ is computed, using

$$\mathbf{eq} = [\, Q \,]$$

where Q is the heat supply per unit volume.

Theory

The element stiffness matrix $\mathbf{K}^e$ and the element load vector $\mathbf{f}_l^e$, stored in `Ke` and `fe`, respectively, are computed according to

$$\mathbf{K}^e = \int_A \mathbf{B}^{eT} \mathbf{D} \mathbf{B}^e t \, dA$$

$$\mathbf{f}_l^e = \int_A \mathbf{N}^{eT} Q t \, dA$$

with the constitutive matrix $\mathbf{D}$ defined by `D`.

The evaluation of the integrals for the 2D isoparametric 8 node element is based on a temperature approximation $T(\xi, \eta)$, expressed in a local coordinates system in terms of the nodal variables T_1 to T_8 as

$$T(\xi, \eta) = \mathbf{N}^e \mathbf{a}^e$$

where

$$\mathbf{N}^e = [\, N_1^e \;\; N_2^e \;\; N_3^e \;\; \ldots \;\; N_8^e \,] \qquad \mathbf{a}^e = [\, T_1 \;\; T_2 \;\; T_3 \;\; \ldots \;\; T_8 \,]^T$$

The element shape functions are given by

$$N_1^e = -\frac{1}{4}(1-\xi)(1-\eta)(1+\xi+\eta) \quad N_5^e = \frac{1}{2}(1-\xi^2)(1-\eta)$$
$$N_2^e = -\frac{1}{4}(1+\xi)(1-\eta)(1-\xi+\eta) \quad N_6^e = \frac{1}{2}(1+\xi)(1-\eta^2)$$
$$N_3^e = -\frac{1}{4}(1+\xi)(1+\eta)(1-\xi-\eta) \quad N_7^e = \frac{1}{2}(1-\xi^2)(1+\eta)$$
$$N_4^e = -\frac{1}{4}(1-\xi)(1+\eta)(1+\xi-\eta) \quad N_8^e = \frac{1}{2}(1-\xi)(1-\eta^2)$$

The $\mathbf{B}^e$-matrix is given by

$$\mathbf{B}^e = \nabla \mathbf{N}^e = \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix} \mathbf{N}^e = (\mathbf{J}^T)^{-1} \begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{bmatrix} \mathbf{N}^e$$

where $\mathbf{J}$ is the Jacobian matrix

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix}$$

Evaluation of the integrals is done by Gauss integration.

8.2.8 flw2i8s

Purpose

Compute heat flux and temperature gradients in an 8 node isoparametric heat flow element.

Syntax

```
[es, et, eci] = flw2i8s(ex, ey, ep, D, ed)
```

Description

`flw2i8s` computes the heat flux vector `es` and the temperature gradient `et` (or corresponding quantities) in an 8 node isoparametric heat flow element.

The input variables `ex`, `ey`, `ep` and the matrix `D` are defined in `flw2i8e`. The vector `ed` contains the nodal temperatures $\mathbf{a}^e$ of the element and is obtained by the function `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\ T_1 \ \ T_2 \ \ T_3 \ \ \ldots \ \ T_8\ ]$$

The output variables

$$\mathbf{es} = \bar{\mathbf{q}}^T = \begin{bmatrix} q_x^1 & q_y^1 \\ q_x^2 & q_y^2 \\ \vdots & \vdots \\ q_x^{n^2} & q_y^{n^2} \end{bmatrix}$$

$$\mathbf{et} = (\bar{\nabla} T)^T = \begin{bmatrix} \frac{\partial T}{\partial x}^1 & \frac{\partial T}{\partial y}^1 \\ \frac{\partial T}{\partial x}^2 & \frac{\partial T}{\partial y}^2 \\ \vdots & \vdots \\ \frac{\partial T}{\partial x}^{n^2} & \frac{\partial T}{\partial y}^{n^2} \end{bmatrix}$$

$$\mathbf{eci} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_{n^2} & y_{n^2} \end{bmatrix}$$

contain the heat flux, the temperature gradient, and the coordinates of the integration points. The index n denotes the number of integration points used within the element, see `flw2i8e`.

Theory

The temperature gradient and the heat flux are computed according to

$$\nabla T = \mathbf{B}^e \, \mathbf{a}^e$$

$$\mathbf{q} = -\mathbf{D} \nabla T$$

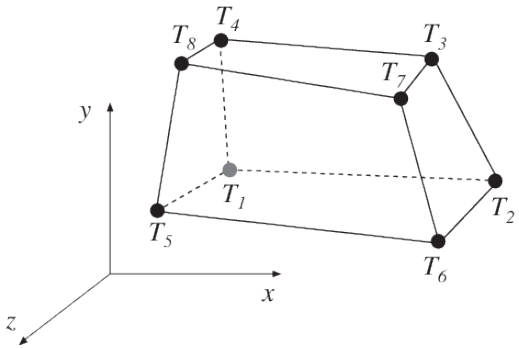
where the matrices $\mathbf{D}$, $\mathbf{B}^e$, and $\mathbf{a}^e$ are described in `flw2i8e`, and where the integration points are chosen as evaluation points.

8.3 3D Heat Flow Functions

8.3.1 flw3i8e

Purpose

Compute element stiffness matrix for an 8 node isoparametric element.



Syntax

```
Ke = flw3i8e(ex, ey, ez, ep, D)
[Ke, fe] = flw3i8e(ex, ey, ez, ep, D, eq)
```

Description

`flw3i8e` provides the element stiffness (conductivity) matrix `Ke` and the element load vector `fe` for an 8 node isoparametric heat flow element.

The element nodal coordinates x_1, y_1, z_1, x_2 etc, are supplied to the function by `ex`, `ey` and `ez`. The number of Gauss points n ($n \times n \times n$ integration points, $n = 1, 2, 3$) are supplied to the function by `ep` and the thermal conductivities (or corresponding quantities) k_{xx}, k_{xy} etc are supplied by `D`.

$$\mathbf{ex} = \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_8 \end{bmatrix}$$
$$\mathbf{ey} = \begin{bmatrix} y_1 & y_2 & y_3 & \dots & y_8 \end{bmatrix}$$
$$\mathbf{ez} = \begin{bmatrix} z_1 & z_2 & z_3 & \dots & z_8 \end{bmatrix}$$

$$\mathbf{ep} = \begin{bmatrix} n \end{bmatrix} \qquad \mathbf{D} = \begin{bmatrix} k_{xx} & k_{xy} & k_{xz} \\ k_{yx} & k_{yy} & k_{yz} \\ k_{zx} & k_{zy} & k_{zz} \end{bmatrix}$$

If the scalar variable `eq` is given in the function, the element load vector **fe** is computed, using

$$\mathbf{eq} = \begin{bmatrix} Q \end{bmatrix}$$

where Q is the heat supply per unit volume.

Theory

The element stiffness matrix $\mathbf{K}^e$ and the element load vector $\mathbf{f}_l^e$,

stored in `Ke` and `fe`, respectively, are computed according to

$$\mathbf{K}^e = \int_V \mathbf{B}^{eT}\mathbf{D}\mathbf{B}^e \, dV$$

$$\mathbf{f}_l^e = \int_V \mathbf{N}^{eT} Q \, dV$$

with the constitutive matrix $\mathbf{D}$ defined by `D`.

The evaluation of the integrals for the 3D isoparametric 8 node element is based on a temperature approximation $T(\xi, \eta, \zeta)$, expressed in a local coordinate system in terms of the nodal variables T_1 to T_8 as

$$T(\xi, \eta, \zeta) = \mathbf{N}^e \mathbf{a}^e$$

$$\mathbf{N}^e = [\, N_1^e \;\; N_2^e \;\; N_3^e \;\; \ldots \;\; N_8^e \,] \qquad \mathbf{a}^e = [\, T_1 \;\; T_2 \;\; T_3 \;\; \ldots \;\; T_8 \,]^T$$

The element shape functions are given by

$$N_1^e = \frac{1}{8}(1-\xi)(1-\eta)(1-\zeta) \qquad N_2^e = \frac{1}{8}(1+\xi)(1-\eta)(1-\zeta)$$

$$N_3^e = \frac{1}{8}(1+\xi)(1+\eta)(1-\zeta) \qquad N_4^e = \frac{1}{8}(1-\xi)(1+\eta)(1-\zeta)$$

$$N_5^e = \frac{1}{8}(1-\xi)(1-\eta)(1+\zeta) \qquad N_6^e = \frac{1}{8}(1+\xi)(1-\eta)(1+\zeta)$$

$$N_7^e = \frac{1}{8}(1+\xi)(1+\eta)(1+\zeta) \qquad N_8^e = \frac{1}{8}(1-\xi)(1+\eta)(1+\zeta)$$

The $\mathbf{B}^e$-matrix is given by

$$\mathbf{B}^e = \nabla \mathbf{N}^e = \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{bmatrix} \mathbf{N}^e = (\mathbf{J}^T)^{-1} \begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \\ \frac{\partial}{\partial \zeta} \end{bmatrix} \mathbf{N}^e$$

where $\mathbf{J}$ is the Jacobian matrix

$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \zeta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} & \frac{\partial y}{\partial \zeta} \\ \frac{\partial z}{\partial \xi} & \frac{\partial z}{\partial \eta} & \frac{\partial z}{\partial \zeta} \end{bmatrix}$$

Evaluation of the integrals is done by Gauss integration.

8.3.2 flw3i8s

Purpose

Compute heat flux and temperature gradients in an 8 node isoparametric heat flow element.

Syntax

```
[es, et, eci] = flw3i8s(ex, ey, ez, ep, D, ed)
```

Description

`flw3i8s` computes the heat flux vector `es` and the temperature gradient `et` (or corresponding quantities) in an 8 node isoparametric heat flow element.

The input variables **ex**, **ey**, **ez**, **ep** and the matrix $\mathbf{D}$ are defined in `flw3i8e`. The vector **ed** contains the nodal temperatures $\mathbf{a}^e$ of the element and is obtained by the function `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\, T_1 \;\; T_2 \;\; T_3 \;\; \ldots \;\; T_8 \,]$$

The output variables

$$\mathbf{es} = \bar{\mathbf{q}}^T = \begin{bmatrix} q_x^1 & q_y^1 & q_z^1 \\ q_x^2 & q_y^2 & q_z^2 \\ \vdots & \vdots & \vdots \\ q_x^{n^3} & q_y^{n^3} & q_z^{n^3} \end{bmatrix}$$

$$\mathbf{et} = (\bar{\nabla} T)^T = \begin{bmatrix} \frac{\partial T}{\partial x}^1 & \frac{\partial T}{\partial y}^1 & \frac{\partial T}{\partial z}^1 \\ \frac{\partial T}{\partial x}^2 & \frac{\partial T}{\partial y}^2 & \frac{\partial T}{\partial z}^2 \\ \vdots & \vdots & \vdots \\ \frac{\partial T}{\partial x}^{n^3} & \frac{\partial T}{\partial y}^{n^3} & \frac{\partial T}{\partial z}^{n^3} \end{bmatrix} \qquad \mathbf{eci} = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_{n^3} & y_{n^3} & z_{n^3} \end{bmatrix}$$

contain the heat flux, the temperature gradient, and the coordinates of the integration points. The index n denotes the number of integration points used within the element, see `flw3i8e`.

Theory

The temperature gradient and the heat flux are computed according to

$$\nabla T = \mathbf{B}^e \, \mathbf{a}^e$$

$$\mathbf{q} = -\mathbf{D}\nabla T$$

where the matrices $\mathbf{D}$, $\mathbf{B}^e$, and $\mathbf{a}^e$ are described in `flw3i8e`, and where the integration points are chosen as evaluation points.

CHAPTER

NINE

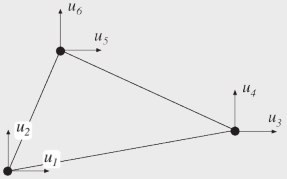
SOLID ELEMENTS FUNCTIONS

Solid elements are available for two dimensional analysis in plane stress (panels) and plane strain, and for general three dimensional analysis. In the two dimensional case there are a triangular three node element, a quadrilateral four node element, two rectangular four node elements, and quadrilateral isoparametric four and eight node elements. For three dimensional analysis there is an eight node isoparametric element.

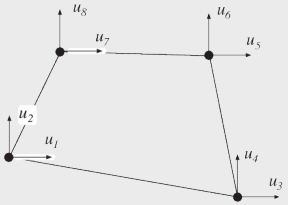
The elements are able to deal with both isotropic and anisotropic materials. The triangular element and the three isoparametric elements can also be used together with a nonlinear material model.

The material properties are specified by supplying the constitutive matrix $\mathbf{D}$ as an input variable to the element functions. This matrix can be formed by the functions described in Section *Material functions*.

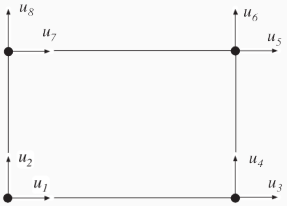
Table 1: **Solid elements**



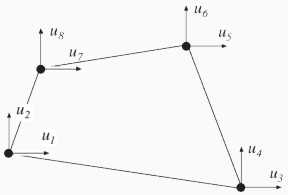
plante



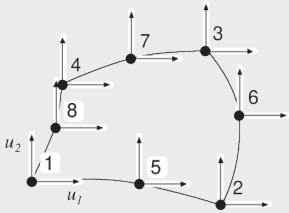
planqe



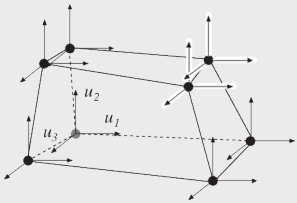
planre plantce



plani4e



plani8e



soli8e

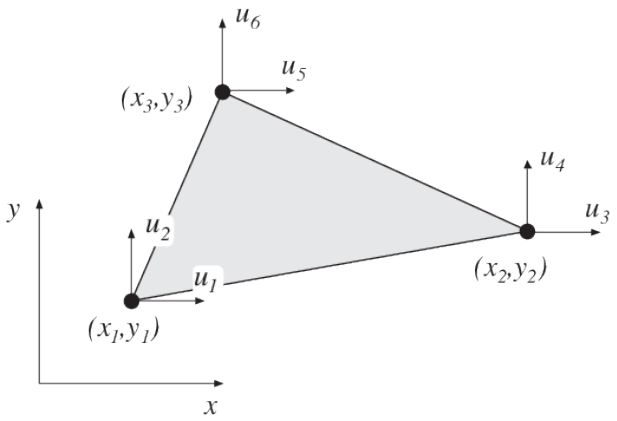
9.1 2D Solid Functions

9.1.1 plante

Purpose

Compute element matrices for a triangular element in plane strain or plane stress.

Syntax




```
Ke = plante(ex, ey, ep, D)
[Ke, fe] = plante(ex, ey, ep, D, eq)
```

Description

`plante` provides an element stiffness matrix `Ke` and an element load vector `fe` for a triangular element in plane strain or plane stress.

The element nodal coordinates $x_1, y_1, x_2, \ldots$ are supplied to the function by $\mathbf{ex}$ and $\mathbf{ey}$. The type of analysis `ptype` and the element thickness `t` are supplied by $\mathbf{ep}$,

$$\begin{array}{ll} ptype = 1 & \text{plane stress} \\ ptype = 2 & \text{plane strain} \end{array}$$

and the material properties are supplied by the constitutive matrix `D`. Any arbitrary $\mathbf{D}$-matrix with dimensions from 3×3 to 6×6 may be given. For an isotropic elastic material the constitutive matrix can be formed by the function `hooke`, see Section *Material functions*.

$$\begin{aligned} \mathbf{ex} &= [\, x_1 \;\; x_2 \;\; x_3 \,] \\ \mathbf{ey} &= [\, y_1 \;\; y_2 \;\; y_3 \,] \\ \mathbf{ep} &= [\, ptype \;\; t \,] \end{aligned}$$

$$\mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} \\ D_{21} & D_{22} & D_{23} \\ D_{31} & D_{32} & D_{33} \end{bmatrix} \quad \text{or} \quad \mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} & D_{14} & [D_{15}] & [\\ D_{21} & D_{22} & D_{23} & D_{24} & [D_{25}] & [\\ D_{31} & D_{32} & D_{33} & D_{34} & [D_{35}] & [\\ D_{41} & D_{42} & D_{43} & D_{44} & [D_{45}] & [\\ [D_{51}] & [D_{52}] & [D_{53}] & [D_{54}] & [D_{55}] & [\\ [D_{61}] & [D_{62}] & [D_{63}] & [D_{64}] & [D_{65}] & [\end{bmatrix}$$

If uniformly distributed loads are applied to the element, the element load vector `fe` is computed. The input variable

$$\text{eq} = \begin{bmatrix} b_x \\ b_y \end{bmatrix}$$

containing loads per unit volume, b_x and b_y, is then given.

Theory

The element stiffness matrix $\mathbf{K}^e$ and the element load vector $\mathbf{f}_l^e$, stored in `Ke` and `fe`, respectively, are computed according to

$$\mathbf{K}^e = (\mathbf{C}^{-1})^T \int_A \bar{\mathbf{B}}^T \mathbf{D} \bar{\mathbf{B}} \, t \, dA \, \mathbf{C}^{-1}$$

$$\mathbf{f}_l^e = (\mathbf{C}^{-1})^T \int_A \bar{\mathbf{N}}^T \mathbf{b} \, t \, dA$$

with the constitutive matrix $\mathbf{D}$ defined by `D`, and the body force vector $\mathbf{b}$ defined by `eq`.

The evaluation of the integrals for the triangular element is based on a linear displacement approximation $\mathbf{u}(x, y)$ and is expressed in terms of the nodal variables $u_1, u_2, \ldots, u_6$ as

$$\mathbf{u}(x, y) = \mathbf{N}^e \mathbf{a}^e = \bar{\mathbf{N}} \, \mathbf{C}^{-1} \mathbf{a}^e$$

where

$$\mathbf{u} = \begin{bmatrix} u_x \\ u_y \end{bmatrix} \quad \bar{\mathbf{N}} = \begin{bmatrix} 1 & x & y & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x & y \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} 1 & x_1 & y_1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x_1 & y_1 \\ 1 & x_2 & y_2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x_2 & y_2 \\ 1 & x_3 & y_3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x_3 & y_3 \end{bmatrix} \quad \mathbf{a}^e = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix}$$

The matrix $\bar{\mathbf{B}}$ is obtained as

$$\bar{\mathbf{B}} = \tilde{\nabla}\bar{\mathbf{N}} \quad \text{where} \quad \tilde{\nabla} = \begin{bmatrix} \dfrac{\partial}{\partial x} & 0 \\ 0 & \dfrac{\partial}{\partial y} \\ \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial x} \end{bmatrix}$$

If a larger $\mathbf{D}$-matrix than 3×3 is used for plane stress ($ptype = 1$), the $\mathbf{D}$-matrix is reduced to a 3×3 matrix by static condensation using $\sigma_{zz} = \sigma_{xz} = \sigma_{yz} = 0$. These stress components are connected with the rows 3, 5 and 6 in the D-matrix respectively.

If a larger $\mathbf{D}$-matrix than 3×3 is used for plane strain ($ptype = 2$), the $\mathbf{D}$-matrix is reduced to a 3×3 matrix using $\varepsilon_{zz} = \gamma_{xz} = \gamma_{yz} = 0$. This implies that a 3×3 $\mathbf{D}$-matrix is created by the rows and the columns 1, 2 and 4 from the original D-matrix.

Evaluation of the integrals for the triangular element yields

$$\mathbf{K}^e = (\mathbf{C}^{-1})^T \bar{\mathbf{B}}^T \mathbf{D} \bar{\mathbf{B}}\, \mathbf{C}^{-1}\, t\, A$$

$$\mathbf{f}_l^e = \frac{At}{3} \begin{bmatrix} b_x & b_y & b_x & b_y & b_x & b_y \end{bmatrix}^T$$

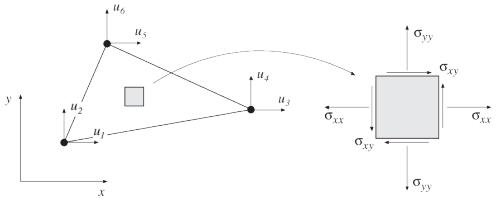
where the element area A is determined as

$$A = \frac{1}{2} \det \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix}$$

9.1.2 plants

Purpose

Compute stresses and strains in a triangular element in plane strain or plane stress.



Syntax

```
[es, et] = plants(ex, ey, ep, D, ed)
```

Description

`plants` computes the stresses `es` and the strains `et` in a triangular element in plane strain or plane stress.

The input variables $\mathbf{ex}$, $\mathbf{ey}$, $\mathbf{ep}$ and $\mathbf{D}$ are defined in `plante`. The vector $\mathbf{ed}$ contains the nodal displacements $\mathbf{a}^e$ of the element and is obtained by the function `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\, u_1 \;\; u_2 \;\; \ldots \;\; u_6 \,]$$

The output variables

$$\text{es} = \boldsymbol{\sigma}^T = [\, \sigma_{xx} \;\; \sigma_{yy} \;\; [\sigma_{zz}] \;\; \sigma_{xy} \;\; [\sigma_{xz}] \;\; [\sigma_{yz}] \,]$$

$$\text{et} = \boldsymbol{\varepsilon}^T = [\, \varepsilon_{xx} \;\; \varepsilon_{yy} \;\; [\varepsilon_{zz}] \;\; \gamma_{xy} \;\; [\gamma_{xz}] \;\; [\gamma_{yz}] \,]$$

contain the stress and strain components. The size of `es` and `et` follows the size of `D`. Note that for plane stress $\varepsilon_{zz} \neq 0$, and for plane strain $\sigma_{zz} \neq 0$.

Theory

The strains and stresses are computed according to

$$\boldsymbol{\varepsilon} = \bar{\mathbf{B}}\, \mathbf{C}^{-1}\, \mathbf{a}^e$$

$$\boldsymbol{\sigma} = \mathbf{D}\, \boldsymbol{\varepsilon}$$

where the matrices $\mathbf{D}$, $\bar{\mathbf{B}}$, $\mathbf{C}$ and $\mathbf{a}^e$ are described in `plante`. Note that both the strains and the stresses are constant in the element.

9.1.3 plantf

Purpose

Compute internal element force vector in a triangular element in plane strain or plane stress.

Syntax

```
ef = plantf(ex, ey, ep, es)
```

Description

`plantf` computes the internal element forces `ef` in a triangular element in plane strain or plane stress.

The input variables $\mathbf{ex}$, $\mathbf{ey}$ and $\mathbf{ep}$ are defined in `plante`, and the input variable $\mathbf{es}$ is defined in `plants`.

The output variable

$$\text{ef} = \mathbf{f}_i^{eT} = [\, f_{i1} \; f_{i2} \; \ldots \; f_{i6} \,]$$

contains the components of the internal force vector.

Theory

The internal force vector is computed according to

$$\mathbf{f}_i^e = (\mathbf{C}^{-1})^T \int_A \bar{\mathbf{B}}^T \boldsymbol{\sigma} \; t \; dA$$

where the matrices $\bar{\mathbf{B}}$ and $\mathbf{C}$ are defined in `plante` and $\boldsymbol{\sigma}$ is defined in `plants`.

Evaluation of the integral for the triangular element yields

$$\mathbf{f}_i^e = (\mathbf{C}^{-1})^T \bar{\mathbf{B}}^T \boldsymbol{\sigma} \; t \; A$$

9.1.4 planqe

Purpose

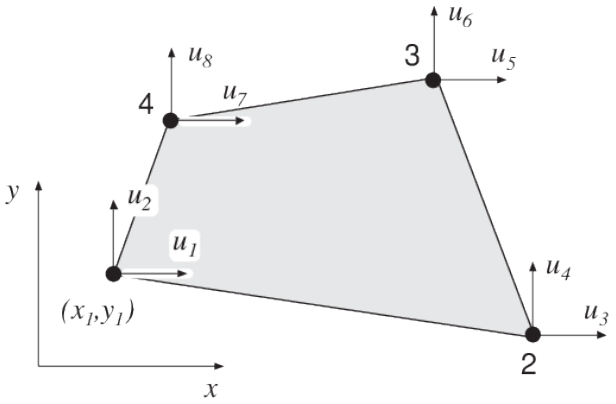
Compute element matrices for a quadrilateral element in plane strain or plane stress.

Syntax

```
Ke = planqe(ex, ey, ep, D)
[Ke, fe] = planqe(ex, ey, ep, D, eq)
```

Description

`planqe` provides an element stiffness matrix Ke and an element load vector fe for a quadrilateral element in plane strain or plane stress.



The element nodal coordinates x_1, y_1, x_2, etc. are supplied to the function by **ex** and **ey**. The type of analysis $ptype$ and the element thickness t are supplied by **ep**:

$ptype = 1$	plane stress
$ptype = 2$	plane strain

The material properties are supplied by the constitutive matrix D. Any arbitrary **D**-matrix with dimensions from 3×3 to 6×6 may be given. For an isotropic elastic material the constitutive matrix can be formed by the function **hooke**, see Section *Material functions*.

$$\mathbf{ex} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix}$$
$$\mathbf{ey} = \begin{bmatrix} y_1 & y_2 & y_3 & y_4 \end{bmatrix}$$
$$\mathbf{ep} = \begin{bmatrix} ptype & t \end{bmatrix}$$

$$\mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} \\ D_{21} & D_{22} & D_{23} \\ D_{31} & D_{32} & D_{33} \end{bmatrix} \quad \text{or} \quad \mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} & D_{14} & D_{15} & D_{16} \\ D_{21} & D_{22} & D_{23} & D_{24} & D_{25} & D_{26} \\ D_{31} & D_{32} & D_{33} & D_{34} & D_{35} & D_{36} \\ D_{41} & D_{42} & D_{43} & D_{44} & D_{45} & D_{46} \\ D_{51} & D_{52} & D_{53} & D_{54} & D_{55} & D_{56} \\ D_{61} & D_{62} & D_{63} & D_{64} & D_{65} & D_{66} \end{bmatrix}$$

If uniformly distributed loads are applied on the element, the element load vector fe is computed. The input variable

$$eq = \begin{bmatrix} b_x \\ b_y \end{bmatrix}$$

contains loads per unit volume, b_x and b_y.

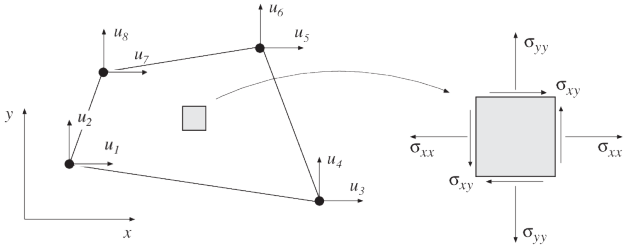
Theory

In computing the element matrices, two more degrees of freedom are introduced. The location of these two degrees of freedom is defined by the mean value of the coordinates at the corner points. Four sets of element matrices are calculated using `plante`. These matrices are then assembled and the two extra degrees of freedom are eliminated by static condensation.

9.1.5 planqs

Purpose

Compute stresses and strains in a quadrilateral element in plane strain or plane stress.



Syntax

```
[es,et]=planqs(ex,ey,ep,D,ed)
[es,et]=planqs(ex,ey,ep,D,ed,eq)
```

Description

`planqs` computes the stresses `es` and the strains `et` in a quadrilateral element in plane strain or plane stress.

The input variables $\mathbf{ex}$, $\mathbf{ey}$, $\mathbf{ep}$, $\mathbf{D}$ and $\mathbf{eq}$ are defined in `planqe`. The vector $\mathbf{ed}$ contains the nodal displacements $\mathbf{a}^e$ of the element and is obtained by the function `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\, u_1 \;\; u_2 \;\; \ldots \;\; u_8 \,]$$

If body forces are applied to the element the variable $\mathbf{eq}$ must be included.

The output variables

$$\text{es} = \boldsymbol{\sigma}^T = [\,\sigma_{xx}\ \sigma_{yy}\ [\sigma_{zz}]\ \sigma_{xy}\ [\sigma_{xz}]\ [\sigma_{yz}]\,]$$

$$\text{et} = \boldsymbol{\varepsilon}^T = [\,\varepsilon_{xx}\ \varepsilon_{yy}\ [\varepsilon_{zz}]\ \gamma_{xy}\ [\gamma_{xz}]\ [\gamma_{yz}]\,]$$

contain the stress and strain components. The size of `es` and `et` follows the size of `D`. Note that for plane stress $\varepsilon_{zz} \neq 0$, and for plane strain $\sigma_{zz} \neq 0$.

Theory

By assembling triangular elements as described in `planqe` a system of equations containing 10 degrees of freedom is obtained. From this system of equations the two unknown displacements at the center of the element are computed. Then according to the description in `plants` the strain and stress components in each of the four triangular elements are produced. Finally the quadrilateral element strains and stresses are computed as area weighted mean values from the values of the four triangular elements. If uniformly distributed loads are applied on the element, the element load vector `eq` is needed for the calculations.

9.1.6 planre

Purpose

Compute element matrices for a rectangular (Melosh) element in plane strain or plane stress.

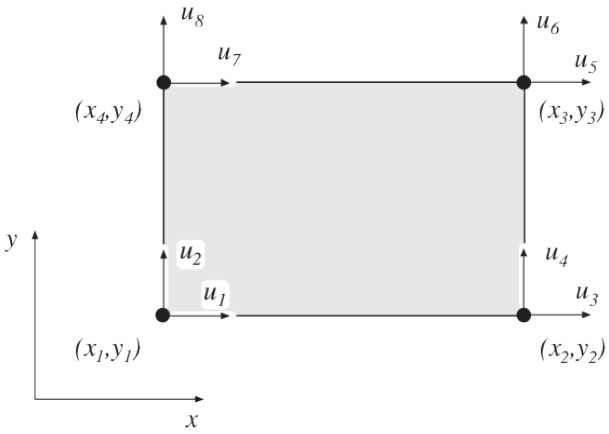
Syntax

```
Ke = planre(ex, ey, ep, D)
[Ke, fe] = planre(ex, ey, ep, D, eq)
```

Description

planre provides an element stiffness matrix Ke and an element load vector fe for a rectangular (Melosh) element in plane strain or plane stress. This element can only be used if the element edges are parallel to the coordinate axis.

The element nodal coordinates (x_1, y_1) and (x_3, y_3) are supplied to the function by ex and ey. The type of analysis ptype and the



element thickness t are supplied by ep,

$$\begin{array}{ll} ptype = 1 & \text{plane stress} \\ ptype = 2 & \text{plane strain} \end{array}$$

and the material properties are supplied by the constitutive matrix D. Any arbitrary D-matrix with dimensions from 3×3 to 6×6 may be given. For an isotropic elastic material the constitutive matrix can be formed by the function hooke, see Section Material.

$$\begin{array}{l} \mathbf{ex} = [\, x_1 \;\; x_3 \,] \\ \mathbf{ey} = [\, y_1 \;\; y_3 \,] \end{array} \qquad \mathbf{ep} = [\, ptype \;\; t \,]$$

$$\mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} \\ D_{21} & D_{22} & D_{23} \\ D_{31} & D_{32} & D_{33} \end{bmatrix} \quad \text{or} \quad \mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} & D_{14} & [D_{15}] & [\\ D_{21} & D_{22} & D_{23} & D_{24} & [D_{25}] & [\\ D_{31} & D_{32} & D_{33} & D_{34} & [D_{35}] & [\\ D_{41} & D_{42} & D_{43} & D_{44} & [D_{45}] & [\\ [D_{51}] & [D_{52}] & [D_{53}] & [D_{54}] & [D_{55}] & [\\ [D_{61}] & [D_{62}] & [D_{63}] & [D_{64}] & [D_{65}] & [\end{bmatrix}$$

If uniformly distributed loads are applied on the element, the element load vector fe is computed. The input variable

$$eq = \begin{bmatrix} b_x \\ b_y \end{bmatrix}$$

containing loads per unit volume, b_x and b_y, is then given.

Theory

The element stiffness matrix $\mathbf{K}^e$ and the element load vector $\mathbf{f}_l^e$, stored in Ke and fe, respectively, are computed according to

$$\mathbf{K}^e = \int_A \mathbf{B}^{eT}\mathbf{D}\mathbf{B}^e t\, dA$$

$$\mathbf{f}_l^e = \int_A \mathbf{N}^{eT}\mathbf{b} t\, dA$$

with the constitutive matrix $\mathbf{D}$ defined by D, and the body force vector $\mathbf{b}$ defined by eq.

The evaluation of the integrals for the rectangular element is based on a bilinear displacement approximation $\mathbf{u}(x, y)$ and is expressed in terms of the nodal variables $u_1, u_2, \ldots, u_8$ as

$$\mathbf{u}(x, y) = \mathbf{N}^e\mathbf{a}^e$$

where

$$\mathbf{u} = \begin{bmatrix} u_x \\ u_y \end{bmatrix} \quad \mathbf{N}^e = \begin{bmatrix} N_1^e & 0 & N_2^e & 0 & N_3^e & 0 & N_4^e & 0 \\ 0 & N_1^e & 0 & N_2^e & 0 & N_3^e & 0 & N_4^e \end{bmatrix} \quad \mathbf{a}^e = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_8 \end{bmatrix}$$

With a local coordinate system located at the center of the element, the element shape functions N_1^e–N_4^e are obtained as

$$N_1^e = \frac{1}{4ab}(x - x_2)(y - y_4)$$
$$N_2^e = -\frac{1}{4ab}(x - x_1)(y - y_3)$$
$$N_3^e = \frac{1}{4ab}(x - x_4)(y - y_2)$$
$$N_4^e = -\frac{1}{4ab}(x - x_3)(y - y_1)$$

where

$$a = \frac{1}{2}(x_3 - x_1) \quad \text{and} \quad b = \frac{1}{2}(y_3 - y_1)$$

The matrix $\mathbf{B}^e$ is obtained as

$$\mathbf{B}^e = \tilde{\nabla}\mathbf{N}^e \qquad \text{where} \quad \tilde{\nabla} = \begin{bmatrix} \dfrac{\partial}{\partial x} & 0 \\ 0 & \dfrac{\partial}{\partial y} \\ \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial x} \end{bmatrix}$$

If a larger D-matrix than 3
$times3$ is used for plane stress ($ptype = 1$), the D-matrix is reduced to a 3
$times3$ matrix by static condensation using $\sigma_{zz} = \sigma_{xz} = \sigma_{yz} = 0$. These stress components are connected with the rows 3, 5 and 6 in the D-matrix respectively.

If a larger D-matrix than 3
$times3$ is used for plane strain ($ptype = 2$), the D-matrix is reduced to a 3
$times3$ matrix using $\varepsilon_{zz} = \gamma_{xz} = \gamma_{yz} = 0$. This implies that a 3
$times3$ D-matrix is created by the rows and the columns 1, 2 and 4 from the original D-matrix.

Evaluation of the integrals for the rectangular element can be done either analytically or numerically by use of a 2
$times2$ point Gauss integration. The element load vector $\mathbf{f}_l^e$ yields

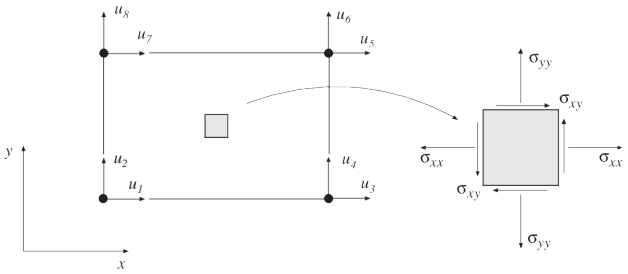
$$\mathbf{f}_l^e = abt \begin{bmatrix} b_x \\ b_y \\ b_x \\ b_y \\ b_x \\ b_y \\ b_x \\ b_y \end{bmatrix}$$

9.1.7 planrs

Purpose

Compute stresses and strains in a rectangular (Melosh) element in plane strain or plane stress.

Syntax




```
[es,et]=planrs(ex,ey,ep,D,ed)
```

Description

`planrs` computes the stresses `es` and the strains `et` in a rectangular (Melosh) element in plane strain or plane stress. The stress and strain components are computed at the center of the element.

The input variables $\mathbf{ex}$, $\mathbf{ey}$, $\mathbf{ep}$ and $\mathbf{D}$ are defined in `planre`. The vector $\mathbf{ed}$ contains the nodal displacements $\mathbf{a}^e$ of the element and is obtained by the function `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\, u_1 \;\; u_2 \;\; \ldots \;\; u_8 \,]$$

The output variables

$$\text{es} = \boldsymbol{\sigma}^T = [\, \sigma_{xx} \;\; \sigma_{yy} \;\; [\sigma_{zz}] \;\; \sigma_{xy} \;\; [\sigma_{xz}] \;\; [\sigma_{yz}] \,]$$

$$\text{et} = \boldsymbol{\varepsilon}^T = [\, \varepsilon_{xx} \;\; \varepsilon_{yy} \;\; [\varepsilon_{zz}] \;\; \gamma_{xy} \;\; [\gamma_{xz}] \;\; [\gamma_{yz}] \,]$$

contain the stress and strain components. The size of `es` and `et` follows the size of `D`. Note that for plane stress $\varepsilon_{zz} \neq 0$, and for plane strain $\sigma_{zz} \neq 0$.

Theory

The strains and stresses are computed according to

$$\boldsymbol{\varepsilon} = \mathbf{B}^e \, \mathbf{a}^e$$

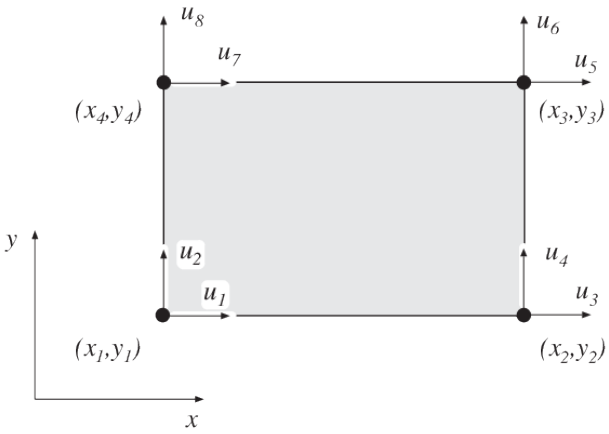
$$\boldsymbol{\sigma} = \mathbf{D} \, \boldsymbol{\varepsilon}$$

where the matrices $\mathbf{D}$, $\mathbf{B}^e$, and $\mathbf{a}^e$ are described in `planre`, and where the evaluation point (x, y) is chosen to be at the center of the element.

9.1.8 plantce

Purpose

Compute element matrices for a rectangular (Turner-Clough) element in plane strain or plane stress.



Syntax

```
Ke = plantce(ex, ey, ep)
[Ke, fe] = plantce(ex, ey, ep, eq)
```

Description

`plantce` provides an element stiffness matrix `Ke` and an element load vector `fe` for a rectangular (Turner-Clough) element in plane strain or plane stress. This element can only be used if the material is isotropic and if the element edges are parallel to the coordinate axis.

The element nodal coordinates (x_1, y_1) and (x_3, y_3) are supplied to the function by $\mathbf{ex}$ and $\mathbf{ey}$. The state of stress `ptype`, the element thickness t and the material properties E and ν are supplied by $\mathbf{ep}$. For plane stress $ptype = 1$ and for plane strain $ptype = 2$.

$$\begin{matrix}\mathbf{ex} = [\,x_1 \;\; x_3\,] \\ \mathbf{ey} = [\,y_1 \;\; y_3\,]\end{matrix} \qquad \mathbf{ep} = [\,ptype \;\; t \;\; E \;\; \nu\,]$$

If uniformly distributed loads are applied to the element, the ele-

ment load vector `fe` is computed. The input variable

$$eq = \begin{bmatrix} b_x \\ b_y \end{bmatrix}$$

containing loads per unit volume, b_x and b_y, is then given.

Theory

The element stiffness matrix $\mathbf{K}^e$ and the element load vector $\mathbf{f}_l^e$, stored in `Ke` and `fe`, respectively, are computed according to

$$\mathbf{K}^e = \int_A \mathbf{B}^{eT}\mathbf{D}\mathbf{B}^e t\, dA$$

$$\mathbf{f}_l^e = \int_A \mathbf{N}^{eT}\mathbf{b} t\, dA$$

where the constitutive matrix $\mathbf{D}$ is described in `hooke`, see Section *Material functions*, and the body force vector $\mathbf{b}$ is defined by `eq`.

The evaluation of the integrals for the Turner-Clough element is based on a displacement field $\mathbf{u}(x, y)$ built up of a bilinear displacement approximation superposed by bubble functions in order to create a linear stress field over the element. The displacement field is expressed in terms of the nodal variables $u_1, u_2, \ldots, u_8$ as

$$\mathbf{u}(x, y) = \mathbf{N}^e\mathbf{a}^e$$

where

$$\mathbf{u} = \begin{bmatrix} u_x \\ u_y \end{bmatrix} \quad \mathbf{N}^e = \begin{bmatrix} N_1^e & N_5^e & N_2^e & -N_5^e & N_3^e & N_5^e & N_4^e & -N_5^e \\ N_6^e & N_1^e & -N_6^e & N_2^e & N_6^e & N_3^e & -N_6^e & N_4^e \end{bmatrix} \quad \mathbf{a}^e =$$

With a local coordinate system located at the center of the element,

the element shape functions N_1^e–N_6^e are obtained as

$$
\begin{aligned}
N_1^e &= \frac{1}{4ab}(a-x)(b-y) \\
N_2^e &= \frac{1}{4ab}(a+x)(b-y) \\
N_3^e &= \frac{1}{4ab}(a+x)(b+y) \\
N_4^e &= \frac{1}{4ab}(a-x)(b+y) \\
N_5^e &= \frac{1}{8ab}\left[(b^2-y^2)+\nu(a^2-x^2)\right] \\
N_6^e &= \frac{1}{8ab}\left[(a^2-x^2)+\nu(b^2-y^2)\right]
\end{aligned}
$$

where

$$
a = \frac{1}{2}(x_3 - x_1) \qquad b = \frac{1}{2}(y_3 - y_1)
$$

The matrix $\mathbf{B}^e$ is obtained as

$$
\mathbf{B}^e = \tilde{\nabla}\mathbf{N}^e \qquad \text{where} \quad \tilde{\nabla} = \begin{bmatrix} \dfrac{\partial}{\partial x} & 0 \\ 0 & \dfrac{\partial}{\partial y} \\ \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial x} \end{bmatrix}
$$

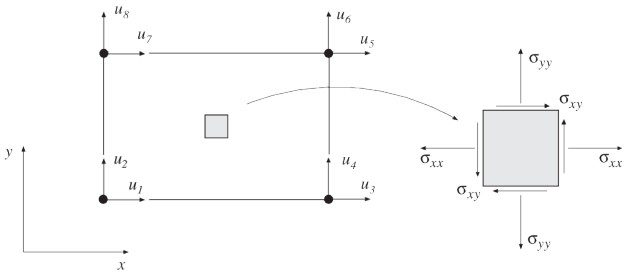
Evaluation of the integrals for the Turner-Clough element can be done either analytically or numerically by use of a 2×2 point Gauss integration. The element load vector $\mathbf{f}_l^e$ yields

$$
\mathbf{f}_l^e = abt \begin{bmatrix} b_x \\ b_y \\ b_x \\ b_y \\ b_x \\ b_y \\ b_x \\ b_y \end{bmatrix}
$$

9.1.9 plantcs

Purpose

Compute stresses and strains in a Turner-Clough element in plane strain or plane stress.



Syntax

```
[es, et] = plantcs(ex, ey, ep, ed)
```

Description

`plantcs` computes the stresses `es` and the strains `et` in a rectangular Turner-Clough element in plane strain or plane stress. The stress and strain components are computed at the center of the element.

The input variables **ex**, **ey**, and **ep** are defined in `plantce`. The vector **ed** contains the nodal displacements $\mathbf{a}^e$ of the element and is obtained by the function `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\, u_1 \;\; u_2 \;\; \ldots \;\; u_8 \,]$$

The output variables

$$\text{es} = \boldsymbol{\sigma}^T = [\, \sigma_{xx} \;\; \sigma_{yy} \;\; [\sigma_{zz}] \;\; \sigma_{xy} \;\; [\sigma_{xz}] \;\; [\sigma_{yz}] \,]$$

$$\text{et} = \boldsymbol{\varepsilon}^T = [\, \varepsilon_{xx} \;\; \varepsilon_{yy} \;\; [\varepsilon_{zz}] \;\; \gamma_{xy} \;\; [\gamma_{xz}] \;\; [\gamma_{yz}] \,]$$

contain the stress and strain components. The size of `es` and `et` follows the size of `D`. Note that for plane stress $\varepsilon_{zz} \neq 0$, and for plane strain $\sigma_{zz} \neq 0$.

Theory

The strains and stresses are computed according to

$$\boldsymbol{\varepsilon} = \mathbf{B}^e \, \mathbf{a}^e$$

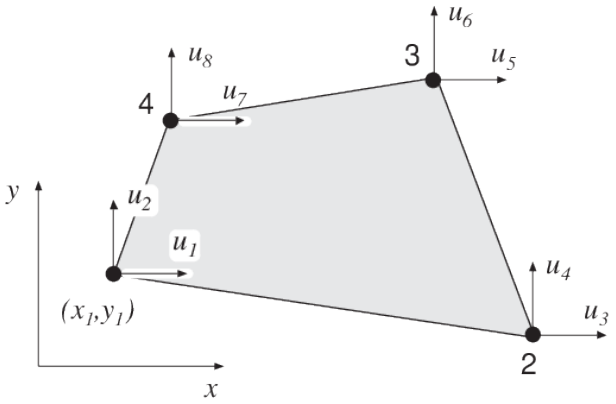
$$\boldsymbol{\sigma} = \mathbf{D}\,\boldsymbol{\varepsilon}$$

where the matrices $\mathbf{D}$, $\mathbf{B}^e$, and $\mathbf{a}^e$ are described in `plantce`, and where the evaluation point (x, y) is chosen to be at the center of the element.

9.1.10 plani4e

Purpose

Compute element matrices for a 4 node isoparametric element in plane strain or plane stress.



Syntax

```
Ke = plani4e(ex, ey, ep, D)
[Ke, fe] = plani4e(ex, ey, ep, D, eq)
```

Description

`plani4e` provides an element stiffness matrix `Ke` and an element load vector `fe` for a 4 node isoparametric element in plane strain or plane stress.

The element nodal coordinates $x_1, y_1, x_2, \ldots$ are supplied to the function by $\mathbf{ex}$ and $\mathbf{ey}$. The type of analysis $ptype$, the element thickness t, and the number of Gauss points n are supplied by $\mathbf{ep}$:

$ptype = 1$	plane stress	$(n \times n)$ integration points
$ptype = 2$	plane strain	$n = 1, 2, 3$

The material properties are supplied by the constitutive matrix D. Any arbitrary $\mathbf{D}$-matrix with dimensions from 3×3 to 6×6 may be given. For an isotropic elastic material the constitutive matrix can be formed by the function `hooke`, see Section *Material functions*.

$$\mathbf{ex} = [\, x_1 \;\; x_2 \;\; x_3 \;\; x_4 \,]$$
$$\mathbf{ey} = [\, y_1 \;\; y_2 \;\; y_3 \;\; y_4 \,]$$
$$\mathbf{ep} = [\, ptype \;\; t \;\; n \,]$$

$$\mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} \\ D_{21} & D_{22} & D_{23} \\ D_{31} & D_{32} & D_{33} \end{bmatrix} \quad \text{or} \quad \mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} & D_{14} & [D_{15}] & [\\ D_{21} & D_{22} & D_{23} & D_{24} & [D_{25}] & [\\ D_{31} & D_{32} & D_{33} & D_{34} & [D_{35}] & [\\ D_{41} & D_{42} & D_{43} & D_{44} & [D_{45}] & [\\ [D_{51}] & [D_{52}] & [D_{53}] & [D_{54}] & [D_{55}] & [\\ [D_{61}] & [D_{62}] & [D_{63}] & [D_{64}] & [D_{65}] & [\end{bmatrix}$$

If different $\mathbf{D}_i$ matrices are used in the Gauss points, these $\mathbf{D}_i$ matrices are stored in a global vector `D`. For numbering of the Gauss points, see `eci` in `plani4s`.

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_1 \\ \mathbf{D}_2 \\ \vdots \\ \mathbf{D}_{n^2} \end{bmatrix}$$

If uniformly distributed loads are applied to the element, the element load vector `fe` is computed. The input variable

$$\mathrm{eq} = \begin{bmatrix} b_x \\ b_y \end{bmatrix}$$

containing loads per unit volume, b_x and b_y, is then given.

Theory

The element stiffness matrix $\mathbf{K}^e$ and the element load vector $\mathbf{f}_l^e$, stored in `Ke` and `fe`, respectively, are computed according to

$$\mathbf{K}^e = \int_A \mathbf{B}^{eT}\mathbf{D}\mathbf{B}^e t \, dA$$
$$\mathbf{f}_l^e = \int_A \mathbf{N}^{eT}\mathbf{b} t \, dA$$

with the constitutive matrix $\mathbf{D}$ defined by `D`, and the body force vector $\mathbf{b}$ defined by `eq`.

The evaluation of the integrals for the isoparametric 4 node element is based on a displacement approximation $\mathbf{u}(\xi, \eta)$, expressed in a local coordinate system in terms of the nodal variables $u_1, u_2, \ldots, u_8$ as

$$\mathbf{u}(\xi, \eta) = \mathbf{N}^e \mathbf{a}^e$$

where

$$\mathbf{u} = \begin{bmatrix} u_x \\ u_y \end{bmatrix} \qquad \mathbf{N}^e = \begin{bmatrix} N_1^e & 0 & N_2^e & 0 & N_3^e & 0 & N_4^e & 0 \\ 0 & N_1^e & 0 & N_2^e & 0 & N_3^e & 0 & N_4^e \end{bmatrix} \qquad \mathbf{a}^e = \begin{bmatrix} u \\ u \\ \\ \\ u \end{bmatrix}$$

The element shape functions are given by

$$N_1^e = \frac{1}{4}(1-\xi)(1-\eta) \qquad N_2^e = \frac{1}{4}(1+\xi)(1-\eta)$$
$$N_3^e = \frac{1}{4}(1+\xi)(1+\eta) \qquad N_4^e = \frac{1}{4}(1-\xi)(1+\eta)$$

The matrix $\mathbf{B}^e$ is obtained as

$$\mathbf{B}^e = \tilde{\nabla}\mathbf{N}^e \qquad \text{where} \quad \tilde{\nabla} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix}$$

and

$$\begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix} = (\mathbf{J}^T)^{-1} \begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{bmatrix} \qquad \mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix}$$

If a larger $\mathbf{D}$-matrix than 3×3 is used for plane stress ($ptype = 1$), the $\mathbf{D}$-matrix is reduced to a 3×3 matrix by static condensation using $\sigma_{zz} = \sigma_{xz} = \sigma_{yz} = 0$. These stress components are connected with the rows 3, 5 and 6 in the D-matrix respectively.

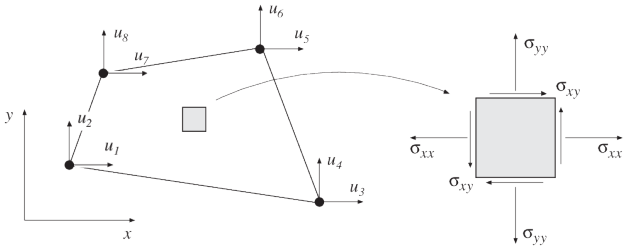
If a larger $\mathbf{D}$-matrix than 3×3 is used for plane strain ($ptype = 2$), the $\mathbf{D}$-matrix is reduced to a 3×3 matrix using $\varepsilon_{zz} = \gamma_{xz} = \gamma_{yz} = 0$. This implies that a 3×3 $\mathbf{D}$-matrix is created by the rows and the columns 1, 2 and 4 from the original D-matrix.

Evaluation of the integrals is done by Gauss integration.

9.1.11 plani4s

Purpose

Compute stresses and strains in a 4 node isoparametric element in plane strain or plane stress.



Syntax

```
[es,et,eci]=plani4s(ex,ey,ep,D,ed)
```

Description

`plani4s` computes stresses `es` and the strains `et` in a 4 node isoparametric element in plane strain or plane stress.

The input variables **ex**, **ey**, **ep** and the matrix **D** are defined in `plani4e`. The vector **ed** contains the nodal displacements $\mathbf{a}^e$ of the element and is obtained by the function `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\, u_1 \;\; u_2 \;\; \ldots \;\; u_8 \,]$$

The output variables

$$\text{es} = \boldsymbol{\sigma}^T = \begin{bmatrix} \sigma_{xx}^1 & \sigma_{yy}^1 & [\sigma_{zz}^1] & \sigma_{xy}^1 & [\sigma_{xz}^1] & [\sigma_{yz}^1] \\ \sigma_{xx}^2 & \sigma_{yy}^2 & [\sigma_{zz}^2] & \sigma_{xy}^2 & [\sigma_{xz}^2] & [\sigma_{yz}^2] \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \sigma_{xx}^{n^2} & \sigma_{yy}^{n^2} & [\sigma_{zz}^{n^2}] & \sigma_{xy}^{n^2} & [\sigma_{xz}^{n^2}] & [\sigma_{yz}^{n^2}] \end{bmatrix}$$

$$\text{et} = \boldsymbol{\varepsilon}^T = \begin{bmatrix} \varepsilon_{xx}^1 & \varepsilon_{yy}^1 & [\varepsilon_{zz}^1] & \gamma_{xy}^1 & [\gamma_{xz}^1] & [\gamma_{yz}^1] \\ \varepsilon_{xx}^2 & \varepsilon_{yy}^2 & [\varepsilon_{zz}^2] & \gamma_{xy}^2 & [\gamma_{xz}^2] & [\gamma_{yz}^2] \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \varepsilon_{xx}^{n^2} & \varepsilon_{yy}^{n^2} & [\varepsilon_{zz}^{n^2}] & \gamma_{xy}^{n^2} & [\gamma_{xz}^{n^2}] & [\gamma_{yz}^{n^2}] \end{bmatrix}$$

$$\mathbf{eci} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_{n^2} & y_{n^2} \end{bmatrix}$$

contain the stress and strain components, and the coordinates of the integration points. The index n denotes the number of integration points used within the element, cf. `plani4e`. The number of columns in `es` and `et` follows the size of `D`. Note that for plane stress $\varepsilon_{zz} \neq 0$, and for plane strain $\sigma_{zz} \neq 0$.

Theory

The strains and stresses are computed according to

$$\boldsymbol{\varepsilon} = \mathbf{B}^e \, \mathbf{a}^e$$

$$\boldsymbol{\sigma} = \mathbf{D} \, \boldsymbol{\varepsilon}$$

where the matrices $\mathbf{D}$, $\mathbf{B}^e$, and $\mathbf{a}^e$ are described in `plani4e`, and where the integration points are chosen as evaluation points.

9.1.12 plani4f

Purpose

Compute internal element force vector in a 4 node isoparametric element in plane strain or plane stress.

Syntax

```
ef = plani4f(ex, ey, ep, es)
```

Description

`plani4f` computes the internal element forces ef in a 4 node isoparametric element in plane strain or plane stress.

The input variables **ex**, **ey** and **ep** are defined in `plani4e`, and the input variable **es** is defined in `plani4s`.

The output variable

$$\text{ef} = \mathbf{f}_i^{eT} = [\, f_{i1} \; f_{i2} \; \ldots \; f_{i8} \,]$$

contains the components of the internal force vector.

Theory

The internal force vector is computed according to

$$\mathbf{f}_i^e = \int_A \mathbf{B}^{eT} \boldsymbol{\sigma} \; t \; dA$$

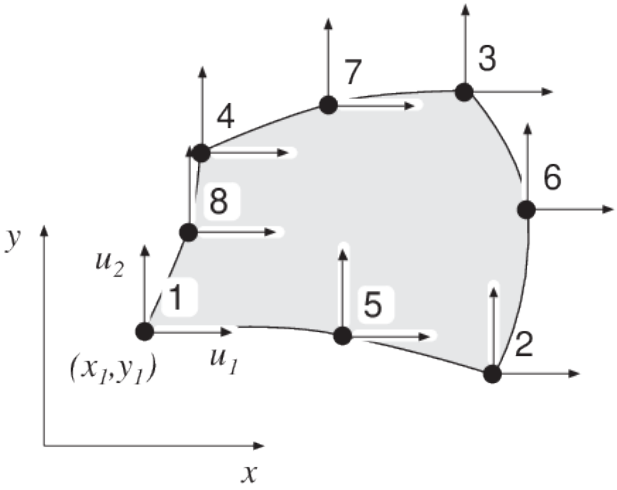
where the matrices $\mathbf{B}^e$ and $\boldsymbol{\sigma}$ are defined in `plani4e` and `plani4s`, respectively.

Evaluation of the integral is done by Gauss integration.

9.1.13 plani8e

Purpose

Compute element matrices for an 8 node isoparametric element in plane strain or plane stress.



Syntax

```
Ke = plani8e(ex, ey, ep, D)
[Ke, fe] = plani8e(ex, ey, ep, D, eq)
```

Description

`plani8e` provides an element stiffness matrix `Ke` and an element

load vector `fe` for an 8 node isoparametric element in plane strain or plane stress.

The element nodal coordinates $x_1, y_1, x_2, \ldots$ are supplied to the function by $\mathbf{ex}$ and $\mathbf{ey}$. The type of analysis $ptype$, the element thickness t, and the number of Gauss points n are supplied by $\mathbf{ep}$:

$$\begin{array}{lll} ptype = 1 & \text{plane stress} & (n \times n) \text{ integration points} \\ ptype = 2 & \text{plane strain} & n = 1, 2, 3 \end{array}$$

The material properties are supplied by the constitutive matrix $\mathbf{D}$. Any arbitrary $\mathbf{D}$-matrix with dimensions from 3×3 to 6×6 may be given. For an isotropic elastic material the constitutive matrix can be formed by the function `hooke`.

$$\begin{aligned} \mathbf{ex} &= [x_1, x_2, \ldots, x_8] \\ \mathbf{ey} &= [y_1, y_2, \ldots, y_8] \\ \mathbf{ep} &= [ptype, t, n] \end{aligned}$$

$$\mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} \\ D_{21} & D_{22} & D_{23} \\ D_{31} & D_{32} & D_{33} \end{bmatrix} \quad \text{or} \quad \mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} & D_{14} & [D_{15}] & [\\ D_{21} & D_{22} & D_{23} & D_{24} & [D_{25}] & [\\ D_{31} & D_{32} & D_{33} & D_{34} & [D_{35}] & [\\ D_{41} & D_{42} & D_{43} & D_{44} & [D_{45}] & [\\ [D_{51}] & [D_{52}] & [D_{53}] & [D_{54}] & [D_{55}] & [\\ [D_{61}] & [D_{62}] & [D_{63}] & [D_{64}] & [D_{65}] & [\end{bmatrix}$$

If different $\mathbf{D}_i$ matrices are used in the Gauss points these $\mathbf{D}_i$ matrices are stored in a global vector `D`. For numbering of the Gauss points, see `eci` in `plani8s`.

$$D = \begin{bmatrix} D_1 \\ D_2 \\ \vdots \\ D_{n^2} \end{bmatrix}$$

If uniformly distributed loads are applied to the element, the element load vector `fe` is computed. The input variable

$$\mathbf{eq} = \begin{bmatrix} b_x \\ b_y \end{bmatrix}$$

containing loads per unit volume, b_x and b_y, is then given.

Theory

The element stiffness matrix $\mathbf{K}^e$ and the element load vector $\mathbf{f}_l^e$, stored in `Ke` and `fe`, respectively, are computed according to

$$\mathbf{K}^e = \int_A \mathbf{B}^{eT}\mathbf{D}\mathbf{B}^e t\, dA$$

$$\mathbf{f}_l^e = \int_A \mathbf{N}^{eT}\mathbf{b} t\, dA$$

with the constitutive matrix $\mathbf{D}$ defined by `D`, and the body force vector $\mathbf{b}$ defined by `eq`.

The evaluation of the integrals for the isoparametric 8 node element is based on a displacement approximation $\mathbf{u}(\xi,\eta)$, expressed in a local coordinate system in terms of the nodal variables $u_1, u_2, \ldots, u_{16}$ as

$$\mathbf{u}(\xi,\eta) = \mathbf{N}^e\mathbf{a}^e$$

where

$$\mathbf{u} = \begin{bmatrix} u_x \\ u_y \end{bmatrix}, \quad \mathbf{N}^e = \begin{bmatrix} N_1^e & 0 & N_2^e & 0 & \cdots & N_8^e & 0 \\ 0 & N_1^e & 0 & N_2^e & \cdots & 0 & N_8^e \end{bmatrix}, \quad \mathbf{a}^e = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{16} \end{bmatrix}$$

The element shape functions are given by

$$N_1^e = -\frac{1}{4}(1-\xi)(1-\eta)(1+\xi+\eta) \qquad N_5^e = \frac{1}{2}(1-\xi^2)(1-\eta)$$

$$N_2^e = -\frac{1}{4}(1+\xi)(1-\eta)(1-\xi+\eta) \qquad N_6^e = \frac{1}{2}(1+\xi)(1-\eta^2)$$

$$N_3^e = -\frac{1}{4}(1+\xi)(1+\eta)(1-\xi-\eta) \qquad N_7^e = \frac{1}{2}(1-\xi^2)(1+\eta)$$

$$N_4^e = -\frac{1}{4}(1-\xi)(1+\eta)(1+\xi-\eta) \qquad N_8^e = \frac{1}{2}(1-\xi)(1-\eta^2)$$

The matrix $\mathbf{B}^e$ is obtained as

$$\mathbf{B}^e = \tilde{\nabla}\mathbf{N}^e \qquad \text{where} \quad \tilde{\nabla} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix}$$

and where

$$\begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{bmatrix} = (\mathbf{J}^T)^{-1} \begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{bmatrix} \qquad \mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{bmatrix}$$

If a larger $\mathbf{D}$-matrix than 3×3 is used for plane stress ($ptype = 1$), the $\mathbf{D}$-matrix is reduced to a 3×3 matrix by static condensation, setting

$$\sigma_{zz} = 0$$
$$\sigma_{xz} = 0$$
$$\sigma_{yz} = 0$$

These stress components correspond to rows and columns 3, 5, and 6 in the D-matrix, respectively.

If a larger $\mathbf{D}$-matrix than 3×3 is used for plane strain ($ptype = 2$), the $\mathbf{D}$-matrix is reduced to a 3×3 matrix by setting

$$\varepsilon_{zz} = 0$$
$$\gamma_{xz} = 0$$
$$\gamma_{yz} = 0$$

This means that the resulting 3×3 $\mathbf{D}$-matrix is formed by extracting rows and columns 1, 2, and 4 from the original D-matrix.

Evaluation of the integrals is done by Gauss integration.

9.1.14 plani8s

Purpose

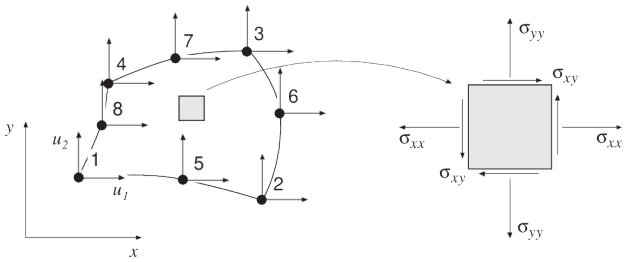
Compute stresses and strains in an 8 node isoparametric element in plane strain or plane stress.

Syntax

```
[es,et,eci]=plani8s(ex,ey,ep,D,ed)
```

Description

`plani8s` computes stresses `es` and the strains `et` in an 8 node isoparametric element in plane strain or plane stress.



The input variables **ex**, **ey**, **ep** and the matrix $\mathbf{D}$ are defined in `plani8e`. The vector **ed** contains the nodal displacements $\mathbf{a}^e$ of the element and is obtained by the function `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\, u_1 \;\; u_2 \;\; \ldots \;\; u_{16} \,]$$

The output variables

$$\text{es} = \boldsymbol{\sigma}^T = \begin{bmatrix} \sigma_{xx}^1 & \sigma_{yy}^1 & [\sigma_{zz}^1] & \sigma_{xy}^1 & [\sigma_{xz}^1] & [\sigma_{yz}^1] \\ \sigma_{xx}^2 & \sigma_{yy}^2 & [\sigma_{zz}^2] & \sigma_{xy}^2 & [\sigma_{xz}^2] & [\sigma_{yz}^2] \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \sigma_{xx}^{n^2} & \sigma_{yy}^{n^2} & [\sigma_{zz}^{n^2}] & \sigma_{xy}^{n^2} & [\sigma_{xz}^{n^2}] & [\sigma_{yz}^{n^2}] \end{bmatrix}$$

$$\text{et} = \boldsymbol{\varepsilon}^T = \begin{bmatrix} \varepsilon_{xx}^1 & \varepsilon_{yy}^1 & [\varepsilon_{zz}^1] & \gamma_{xy}^1 & [\gamma_{xz}^1] & [\gamma_{yz}^1] \\ \varepsilon_{xx}^2 & \varepsilon_{yy}^2 & [\varepsilon_{zz}^2] & \gamma_{xy}^2 & [\gamma_{xz}^2] & [\gamma_{yz}^2] \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \varepsilon_{xx}^{n^2} & \varepsilon_{yy}^{n^2} & [\varepsilon_{zz}^{n^2}] & \gamma_{xy}^{n^2} & [\gamma_{xz}^{n^2}] & [\gamma_{yz}^{n^2}] \end{bmatrix} \qquad \mathbf{eci} = \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \\ x_{n^2} & y_{n^2} \end{bmatrix}$$

contain the stress and strain components, and the coordinates of the integration points. The index n denotes the number of integration points used within the element, cf. `plani8e`. The number of columns in `es` and `et` follows the size of `D`. Note that for plane stress $\varepsilon_{zz} \neq 0$, and for plane strain $\sigma_{zz} \neq 0$.

Theory

The strains and stresses are computed according to

$$\boldsymbol{\varepsilon} = \mathbf{B}^e \, \mathbf{a}^e$$

$$\boldsymbol{\sigma} = \mathbf{D} \, \boldsymbol{\varepsilon}$$

where the matrices $\mathbf{D}$, $\mathbf{B}^e$, and $\mathbf{a}^e$ are described in `plani8e`, and where the integration points are chosen as evaluation points.

9.1.15 plani8f

Purpose

Compute internal element force vector in an 8 node isoparametric element in plane strain or plane stress.

Syntax

```
ef = plani8f(ex, ey, ep, es)
```

Description

`plani8f` computes the internal element forces ef in an 8 node isoparametric element in plane strain or plane stress.

The input variables **ex**, **ey** and **ep** are defined in `plani8e`, and the input variable **es** is defined in `plani8s`.

The output variable

$$\text{ef} = \mathbf{f}_i^{eT} = [\, f_{i1} \;\; f_{i2} \;\; \ldots \;\; f_{i16} \,]$$

contains the components of the internal force vector.

Theory

The internal force vector is computed according to

$$\mathbf{f}_i^e = \int_A \mathbf{B}^{eT} \boldsymbol{\sigma} \; t \; dA$$

where the matrices $\mathbf{B}^e$ and $\boldsymbol{\sigma}$ are defined in `plani8e` and `plani8s`, respectively.

Evaluation of the integral is done by Gauss integration.

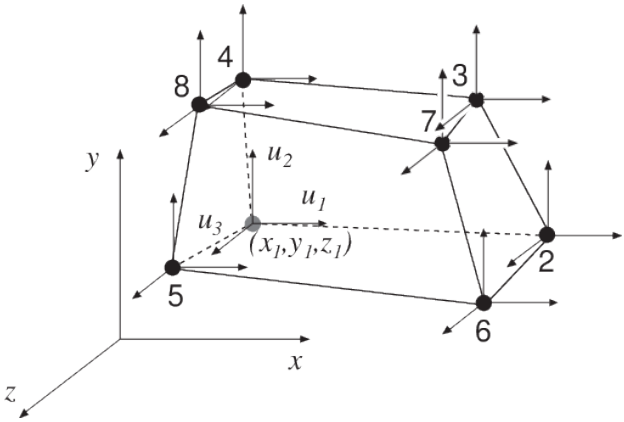
9.2 3D Solid Functions

9.2.1 soli8e

Purpose

Compute element matrices for an 8 node isoparametric solid element.

Syntax




```
Ke = soli8e(ex, ey, ez, ep, D)
[Ke, fe] = soli8e(ex, ey, ez, ep, D, eq)
```

Description

`soli8e` provides an element stiffness matrix `Ke` and an element load vector `fe` for an 8 node isoparametric solid element.

The element nodal coordinates x_1, y_1, z_1, x_2 etc. are supplied to the function by **ex**, **ey** and **ez**, and the number of Gauss points n are supplied by **ep**.

$(n \times n \times n)$ integration points, $n = 1, 2, 3$

The material properties are supplied by the constitutive matrix `D`. Any arbitrary **D**-matrix with dimensions (6×6) may be given. For an isotropic elastic material the constitutive matrix can be formed by the function `hooke`, see Section *Material functions*.

$$\mathbf{ex} = [\, x_1,\, x_2,\, \ldots,\, x_8 \,]$$
$$\mathbf{ey} = [\, y_1,\, y_2,\, \ldots,\, y_8 \,]$$
$$\mathbf{ez} = [\, z_1,\, z_2,\, \ldots,\, z_8 \,]$$

$$\mathbf{ep} = [\, n \,]$$

$$\mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & \cdots & D_{16} \\ D_{21} & D_{22} & \cdots & D_{26} \\ \vdots & \vdots & \ddots & \vdots \\ D_{61} & D_{62} & \cdots & D_{66} \end{bmatrix}$$

If different $\mathbf{D}_i$ matrices are used in the Gauss points these $\mathbf{D}_i$ matrices are stored in a global vector `D`. For numbering of the Gauss points, see `eci` in `soli8s`.

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_1 \\ \mathbf{D}_2 \\ \vdots \\ \mathbf{D}_{n^3} \end{bmatrix}$$

If uniformly distributed loads are applied to the element, the element load vector `fe` is computed. The input variable

$$\text{eq} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}$$

containing loads per unit volume, b_x, b_y, and b_z, is then given.

Theory

The element stiffness matrix $\mathbf{K}^e$ and the element load vector $\mathbf{f}_l^e$, stored in `Ke` and `fe`, respectively, are computed according to

$$\mathbf{K}^e = \int_V \mathbf{B}^{eT}\mathbf{D}\mathbf{B}^e \, dV$$

$$\mathbf{f}_l^e = \int_V \mathbf{N}^{eT}\mathbf{b} \, dV$$

with the constitutive matrix $\mathbf{D}$ defined by `D`, and the body force vector $\mathbf{b}$ defined by `eq`.

The evaluation of the integrals for the isoparametric 8 node solid element is based on a displacement approximation $\mathbf{u}(\xi, \eta, \zeta)$, expressed in a local coordinate system in terms of the nodal variables

$u_1, u_2, \ldots, u_{24}$ as

$$\mathbf{u}(\xi,\eta$$

$$\mathbf{u} = \begin{bmatrix} u_x \\ u_y \\ u_z \end{bmatrix} \quad \mathbf{N}^e = \begin{bmatrix} N_1^e & 0 & 0 & N_2^e & 0 & 0 & \ldots & N_8^e & 0 & 0 \\ 0 & N_1^e & 0 & 0 & N_2^e & 0 & \ldots & 0 & N_8^e & 0 \\ 0 & 0 & N_1^e & 0 & 0 & N_2^e & \ldots & 0 & 0 & N_8^e \end{bmatrix} \quad \mathbf{a}$$

The element shape functions are given by

$$N_1^e = \frac{1}{8}(1-\xi)(1-\eta)(1-\zeta) \quad N_5^e = \frac{1}{8}(1-\xi)(1-\eta)(1+\zeta)$$
$$N_2^e = \frac{1}{8}(1+\xi)(1-\eta)(1-\zeta) \quad N_6^e = \frac{1}{8}(1+\xi)(1-\eta)(1+\zeta)$$
$$N_3^e = \frac{1}{8}(1+\xi)(1+\eta)(1-\zeta) \quad N_7^e = \frac{1}{8}(1+\xi)(1+\eta)(1+\zeta)$$
$$N_4^e = \frac{1}{8}(1-\xi)(1+\eta)(1-\zeta) \quad N_8^e = \frac{1}{8}(1-\xi)(1+\eta)(1+\zeta)$$

The $\mathbf{B}^e$-matrix is obtained as

$$\mathbf{B}^e = \tilde{\nabla}\mathbf{N}^e$$

where

$$\tilde{\nabla} = \begin{bmatrix} \frac{\partial}{\partial x} & 0 & 0 \\ 0 & \frac{\partial}{\partial y} & 0 \\ 0 & 0 & \frac{\partial}{\partial z} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} & 0 \\ \frac{\partial}{\partial z} & 0 & \frac{\partial}{\partial x} \\ 0 & \frac{\partial}{\partial z} & \frac{\partial}{\partial y} \end{bmatrix} \qquad \begin{bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \\ \frac{\partial}{\partial z} \end{bmatrix} = (\mathbf{J}^T)^{-1} \begin{bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \\ \frac{\partial}{\partial \zeta} \end{bmatrix}$$

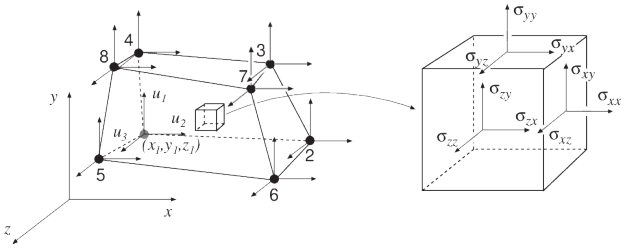
$$\mathbf{J} = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} & \frac{\partial x}{\partial \zeta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} & \frac{\partial y}{\partial \zeta} \\ \frac{\partial z}{\partial \xi} & \frac{\partial z}{\partial \eta} & \frac{\partial z}{\partial \zeta} \end{bmatrix}$$

Evaluation of the integrals is done by Gauss integration.

9.2.2 soli8s

Purpose

Compute stresses and strains in an 8 node isoparametric solid element.



Syntax

```
[es,et,eci]=soli8s(ex,ey,ez,ep,D,ed)
```

Description

`soli8s` computes stresses `es` and the strains `et` in an 8 node isoparametric solid element.

The input variables **ex**, **ey**, **ez**, **ep** and the matrix $\mathbf{D}$ are defined in `soli8e`. The vector **ed** contains the nodal displacements $\mathbf{a}^e$ of the element and is obtained by the function `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\ u_1\ \ u_2\ \ \ldots\ \ u_{24}\ ]$$

The output variables

$$\text{es} = \boldsymbol{\sigma}^T = \begin{bmatrix} \sigma_{xx}^1 & \sigma_{yy}^1 & \sigma_{zz}^1 & \sigma_{xy}^1 & \sigma_{xz}^1 & \sigma_{yz}^1 \\ \sigma_{xx}^2 & \sigma_{yy}^2 & \sigma_{zz}^2 & \sigma_{xy}^2 & \sigma_{xz}^2 & \sigma_{yz}^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \sigma_{xx}^{n^3} & \sigma_{yy}^{n^3} & \sigma_{zz}^{n^3} & \sigma_{xy}^{n^3} & \sigma_{xz}^{n^3} & \sigma_{yz}^{n^3} \end{bmatrix}$$

$$\text{et} = \boldsymbol{\varepsilon}^T = \begin{bmatrix} \varepsilon_{xx}^1 & \varepsilon_{yy}^1 & \varepsilon_{zz}^1 & \gamma_{xy}^1 & \gamma_{xz}^1 & \gamma_{yz}^1 \\ \varepsilon_{xx}^2 & \varepsilon_{yy}^2 & \varepsilon_{zz}^2 & \gamma_{xy}^2 & \gamma_{xz}^2 & \gamma_{yz}^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \varepsilon_{xx}^{n^3} & \varepsilon_{yy}^{n^3} & \varepsilon_{zz}^{n^3} & \gamma_{xy}^{n^3} & \gamma_{xz}^{n^3} & \gamma_{yz}^{n^3} \end{bmatrix} \qquad \mathbf{eci} = \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ \vdots & \vdots & \vdots \\ x_{n^3} & y_{n^3} & z_{n^3} \end{bmatrix}$$

contain the stress and strain components, and the coordinates of the integration points. The index n denotes the number of integration points used within the element, cf. `soli8e`.

Theory

The strains and stresses are computed according to

$$\boldsymbol{\varepsilon} = \mathbf{B}^e \mathbf{a}^e$$

$$\boldsymbol{\sigma} = \mathbf{D}\boldsymbol{\varepsilon}$$

where the matrices $\mathbf{D}$, $\mathbf{B}^e$, and $\mathbf{a}^e$ are described in `soli8e`, and where the integration points are chosen as evaluation points.

9.2.3 soli8f

Purpose

Compute internal element force vector in an 8 node isoparametric solid element.

Syntax

```
ef = soli8f(ex, ey, ez, ep, es)
```

Description

`soli8f` computes the internal element forces ef in an 8 node isoparametric solid element.

The input variables $\mathbf{ex}$, $\mathbf{ey}$, $\mathbf{ez}$ and $\mathbf{ep}$ are defined in `soli8e`, and the input variable $\mathbf{es}$ is defined in `soli8s`.

The output variable

$$\mathrm{ef} = \mathbf{f}_i^{eT} = [\, f_{i1} \;\; f_{i2} \;\; \ldots \;\; f_{i24} \,]$$

contains the components of the internal force vector.

Theory

The internal force vector is computed according to

$$\mathbf{f}_i^e = \int_V \mathbf{B}^{eT} \boldsymbol{\sigma} \; dV$$

where the matrices $\mathbf{B}$ and $\boldsymbol{\sigma}$ are defined in `soli8e` and `soli8s`, respectively.

Evaluation of the integral is done by Gauss integration.

CHAPTER

TEN

BEAM ELEMENT FUNCTIONS

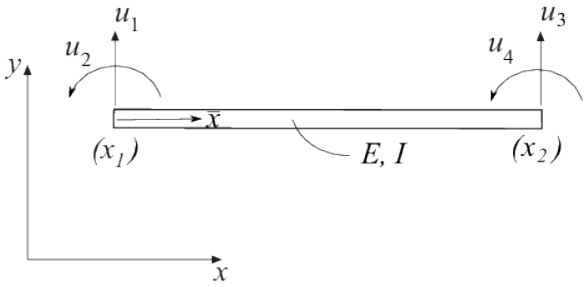
Beam elements are available for one, two, and three dimensional linear static analysis. Two dimensional beam elements for nonlinear geometric and dynamic analysis are also available.

10.1 1D beam elements

10.1.1 beam1e

Purpose

Compute element stiffness matrix for a one dimensional beam element.



Syntax

```
Ke = beam1e(ex, ep)
[Ke, fe] = beam1e(ex, ep, eq)
```

Description

`beam1e` provides the global element stiffness matrix `Ke` for a one dimensional beam element.

The input variables

`ex = [x1 x2]`

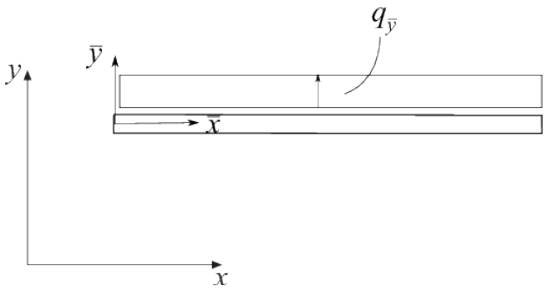
`ep = [E I]`

supply the element nodal coordinates x_1 and x_2, the modulus of elasticity E and the moment of inertia I.

The element load vector `fe` can also be computed if a uniformly distributed load is applied to the element. The optional input variable

`eq = [q_ybar]`

then contains the distributed load per unit length, $q_{\bar{y}}$.



Theory

The element stiffness matrix $\bar{\mathbf{K}}^e$, stored in `Ke`, is computed according to

$$\bar{\mathbf{K}}^e = \frac{D_{EI}}{L^3} \begin{bmatrix} 12 & 6L & -12 & 6L \\ 6L & 4L^2 & -6L & 2L^2 \\ -12 & -6L & 12 & -6L \\ 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}$$

where the bending stiffness D_{EI} and the length L are given by

$$D_{EI} = EI; \quad L = x_2 - x_1$$

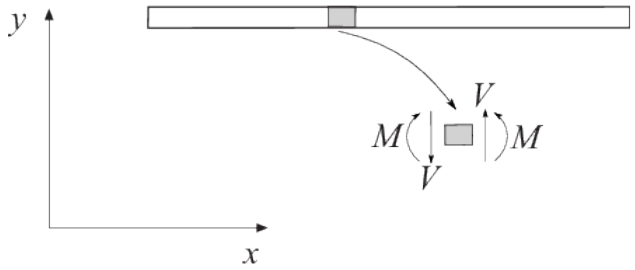
The element loads $\bar{\mathbf{f}}_l^e$ stored in the variable `fe` are computed according to

$$\bar{\mathbf{f}}_l^e = q_{\bar{y}} \begin{bmatrix} \frac{L}{2} \\ \frac{L^2}{12} \\ \frac{L}{2} \\ -\frac{L^2}{12} \end{bmatrix}$$

10.1.2 beam1s

Purpose

Compute section forces in a one dimensional beam element.



Syntax

```
es = beam1s(ex, ep, ed)
es = beam1s(ex, ep, ed, eq)
[es, edi, eci] = beam1s(ex, ep, ed, eq, n)
```

Description

`beam1s` computes the section forces and displacements in local directions along the beam element `beam1e`.

The input variables `ex`, `ep` and `eq` are defined in `beam1e`, and the element displacements, stored in `ed`, are obtained by the function

`extract`. If distributed loads are applied to the element, the variable `eq` must be included. The number of evaluation points for section forces and displacements are determined by `n`. If `n` is omitted, only the ends of the beam are evaluated.

The output variables

$$\text{es} = \begin{bmatrix} V(0) & M(0) \\ V(\bar{x}_2) & M(\bar{x}_2) \\ \vdots & \vdots \\ V(\bar{x}_{n-1}) & M(\bar{x}_{n-1}) \\ V(L) & M(L) \end{bmatrix} \qquad \text{edi} = \begin{bmatrix} v(0) \\ v(\bar{x}_2) \\ \vdots \\ v(\bar{x}_{n-1}) \\ v(L) \end{bmatrix} \qquad \text{eci} = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

contain the section forces, the displacements, and the evaluation points on the local $\bar{x}$-axis. L is the length of the beam element.

Theory

The nodal displacements in local coordinates are given by

$$\bar{\mathbf{a}}^e = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_4 \end{bmatrix}$$

where the transpose of $\mathbf{a}^e$ is stored in `ed`.

The displacement $v(\bar{x})$, the bending moment $M(\bar{x})$ and the shear force $V(\bar{x})$ are computed from

$$v(\bar{x}) = \mathbf{N}\bar{\mathbf{a}}^e + v_p(\bar{x})$$

$$M(\bar{x}) = D_{EI}\mathbf{B}\bar{\mathbf{a}}^e + M_p(\bar{x})$$

$$V(\bar{x}) = -D_{EI}\frac{d\mathbf{B}}{dx}\bar{\mathbf{a}}^e + V_p(\bar{x})$$

where

$$\mathbf{N} = \begin{bmatrix} 1 & \bar{x} & \bar{x}^2 & \bar{x}^3 \end{bmatrix} \mathbf{C}^{-1}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 2 & 6\bar{x} \end{bmatrix} \mathbf{C}^{-1}$$

$$\frac{d\mathbf{B}}{dx} = \begin{bmatrix} 0 & 0 & 0 & 6 \end{bmatrix} \mathbf{C}^{-1}$$

$$v_p(\bar{x}) = \frac{q_{\bar{y}}}{D_{EI}} \left(\frac{\bar{x}^4}{24} - \frac{L\bar{x}^3}{12} + \frac{L^2\bar{x}^2}{24} \right)$$

$$M_p(\bar{x}) = q_{\bar{y}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} + \frac{L^2}{12} \right)$$

$$V_p(\bar{x}) = -q_{\bar{y}} \left(\bar{x} - \frac{L}{2} \right)$$

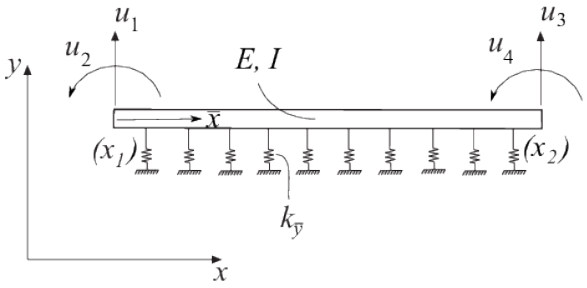
in which D_{EI}, L, and $q_{\bar{y}}$ are defined in `beam1e` and

$$\mathbf{C}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{3}{L^2} & -\frac{2}{L} & \frac{3}{L^2} & -\frac{1}{L} \\ \frac{2}{L^3} & \frac{1}{L^2} & -\frac{2}{L^3} & \frac{1}{L^2} \end{bmatrix}$$

10.1.3 beam1we

Purpose

Compute element stiffness matrix for a one dimensional beam element on elastic support.



Syntax

```
Ke = beam1we(ex, ep)
[Ke, fe] = beam1we(ex, ep, eq)
```

Description

beam1we provides the global element stiffness matrix **Ke** for a one dimensional beam element with elastic support.

The input variables

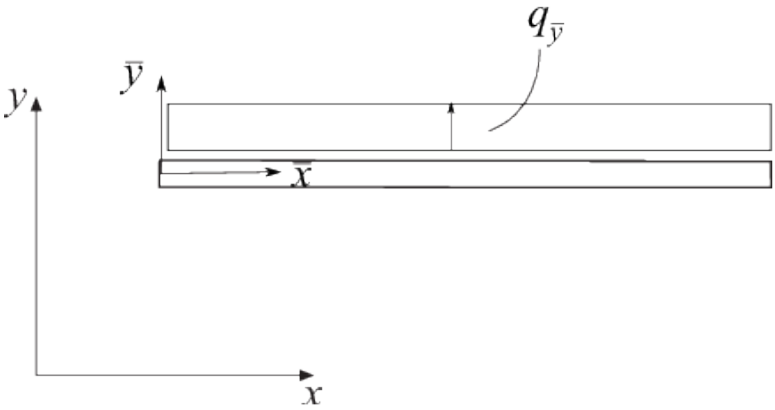
$$\text{ex} = [x_1 \;\; x_2] \qquad \text{ep} = [E \;\; I \;\; k_{\bar{y}}]$$

supply the element nodal coordinates x_1 and x_2, the modulus of elasticity E, the moment of inertia I, and the spring stiffness in the transverse direction $k_{\bar{y}}$.

The element load vector **fe** can also be computed if a uniformly distributed load is applied to the element. The optional input variable

$$\text{eq} = \left[q_{\bar{y}}\right]$$

contains the distributed load per unit length, $q_{\bar{y}}$.



Theory

The element stiffness matrix $\bar{\mathbf{K}}^e$, stored in **Ke**, is computed according to

$$\bar{\mathbf{K}}^e = \bar{\mathbf{K}}_0^e + \bar{\mathbf{K}}_s^e$$

where

$$\bar{\mathbf{K}}_0^e = \frac{D_{EI}}{L^3} \begin{bmatrix} 12 & 6L & -12 & 6L \\ 6L & 4L^2 & -6L & 2L^2 \\ -12 & -6L & 12 & -6L \\ 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}$$

and

$$\bar{\mathbf{K}}_s^e = \frac{k_{\bar{y}} L}{420} \begin{bmatrix} 156 & 22L & 54 & -13L \\ 22L & 4L^2 & 13L & -3L^2 \\ 54 & 13L & 156 & -22L \\ -13L & -3L^2 & -22L & 4L^2 \end{bmatrix}$$

where the bending stiffness D_{EI} and the length L are given by

$$D_{EI} = EI \qquad L = x_2 - x_1$$

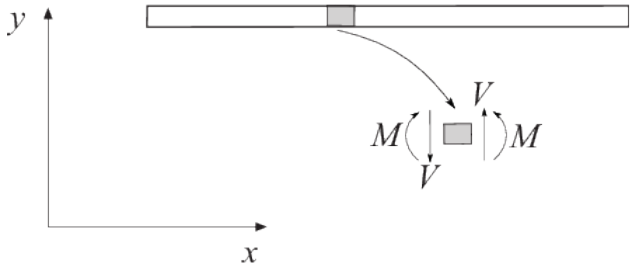
The element loads $\bar{\mathbf{f}}_l^e$ stored in the variable **fe** are computed according to

$$\bar{\mathbf{f}}_l^e = q_{\bar{y}} \begin{bmatrix} \frac{L}{2} \\ \frac{L^2}{12} \\ \frac{L}{2} \\ -\frac{L^2}{12} \end{bmatrix}$$

10.1.4 beam1ws

Purpose

Compute section forces in a one dimensional beam element with elastic support.



Syntax

```
es = beam1ws(ex, ep, ed)
es = beam1ws(ex, ep, ed, eq)
[es, edi, eci] = beam1ws(ex, ep, ed, eq, n)
```

Description

`beam1ws` computes the section forces and displacements in local directions along the beam element `beam1we`.

The input variables `ex`, `ep` and `eq` are defined in `beam1we`, and the element displacements, stored in `ed`, are obtained by the function `extract`. If distributed loads are applied to the element, the variable `eq` must be included. The number of evaluation points for section forces and displacements are determined by `n`. If `n` is omitted, only the ends of the beam are evaluated.

The output variables

$$\text{es} = \begin{bmatrix} V(0) & M(0) \\ V(\bar{x}_2) & M(\bar{x}_2) \\ \vdots & \vdots \\ V(\bar{x}_{n-1}) & M(\bar{x}_{n-1}) \\ V(L) & M(L) \end{bmatrix}$$

$$\text{edi} = \begin{bmatrix} v(0) \\ v(\bar{x}_2) \\ \vdots \\ v(\bar{x}_{n-1}) \\ v(L) \end{bmatrix}$$

$$\text{eci} = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

contain the section forces, the displacements, and the evaluation points on the local $\bar{x}$-axis. L is the length of the beam element.

Theory

The nodal displacements in local coordinates are given by

$$\bar{\mathbf{a}}^e = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_4 \end{bmatrix}$$

where the transpose of $\mathbf{a}^e$ is stored in `ed`.

The displacement $v(\bar{x})$, the bending moment $M(\bar{x})$ and the shear force $V(\bar{x})$ are computed from

$$v(\bar{x}) = \mathbf{N}\bar{\mathbf{a}}^e + v_p(\bar{x})$$

$$M(\bar{x}) = D_{EI}\mathbf{B}\bar{\mathbf{a}}^e + M_p(\bar{x})$$

$$V(\bar{x}) = -D_{EI}\frac{d\mathbf{B}}{dx}\bar{\mathbf{a}}^e + V_p(\bar{x})$$

where

$$\mathbf{N} = \begin{bmatrix} 1 & \bar{x} & \bar{x}^2 & \bar{x}^3 \end{bmatrix} \mathbf{C}^{-1}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 2 & 6\bar{x} \end{bmatrix} \mathbf{C}^{-1}$$

$$\frac{d\mathbf{B}}{dx} = \begin{bmatrix} 0 & 0 & 0 & 6 \end{bmatrix} \mathbf{C}^{-1}$$

$$v_p(\bar{x}) = -\frac{k_{\bar{y}}}{D_{EI}} \begin{bmatrix} \frac{\bar{x}^4 - 2L\bar{x}^3 + L^2\bar{x}^2}{24} \\ \frac{\bar{x}^5 - 3L^2\bar{x}^3 + 2L^3\bar{x}^2}{120} \\ \frac{\bar{x}^6 - 4L^3\bar{x}^3 + 3L^4\bar{x}^2}{360} \\ \frac{\bar{x}^7 - 5L^4\bar{x}^3 + 4L^5\bar{x}^2}{840} \end{bmatrix}^T \mathbf{C}^{-1}\bar{\mathbf{a}}^e + \frac{q_{\bar{y}}}{D_{EI}}\left(\frac{\bar{x}^4}{24} - \frac{L\bar{x}^3}{12} + \frac{L^2\bar{x}^2}{24}\right)$$

$$M_p(\bar{x}) = -k_{\bar{y}} \begin{bmatrix} \frac{6\bar{x}^2 - 6L\bar{x} + L^2}{12} \\ \frac{10\bar{x}^3 - 9L^2\bar{x} + 2L^3}{60} \\ \frac{5\bar{x}^4 - 4L^3\bar{x} + L^4}{60} \\ \frac{21\bar{x}^5 - 15L^4\bar{x} + 4L^5}{420} \end{bmatrix}^T \mathbf{C}^{-1}\bar{\mathbf{a}}^e + q_{\bar{y}}\left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} + \frac{L^2}{12}\right)$$

$$V_p(\bar{x}) = k_{\bar{y}} \begin{bmatrix} \frac{2\bar{x} - L}{2} \\ \frac{10\bar{x}^2 - 3L^2}{20} \\ \frac{5\bar{x}^3 - L^3}{15} \\ \frac{7\bar{x}^4 - L^4}{28} \end{bmatrix}^T \mathbf{C}^{-1}\bar{\mathbf{a}}^e - q_{\bar{y}}\left(\bar{x} - \frac{L}{2}\right)$$

in which D_{EI}, $k_{\bar{y}}$, L, and $q_{\bar{y}}$ are defined in `beam1we` and

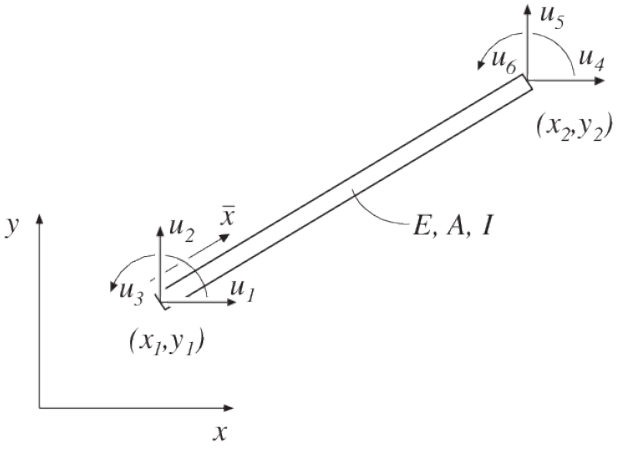
$$\mathbf{C}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{3}{L^2} & -\frac{2}{L} & \frac{3}{L^2} & -\frac{1}{L} \\ \frac{2}{L^3} & \frac{1}{L^2} & -\frac{2}{L^3} & \frac{1}{L^2} \end{bmatrix}$$

10.2 2D beam elements

10.2.1 beam2e

Purpose

Compute element stiffness matrix for a two-dimensional beam element.



Syntax

```
Ke = beam2e(ex, ey, ep)
[Ke, fe] = beam2e(ex, ey, ep, eq)
```

Description

beam2e provides the global element stiffness matrix **Ke** for a two-dimensional beam element.

The input variables:

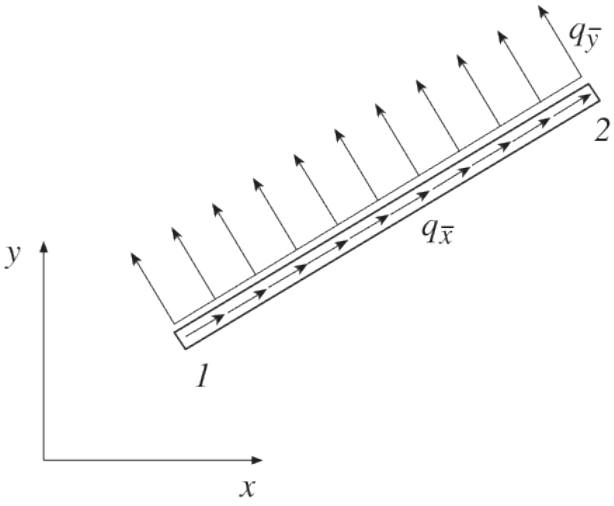
$$\text{ex} = [x_1, x_2]$$
$$\text{ey} = [y_1, y_2]$$
$$\text{ep} = [E, A, I]$$

supply the element nodal coordinates x_1, y_1, x_2, and y_2, the modulus of elasticity E, the cross-section area A, and the moment of inertia I.

The element load vector **fe** can also be computed if a uniformly distributed transverse load is applied to the element. The optional input variable:

$$\text{eq} = [q_{\bar{x}}, q_{\bar{y}}]$$

contains the distributed loads per unit length, $q_{\bar{x}}$ and $q_{\bar{y}}$.



Theory

The element stiffness matrix $\mathbf{K}^e$, stored in **Ke**, is computed according to:

$$\mathbf{K}^e = \mathbf{G}^T \bar{\mathbf{K}}^e \mathbf{G}$$

where:

$$\bar{\mathbf{K}}^e = \begin{bmatrix} \frac{D_{EA}}{L} & 0 & 0 & -\frac{D_{EA}}{L} & 0 & 0 \\ 0 & \frac{12D_{EI}}{L^3} & \frac{6D_{EI}}{L^2} & 0 & -\frac{12D_{EI}}{L^3} & \frac{6D_{EI}}{L^2} \\ 0 & \frac{6D_{EI}}{L^2} & \frac{4D_{EI}}{L} & 0 & -\frac{6D_{EI}}{L^2} & \frac{2D_{EI}}{L} \\ -\frac{D_{EA}}{L} & 0 & 0 & \frac{D_{EA}}{L} & 0 & 0 \\ 0 & -\frac{12D_{EI}}{L^3} & -\frac{6D_{EI}}{L^2} & 0 & \frac{12D_{EI}}{L^3} & -\frac{6D_{EI}}{L^2} \\ 0 & \frac{6D_{EI}}{L^2} & \frac{2D_{EI}}{L} & 0 & -\frac{6D_{EI}}{L^2} & \frac{4D_{EI}}{L} \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} n_{x\bar{x}} & n_{y\bar{x}} & 0 & 0 & 0 & 0 \\ n_{x\bar{y}} & n_{y\bar{y}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & n_{x\bar{x}} & n_{y\bar{x}} & 0 \\ 0 & 0 & 0 & n_{x\bar{y}} & n_{y\bar{y}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

where the axial stiffness D_{EA}, the bending stiffness D_{EI}, and the length L are given by:

$$D_{EA} = EA, \quad D_{EI} = EI, \quad L = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

The transformation matrix $\mathbf{G}$ contains the direction cosines:

$$n_{x\bar{x}} = n_{y\bar{y}} = \frac{x_2 - x_1}{L}, \quad n_{y\bar{x}} = -n_{x\bar{y}} = \frac{y_2 - y_1}{L}$$

The element loads $\mathbf{f}_l^e$, stored in the variable **fe**, are computed according to:

$$\mathbf{f}_l^e = \mathbf{G}^T \bar{\mathbf{f}}_l^e$$

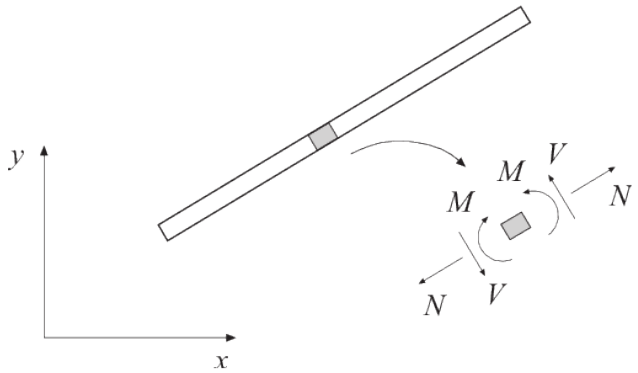
where:

$$\bar{\mathbf{f}}_l^e = \begin{bmatrix} \frac{q_{\bar{x}} L}{2} \\ \frac{q_{\bar{y}} L}{2} \\ \frac{q_{\bar{y}} L^2}{12} \\ \frac{q_{\bar{x}} L}{2} \\ \frac{q_{\bar{y}} L}{2} \\ -\frac{q_{\bar{y}} L^2}{12} \end{bmatrix}$$

10.2.2 beam2s

Purpose

Compute section forces in a two-dimensional beam element.



Syntax

```
[es] = beam2s(ex, ey, ep, ed)
[es] = beam2s(ex, ey, ep, ed, eq)
[es, edi] = beam2s(ex, ey, ep, ed, eq, n)
[es, edi, eci] = beam2s(ex, ey, ep, ed, eq, n)
```

Description

beam2s computes the section forces and displacements in local directions along the beam element **beam2e**.

The input variables **ex**, **ey**, **ep**, and **eq** are defined in **beam2e**.

The element displacements, stored in **ed**, are obtained by the function **extract**. If a distributed load is applied to the element, the variable **eq** must be included. The number of evaluation points for section forces and displacements is determined by **n**. If **n** is omitted, only the ends of the beam are evaluated.

The output variables:

$$es = \begin{bmatrix} N(0) & V(0) & M(0) \\ N(\bar{x}_2) & V(\bar{x}_2) & M(\bar{x}_2) \\ \vdots & \vdots & \vdots \\ N(\bar{x}_{n-1}) & V(\bar{x}_{n-1}) & M(\bar{x}_{n-1}) \\ N(L) & V(L) & M(L) \end{bmatrix}$$

$$edi = \begin{bmatrix} u(0) & v(0) \\ u(\bar{x}_2) & v(\bar{x}_2) \\ \vdots & \vdots \\ u(\bar{x}_{n-1}) & v(\bar{x}_{n-1}) \\ u(L) & v(L) \end{bmatrix}$$

$$eci = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

contain the section forces, the displacements, and the evaluation points on the local $\bar{x}$-axis. L is the length of the beam element.

Theory

The nodal displacements in local coordinates are given by:

$$\bar{\mathbf{a}}^e = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_4 \\ \bar{u}_5 \\ \bar{u}_6 \end{bmatrix} = \mathbf{G}\mathbf{a}^e$$

where $\mathbf{G}$ is described in **beam2e** and the transpose of $\mathbf{a}^e$ is stored in **ed**.

The displacements associated with bar action and beam action are determined as:

$$\bar{\mathbf{a}}^e_{\text{bar}} = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_4 \end{bmatrix}, \quad \bar{\mathbf{a}}^e_{\text{beam}} = \begin{bmatrix} \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_5 \\ \bar{u}_6 \end{bmatrix}$$

The displacement $u(\bar{x})$ and the normal force $N(\bar{x})$ are computed from:

$$u(\bar{x}) = \mathbf{N}_{\text{bar}}\bar{\mathbf{a}}^e_{\text{bar}} + u_p(\bar{x})$$

$$N(\bar{x}) = D_{EA}\mathbf{B}_{\text{bar}}\bar{\mathbf{a}}^e + N_p(\bar{x})$$

where:

$$\mathbf{N}_{\text{bar}} = \begin{bmatrix}1 & \bar{x}\end{bmatrix}\mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix}1 - \frac{\bar{x}}{L} & \frac{\bar{x}}{L}\end{bmatrix}$$

$$\mathbf{B}_{\text{bar}} = \begin{bmatrix}0 & 1\end{bmatrix}\mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix}-\frac{1}{L} & \frac{1}{L}\end{bmatrix}$$

$$u_p(\bar{x}) = -\frac{q_{\bar{x}}}{D_{EA}}\left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2}\right)$$

$$N_p(\bar{x}) = -q_{\bar{x}}\left(\bar{x} - \frac{L}{2}\right)$$

where D_{EA}, L, and $q_{\bar{x}}$ are defined in **beam2e**, and:

$$\mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix}1 & 0\\ -\frac{1}{L} & \frac{1}{L}\end{bmatrix}$$

The displacement $v(\bar{x})$, the bending moment $M(\bar{x})$, and the shear force $V(\bar{x})$ are computed from:

$$v(\bar{x}) = \mathbf{N}_{\text{beam}}\bar{\mathbf{a}}^e_{\text{beam}} + v_p(\bar{x})$$

$$M(\bar{x}) = D_{EI}\mathbf{B}_{\text{beam}}\bar{\mathbf{a}}^e_{\text{beam}} + M_p(\bar{x})$$

$$V(\bar{x}) = -D_{EI}\frac{d\mathbf{B}_{\text{beam}}}{d\bar{x}}\bar{\mathbf{a}}^e_{\text{beam}} + V_p(\bar{x})$$

where:

$$\mathbf{N}_{\text{beam}} = \begin{bmatrix}1 & \bar{x} & \bar{x}^2 & \bar{x}^3\end{bmatrix}\mathbf{C}^{-1}_{\text{beam}}$$

$$\mathbf{B}_{\text{beam}} = \begin{bmatrix}0 & 0 & 2 & 6\bar{x}\end{bmatrix}\mathbf{C}^{-1}_{\text{beam}}$$

$$\frac{d\mathbf{B}_{\text{beam}}}{d\bar{x}} = \begin{bmatrix}0 & 0 & 0 & 6\end{bmatrix}\mathbf{C}^{-1}_{\text{beam}}$$

$$v_p(\bar{x}) = \frac{q_{\bar{y}}}{D_{EI}}\left(\frac{\bar{x}^4}{24} - \frac{L\bar{x}^3}{12} + \frac{L^2\bar{x}^2}{24}\right)$$

$$M_p(\bar{x}) = q_{\bar{y}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} + \frac{L^2}{12} \right)$$

$$V_p(\bar{x}) = -q_{\bar{y}} \left(\bar{x} - \frac{L}{2} \right)$$

where D_{EI}, L, and $q_{\bar{y}}$ are defined in **beam2e**, and:

$$\mathbf{C}_{\text{beam}}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{3}{L^2} & -\frac{2}{L} & \frac{3}{L^2} & -\frac{1}{L} \\ \frac{2}{L^3} & \frac{1}{L^2} & -\frac{2}{L^3} & \frac{1}{L^2} \end{bmatrix}$$

10.2.3 beam2te

Purpose

Compute element stiffness matrix for a two dimensional Timoshenko beam element.

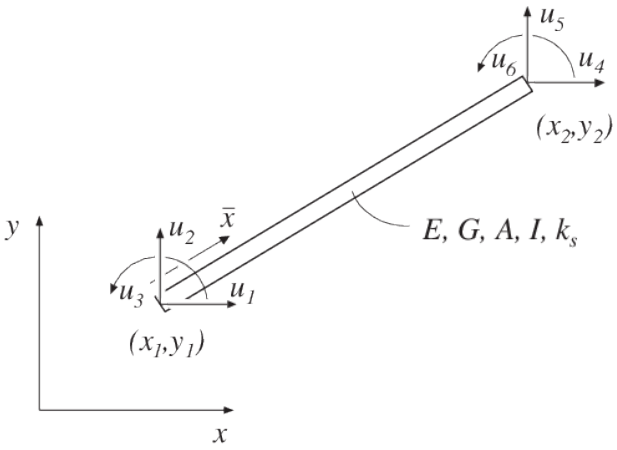


Fig. 1: Two dimensional beam element

Syntax

```
Ke = beam2te(ex, ey, ep)
[Ke, fe] = beam2te(ex, ey, ep, eq)
```

Description

`beam2te` provides the global element stiffness matrix `Ke` for a two dimensional Timoshenko beam element.

The input variables

$$\mathbf{ex} = [x_1 \;\; x_2]$$
$$\mathbf{ey} = [y_1 \;\; y_2]$$
$$\mathbf{ep} = [E \;\; G \;\; A \;\; I \;\; k_s]$$

supply the element nodal coordinates x_1, y_1, x_2, and y_2, the modulus of elasticity E, the shear modulus G, the cross section area A, the moment of inertia I and the shear correction factor k_s.

The element load vector `fe` can also be computed if uniformly distributed loads are applied to the element. The optional input variable

$$\mathrm{eq} = [q_{\bar{x}} \;\; q_{\bar{y}}]$$

contains the distributed loads per unit length, $q_{\bar{x}}$ and $q_{\bar{y}}$.

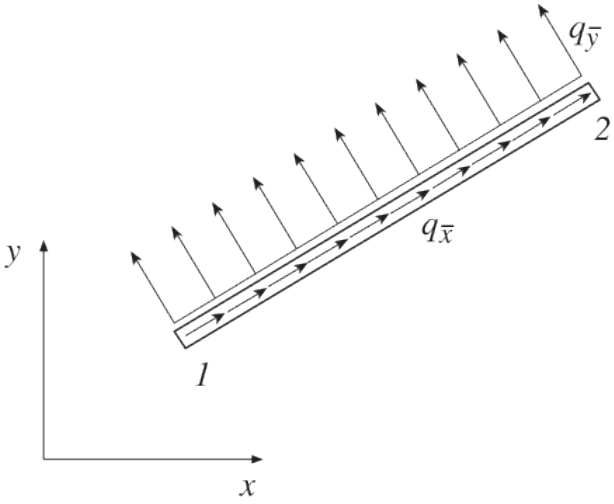


Fig. 2: Uniformly distributed load

Theory

The element stiffness matrix $\mathbf{K}^e$, stored in `Ke`, is computed according to

$$\mathbf{K}^e = \mathbf{G}^T \bar{\mathbf{K}}^e \mathbf{G}$$

where $\mathbf{G}$ is described in `beam2e`, and $\bar{\mathbf{K}}^e$ is given by

$$\bar{\mathbf{K}}^e = \begin{bmatrix} \frac{D_{EA}}{L} & 0 & 0 & -\frac{D_{EA}}{L} & 0 & 0 \\ 0 & \frac{12D_{EI}}{L^3(1+\mu)} & \frac{6D_{EI}}{L^2(1+\mu)} & 0 & -\frac{12D_{EI}}{L^3(1+\mu)} & \frac{6D_{EI}}{L^2(1+\mu)} \\ 0 & \frac{6D_{EI}}{L^2(1+\mu)} & \frac{4D_{EI}(1+\frac{\mu}{4})}{L(1+\mu)} & 0 & -\frac{6D_{EI}}{L^2(1+\mu)} & \frac{2D_{EI}(1-\frac{\mu}{2})}{L(1+\mu)} \\ -\frac{D_{EA}}{L} & 0 & 0 & \frac{D_{EA}}{L} & 0 & 0 \\ 0 & -\frac{12D_{EI}}{L^3(1+\mu)} & -\frac{6D_{EI}}{L^2(1+\mu)} & 0 & \frac{12D_{EI}}{L^3(1+\mu)} & -\frac{6D_{EI}}{L^2(1+\mu)} \\ 0 & \frac{6D_{EI}}{L^2(1+\mu)} & \frac{2D_{EI}(1-\frac{\mu}{2})}{L(1+\mu)} & 0 & -\frac{6D_{EI}}{L^2(1+\mu)} & \frac{4D_{EI}(1+\frac{\mu}{4})}{L(1+\mu)} \end{bmatrix}$$

where the axial stiffness D_{EA}, the bending stiffness D_{EI}, and the length L are given by

$$D_{EA} = EA \qquad D_{EI} = EI \qquad L = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

and where

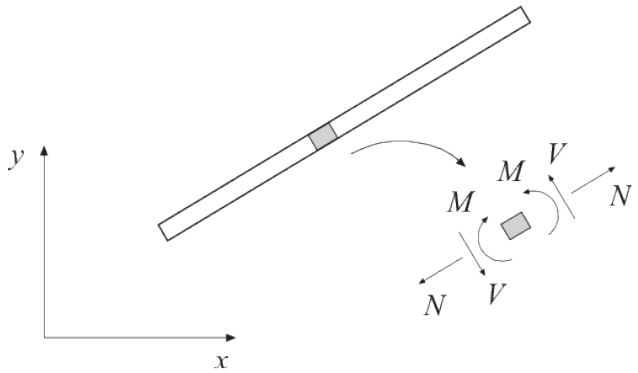
$$\mu = \frac{12D_{EI}}{L^2 G A k_s}$$

The element loads $\mathbf{f}_l^e$ stored in the variable `fe` are computed according to

$$\mathbf{f}_l^e = \mathbf{G}^T \bar{\mathbf{f}}_l^e$$

where

$$\bar{\mathbf{f}}_l^e = \begin{bmatrix} \frac{q_{\bar{x}} L}{2} \\ \frac{q_{\bar{y}} L}{2} \\ \frac{q_{\bar{y}} L^2}{12} \\ \frac{q_{\bar{x}} L}{2} \\ \frac{q_{\bar{y}} L}{2} \\ -\frac{q_{\bar{y}} L^2}{12} \end{bmatrix}$$



10.2.4 beam2ts

Purpose

Compute section forces in a two dimensional Timoshenko beam element.

Syntax

```
es = beam2ts(ex, ey, ep, ed)
es = beam2ts(ex, ey, ep, ed, eq)
[es, edi, eci] = beam2ts(ex, ey, ep, ed, eq, n)
```

Description

`beam2ts` computes the section forces and displacements in local directions along the beam element `beam2te`.

The input variables `ex`, `ey`, `ep` and `eq` are defined in `beam2te`. The element displacements, stored in `ed`, are obtained by the function `extract`. If distributed loads are applied to the element, the variable `eq` must be included. The number of evaluation points for section forces and displacements are determined by `n`. If `n` is omitted, only the ends of the beam are evaluated.

The output variables

$$\begin{aligned} \text{es} &= [\ \mathbf{N}\ \mathbf{V}\ \mathbf{M}\ ] \\ \text{edi} &= [\ \mathbf{u}\ \mathbf{v}\ \boldsymbol{\theta}\ ] \\ \text{eci} &= [\bar{\mathbf{x}}] \end{aligned}$$

consist of column matrices that contain the section forces, the displacements and rotation of the cross section (note that the rotation θ is not equal to $\frac{d\bar{v}}{d\bar{x}}$), and the evaluation points on the local $\bar{x}$-axis. The explicit matrix expressions are

$$\text{es} = \begin{bmatrix} N_1 & V_1 & M_1 \\ N_2 & V_2 & M_2 \\ \vdots & \vdots & \vdots \\ N_n & V_n & M_n \end{bmatrix} \qquad \text{edi} = \begin{bmatrix} u_1 & v_1 & \theta_1 \\ u_2 & v_2 & \theta_2 \\ \vdots & \vdots & \vdots \\ u_n & v_n & \theta_n \end{bmatrix} \qquad \text{eci} = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

where L is the length of the beam element.

Theory

The nodal displacements in local coordinates are given by

$$\bar{\mathbf{a}}^e = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_4 \\ \bar{u}_5 \\ \bar{u}_6 \end{bmatrix} = \mathbf{G}\mathbf{a}^e$$

where $\mathbf{G}$ is described in `beam2e` and the transpose of $\mathbf{a}^e$ is stored in `ed`. The displacements associated with bar action and beam action are determined as

$$\bar{\mathbf{a}}^e_{\text{bar}} = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_4 \end{bmatrix} \qquad \bar{\mathbf{a}}^e_{\text{beam}} = \begin{bmatrix} \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_5 \\ \bar{u}_6 \end{bmatrix}$$

The displacement $u(\bar{x})$ and the normal force $N(\bar{x})$ are computed from

$$u(\bar{x}) = \mathbf{N}_{\text{bar}}\bar{\mathbf{a}}^e_{\text{bar}} + u_p(\bar{x})$$

$$N(\bar{x}) = D_{EA}\mathbf{B}_{\text{bar}}\bar{\mathbf{a}}^e + N_p(\bar{x})$$

where

$$\mathbf{N}_{\text{bar}} = \begin{bmatrix} 1 & \bar{x} \end{bmatrix} \mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix} 1 - \frac{\bar{x}}{L} & \frac{\bar{x}}{L} \end{bmatrix}$$

$$\mathbf{B}_{\text{bar}} = \begin{bmatrix} 0 & 1 \end{bmatrix} \mathbf{C}_{\text{bar}}^{-1} = \begin{bmatrix} -\frac{1}{L} & \frac{1}{L} \end{bmatrix}$$

$$u_p(\bar{x}) = -\frac{q_{\bar{x}}}{D_{EA}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} \right)$$

$$N_p(\bar{x}) = -q_{\bar{x}} \left(\bar{x} - \frac{L}{2} \right)$$

in which D_{EA}, L, and $q_{\bar{x}}$ are defined in `beam2te` and

$$\mathbf{C}_{\text{bar}}^{-1} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix}$$

The displacement $v(\bar{x})$, the rotation $\theta(\bar{x})$, the bending moment $M(\bar{x})$ and the shear force $V(\bar{x})$ are computed from

$$v(\bar{x}) = \mathbf{N}_{\text{beam},v} \bar{\mathbf{a}}_{\text{beam}}^e + v_p(\bar{x})$$

$$\theta(\bar{x}) = \mathbf{N}_{\text{beam},\theta} \bar{\mathbf{a}}_{\text{beam}}^e + \theta_p(\bar{x})$$

$$M(\bar{x}) = D_{EI} \frac{d\theta}{dx} = D_{EI} \frac{d\mathbf{N}_{\text{beam},\theta}}{d\bar{x}} \bar{\mathbf{a}}_{\text{beam}}^e + M_p(\bar{x})$$

$$V(\bar{x}) = D_{GA} k_s \left(\frac{dv}{dx} - \theta \right) = D_{GA} k_s \left(\frac{d\mathbf{N}_{\text{beam},v}}{d\bar{x}} - \mathbf{N}_{\text{beam},\theta} \right) \bar{\mathbf{a}}_{\text{beam}}^e + V_p(\bar{x})$$

where

$$\mathbf{N}_{\text{beam},v} = \begin{bmatrix} 1 & \bar{x} & \bar{x}^2 & \bar{x}^3 \end{bmatrix} \mathbf{C}_{\text{beam}}^{-1}$$

$$\frac{d\mathbf{N}_{\text{beam},v}}{d\bar{x}} = \begin{bmatrix} 0 & 1 & 2\bar{x} & 3\bar{x}^2 \end{bmatrix} \mathbf{C}_{\text{beam}}^{-1}$$

$$\mathbf{N}_{\text{beam},\theta} = \begin{bmatrix} 0 & 1 & 2\bar{x} & 3\bar{x}^2 + 6\alpha \end{bmatrix} \mathbf{C}_{\text{beam}}^{-1}$$

$$\frac{d\mathbf{N}_{\text{beam},\theta}}{d\bar{x}} = \begin{bmatrix} 0 & 0 & 2 & 6\bar{x} \end{bmatrix} \mathbf{C}_{\text{beam}}^{-1}$$

$$v_p(\bar{x}) = \frac{q_{\bar{y}}}{D_{EI}} \left(\frac{\bar{x}^4}{24} - \frac{L\bar{x}^3}{12} + \frac{L^2\bar{x}^2}{2} \right) + \frac{q_{\bar{y}}}{D_{GA}k_s} \left(-\frac{\bar{x}^2}{2} + \frac{L\bar{x}}{2} \right)$$

$$\theta_p(\bar{x}) = \frac{q_{\bar{y}}}{D_{EI}} \left(\frac{\bar{x}^3}{6} - \frac{L\bar{x}^2}{4} + \frac{L^2\bar{x}}{12} \right)$$

$$M_p(\bar{x}) = q_{\bar{y}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} + \frac{L^2}{12} \right)$$

$$V_p(\bar{x}) = -q_{\bar{y}} \left(\bar{x} - \frac{L}{2} \right)$$

in which D_{EI}, D_{GA}, k_s, L, and $q_{\bar{y}}$ are defined in `beam2te` and

$$\mathbf{C}_{\text{beam}}^{-1} = \frac{1}{L^2 + 12\alpha} \begin{bmatrix} L^2 + 12\alpha & 0 & 0 & 0 \\ -\frac{12\alpha}{L} & L^2 + 6\alpha & \frac{12\alpha}{L} & -6\alpha \\ -3 & -2L - \frac{6\alpha}{L} & 3 & -L + \frac{6\alpha}{L} \\ \frac{2}{L} & 1 & -\frac{2}{L} & 1 \end{bmatrix}$$

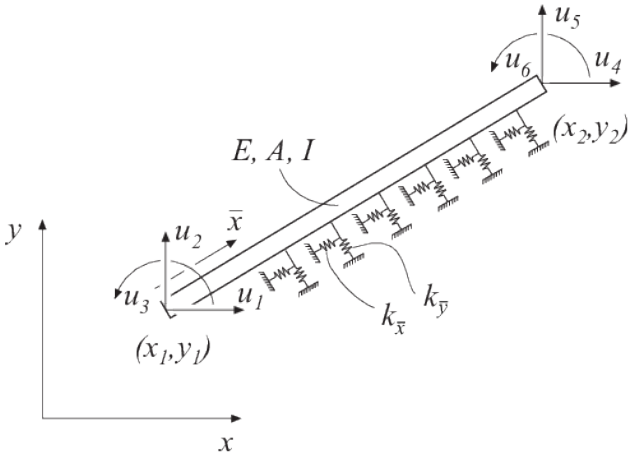
with

$$\alpha = \frac{D_{EI}}{D_{GA} k_s}$$

10.2.5 beam2we

Purpose

Compute element stiffness matrix for a two dimensional beam element on elastic support.



Syntax

```
Ke = beam2we(ex, ey, ep)
[Ke, fe] = beam2we(ex, ey, ep, eq)
```

Description

`beam2we` provides the global element stiffness matrix `Ke` for a two dimensional beam element with elastic support.

The input variables

$$\begin{aligned}\text{ex} &= [x_1 \;\; x_2] \\ \text{ey} &= [y_1 \;\; y_2] \\ \text{ep} &= [E \;\; A \;\; I \;\; k_{\bar{x}} \;\; k_{\bar{y}}]\end{aligned}$$

supply the element nodal coordinates x_1, x_2, y_1, and y_2, the modulus of elasticity E, the cross section area A, the moment of inertia I, the spring stiffness in the axial direction $k_{\bar{x}}$, and the spring stiffness in the transverse direction $k_{\bar{y}}$.

The element load vector `fe` can also be computed if uniformly distributed loads are applied to the element. The optional input variable

$$\text{eq} = [q_{\bar{x}} \;\; q_{\bar{y}}]$$

contains the distributed load per unit length, $q_{\bar{x}}$ and $q_{\bar{y}}$.

Theory

The element stiffness matrix $\mathbf{K}^e$, stored in `Ke`, is computed according to

$$\mathbf{K}^e = \mathbf{G}^T \bar{\mathbf{K}}^e \mathbf{G}$$

where

$$\bar{\mathbf{K}}^e = \bar{\mathbf{K}}_0^e + \bar{\mathbf{K}}_s^e$$

$$\bar{\mathbf{K}}_0^e = \begin{bmatrix} \frac{D_{EA}}{L} & 0 & 0 & -\frac{D_{EA}}{L} & 0 & 0 \\ 0 & \frac{12D_{EI}}{L^3} & \frac{6D_{EI}}{L^2} & 0 & -\frac{12D_{EI}}{L^3} & \frac{6D_{EI}}{L^2} \\ 0 & \frac{6D_{EI}}{L^2} & \frac{4D_{EI}}{L} & 0 & -\frac{6D_{EI}}{L^2} & \frac{2D_{EI}}{L} \\ -\frac{D_{EA}}{L} & 0 & 0 & \frac{D_{EA}}{L} & 0 & 0 \\ 0 & -\frac{12D_{EI}}{L^3} & -\frac{6D_{EI}}{L^2} & 0 & \frac{12D_{EI}}{L^3} & -\frac{6D_{EI}}{L^2} \\ 0 & \frac{6D_{EI}}{L^2} & \frac{2D_{EI}}{L} & 0 & -\frac{6D_{EI}}{L^2} & \frac{4D_{EI}}{L} \end{bmatrix}$$

$$\bar{\mathbf{K}}_s^e = \frac{L}{420} \begin{bmatrix} 140k_{\bar{x}} & 0 & 0 & 70k_{\bar{x}} & 0 & 0 \\ 0 & 156k_{\bar{y}} & 22k_{\bar{y}}L & 0 & 54k_{\bar{y}} & -13k_{\bar{y}}L \\ 0 & 22k_{\bar{y}}L & 4k_{\bar{y}}L^2 & 0 & 13k_{\bar{y}}L & -3k_{\bar{y}}L^2 \\ 70k_{\bar{x}} & 0 & 0 & 140k_{\bar{x}} & 0 & 0 \\ 0 & 54k_{\bar{y}} & 13k_{\bar{y}}L & 0 & 156k_{\bar{y}} & -22k_{\bar{y}}L \\ 0 & -13k_{\bar{y}}L & -3k_{\bar{y}}L^2 & 0 & -22k_{\bar{y}}L & 4k_{\bar{y}}L^2 \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} n_{x\bar{x}} & n_{y\bar{x}} & 0 & 0 & 0 & 0 \\ n_{x\bar{y}} & n_{y\bar{y}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & n_{x\bar{x}} & n_{y\bar{x}} & 0 \\ 0 & 0 & 0 & n_{x\bar{y}} & n_{y\bar{y}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

where the axial stiffness D_{EA}, the bending stiffness D_{EI} and the length L are given by

$$D_{EA} = EA; \quad D_{EI} = EI; \quad L = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

The transformation matrix $\mathbf{G}$ contains the direction cosines

$$n_{x\bar{x}} = n_{y\bar{y}} = \frac{x_2 - x_1}{L} \qquad n_{y\bar{x}} = -n_{x\bar{y}} = \frac{y_2 - y_1}{L}$$

The element loads $\mathbf{f}_l^e$ stored in the variable `fe` are computed according to

$$\mathbf{f}_l^e = \mathbf{G}^T \bar{\mathbf{f}}_l^e$$

where

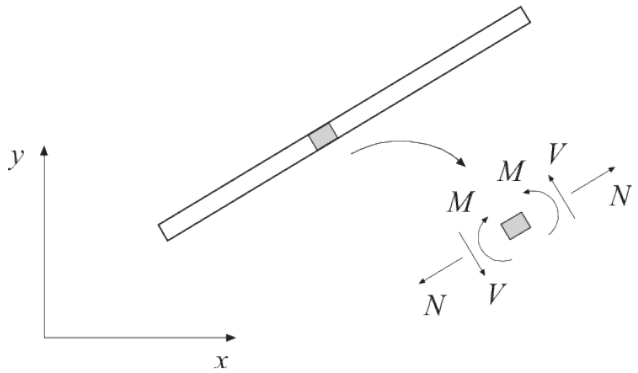
$$\bar{\mathbf{f}}_l^e = \begin{bmatrix} \frac{q_{\bar{x}} L}{2} \\ \frac{q_{\bar{y}} L}{2} \\ \frac{q_{\bar{y}} L^2}{12} \\ \frac{q_{\bar{x}} L}{2} \\ \frac{q_{\bar{y}} L}{2} \\ -\frac{q_{\bar{y}} L^2}{12} \end{bmatrix}$$

10.2.6 beam2ws

Purpose

Compute section forces in a two dimensional beam element with elastic support.

Syntax




```
es = beam2ws(ex, ey, ep, ed)
es = beam2ws(ex, ey, ep, ed, eq)
es, edi, eci = beam2ws(ex, ey, ep, ed, eq, n)
```

Description

`beam2ws` computes the section forces and displacements in local directions along the beam element `beam2we`.

The input variables `ex`, `ey`, `ep` and `eq` are defined in `beam2we`, and the element displacements, stored in `ed`, are obtained by the function `extract`. If distributed loads are applied to the element, the variable `eq` must be included. The number of evaluation points for section forces and displacements are determined by `n`. If `n` is omitted, only the ends of the beam are evaluated.

The output variables

$$\text{es} = \begin{bmatrix} N(0) & V(0) & M(0) \\ N(\bar{x}_2) & V(\bar{x}_2) & M(\bar{x}_2) \\ \vdots & \vdots & \vdots \\ N(\bar{x}_{n-1}) & V(\bar{x}_{n-1}) & M(\bar{x}_{n-1}) \\ N(L) & V(L) & M(L) \end{bmatrix}$$

$$\text{edi} = \begin{bmatrix} u(0) & v(0) \\ u(\bar{x}_2) & v(\bar{x}_2) \\ \vdots & \vdots \\ u(\bar{x}_{n-1}) & v(\bar{x}_{n-1}) \\ u(L) & v(L) \end{bmatrix}$$

$$\text{eci} = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

contain the section forces, the displacements, and the evaluation points on the local $\bar{x}$-axis. L is the length of the beam element.

Theory

The nodal displacements in local coordinates are given by

$$\bar{\mathbf{a}}^e = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_4 \\ \bar{u}_5 \\ \bar{u}_6 \end{bmatrix} = \mathbf{G}\mathbf{a}^e$$

where $\mathbf{G}$ is described in `beam2we` and the transpose of $\mathbf{a}^e$ is stored in `ed`. The displacements associated with bar action and beam action are determined as

$$\bar{\mathbf{a}}^e_{\text{bar}} = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_4 \end{bmatrix} \qquad \bar{\mathbf{a}}^e_{\text{beam}} = \begin{bmatrix} \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_5 \\ \bar{u}_6 \end{bmatrix}$$

The displacement $u(\bar{x})$ and the normal force $N(\bar{x})$ are computed from

$$u(\bar{x}) = \mathbf{N}_{\text{bar}}\bar{\mathbf{a}}^e_{\text{bar}} + u_p(\bar{x})$$

$$N(\bar{x}) = D_{EA}\mathbf{B}_{\text{bar}}\bar{\mathbf{a}}^e + N_p(\bar{x})$$

where

$$\mathbf{N}_{\text{bar}} = \begin{bmatrix} 1 & \bar{x} \end{bmatrix} \mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix} 1 - \frac{\bar{x}}{L} & \frac{\bar{x}}{L} \end{bmatrix}$$

$$\mathbf{B}_{\text{bar}} = \begin{bmatrix} 0 & 1 \end{bmatrix} \mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix} -\frac{1}{L} & \frac{1}{L} \end{bmatrix}$$

$$u_p(\bar{x}) = \frac{k_{\bar{x}}}{D_{EA}} \begin{bmatrix} \frac{\bar{x}^2 - L\bar{x}}{2} & \frac{\bar{x}^3 - L^2\bar{x}}{6} \end{bmatrix} \mathbf{C}^{-1}_{\text{bar}}\bar{\mathbf{a}}^e_{\text{bar}} - \frac{q_{\bar{x}}}{D_{EA}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} \right)$$

$$N_p(\bar{x}) = k_{\bar{x}} \begin{bmatrix} \frac{2\bar{x} - L}{2} & \frac{3\bar{x}^2 - L^2}{6} \end{bmatrix} \mathbf{C}^{-1}_{\text{bar}}\bar{\mathbf{a}}^e_{\text{bar}} - q_{\bar{x}} \left(\bar{x} - \frac{L}{2} \right)$$

in which D_{EA}, $k_{\bar{x}}$, L, and $q_{\bar{x}}$ are defined in `beam2we` and

$$\mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix}$$

The displacement $v(\bar{x})$, the bending moment $M(\bar{x})$ and the shear force $V(\bar{x})$ are computed from

$$v(\bar{x}) = \mathbf{N}_{\text{beam}}\bar{\mathbf{a}}^e_{\text{beam}} + v_p(\bar{x})$$

$$M(\bar{x}) = D_{EI}\mathbf{B}_{\text{beam}}\bar{\mathbf{a}}^e_{\text{beam}} + M_p(\bar{x})$$

$$V(\bar{x}) = -D_{EI}\frac{d\mathbf{B}_{\text{beam}}}{dx}\bar{\mathbf{a}}^e_{\text{beam}} + V_p(\bar{x})$$

where

$$\mathbf{N}_{\text{beam}} = \begin{bmatrix} 1 & \bar{x} & \bar{x}^2 & \bar{x}^3 \end{bmatrix} \mathbf{C}^{-1}_{\text{beam}}$$

$$\mathbf{B}_{\text{beam}} = \begin{bmatrix} 0 & 0 & 2 & 6\bar{x} \end{bmatrix} \mathbf{C}^{-1}_{\text{beam}}$$

$$\frac{d\mathbf{B}_{\text{beam}}}{dx} = \begin{bmatrix} 0 & 0 & 0 & 6 \end{bmatrix} \mathbf{C}^{-1}_{\text{beam}}$$

$$v_p(\bar{x}) = -\frac{k_{\bar{y}}}{D_{EI}} \begin{bmatrix} \frac{\bar{x}^4 - 2L\bar{x}^3 + L^2\bar{x}^2}{24} \\ \frac{\bar{x}^5 - 3L^2\bar{x}^3 + 2L^3\bar{x}^2}{120} \\ \frac{\bar{x}^6 - 4L^3\bar{x}^3 + 3L^4\bar{x}^2}{360} \\ \frac{\bar{x}^7 - 5L^4\bar{x}^3 + 4L^5\bar{x}^2}{840} \end{bmatrix}^T \mathbf{C}_{\text{beam}}^{-1} \bar{\mathbf{a}}_{\text{beam}}^e + \frac{q_{\bar{y}}}{D_{EI}} \left(\frac{\bar{x}^4}{24} - \frac{L\bar{x}^3}{12} + \frac{L^2\bar{x}^2}{24} \right)$$

$$M_p(\bar{x}) = -k_{\bar{y}} \begin{bmatrix} \frac{6\bar{x}^2 - 6L\bar{x} + L^2}{12} \\ \frac{10\bar{x}^3 - 9L^2\bar{x} + 2L^3}{60} \\ \frac{5\bar{x}^4 - 4L^3\bar{x} + L^4}{60} \\ \frac{21\bar{x}^5 - 15L^4\bar{x} + 4L^5}{420} \end{bmatrix}^T \mathbf{C}_{\text{beam}}^{-1} \bar{\mathbf{a}}_{\text{beam}}^e + q_{\bar{y}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} + \frac{L^2}{12} \right)$$

$$V_p(\bar{x}) = k_{\bar{y}} \begin{bmatrix} \frac{2\bar{x} - L}{2} \\ \frac{10\bar{x}^2 - 3L^2}{20} \\ \frac{5\bar{x}^3 - L^3}{15} \\ \frac{7\bar{x}^4 - L^4}{28} \end{bmatrix}^T \mathbf{C}_{\text{beam}}^{-1} \bar{\mathbf{a}}_{\text{beam}}^e - q_{\bar{y}} \left(\bar{x} - \frac{L}{2} \right)$$

in which D_{EI}, $k_{\bar{y}}$, L, and $q_{\bar{y}}$ are defined in `beam2we` and

$$\mathbf{C}_{\text{beam}}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{3}{L^2} & -\frac{2}{L} & \frac{3}{L^2} & -\frac{1}{L} \\ \frac{2}{L^3} & \frac{1}{L^2} & -\frac{2}{L^3} & \frac{1}{L^2} \end{bmatrix}$$

10.2.7 beam2ge

Purpose

Compute element stiffness matrix for a two dimensional nonlinear beam element with respect to geometrical nonlinearity.

Syntax

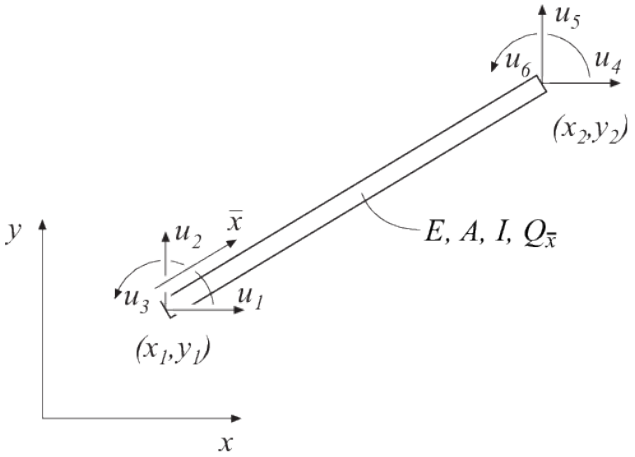
```
Ke = beam2ge(ex, ey, ep, Qx)
[Ke, fe] = beam2ge(ex, ey, ep, Qx, eq)
```

Description

`beam2ge` provides the global element stiffness matrix `Ke` for a two dimensional beam element with respect to geometrical nonlinearity.

The input variables:

- `ex = [x1 x2]`



- `ey = [y1 y2]`
- `ep = [E A I]`

supply the element nodal coordinates `x1`, `y1`, `x2`, and `y2`, the modulus of elasticity `E`, the cross section area `A`, and the moment of inertia `I`.

- `Qx = [Q_xbar]`

contains the value of the predefined axial force `Q_xbar`, which is positive in tension.

The element load vector `fe` can also be computed if a uniformly distributed transverse load is applied to the element. The optional input variable

- `eq = [q_ybar]`

contains the distributed transverse load per unit length, `q_ybar`. Note that `eq` is a scalar and not a vector as in `beam2e`.

Theory

The element stiffness matrix $\mathbf{K}^e$, stored in the variable `Ke`, is computed according to

$$\mathbf{K}^e = \mathbf{G}^T \bar{\mathbf{K}}^e \mathbf{G}$$

where $\bar{\mathbf{K}}^e$ is given by

$$\bar{\mathbf{K}}^e = \bar{\mathbf{K}}_0^e + \bar{\mathbf{K}}_\sigma^e$$

with

$$\bar{\mathbf{K}}_0^e = \begin{bmatrix} \frac{D_{EA}}{L} & 0 & 0 & -\frac{D_{EA}}{L} & 0 & 0 \\ 0 & \frac{12D_{EI}}{L^3} & \frac{6D_{EI}}{L^2} & 0 & -\frac{12D_{EI}}{L^3} & \frac{6D_{EI}}{L^2} \\ 0 & \frac{6D_{EI}}{L^2} & \frac{4D_{EI}}{L} & 0 & -\frac{6D_{EI}}{L^2} & \frac{2D_{EI}}{L} \\ -\frac{D_{EA}}{L} & 0 & 0 & \frac{D_{EA}}{L} & 0 & 0 \\ 0 & -\frac{12D_{EI}}{L^3} & -\frac{6D_{EI}}{L^2} & 0 & \frac{12D_{EI}}{L^3} & -\frac{6D_{EI}}{L^2} \\ 0 & \frac{6D_{EI}}{L^2} & \frac{2D_{EI}}{L} & 0 & -\frac{6D_{EI}}{L^2} & \frac{4D_{EI}}{L} \end{bmatrix}$$

$$\bar{\mathbf{K}}_\sigma^e = Q_{\bar{x}} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{6}{5L} & \frac{1}{10} & 0 & -\frac{6}{5L} & \frac{1}{10} \\ 0 & \frac{1}{10} & \frac{2L}{15} & 0 & -\frac{1}{10} & -\frac{L}{30} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{6}{5L} & -\frac{1}{10} & 0 & \frac{6}{5L} & -\frac{1}{10} \\ 0 & \frac{1}{10} & -\frac{L}{30} & 0 & -\frac{1}{10} & \frac{2L}{15} \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} n_{x\bar{x}} & n_{y\bar{x}} & 0 & 0 & 0 & 0 \\ n_{x\bar{y}} & n_{y\bar{y}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & n_{x\bar{x}} & n_{y\bar{x}} & 0 \\ 0 & 0 & 0 & n_{x\bar{y}} & n_{y\bar{y}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

where the axial stiffness D_{EA}, the bending stiffness D_{EI} and the length L are given by

$$D_{EA} = EA; \quad D_{EI} = EI; \quad L = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

The transformation matrix $\mathbf{G}$ contains the direction cosines

$$n_{x\bar{x}} = n_{y\bar{y}} = \frac{x_2 - x_1}{L} \qquad n_{y\bar{x}} = -n_{x\bar{y}} = \frac{y_2 - y_1}{L}$$

The element loads $\mathbf{f}_l^e$ stored in `fe` are computed according to

$$\mathbf{f}_l^e = \mathbf{G}^T \bar{\mathbf{f}}_l^e$$

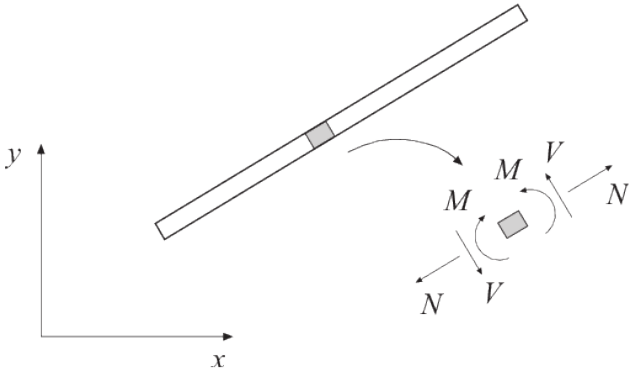
where

$$\bar{\mathbf{f}}_l^e = q_{\bar{y}} \begin{bmatrix} 0 \\ \frac{L}{2} \\ \frac{L^2}{12} \\ 0 \\ \frac{L}{2} \\ -\frac{L^2}{12} \end{bmatrix}$$

10.2.8 beam2gs

Purpose

Compute section forces in a two dimensional nonlinear beam element with geometrical nonlinearity.



Syntax

```
[es, Qx] = beam2gs(ex, ey, ep, ed, Qx)
[es, Qx] = beam2gs(ex, ey, ep, ed, Qx, eq)
[es, Qx, edi] = beam2gs(ex, ey, ep, ed, Qx, eq, n)
[es, Qx, edi, eci] = beam2gs(ex, ey, ep, ed, Qx,␣
↪eq, n)
```

Description

`beam2gs` computes the section forces and displacements in local directions along the geometric nonlinear beam element `beam2ge`.

The input variables `ex`, `ey`, `ep`, `Qx`, and `eq` are described in `beam2ge`. The element displacements, stored in `ed`, are obtained

by the function `extract`. If a distributed transversal load is applied to the element, the variable `eq` must be included. The number of evaluation points for section forces and displacements are determined by `n`. If `n` is omitted, only the ends of the beam are evaluated.

The output variable `Qx` contains $Q_{\bar{x}}$ and the output variables

$$\text{es} = \begin{bmatrix} N(0) & V(0) & M(0) \\ N(\bar{x}_2) & V(\bar{x}_2) & M(\bar{x}_2) \\ \vdots & \vdots & \vdots \\ N(\bar{x}_{n-1}) & V(\bar{x}_{n-1}) & M(\bar{x}_{n-1}) \\ N(L) & V(L) & M(L) \end{bmatrix}$$

$$\text{edi} = \begin{bmatrix} u(0) & v(0) \\ u(\bar{x}_2) & v(\bar{x}_2) \\ \vdots & \vdots \\ u(\bar{x}_{n-1}) & v(\bar{x}_{n-1}) \\ u(L) & v(L) \end{bmatrix}$$

$$\text{eci} = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

contain the section forces, the displacements, and the evaluation points on the local $\bar{x}$-axis. L is the length of the beam element.

Theory

The nodal displacements in local coordinates are given by

$$\bar{\mathbf{a}}^e = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_4 \\ \bar{u}_5 \\ \bar{u}_6 \end{bmatrix} = \mathbf{G}\mathbf{a}^e$$

where $\mathbf{G}$ is described in `beam2ge` and the transpose of $\mathbf{a}^e$ is stored in `ed`.

The displacements associated with bar action and beam action are determined as

$$\bar{\mathbf{a}}^e_{\text{bar}} = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_4 \end{bmatrix}; \qquad \bar{\mathbf{a}}^e_{\text{beam}} = \begin{bmatrix} \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_5 \\ \bar{u}_6 \end{bmatrix}$$

The displacement $u(\bar{x})$ is computed from

$$u(\bar{x}) = \mathbf{N}_{\text{bar}} \bar{\mathbf{a}}^e_{\text{bar}}$$

where

$$\mathbf{N}_{\text{bar}} = \begin{bmatrix} 1 & \bar{x} \end{bmatrix} \mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix} 1 - \frac{\bar{x}}{L} & \frac{\bar{x}}{L} \end{bmatrix}$$

where L is defined in `beam2ge` and

$$\mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix}$$

The displacement $v(\bar{x})$, the rotation $\theta(\bar{x})$, the bending moment $M(\bar{x})$ and the shear force $V(\bar{x})$ are computed from

$$v(\bar{x}) = \mathbf{N}_{\text{beam}} \bar{\mathbf{a}}^e_{\text{beam}} + v_p(\bar{x})$$

$$\theta(\bar{x}) = \frac{d\mathbf{N}_{\text{beam}}}{dx} \bar{\mathbf{a}}^e_{\text{beam}} + \theta_p(\bar{x})$$

$$M(\bar{x}) = D_{EI} \mathbf{B}_{\text{beam}} \bar{\mathbf{a}}^e_{\text{beam}} + M_p(\bar{x})$$

$$V(\bar{x}) = -D_{EI} \frac{d\mathbf{B}_{\text{beam}}}{dx} \bar{\mathbf{a}}^e_{\text{beam}} + V_p(\bar{x})$$

where

$$\mathbf{N}_{\text{beam}} = \begin{bmatrix} 1 & \bar{x} & \bar{x}^2 & \bar{x}^3 \end{bmatrix} \mathbf{C}^{-1}_{\text{beam}}$$

$$\frac{d\mathbf{N}_{\text{beam}}}{dx} = \begin{bmatrix} 0 & 1 & 2\bar{x} & 3\bar{x}^2 \end{bmatrix} \mathbf{C}^{-1}_{\text{beam}}$$

$$\mathbf{B}_{\text{beam}} = \begin{bmatrix} 0 & 0 & 2 & 6\bar{x} \end{bmatrix} \mathbf{C}^{-1}_{\text{beam}}$$

$$\frac{d\mathbf{B}_{\text{beam}}}{dx} = \begin{bmatrix} 0 & 0 & 0 & 6 \end{bmatrix} \mathbf{C}^{-1}_{\text{beam}}$$

$$v_p(\bar{x}) = -\frac{Q_{\bar{x}}}{D_{EI}} \begin{bmatrix} 0 \\ 0 \\ \frac{\bar{x}^4}{12} - \frac{L\bar{x}^3}{6} + \frac{L^2\bar{x}^2}{12} \\ \frac{\bar{x}^5}{20} - \frac{3L^2\bar{x}^3}{20} + \frac{L^3\bar{x}^2}{10} \end{bmatrix}^T \mathbf{C}_{\text{beam}}^{-1}\bar{\mathbf{a}}_{\text{beam}}^e + \frac{q_{\bar{y}}}{D_{EI}} \left(\frac{\bar{x}^4}{24} - \frac{L\bar{x}^3}{12} + \frac{L^2}{2} \right.$$

$$\theta_p(\bar{x}) = -\frac{Q_{\bar{x}}}{D_{EI}} \begin{bmatrix} 0 \\ 0 \\ \frac{\bar{x}^3}{3} - \frac{L\bar{x}^2}{2} + \frac{L^2\bar{x}}{6} \\ \frac{\bar{x}^4}{4} - \frac{9L^2\bar{x}^2}{20} + \frac{L^3\bar{x}}{5} \end{bmatrix}^T \mathbf{C}_{\text{beam}}^{-1}\bar{\mathbf{a}}_{\text{beam}}^e + \frac{q_{\bar{y}}}{D_{EI}} \left(\frac{\bar{x}^3}{6} - \frac{L\bar{x}^2}{4} + \frac{L^2\bar{x}}{12} \right.$$

$$M_p(\bar{x}) = -Q_{\bar{x}} \begin{bmatrix} 0 \\ 0 \\ \bar{x}^2 - L\bar{x} + \frac{L^2}{6} \\ \bar{x}^3 - \frac{9L^2\bar{x}}{10} + \frac{L^3}{5} \end{bmatrix}^T \mathbf{C}_{\text{beam}}^{-1}\bar{\mathbf{a}}_{\text{beam}}^e + q_{\bar{y}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} + \frac{L^2}{12} \right)$$

$$V_p(\bar{x}) = Q_{\bar{x}} \begin{bmatrix} 0 \\ 0 \\ 2\bar{x} - L \\ 3\bar{x}^2 - \frac{9L^2}{10} \end{bmatrix}^T \mathbf{C}_{\text{beam}}^{-1}\bar{\mathbf{a}}_{\text{beam}}^e - q_{\bar{y}} \left(\bar{x} - \frac{L}{2} \right)$$

in which D_{EI}, L, and $q_{\bar{y}}$ are defined in `beam2ge` and

$$\mathbf{C}_{\text{beam}}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{3}{L^2} & -\frac{2}{L} & \frac{3}{L^2} & -\frac{1}{L} \\ \frac{2}{L^3} & \frac{1}{L^2} & -\frac{2}{L^3} & \frac{1}{L^2} \end{bmatrix}$$

An updated value of the axial force is computed as

$$Q_{\bar{x}} = D_{EA} \begin{bmatrix} 0 & 1 \end{bmatrix} \mathbf{C}_{\text{bar}}^{-1}\bar{\mathbf{a}}_{\text{bar}}^e$$

The normal force $N(\bar{x})$ is then computed as

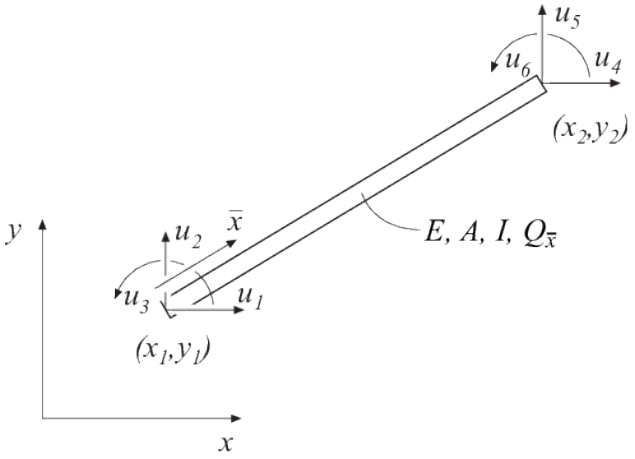
$$N(\bar{x}) = Q_{\bar{x}} + \theta(\bar{x})V(\bar{x})$$

10.2.9 beam2gxe

Purpose

Compute element stiffness matrix for a two dimensional nonlinear beam element with exact solution.

Syntax




```
Ke = beam2gxe(ex, ey, ep, Qx)
[Ke, fe] = beam2gxe(ex, ey, ep, Qx, eq)
```

Description

`beam2gxe` provides the global element stiffness matrix `Ke` for a two dimensional beam element with respect to geometrical nonlinearity considering exact solution.

The input variables:

- `ex = [x1 x2]`
- `ey = [y1 y2]`
- `ep = [E A I]`

supply the element nodal coordinates x_1, y_1, x_2, and y_2, the modulus of elasticity E, the cross section area A, and the moment of inertia I.

- `Qx = [Q_xbar]`

contains the value of the predefined axial force $Q_{\bar{x}}$, which is positive in tension.

The element load vector `fe` can also be computed if a uniformly distributed transverse load is applied to the element. The optional

input variable

- `eq = [q_ybar]`

then contains the distributed transverse load per unit length, $q_{\bar{y}}$. Note that `eq` is a scalar and not a vector as in `beam2e`.

Theory

The element stiffness matrix $\mathbf{K}^e$, stored in the variable `Ke`, is computed according to

$$\mathbf{K}^e = \mathbf{G}^T \bar{\mathbf{K}}^e \mathbf{G}$$

with

$$\bar{\mathbf{K}}^e = \begin{bmatrix} \frac{D_{EA}}{L} & 0 & 0 & -\frac{D_{EA}}{L} & 0 & 0 \\ 0 & \frac{12D_{EI}}{L^3}\phi_5 & \frac{6D_{EI}}{L^2}\phi_2 & 0 & -\frac{12D_{EI}}{L^3}\phi_5 & \frac{6D_{EI}}{L^2}\phi_2 \\ 0 & \frac{6D_{EI}}{L^2}\phi_2 & \frac{4D_{EI}}{L}\phi_3 & 0 & -\frac{6D_{EI}}{L^2}\phi_2 & \frac{2D_{EI}}{L}\phi_4 \\ -\frac{D_{EA}}{L} & 0 & 0 & \frac{D_{EA}}{L} & 0 & 0 \\ 0 & -\frac{12D_{EI}}{L^3}\phi_5 & -\frac{6D_{EI}}{L^2}\phi_2 & 0 & \frac{12D_{EI}}{L^3}\phi_5 & -\frac{6D_{EI}}{L^2}\phi_2 \\ 0 & \frac{6D_{EI}}{L^2}\phi_2 & \frac{2D_{EI}}{L}\phi_4 & 0 & -\frac{6D_{EI}}{L^2}\phi_2 & \frac{4D_{EI}}{L}\phi_3 \end{bmatrix}$$

$$\mathbf{G} = \begin{bmatrix} n_{x\bar{x}} & n_{y\bar{x}} & 0 & 0 & 0 & 0 \\ n_{x\bar{y}} & n_{y\bar{y}} & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & n_{x\bar{x}} & n_{y\bar{x}} & 0 \\ 0 & 0 & 0 & n_{x\bar{y}} & n_{y\bar{y}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

where the axial stiffness D_{EA}, the bending stiffness D_{EI} and the length L are given by

$$D_{EA} = EA; \quad D_{EI} = EI; \quad L = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

The transformation matrix $\mathbf{G}$ contains the direction cosines

$$n_{x\bar{x}} = n_{y\bar{y}} = \frac{x_2 - x_1}{L} \qquad n_{y\bar{x}} = -n_{x\bar{y}} = \frac{y_2 - y_1}{L}$$

For axial compression ($Q_{\bar{x}} < 0$):

$$\phi_2 = \frac{1}{12}\frac{k^2 L^2}{1 - \phi_1} \qquad \phi_3 = \frac{1}{4}\phi_1 + \frac{3}{4}\phi_2$$

$$\phi_4 = -\frac{1}{2}\phi_1 + \frac{3}{2}\phi_2 \qquad \phi_5 = \phi_1 \phi_2$$

with

$$k = \sqrt{\frac{-Q_{\bar{x}}}{D_{EI}}} \qquad \phi_1 = \frac{kL}{2}\cot\frac{kL}{2}$$

For axial tension ($Q_{\bar{x}} > 0$):

$$\phi_2 = -\frac{1}{12}\frac{k^2L^2}{1-\phi_1} \qquad \phi_3 = \frac{1}{4}\phi_1 + \frac{3}{4}\phi_2$$

$$\phi_4 = -\frac{1}{2}\phi_1 + \frac{3}{2}\phi_2 \qquad \phi_5 = \phi_1\phi_2$$

with

$$k = \sqrt{\frac{Q_{\bar{x}}}{D_{EI}}} \qquad \phi_1 = \frac{kL}{2}\coth\frac{kL}{2}$$

The element loads $\mathbf{f}_l^e$ stored in the variable `fe` are computed according to

$$\mathbf{f}_l^e = \mathbf{G}^T\bar{\mathbf{f}}_l^e$$

where

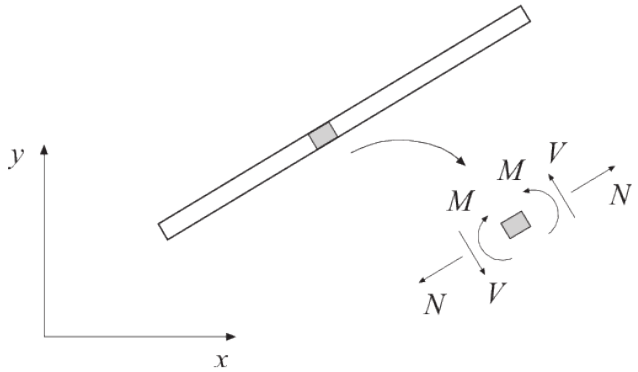
$$\bar{\mathbf{f}}_l^e = qL\begin{bmatrix} 0 \\ \frac{1}{2} \\ \frac{L}{12}\psi \\ 0 \\ \frac{1}{2} \\ -\frac{L}{12}\psi \end{bmatrix}$$

For an axial compressive force ($Q_{\bar{x}} < 0$):

$$\psi = 6\left(\frac{2}{(kL)^2} - \frac{1+\cos kL}{kL\sin kL}\right)$$

and for an axial tensile force ($Q_{\bar{x}} > 0$):

$$\psi = -6\left(\frac{2}{(kL)^2} - \frac{1+\cosh kL}{kL\sinh kL}\right)$$



10.2.10 beam2gxs

Purpose

Compute section forces in a two dimensional geometric nonlinear beam element with exact solution.

Syntax

```
[es,Qx] = beam2gxs(ex, ey, ep, ed, Qx)
[es,Qx] = beam2gxs(ex, ey, ep, ed, Qx, eq)
[es,Qx,edi] = beam2gxs(ex, ey, ep, ed, Qx, eq, n)
[es,Qx,edi,eci] = beam2gxs(ex, ey, ep, ed, Qx, eq,
↪ n)
```

Description

`beam2gxs` computes the section forces and displacements in local directions along the geometric nonlinear beam element `beam2gxe`.

The input variables `ex`, `ey`, `ep`, `Qx`, and `eq` are described in `beam2gxe`. The element displacements, stored in `ed`, are obtained by the function `extract`. If a distributed transversal load is applied to the element, the variable `eq` must be included. The number of evaluation points for section forces and displacements are determined by `n`. If `n` is omitted, only the ends of the beam are evaluated.

The output variable `Qx` contains $Q_{\bar{x}}$ and the output variables

$$\text{es} = \begin{bmatrix} N(0) & V(0) & M(0) \\ N(\bar{x}_2) & V(\bar{x}_2) & M(\bar{x}_2) \\ \vdots & \vdots & \vdots \\ N(\bar{x}_{n-1}) & V(\bar{x}_{n-1}) & M(\bar{x}_{n-1}) \\ N(L) & V(L) & M(L) \end{bmatrix}$$

$$\text{edi} = \begin{bmatrix} u(0) & v(0) \\ u(\bar{x}_2) & v(\bar{x}_2) \\ \vdots & \vdots \\ u(\bar{x}_{n-1}) & v(\bar{x}_{n-1}) \\ u(L) & v(L) \end{bmatrix}$$

$$\text{eci} = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

contain the section forces, the displacements, and the evaluation points on the local $\bar{x}$-axis. L is the length of the beam element.

Theory

The nodal displacements in local coordinates are given by

$$\bar{\mathbf{a}}^e = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_4 \\ \bar{u}_5 \\ \bar{u}_6 \end{bmatrix} = \mathbf{G}\mathbf{a}^e$$

where $\mathbf{G}$ is described in `beam2ge` and the transpose of $\mathbf{a}^e$ is stored in `ed`. The displacements associated with bar action and beam action are determined as

$$\bar{\mathbf{a}}^e_{\text{bar}} = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_4 \end{bmatrix}; \quad \bar{\mathbf{a}}^e_{\text{beam}} = \begin{bmatrix} \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_5 \\ \bar{u}_6 \end{bmatrix}$$

The displacement $u(\bar{x})$ is computed from

$$u(\bar{x}) = \mathbf{N}_{\text{bar}}\bar{\mathbf{a}}^e_{\text{bar}}$$

where

$$\mathbf{N}_{\text{bar}} = \begin{bmatrix} 1 & \bar{x} \end{bmatrix} \mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix} 1 - \frac{\bar{x}}{L} & \frac{\bar{x}}{L} \end{bmatrix}$$

where L is defined in `beam2gxe` and

$$\mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix} 1 & 0 \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix}$$

The displacement $v(\bar{x})$, the rotation $\theta(\bar{x})$, the bending moment $M(\bar{x})$ and the shear force $V(\bar{x})$ are computed from

$$v(\bar{x}) = \mathbf{N}_{\text{beam}}\bar{\mathbf{a}}^e_{\text{beam}} + v_p(\bar{x})$$

$$\theta(\bar{x}) = \frac{d\mathbf{N}_{\text{beam}}}{dx}\bar{\mathbf{a}}^e_{\text{beam}} + \theta_p(\bar{x})$$

$$M(\bar{x}) = D_{EI}\mathbf{B}_{\text{beam}}\bar{\mathbf{a}}^e_{\text{beam}} + M_p(\bar{x})$$

$$V(\bar{x}) = -D_{EI}\frac{d\mathbf{B}_{\text{beam}}}{dx}\bar{\mathbf{a}}^e_{\text{beam}} + V_p(\bar{x})$$

For an axial compressive force ($Q_{\bar{x}} < 0$) we have

$$\mathbf{N}_{\text{beam}} = \begin{bmatrix} 1 & \bar{x} & \cos k\bar{x} & \sin k\bar{x} \end{bmatrix} \mathbf{C}^{-1}_{\text{beam}}$$

$$\frac{d\mathbf{N}_{\text{beam}}}{dx} = \begin{bmatrix} 0 & 1 & -k\sin k\bar{x} & k\cos k\bar{x} \end{bmatrix} \mathbf{C}^{-1}_{\text{beam}}$$

$$\mathbf{B}_{\text{beam}} = \begin{bmatrix} 0 & 0 & -k^2\cos k\bar{x} & -k^2\sin k\bar{x} \end{bmatrix} \mathbf{C}^{-1}_{\text{beam}}$$

$$\frac{d\mathbf{B}_{\text{beam}}}{dx} = \begin{bmatrix} 0 & 0 & k^3\sin k\bar{x} & -k^3\cos k\bar{x} \end{bmatrix} \mathbf{C}^{-1}_{\text{beam}}$$

$$v_p(\bar{x}) = \frac{q_{\bar{y}}L^4}{2D_{EI}}\left[\frac{1+\cos kL}{(kL)^3\sin kL}(-1+\cos k\bar{x}) - \frac{1}{(kL)^3}\sin k\bar{x} + \frac{1}{(kL)^2}\left(\frac{\bar{x}^2}{L^2} - \frac{\bar{x}}{L}\right)\right]$$

$$\theta_p(\bar{x}) = \frac{q_{\bar{y}}L^3}{2D_{EI}}\left[-\frac{1+\cos kL}{(kL)^2\sin kL}\sin k\bar{x} - \frac{1}{(kL)^2}\cos k\bar{x} + \frac{1}{(kL)^2}\left(\frac{2\bar{x}}{L} - 1\right)\right]$$

$$M_p(\bar{x}) = \frac{q_{\bar{y}}L^2}{2}\left[-\frac{1+\cos kL}{(kL)\sin kL}\cos k\bar{x} + \frac{1}{(kL)}\sin k\bar{x} + \frac{2}{(kL)^2}\right]$$

$$V_p(\bar{x}) = Q_{\bar{x}} \begin{bmatrix} 0 \\ 0 \\ 2\bar{x} - L \\ 3\bar{x}^2 - \frac{9L^2}{10} \end{bmatrix}^T \mathbf{C}_{\text{beam}}^{-1} \bar{\mathbf{a}}_{\text{beam}}^e - q_{\bar{y}} \left(\bar{x} - \frac{L}{2} \right)$$

in which D_{EI}, L, and $q_{\bar{y}}$ are defined in `beam2gxe` and

$$\mathbf{C}_{\text{beam}}^{-1} = \begin{bmatrix} k(kL\sin kL + \cos kL - 1) & -kL\cos kL + \sin kL & -k(1-\cos kL \\ -k^2 \sin kL & -k(1-\cos kL) & k^2 \sin kL \\ -k(1-\cos kL) & kL\cos kL - \sin kL & k(1-\cos kL) \\ k\sin kL & kL\sin kL + \cos kL - 1 & -k\sin kL \end{bmatrix}$$

An updated value of the axial force is computed as

$$Q_{\bar{x}} = D_{EA} \begin{bmatrix} 0 & 1 \end{bmatrix} \mathbf{C}_{\text{bar}}^{-1} \bar{\mathbf{a}}_{\text{bar}}^e$$

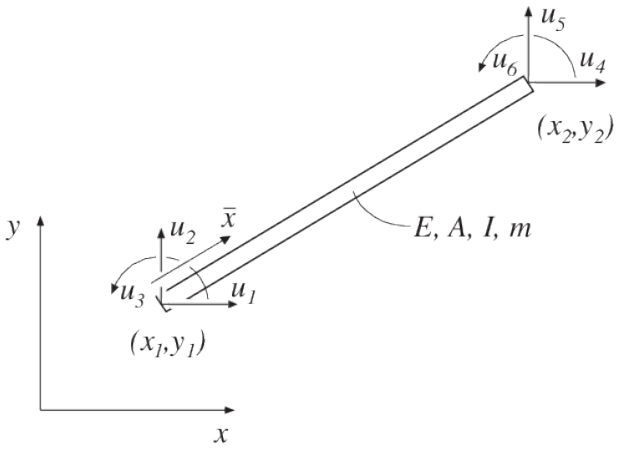
The normal force $N(\bar{x})$ is then computed as

$$N(\bar{x}) = Q_{\bar{x}} + \theta(\bar{x}) V(\bar{x})$$

10.2.11 beam2de

Purpose

Compute element stiffness, mass and damping matrices for a two dimensional beam element.



Syntax

```
[Ke, Me] = beam2de(ex, ey, ep)
[Ke, Me, Ce] = beam2de(ex, ey, ep)
```

Description

`beam2de` provides the global element stiffness matrix `Ke`, the global element mass matrix `Me`, and the global element damping matrix `Ce`, for a two dimensional beam element.

The input variables `ex` and `ey` are described in `beam2e`, and

$$ep = [E,\ A,\ I,\ m,\ [a_0,\ a_1]]$$

contains the modulus of elasticity E, the cross section area A, the moment of inertia I, the mass per unit length m, and the Rayleigh damping coefficients a_0 and a_1. If a_0 and a_1 are omitted, the element damping matrix `Ce` is not computed.

Theory

The element stiffness matrix $\mathbf{K}^e$, the element mass matrix $\mathbf{M}^e$ and the element damping matrix $\mathbf{C}^e$, stored in the variables `Ke`, `Me` and `Ce`, respectively, are computed according to

$$\mathbf{K}^e = \mathbf{G}^T\bar{\mathbf{K}}^e\mathbf{G} \qquad \mathbf{M}^e = \mathbf{G}^T\bar{\mathbf{M}}^e\mathbf{G} \qquad \mathbf{C}^e = \mathbf{G}^T\bar{\mathbf{C}}^e\mathbf{G}$$

where $\mathbf{G}$ and $\bar{\mathbf{K}}^e$ are described in `beam2e`.

The matrix $\bar{\mathbf{M}}^e$ is given by

$$\bar{\mathbf{M}}^e = \frac{mL}{420}\begin{bmatrix} 140 & 0 & 0 & 70 & 0 & 0 \\ 0 & 156 & 22L & 0 & 54 & -13L \\ 0 & 22L & 4L^2 & 0 & 13L & -3L^2 \\ 70 & 0 & 0 & 140 & 0 & 0 \\ 0 & 54 & 13L & 0 & 156 & -22L \\ 0 & -13L & -3L^2 & 0 & -22L & 4L^2 \end{bmatrix}$$

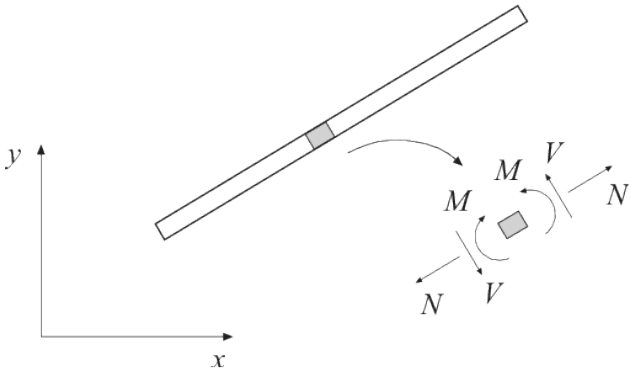
and the matrix $\bar{\mathbf{C}}^e$ is computed by combining $\bar{\mathbf{K}}^e$ and $\bar{\mathbf{M}}^e$:

$$\bar{\mathbf{C}}^e = a_0\bar{\mathbf{M}}^e + a_1\bar{\mathbf{K}}^e$$

10.2.12 beam2ds

Purpose

Compute section forces for a two dimensional beam element in dynamic analysis.



Syntax

```
es = beam2ds(ex, ey, ep, ed, ev, ea)
[es, edi, eci] = beam2gs(ex, ey, ep, ed, ev, ea,␣
↪n)
```

Description

`beam2ds` computes the section forces at the ends of the dynamic beam element `beam2de`.

The input variables `ex`, `ey`, and `ep` are defined in `beam2de`. The element displacements, velocities, and accelerations, stored in `ed`, `ev`, and `ea` respectively, are obtained by the function `extract`.

The output variable `es` contains the section forces at the ends of the beam:

$$es = \begin{bmatrix} N_1 & V_1 & M_1 \\ N_2 & V_2 & M_2 \end{bmatrix}$$

Theory

The section forces at the ends of the beam are obtained from the element force vector:

$$\bar{\mathbf{P}} = \begin{bmatrix} -N_1 & -V_1 & -M_1 & N_2 & V_2 & M_2 \end{bmatrix}^T$$

computed according to:

$$\bar{\mathbf{P}} = \bar{\mathbf{K}}^e \mathbf{G}\mathbf{a}^e + \bar{\mathbf{C}}^e \mathbf{G}\dot{\mathbf{a}}^e + \bar{\mathbf{M}}^e \mathbf{G}\ddot{\mathbf{a}}^e$$

The matrices $\bar{\mathbf{K}}^e$ and $\mathbf{G}$ are described in `beam2e`, and the matrices $\bar{\mathbf{M}}^e$ and $\bar{\mathbf{C}}^e$ are described in `beam2d`.

The nodal displacements:

$$\mathbf{a}^e = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \end{bmatrix}^T$$

shown in `beam2de` also define the directions of the nodal velocities:

$$\dot{\mathbf{a}}^e = \begin{bmatrix} \dot{u}_1 & \dot{u}_2 & \dot{u}_3 & \dot{u}_4 & \dot{u}_5 & \dot{u}_6 \end{bmatrix}^T$$

and the nodal accelerations:

$$\ddot{\mathbf{a}}^e = \begin{bmatrix} \ddot{u}_1 & \ddot{u}_2 & \ddot{u}_3 & \ddot{u}_4 & \ddot{u}_5 & \ddot{u}_6 \end{bmatrix}^T$$

Note that the transposes of $\mathbf{a}^e$, $\dot{\mathbf{a}}^e$, and $\ddot{\mathbf{a}}^e$ are stored in `ed`, `ev`, and `ea` respectively.

10.3 3D beam elements

10.3.1 beam3e

Purpose

Compute element stiffness matrix for a three dimensional beam element.

Syntax

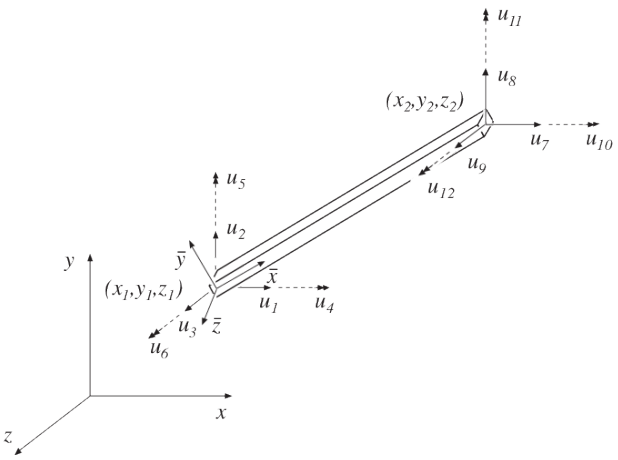
```
Ke = beam3e(ex, ey, ez, eo, ep)
[Ke, fe] = beam3e(ex, ey, ez, eo, ep, eq)
```

Description

`beam3e` provides the global element stiffness matrix `Ke` for a three dimensional beam element.

The input variables

$$\begin{aligned} \text{ex} &= [x_1 \;\; x_2] \\ \text{ey} &= [y_1 \;\; y_2] \qquad \text{eo} = [x_{\bar{z}} \;\; y_{\bar{z}} \;\; z_{\bar{z}}] \\ \text{ez} &= [z_1 \;\; z_2] \end{aligned}$$



supply the element nodal coordinates x_1, y_1, etc. as well as the direction of the local beam coordinate system $(\bar{x}, \bar{y}, \bar{z})$. By giving a global vector $(x_{\bar{z}}, y_{\bar{z}}, z_{\bar{z}})$ parallel with the positive local $\bar{z}$ axis of the beam, the local beam coordinate system is defined.

The variable

$$\text{ep} = [E \;\; G \;\; A \;\; I_{\bar{y}} \;\; I_{\bar{z}} \;\; K_v]$$

supplies the modulus of elasticity E, the shear modulus G, the cross section area A, the moment of inertia with respect to the $\bar{y}$ axis $I_{\bar{y}}$, the moment of inertia with respect to the $\bar{z}$ axis $I_{\bar{z}}$, and St. Venant torsion constant K_v.

The element load vector `fe` can also be computed if uniformly distributed loads are applied to the element. The optional input variable

$$\text{eq} = [q_{\bar{x}} \;\; q_{\bar{y}} \;\; q_{\bar{z}} \;\; q_{\bar{\omega}}]$$

then contains the distributed loads. The positive directions of $q_{\bar{x}}$, $q_{\bar{y}}$, and $q_{\bar{z}}$ follow the local beam coordinate system. The distributed torque $q_{\bar{\omega}}$ is positive if directed in the local $\bar{x}$-direction, i.e. from local $\bar{y}$ to local $\bar{z}$. All the loads are per unit length.

Theory

The element stiffness matrix $\mathbf{K}^e$ is computed according to

$$\mathbf{K}^e = \mathbf{G}^T \bar{\mathbf{K}}^e \mathbf{G}$$

where

$$\bar{\mathbf{K}}^e = \begin{bmatrix}
\frac{D_{EA}}{L} & 0 & 0 & 0 & 0 & 0 & -\frac{D_{EA}}{L} \\
0 & \frac{12D_{EI_{\bar{z}}}}{L^3} & 0 & 0 & 0 & \frac{6D_{EI_{\bar{z}}}}{L^2} & 0 \\
0 & 0 & \frac{12D_{EI_{\bar{y}}}}{L^3} & 0 & -\frac{6D_{EI_{\bar{y}}}}{L^2} & 0 & 0 \\
0 & 0 & 0 & \frac{D_{GK}}{L} & 0 & 0 & 0 \\
0 & 0 & -\frac{6D_{EI_{\bar{y}}}}{L^2} & 0 & \frac{4D_{EI_{\bar{y}}}}{L} & 0 & 0 \\
0 & \frac{6D_{EI_{\bar{z}}}}{L^2} & 0 & 0 & 0 & \frac{4D_{EI_{\bar{z}}}}{L} & 0 \\
-\frac{D_{EA}}{L} & 0 & 0 & 0 & 0 & 0 & \frac{D_{EA}}{L} \\
0 & -\frac{12D_{EI_{\bar{z}}}}{L^3} & 0 & 0 & 0 & -\frac{6D_{EI_{\bar{z}}}}{L^2} & 0 \\
0 & 0 & -\frac{12D_{EI_{\bar{y}}}}{L^3} & 0 & \frac{6D_{EI_{\bar{y}}}}{L^2} & 0 & 0 \\
0 & 0 & 0 & -\frac{D_{GK}}{L} & 0 & 0 & 0 \\
0 & 0 & -\frac{6D_{EI_{\bar{y}}}}{L^2} & 0 & \frac{2D_{EI_{\bar{y}}}}{L} & 0 & 0 \\
0 & \frac{6D_{EI_{\bar{z}}}}{L^2} & 0 & 0 & 0 & \frac{2D_{EI_{\bar{z}}}}{L} & 0
\end{bmatrix}$$

and

$$\mathbf{G} = \begin{bmatrix}
n_{x\bar{x}} & n_{y\bar{x}} & n_{z\bar{x}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
n_{x\bar{y}} & n_{y\bar{y}} & n_{z\bar{y}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
n_{x\bar{z}} & n_{y\bar{z}} & n_{z\bar{z}} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & n_{x\bar{x}} & n_{y\bar{x}} & n_{z\bar{x}} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & n_{x\bar{y}} & n_{y\bar{y}} & n_{z\bar{y}} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & n_{x\bar{z}} & n_{y\bar{z}} & n_{z\bar{z}} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & n_{x\bar{x}} & n_{y\bar{x}} & n_{z\bar{x}} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & n_{x\bar{y}} & n_{y\bar{y}} & n_{z\bar{y}} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & n_{x\bar{z}} & n_{y\bar{z}} & n_{z\bar{z}} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n_{x\bar{x}} & n_{y\bar{x}} & n_{z\bar{x}} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n_{x\bar{y}} & n_{y\bar{y}} & n_{z\bar{y}} \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & n_{x\bar{z}} & n_{y\bar{z}} & n_{z\bar{z}}
\end{bmatrix}$$

where the axial stiffness D_{EA}, the bending stiffness $D_{EI_{\bar{z}}}$, the bending stiffness $D_{EI_{\bar{y}}}$, and the St. Venant torsion stiffness D_{GK} are given by

$$D_{EA} = EA; \quad D_{EI_{\bar{z}}} = EI_{\bar{z}}; \quad D_{EI_{\bar{y}}} = EI_{\bar{y}}; \quad D_{GK} = GK_v$$

The length L is given by

$$L = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

The transformation matrix $\mathbf{G}$ contains direction cosines computed as

$$
\begin{aligned}
n_{x\bar{x}} &= \frac{x_2 - x_1}{L} \qquad n_{y\bar{x}} = \frac{y_2 - y_1}{L} \qquad n_{z\bar{x}} = \frac{z_2 - z_1}{L} \\
n_{x\bar{z}} &= \frac{x_{\bar{z}}}{L_{\bar{z}}} \qquad n_{y\bar{z}} = \frac{y_{\bar{z}}}{L_{\bar{z}}} \qquad n_{z\bar{z}} = \frac{z_{\bar{z}}}{L_{\bar{z}}} \\
n_{x\bar{y}} &= n_{y\bar{z}} n_{z\bar{x}} - n_{z\bar{z}} n_{y\bar{x}} \\
n_{y\bar{y}} &= n_{z\bar{z}} n_{x\bar{x}} - n_{x\bar{z}} n_{z\bar{x}} \\
n_{z\bar{y}} &= n_{x\bar{z}} n_{y\bar{x}} - n_{y\bar{z}} n_{x\bar{x}}
\end{aligned}
$$

where

$$
L_{\bar{z}} = \sqrt{x_{\bar{z}}^2 + y_{\bar{z}}^2 + z_{\bar{z}}^2}
$$

The element load vector $\mathbf{f}_l^e$, stored in `fe`, is computed according to

$$
\mathbf{f}_l^e = \mathbf{G}^T \bar{\mathbf{f}}_l^e
$$

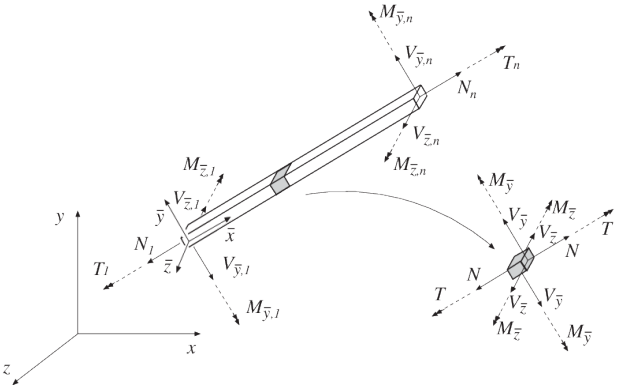
where

$$
\bar{\mathbf{f}}_l^e = \begin{bmatrix}
\frac{q_{\bar{x}} L}{2} \\
\frac{q_{\bar{y}} L}{2} \\
\frac{q_{\bar{z}} L}{2} \\
\frac{q_{\bar{\omega}} L}{2} \\
-\frac{q_{\bar{z}} L^2}{12} \\
\frac{q_{\bar{y}} L^2}{12} \\
\frac{q_{\bar{x}} L}{2} \\
\frac{q_{\bar{y}} L}{2} \\
\frac{q_{\bar{z}} L}{2} \\
\frac{q_{\bar{\omega}} L}{2} \\
\frac{q_{\bar{z}} L^2}{12} \\
-\frac{q_{\bar{y}} L^2}{12}
\end{bmatrix}
$$

10.3.2 beam3s

Purpose

Compute section forces in a three dimensional beam element.



Syntax

```
[es]=beam3s(ex,ey,ez,eo,ep,ed)
[es]=beam3s(ex,ey,ez,eo,ep,ed,eq)
[es,edi]=beam3s(ex,ey,ez,eo,ep,ed,eq,n)
[es,edi,eci]=beam3s(ex,ey,ez,eo,ep,ed,eq,n)
```

Description

`beam3s` computes the section forces and displacements in local directions along the beam element `beam3e`.

The input variables `ex`, `ey`, `ez`, `eo`, `ep`, and `eq` are defined in `beam3e`.

The element displacements, stored in `ed`, are obtained by the function `extract`. If a distributed load is applied to the element, the variable `eq` must be included. The number of evaluation points for section forces and displacements are determined by `n`. If `n` is omitted, only the ends of the beam are evaluated.

The output variables:

$$\text{es} = \begin{bmatrix} N(0) & V_{\bar{y}}(0) & V_{\bar{z}}(0) & T(0) & M_{\bar{y}}(0) & M_{\bar{z}}(0) \\ N(\bar{x}_2) & V_{\bar{y}}(\bar{x}_2) & V_{\bar{z}}(\bar{x}_2) & T(\bar{x}_2) & M_{\bar{y}}(\bar{x}_2) & M_{\bar{z}}(\bar{x}_2) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ N(\bar{x}_{n-1}) & V_{\bar{y}}(\bar{x}_{n-1}) & V_{\bar{z}}(\bar{x}_{n-1}) & T(\bar{x}_{n-1}) & M_{\bar{y}}(\bar{x}_{n-1}) & M_{\bar{z}}(\bar{x}_{n-1}) \\ N(L) & V_{\bar{y}}(L) & V_{\bar{z}}(L) & T(\bar{x}_{n-1}) & M_{\bar{y}}(L) & M_{\bar{z}}(L) \end{bmatrix}$$

$$\text{edi} = \begin{bmatrix} u(0) & v(0) & w(0) & \varphi(0) \\ u(\bar{x}_2) & v(\bar{x}_2) & w(\bar{x}_2) & \varphi(\bar{x}_2) \\ \vdots & \vdots & \vdots & \vdots \\ u(\bar{x}_{n-1}) & v(\bar{x}_{n-1}) & w(\bar{x}_{n-1}) & \varphi(\bar{x}_{n-1}) \\ u(L) & v(L) & w(L) & \varphi(L) \end{bmatrix} \qquad \text{eci} = \begin{bmatrix} 0 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_{n-1} \\ L \end{bmatrix}$$

contain the section forces, the displacements, and the evaluation points on the local $\bar{x}$-axis. L is the length of the beam element.

Theory

The nodal displacements in local coordinates are given by

$$\bar{\mathbf{a}}^e = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_2 \\ \bar{u}_3 \\ \bar{u}_4 \\ \bar{u}_5 \\ \bar{u}_6 \\ \bar{u}_7 \\ \bar{u}_8 \\ \bar{u}_9 \\ \bar{u}_{10} \\ \bar{u}_{11} \\ \bar{u}_{12} \end{bmatrix} = \mathbf{G}\mathbf{a}^e$$

where $\mathbf{G}$ is described in `beam3e` and the transpose of $\mathbf{a}^e$ is stored in `ed`.

The displacements associated with bar action, beam action in the $\bar{x}\bar{y}$-plane, beam action in the $\bar{x}\bar{z}$-plane, and torsion are determined as

$$\bar{\mathbf{a}}^e_{\text{bar}} = \begin{bmatrix} \bar{u}_1 \\ \bar{u}_7 \end{bmatrix}; \quad \bar{\mathbf{a}}^e_{\text{beam},\bar{z}} = \begin{bmatrix} \bar{u}_2 \\ \bar{u}_6 \\ \bar{u}_8 \\ \bar{u}_{12} \end{bmatrix}; \quad \bar{\mathbf{a}}^e_{\text{beam},\bar{y}} = \begin{bmatrix} \bar{u}_3 \\ -\bar{u}_5 \\ \bar{u}_9 \\ -\bar{u}_{11} \end{bmatrix}; \quad \bar{\mathbf{a}}^e_{\text{torsion}} = \begin{bmatrix} \bar{u}_4 \\ \bar{u}_{10} \end{bmatrix}$$

The displacement $u(\bar{x})$ and the normal force $N(\bar{x})$ are computed from

$$u(\bar{x}) = \mathbf{N}_{\text{bar}}\bar{\mathbf{a}}^e_{\text{bar}} + u_p(\bar{x})$$

$$N(\bar{x}) = D_{EA}\mathbf{B}_{\text{bar}}\bar{\mathbf{a}}^e + N_p(\bar{x})$$

where

$$\mathbf{N}_{\text{bar}} = \begin{bmatrix}1 & \bar{x}\end{bmatrix}\mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix}1 - \frac{\bar{x}}{L} & \frac{\bar{x}}{L}\end{bmatrix}$$

$$\mathbf{B}_{\text{bar}} = \begin{bmatrix}0 & 1\end{bmatrix}\mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix}-\frac{1}{L} & \frac{1}{L}\end{bmatrix}$$

$$u_p(\bar{x}) = -\frac{q_{\bar{x}}}{D_{EA}}\left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2}\right)$$

$$N_p(\bar{x}) = -q_{\bar{x}}\left(\bar{x} - \frac{L}{2}\right)$$

in which D_{EA}, L, and $q_{\bar{x}}$ are defined in `beam3e` and

$$\mathbf{C}^{-1}_{\text{bar}} = \begin{bmatrix}1 & 0\\ -\frac{1}{L} & \frac{1}{L}\end{bmatrix}$$

The displacement $v(\bar{x})$, the bending moment $M_{\bar{z}}(\bar{x})$ and the shear force $V_{\bar{y}}(\bar{x})$ are computed from

$$v(\bar{x}) = \mathbf{N}_{\text{beam}}\bar{\mathbf{a}}^e_{\text{beam},\bar{z}} + v_p(\bar{x})$$

$$M_{\bar{z}}(\bar{x}) = D_{EI_{\bar{z}}}\mathbf{B}_{\text{beam}}\bar{\mathbf{a}}^e_{\text{beam},\bar{z}} + M_{\bar{z},p}(\bar{x})$$

$$V_{\bar{y}}(\bar{x}) = -D_{EI_{\bar{z}}}\frac{d\mathbf{B}_{\text{beam}}}{dx}\bar{\mathbf{a}}^e_{\text{beam},\bar{z}} + V_{\bar{y},p}(\bar{x})$$

where

$$\mathbf{N}_{\text{beam}} = \begin{bmatrix}1 & \bar{x} & \bar{x}^2 & \bar{x}^3\end{bmatrix}\mathbf{C}^{-1}_{\text{beam}}$$

$$\mathbf{B}_{\text{beam}} = \begin{bmatrix}0 & 0 & 2 & 6\bar{x}\end{bmatrix}\mathbf{C}^{-1}_{\text{beam}}$$

$$\frac{d\mathbf{B}_{\text{beam}}}{dx} = \begin{bmatrix}0 & 0 & 0 & 6\end{bmatrix}\mathbf{C}^{-1}_{\text{beam}}$$

$$v_p(\bar{x}) = \frac{q_{\bar{y}}}{D_{EI_{\bar{z}}}}\left(\frac{\bar{x}^4}{24} - \frac{L\bar{x}^3}{12} + \frac{L^2\bar{x}^2}{24}\right)$$

$$M_{\bar{z},p}(\bar{x}) = q_{\bar{y}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} + \frac{L^2}{12} \right)$$

$$V_{\bar{y},p}(\bar{x}) = -q_{\bar{y}} \left(\bar{x} - \frac{L}{2} \right)$$

in which $D_{EI_{\bar{z}}}$, L, and $q_{\bar{y}}$ are defined in `beam3e` and

$$\mathbf{C}_{\text{beam}}^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{3}{L^2} & -\frac{2}{L} & \frac{3}{L^2} & -\frac{1}{L} \\ \frac{2}{L^3} & \frac{1}{L^2} & -\frac{2}{L^3} & \frac{1}{L^2} \end{bmatrix}$$

The displacement $w(\bar{x})$, the bending moment $M_{\bar{y}}(\bar{x})$ and the shear force $V_{\bar{z}}(\bar{x})$ are computed from

$$w(\bar{x}) = \mathbf{N}_{\text{beam}} \bar{\mathbf{a}}_{\text{beam},\bar{y}}^e + w_p(\bar{x})$$

$$M_{\bar{y}}(\bar{x}) = -D_{EI_{\bar{y}}} \mathbf{B}_{\text{beam}} \bar{\mathbf{a}}_{\text{beam},\bar{y}}^e + M_{\bar{y},p}(\bar{x})$$

$$V_{\bar{z}}(\bar{x}) = -D_{EI_{\bar{y}}} \frac{d\mathbf{B}_{\text{beam}}}{dx} \bar{\mathbf{a}}_{\text{beam},\bar{y}}^e + V_{\bar{z},p}(\bar{x})$$

where

$$w_p(\bar{x}) = \frac{q_{\bar{z}}}{D_{EI_{\bar{y}}}} \left(\frac{\bar{x}^4}{24} - \frac{L\bar{x}^3}{12} + \frac{L^2\bar{x}^2}{24} \right)$$

$$M_{\bar{y},p}(\bar{x}) = -q_{\bar{z}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} + \frac{L^2}{12} \right)$$

$$V_{\bar{z},p}(\bar{x}) = -q_{\bar{z}} \left(\bar{x} - \frac{L}{2} \right)$$

in which $D_{EI_{\bar{y}}}$, L, and $q_{\bar{z}}$ are defined in `beam3e` and $\mathbf{N}_{\text{beam}}$, $\mathbf{B}_{\text{beam}}$, and $\frac{d\mathbf{B}_{\text{beam}}}{dx}$ are given above.

The displacement $\varphi(\bar{x})$ and the torque $T(\bar{x})$ are computed from

$$\varphi(\bar{x}) = \mathbf{N}_{\text{torsion}} \bar{\mathbf{a}}_{\text{torsion}}^e + \varphi_p(\bar{x})$$

$$T(\bar{x}) = D_{GK} \mathbf{B}_{\text{torsion}} \bar{\mathbf{a}}^e + T_p(\bar{x})$$

where

$$\mathbf{N}_{\text{torsion}} = \mathbf{N}_{\text{bar}}$$

$$\mathbf{B}_{\text{torsion}} = \mathbf{B}_{\text{bar}}$$

$$\varphi_p(\bar{x}) = -\frac{q_\omega}{D_{GK}} \left(\frac{\bar{x}^2}{2} - \frac{L\bar{x}}{2} \right)$$

$$T_p(\bar{x}) = -q_\omega \left(\bar{x} - \frac{L}{2} \right)$$

in which D_{GK}, L, and q_ω are defined in `beam3e`.

CHAPTER

ELEVEN

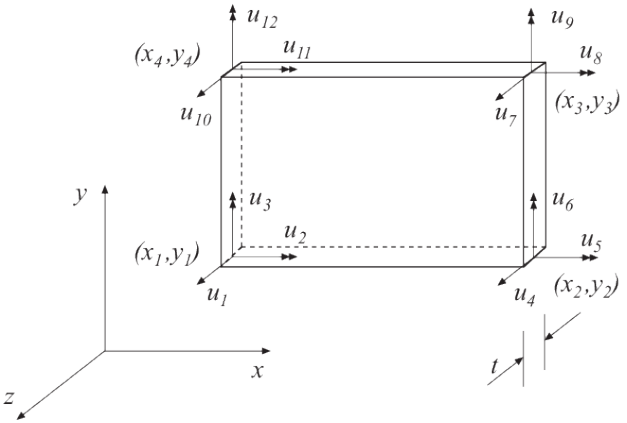
PLATE ELEMENT FUNCTIONS

Only one plate element is currently available, a rectangular 12 dof element. The element presumes a linear elastic material which can be isotropic or anisotropic.

11.1 platre

Purpose

Compute element stiffness matrix for a rectangular plate element.



Syntax

```
Ke = platre(ex, ey, ep, D)
[Ke, fe] = platre(ex, ey, ep, D, eq)
```

Description

`platre` provides an element stiffness matrix Ke, and an element load vector fe, for a rectangular plate element. This element can only be used if the element edges are parallel to the coordinate axes.

The element nodal coordinates x_1, y_1, x_2 etc. are supplied to the function by $\mathbf{ex}$ and $\mathbf{ey}$, the element thickness t by $\mathbf{ep}$, and the material properties by the constitutive matrix D. Any arbitrary $\mathbf{D}$-matrix with dimensions (3×3) and valid for plane stress may be given. For an isotropic elastic material the constitutive matrix can be formed by the function `hooke` (see Section *Material functions*).

$$\mathbf{ex} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \end{bmatrix} \qquad \mathbf{ep} = \begin{bmatrix} t \end{bmatrix} \qquad \mathbf{D} = \begin{bmatrix} D_{11} & D_{12} & D_{13} \\ D_{21} & D_{22} & D_{23} \\ D_{31} & D_{32} & D_{33} \end{bmatrix}$$
$$\mathbf{ey} = \begin{bmatrix} y_1 & y_2 & y_3 & y_4 \end{bmatrix}$$

If a uniformly distributed load is applied to the element, the element load vector fe is computed. The input variable

$$\mathbf{eq} = \begin{bmatrix} q_z \end{bmatrix}$$

then contains the load q_z per unit area in the z-direction.

Theory

The element stiffness matrix $\mathbf{K}^e$ and the element load vector $\mathbf{f}_l^e$, stored in Ke and fe respectively, are computed according to

$$\mathbf{K}^e = (\mathbf{C}^{-1})^T \int_A \bar{\mathbf{B}}^T \tilde{\mathbf{D}} \, \bar{\mathbf{B}} \, dA \, \mathbf{C}^{-1}$$

$$\mathbf{f}_l^e = (\mathbf{C}^{-1})^T \int_A \bar{\mathbf{N}}^T q_z \, dA$$

where the constitutive matrix

$$\tilde{\mathbf{D}} = \frac{t^3}{12} \mathbf{D}$$

and where $\mathbf{D}$ is defined by D.

The evaluation of the integrals for the rectangular plate element is based on the displacement approximation $w(x, y)$ and is expressed in terms of the nodal variables $u_1, u_2, \ldots, u_{12}$ as

$$w(x, y) = \mathbf{N}^e \mathbf{a}^e = \bar{\mathbf{N}} \mathbf{C}^{-1} \mathbf{a}^e$$

where

$$\bar{\mathbf{N}} = [\,1 \;\; x \;\; y \;\; x^2 \;\; xy \;\; y^2 \;\; x^3 \;\; x^2y \;\; xy^2 \;\; y^3 \;\; x^3y \;\; xy^3\,]$$

$$\mathbf{C} = \begin{bmatrix}
1 & -a & -b & a^2 & ab & b^2 & -a^3 & -a^2b & -ab^2 & -b^3 & a^3b \\
0 & 0 & 1 & 0 & -a & -2b & 0 & a^2 & 2ab & 3b^2 & -a^3 \\
0 & -1 & 0 & 2a & b & 0 & -3a^2 & -2ab & -b^2 & 0 & 3a^2b \\
1 & a & -b & a^2 & -ab & b^2 & a^3 & -a^2b & ab^2 & -b^3 & -a^3b \\
0 & 0 & 1 & 0 & a & -2b & 0 & a^2 & -2ab & 3b^2 & a^3 \\
0 & -1 & 0 & -2a & b & 0 & -3a^2 & 2ab & -b^2 & 0 & 3a^2b \\
1 & a & b & a^2 & ab & b^2 & a^3 & a^2b & ab^2 & b^3 & a^3b \\
0 & 0 & 1 & 0 & a & 2b & 0 & a^2 & 2ab & 3b^2 & a^3 \\
0 & -1 & 0 & -2a & -b & 0 & -3a^2 & -2ab & -b^2 & 0 & -3a^2b \\
1 & -a & b & a^2 & -ab & b^2 & -a^3 & a^2b & -ab^2 & b^3 & -a^3b \\
0 & 0 & 1 & 0 & -a & 2b & 0 & a^2 & -2ab & 3b^2 & -a^3 \\
0 & -1 & 0 & 2a & -b & 0 & -3a^2 & 2ab & -b^2 & 0 & -3a^2b
\end{bmatrix}$$

$$\mathbf{a}^e = [\,u_1 \;\; u_2 \;\; \ldots \;\; u_{12}\,]^T$$

and where

$$a = \frac{1}{2}(x_3 - x_1) \qquad b = \frac{1}{2}(y_3 - y_1)$$

The matrix $\bar{\mathbf{B}}$ is obtained as

$$\bar{\mathbf{B}} = \overset{*}{\nabla}\,\bar{\mathbf{N}} = \begin{bmatrix}
0 & 0 & 0 & 2 & 0 & 0 & 6x & 2y & 0 & 0 & 6xy & 0 \\
0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 2x & 6y & 0 & 6xy \\
0 & 0 & 0 & 0 & 2 & 0 & 0 & 4x & 4y & 0 & 6x^2 & 6y^2
\end{bmatrix}$$

where

$$\overset{*}{\nabla} = \begin{bmatrix}
\frac{\partial^2}{\partial x^2} \\
\frac{\partial^2}{\partial y^2} \\
2\frac{\partial^2}{\partial x \partial y}
\end{bmatrix}$$

Evaluation of the integrals for the rectangular plate element is done analytically. Computation of the integrals for the element load vec-

tor $\mathbf{f}_l^e$ yields

$$\mathbf{f}_l^e = q_z L_x L_y \begin{bmatrix} \frac{1}{4} \\ \frac{L_y}{24} \\ -\frac{L_x}{24} \\ \frac{1}{4} \\ \frac{L_y}{24} \\ \frac{L_x}{24} \\ \frac{1}{4} \\ -\frac{L_y}{24} \\ \frac{L_x}{24} \\ \frac{1}{4} \\ -\frac{L_y}{24} \\ -\frac{L_x}{24} \end{bmatrix}$$

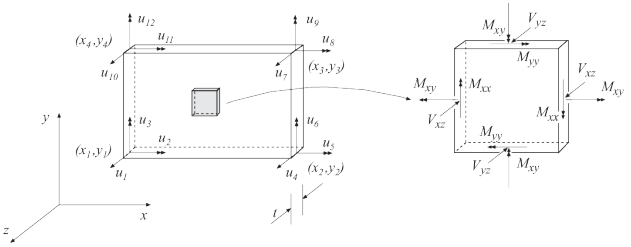
where

$$L_x = x_3 - x_1 \qquad L_y = y_3 - y_1$$

11.2 platrs

Purpose

Compute section forces in a rectangular plate element.



Syntax

```
[es,et]=platrs(ex,ey,ep,D,ed)
```

Description

`platrs` computes the section forces `es` and the curvature matrix `et` in a rectangular plate element. The section forces and the curvatures are computed at the center of the element.

The input variables $\mathbf{ex}$, $\mathbf{ey}$, $\mathbf{ep}$ and $\mathbf{D}$ are defined in `platre`. The vector $\mathbf{ed}$ contains the nodal displacements $\mathbf{a}^e$ of the element and is obtained by the function `extract` as

$$\mathbf{ed} = (\mathbf{a}^e)^T = [\, u_1 \;\; u_2 \;\; \ldots \;\; u_{12} \,]$$

The output variables

$$\mathtt{es} = \left[\, \mathbf{M}^T \; \mathbf{V}^T \,\right] = [\, M_{xx} \; M_{yy} \; M_{xy} \; V_{xz} \; V_{yz} \,]$$

$$\mathtt{et} = \boldsymbol{\kappa}^T = [\, \kappa_{xx} \; \kappa_{yy} \; \kappa_{xy} \,]$$

contain the section forces and curvatures in global directions.

Theory

The curvatures and the section forces are computed according to

$$\boldsymbol{\kappa} = \begin{bmatrix} \kappa_{xx} \\ \kappa_{yy} \\ \kappa_{xy} \end{bmatrix} = \bar{\mathbf{B}}\, \mathbf{C}^{-1}\, \mathbf{a}^e$$

$$\mathbf{M} = \begin{bmatrix} M_{xx} \\ M_{yy} \\ M_{xy} \end{bmatrix} = \tilde{\mathbf{D}}\, \boldsymbol{\kappa}$$

$$\mathbf{V} = \begin{bmatrix} V_{xz} \\ V_{yz} \end{bmatrix} = \tilde{\nabla}\, \mathbf{M}$$

where the matrices $\tilde{\mathbf{D}}$, $\bar{\mathbf{B}}$, $\mathbf{C}$ and $\mathbf{a}^e$ are described in `platre`, and where

$$\tilde{\nabla} = \begin{bmatrix} \dfrac{\partial}{\partial x} & 0 & \dfrac{\partial}{\partial y} \\ 0 & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial x} \end{bmatrix}$$

CHAPTER

TWELVE

SYSTEM FUNCTIONS

The group of system functions comprises functions for the setting up, solving, and elimination of systems of equations. The functions are separated in two groups:

- Static system functions
- Dynamic system functions

Static system functions concern the linear system of equations

$$\mathbf{Ka} = \mathbf{f}$$

where $\mathbf{K}$ is the global stiffness matrix and $\mathbf{f}$ is the global load vector. Often used static system functions are `assem` and `solveq`. The function `assem` assembles the global stiffness matrix and `solveq` computes the global displacement vector $\mathbf{a}$ considering the boundary conditions. It should be noted that $\mathbf{K}$, $\mathbf{f}$, and $\mathbf{a}$ also represent analogous quantities in systems other than structural mechanical systems. For example, in a heat flow problem $\mathbf{K}$ represents the conductivity matrix, $\mathbf{f}$ the heat flow, and $\mathbf{a}$ the temperature.

Dynamic system functions are related to different aspects of linear dynamic systems of coupled ordinary differential equations according to

$$\mathbf{C}\dot{\mathbf{a}} + \mathbf{Ka} = \mathbf{f}$$

for first-order systems and

$$\mathbf{M}\ddot{\mathbf{a}} + \mathbf{C}\dot{\mathbf{a}} + \mathbf{Ka} = \mathbf{f}$$

for second-order systems. First-order systems occur typically in transient heat conduction and second-order systems occur in structural dynamics.

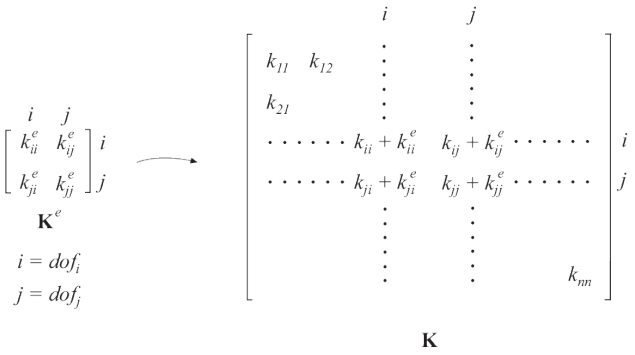
12.1 Static system functions

The group of static system functions comprises functions for setting up and solving the global system of equations. It also contains a function for eigenvalue analysis, a function for static condensation, a function for extraction of element displacements from the global displacement vector and a function for extraction of element coordinates.

12.1.1 assem

Purpose

Assemble element matrices.



Syntax

```
K = assem(edof, K, Ke)
[K, f] = assem(edof, K, Ke, f, fe)
```

Description

`assem` adds the element stiffness matrix $\mathbf{K}^e$, stored in `Ke`, to the structure stiffness matrix $\mathbf{K}$, stored in `K`, according to the topology matrix $\mathbf{edof}$.

The element topology matrix $\mathbf{edof}$ is defined as

$$\mathbf{edof} = [el \quad \underbrace{dof_1 \quad dof_2 \quad \ldots \quad dof_{ned}}_{\text{global dof.}}]$$

where the first column contains the element number, and columns

2 to $(ned+1)$ contain the corresponding global degrees of freedom (ned = number of element degrees of freedom).

In the case where the matrix $\mathbf{K}^e$ is identical for several elements, assembling of these can be carried out simultaneously. Each row in **Edof** then represents one element, i.e. nel is the total number of considered elements.

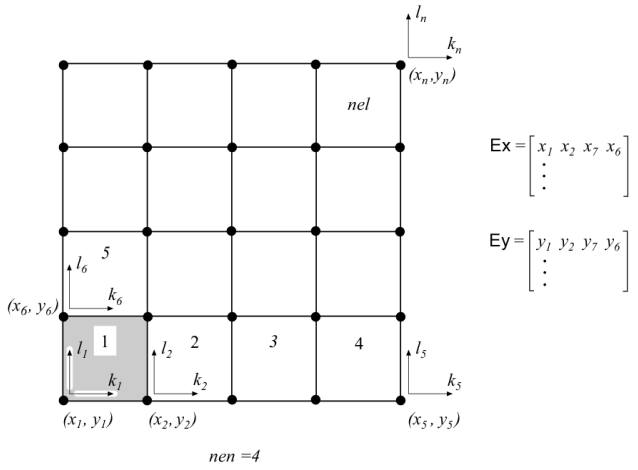
$$\mathbf{Edof} = \left[\begin{array}{ccccccc} el_1 & dof_1 & dof_2 & \cdots & \cdots & \cdots & dof_{ned} \\ el_2 & dof_1 & dof_2 & \cdots & \cdots & \cdots & dof_{ned} \\ \vdots & \vdots & \vdots & & & & \vdots \\ el_{nel} & dof_1 & dof_2 & \cdots & \cdots & \cdots & dof_{ned} \end{array}\right] \Bigg\} \text{one row for each}$$

If **fe** and $\mathbf{f}$ are given in the function, the element load vector $\mathbf{f}^e$ is also added to the global load vector $\mathbf{f}$.

12.1.2 coordxtr

Purpose

Extract element coordinates from a global coordinate matrix.



Syntax

```
[Ex, Ey, Ez] = coordxtr(Edof, Coord, Dof, nen)
```

Description

`coordxtr` extracts element nodal coordinates from the global coordinate matrix `Coord` for elements with equal numbers of element nodes and dof's.

Input variables are the element topology matrix `Edof`, defined in `assem`, the global coordinate matrix `Coord`, the global topology matrix `Dof`, and the number of element nodes `nen` in each element.

$$\mathbf{Coord} = \begin{bmatrix} x_1 & y_1 & [z_1] \\ x_2 & y_2 & [z_2] \\ x_3 & y_3 & [z_3] \\ \vdots & \vdots & \vdots \\ x_n & y_n & [z_n] \end{bmatrix} \qquad \mathbf{Dof} = \begin{bmatrix} k_1 & l_1 & \dots & m_1 \\ k_2 & l_2 & \dots & m_2 \\ k_3 & l_3 & \dots & m_3 \\ \vdots & \vdots & \dots & \vdots \\ k_n & l_n & \dots & m_n \end{bmatrix} \qquad nen = [\ nen$$

The nodal coordinates defined in row *i* of `Coord` correspond to the degrees of freedom of row *i* in `Dof`. The components k_i, l_i and m_i define the degrees of freedom of node *i*, and *n* is the number of global nodes for the considered part of the FE-model.

The output variables `Ex`, `Ey`, and `Ez` are matrices defined according to

$$\mathbf{Ex} = \begin{bmatrix} x_1^1 & x_2^1 & x_3^1 & \dots & x_{nen}^1 \\ x_1^2 & x_2^2 & x_3^2 & \dots & x_{nen}^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{nel} & x_2^{nel} & x_3^{nel} & \dots & x_{nen}^{nel} \end{bmatrix}$$

where row *i* gives the *x*-coordinates of the element defined in row *i* of `Edof`, and where *nel* is the number of considered elements.

The element coordinate data extracted by the function `coordxtr` can be used for plotting purposes and to create input data for the element stiffness functions.

> **Note**
>
> For the two dimensional beam element, the `extract` function will extract six nodal displacements for each element given in the first column vector in `Edof` and store them in the variable

`ed` as

$$\mathbf{ed} = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \\ u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \end{bmatrix}$$

12.1.3 eigen

Purpose

Solve the generalized eigenvalue problem.

Syntax

```
L = eigen(K, M)
L = eigen(K, M, b)
[L, X] = eigen(K, M)
[L, X] = eigen(K, M, b)
```

Description

`eigen` solves the eigenvalue problem

$$|\mathbf{K} - \lambda\mathbf{M}| = 0$$

where $\mathbf{K}$ and $\mathbf{M}$ are square matrices. The eigenvalues λ are stored in the vector $\mathbf{L}$ and the corresponding eigenvectors in the matrix $\mathbf{X}$.

If certain rows and columns in matrices $\mathbf{K}$ and $\mathbf{M}$ are to be eliminated in computing the eigenvalues, $\mathbf{b}$ must be given in the function. The rows (and columns) that are to be eliminated are described in the vector $\mathbf{b}$ defined as

$$\mathbf{b} = \begin{bmatrix} dof_1 \\ dof_2 \\ \vdots \\ dof_{nb} \end{bmatrix}$$

The computed eigenvalues are given in order ranging from the smallest to the largest. The eigenvectors are normalized so that

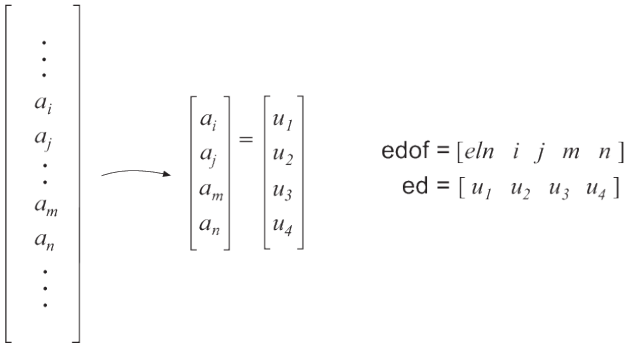
$$\mathbf{X}^T\mathbf{M}\mathbf{X} = \mathbf{I}$$

where $\mathbf{I}$ is the identity matrix.

12.1.4 extract_ed

Purpose

Extract element nodal quantities from a global solution vector.



Syntax

```
ed = extract_ed(edof, a)
```

Description

The `extract_ed` function extracts element displacements or corresponding quantities $\mathbf{a}^e$ from the global solution vector $\mathbf{a}$, stored in `a`.

Input variables are the element topology matrix $\mathbf{edof}$, defined in `assem`, and the global solution vector `a`.

The output variable

$$\mathbf{ed} = (\mathbf{a}^e)^T$$

contains the element displacement vector.

If $\mathbf{Edof}$ contains more than one element, $\mathbf{Ed}$ will be a matrix

$$\mathbf{Ed} = \begin{bmatrix} (\mathbf{a}^e)_1^T \\ (\mathbf{a}^e)_2^T \\ \vdots \\ (\mathbf{a}^e)_{nel}^T \end{bmatrix}$$

where row *i* gives the element displacements for the element defined in row *i* of `Edof`, and *nel* is the total number of considered elements.

Example

For the two-dimensional beam element, the `extract` function will extract six nodal displacements for each element given in $\mathbf{Edof}$, and create a matrix `Ed` of size *(nel × 6)*.

$$\mathbf{Ed} = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \\ u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_1 & u_2 & u_3 & u_4 & u_5 & u_6 \end{bmatrix}$$

12.1.5 red

Purpose

Reduce the size of a square matrix by omitting rows and columns.

Syntax

```
B = red(A, b)
[B, b] = red(A, b)
```

Description

`B = red(A, b)` reduces the square matrix `A` to a smaller matrix `B` by omitting rows and columns of `A`. The indices for rows and columns to be omitted are specified by the column vector `b`.

Example

Assume that the matrix `A` is defined as

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{bmatrix}$$

and `b` as

$$b = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$$

The statement `B = red(A, b)` results in the matrix

$$B = \begin{bmatrix} 1 & 3 \\ 9 & 11 \end{bmatrix}$$

12.1.6 solveq

Purpose

Solve equation system.

Syntax

```
a = solveq(K, f)
a = solveq(K, f, bc)
[a, r] = solveq(K, f, bc)
```

Description

The function `solveq` solves the equation system

$$\mathbf{K}\,\mathbf{a} = \mathbf{f}$$

where $\mathbf{K}$ is a matrix and $\mathbf{a}$ and $\mathbf{f}$ are vectors.

The matrix $\mathbf{K}$ and the vector $\mathbf{f}$ must be predefined. The solution of the system of equations is stored in a vector $\mathbf{a}$ which is created by the function.

If some values of $\mathbf{a}$ are to be prescribed, the row number and the corresponding values are given in the boundary condition matrix

$$\mathbf{bc} = \begin{bmatrix} dof_1 & u_1 \\ dof_2 & u_2 \\ \vdots & \vdots \\ dof_{nbc} & u_{nbc} \end{bmatrix}$$

where the first column contains the row numbers and the second column the corresponding prescribed values.

If `r` is given in the function, support forces are computed according to

$$\mathbf{r} = \mathbf{K}\,\mathbf{a} - \mathbf{f}$$

12.1.7 statcon

Purpose

Reduce system of equations by static condensation.

Syntax

```
[K1, f1] = statcon(K, f)
[K1, f1] = statcon(K, f, b)
[K1, f1] = statcon(K, f, b, bc)
```

Description

`statcon` reduces a system of equations

$$\mathbf{K}\ \mathbf{a} = \mathbf{f}$$

by static condensation.

The degrees of freedom to be eliminated are supplied to the function by the vector

$$\mathbf{b} = \begin{bmatrix} dof_1 \\ dof_2 \\ \vdots \\ dof_{nb} \end{bmatrix}$$

where each row in `b` contains one degree of freedom to be eliminated.

The elimination gives the reduced system of equations

$$\mathbf{K}_1\ \mathbf{a}_1 = \mathbf{f}_1$$

where $\mathbf{K}_1$ and $\mathbf{f}_1$ are stored in `K1` and `f1` respectively.

12.2 Dynamic system functions

The group of system functions comprises functions for solving linear dynamic systems by time stepping or modal analysis, functions for frequency domain analysis, etc.

12.2.1 dyna2

Compute the dynamic solution to a set of uncoupled second-order differential equations.

Syntax

```
X = dyna2(w2, xi, f, g, dt)
```

Description

dyna2 computes the solution to the set

$$\ddot{x}_i + 2\xi_i\omega_i\dot{x}_i + \omega_i^2 x_i = f_i g(t), \qquad i = 1, \ldots, m$$

of differential equations, where $g(t)$ is a piecewise linear time function.

The vectors w2, xi, and f contain the squared circular frequencies ω_i^2, the damping ratios ξ_i, and the applied forces f_i, respectively.

The vector g defines the load function in terms of straight line segments between equally spaced points in time. This function may have been formed by the command gfunc.

The dynamic solution is computed at equal time increments defined by dt. Including the initial zero vector as the first column vector, the result is stored in the $m \times n$ matrix X, where $n-1$ is the number of time steps.

Note

The accuracy of the solution is *not* a function of the output time increment dt, since the command produces the exact solution for straight line segments in the loading time function.

See also

gfunc

12.2.2 dyna2f

Purpose

Compute the dynamic solution to a set of uncoupled second-order differential equations.

Syntax

```
Y = dyna2f(w2, xi, f, p, dt)
```

Description

`dyna2f` computes the solution to the set of differential equations:

$$\ddot{x}_i + 2\xi_i\omega_i\dot{x}_i + \omega_i^2 x_i = f_i g(t), \qquad i = 1, \ldots, m$$

The vectors `w2`, `xi` and `f` are the squared circular frequencies ω_i^2, the damping ratios ξ_i, and the applied forces f_i, respectively. The force vector `p` contains the Fourier coefficients $p(k)$ formed by the command `fft`.

The solution in the frequency domain is computed at equal time increments defined by `dt`. The result is stored in the $m \times n$ matrix `Y`, where `m` is the number of equations and `n` is the number of frequencies resulting from the `fft` command. The dynamic solution in the time domain is achieved by the use of the command `ifft`.

Example

The dynamic solution to a set of uncoupled second-order differential equations can be computed by the following sequence of commands:

```
>> g = gfunc(G, dt);
>> p = fft(g);
>> Y = dyna2f(w2, xi, f, p, dt);
>> X = (real(ifft(Y.')))';
```

where it is assumed that the input variables `G`, `dt`, `w2`, `xi` and `f` are properly defined. Note that the `ifft` command operates on column vectors if `Y` is a matrix; therefore use the transpose of `Y`. The output from the `ifft` command is complex. Therefore use `Y.'` to transpose rows and columns in `Y` in order to avoid the complex conjugate transpose of `Y`.

The time response is represented by the real part of the output from the `ifft` command. If the transpose is used and the result is stored in a matrix `X`, each row will represent the time response for each equation as the output from the command `dyna2`.

See also

gfunc, fft, ifft

12.2.3 fft

Purpose

Transform functions in time domain to frequency domain.

Syntax

```
p = fft(g)
p = fft(g, N)
```

Description

The `fft` function transforms a time dependent function to the frequency domain.

The function to be transformed is stored in the vector `g`. Each row in `g` contains the value of the function at equal time intervals. The function represents a span $-\infty \leq t \leq +\infty$; however, only the values within a typical period are specified by `g`.

The `fft` command can be used with one or two input arguments. If `N` is not specified, the number of frequencies used in the transformation is equal to the number of points in the time domain (i.e., the length of the variable `g`), and the output will be a vector of the same size containing complex values representing the frequency content of the input signal.

The scalar variable `N` can be used to specify the number of frequencies used in the Fourier transform. The size of the output vector in this case will be equal to `N`. It should be remembered that the highest harmonic component in the time signal that can be identified by the Fourier transform corresponds to half the sampling frequency. The sampling frequency is equal to $1/dt$, where dt is the time increment of the time signal.

The complex Fourier coefficients $p(k)$ are stored in the vector `p`, and are computed according to

$$p(k) = \sum_{j=1}^{N} x(j)\omega_N^{(j-1)(k-1)},$$

where

$$\omega_N = e^{-2\pi i/N}.$$

> **Note**
>
> This is a MATLAB built-in function.

12.2.4 freqresp

Purpose

Compute frequency response of a known discrete time response.

Syntax

```
[Freq, Resp] = freqresp(D, dt)
```

Description

`freqresp` computes the frequency response of a discrete dynamic system.

- `D` is the time history function.
- `dt` is the sampling time increment, i.e., the time increment used in the time integration procedure.
- `Resp` contains the computed response as a function of frequency.
- `Freq` contains the corresponding frequencies.

Example

The result can be visualized by:

```
plot(Freq, Resp)
xlabel('frequency (Hz)')
```

or:

```
semilogy(Freq, Resp)
xlabel('frequency (Hz)')
```

The dimension of `Resp` is the same as that of the original time history function.

Note

The time history function of a discrete system computed by direct integration often behaves in an unstructured manner. The reason for this is that the time history is a mixture of several participating eigenmodes at different eigenfrequencies. By using a Fourier transform, however, the response as a function of frequency can be computed efficiently. In particular, it is possible to identify the participating frequencies.

12.3 gfunc

Purpose

Form vector with function values at equally spaced points by linear interpolation.

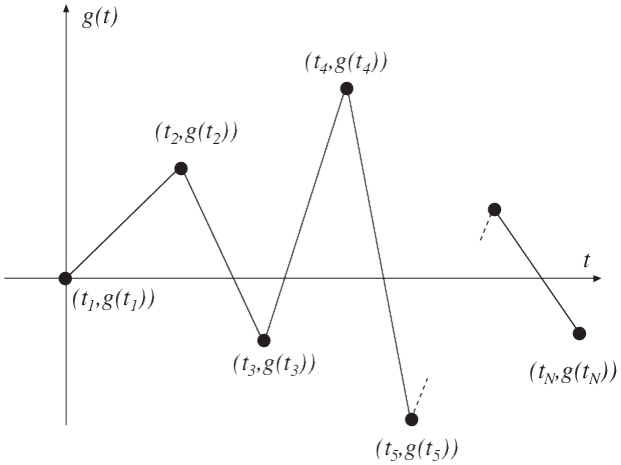


Fig. 1: Piecewise linear time dependent function

Syntax

```
[t, g] = gfunc(G, dt)
```

Description

`gfunc` uses linear interpolation to compute values at equally spaced points for a discrete function g given by

$$G = \begin{bmatrix} t_1 & g(t_1) \\ t_2 & g(t_2) \\ \vdots & \\ t_N & g(t_N) \end{bmatrix},$$

as shown in the figure above.

Function values are computed in the range $t_1 \leq t \leq t_N$, at equal increments, `dt` being defined by the variable `dt`. The number of linear segments (steps) is $(t_N - t_1)/dt$. The corresponding vector `t` is also computed. The result can be plotted by using the command `plot(t, g)`.

12.3.1 ifft

Purpose

Transform function in frequency domain to time domain.

Syntax

```
x = ifft(y)
x = ifft(y, N)
```

Description

`ifft` transforms a function in the frequency domain to a function in the time domain.

The function to be transformed is given in the vector `y`. Each row in `y` contains Fourier terms in the interval $-\infty \leq \omega \leq +\infty$.

The `ifft` command can be used with one or two input arguments. The scalar variable `N` can be used to specify the number of frequencies used in the Fourier transform. The size of the output vector in this case will be equal to `N`. See also the description of the command `fft`.

The inverse Fourier coefficients $x(j)$, stored in the variable `x`, are computed according to

$$x(j) = \frac{1}{N} \sum_{k=1}^{N} y(k) \omega_N^{-(j-1)(k-1)},$$

where

$$\omega_N = e^{-2\pi i/N}.$$

> **Note**
>
> This is a MATLAB built-in function.

See also
fft

12.3.2 ritz

Purpose
Compute approximative eigenvalues and eigenvectors by the Lanczos method.

Syntax

```
L = ritz(K, M, f, m)
L = ritz(K, M, f, m, b)
[L, X] = ritz(K, M, f, m)
[L, X] = ritz(K, M, f, m, b)
```

Description
`ritz` computes, by the use of the Lanczos algorithm, `m` approximative eigenvalues and `m` corresponding eigenvectors for a given pair of *n*-by-*n* matrices `K` and `M` and a given non-zero starting vector `f`.

If certain rows and columns in matrices $\mathbf{K}$ and $\mathbf{M}$ are to be eliminated in computing the eigenvalues, $\mathbf{b}$ must be given in the command. The rows (and columns) to be eliminated are described in the vector $\mathbf{b}$ defined as

$$\mathbf{b} = \begin{bmatrix} dof_1 \\ dof_2 \\ \vdots \\ dof_{nb} \end{bmatrix}$$

> **Note**
>
> If the number of vectors, `m`, is chosen less than the total number of degrees-of-freedom, n, only about the first `m/2` Ritz vectors are good approximations of the true eigenvectors. Recall that the Ritz vectors satisfy the `M`-orthonormality condition

$$\mathbf{X}^T\mathbf{M}\mathbf{X} = \mathbf{I}$$

where $\mathbf{I}$ is the identity matrix.

12.3.3 spectra

Purpose

Compute seismic response spectra for elastic design.

Syntax

```
s = spectra(a, xi, dt, f)
```

Description

The `spectra` function computes the seismic response spectrum for a known acceleration history function.

- The computation is based on the vector `a`, which contains an acceleration time history function defined at equal time steps.
- The time step is specified by the variable `dt`.
- The value of the damping ratio is given by the variable `xi`.
- Output from the computation, stored in the vector `s`, is achieved at frequencies specified by the column vector `f`.

Example

The following procedure can be used to produce a seismic response spectrum for a damping ratio $\xi = 0.05$, defined at 34 logarithmically spaced frequency points. The acceleration time history `a` has been sampled at a frequency of 50 Hz, corresponding to a time increment `dt = 0.02` between collected points:

```
freq = logspace(0, log10(2^(33/6)), 34);
xi = 0.05;
dt = 0.02;
s = spectra(a, xi, dt, freq');
```

The resulting spectrum can be plotted by the command:

```
loglog(freq, s, '*')
```

12.3.4 step1

Purpose

Compute the dynamic solution to a set of first order differential equations.

Syntax

```
[a,da] = step1(K, C, f, a0, bc, ip)
[a,da] = step1(K, C, f, a0, bc, ip, times)
[a,da, ahist, dahist] = step1(K, C, f, a0, bc, ip,
↪ times, dofs)
```

Description

`step1` computes at equal time steps the solution to a set of first order differential equations of the form:

$$\mathbf{C}\dot{\mathbf{a}} + \mathbf{K}\mathbf{a} = \mathbf{f}(x,t),$$
$$\mathbf{a}(0) = \mathbf{a}_0$$

The command solves transient field problems. In the case of heat conduction, `K` and `C` represent the $n \times n$ conductivity and capacity matrices, respectively. `a` is the temperature and `da` (i.e., $\dot{\mathbf{a}}$) is the time derivative of the temperature.

The matrix `f` contains the time-dependent load vectors. If no external loads are active, use `[]` for `f`. The matrix `f` is organized as follows:

```
f = [
time history of the load at dof_1
```

(continues on next page)

(continued from previous page)

```
time history of the load at dof_2
...
time history of the load at dof_n
]
```

The dimension of `f` is:

> (number of degrees-of-freedom) × (number of timesteps + 1)

The initial conditions are given by the vector `a0` containing initial values of `a`.

The matrix `bc` contains the time-dependent prescribed values of the field variable `a`. If no field variables are prescribed, use `[]` for `bc`. The matrix `bc` is organized as follows:

```
bc = [
dof_1   time history of the field variable
dof_2   time history of the field variable
...
dof_m2  time history of the field variable
]
```

The dimension of `bc` is:

> (number of dofs with prescribed field values) × (number of timesteps + 2)

The time integration procedure is governed by the parameters given in the vector `ip` defined as:

```
ip = [dt, T, alpha]
```

where `dt` specifies the length of the time increment, `T` is the total time, and `alpha` is the time integration constant. Frequently used values of `alpha` are:

alpha	Method
0	Forward difference; forward Euler
0.5	Trapezoidal rule; Crank-Nicholson
1	Backward difference; backward Euler

The computed values of `a` and `da` are stored in `a` and `da`, respectively. The first column contains the initial values, and the following columns contain the values for each time step. The dimension is:

> (number of degrees-of-freedom) × (number of time steps + 1)

If the values are to be stored only for specific times, the parameter `times` specifies at which times the solution will be stored. The values are stored in `a` and `da`, one column for each requested time according to `times`. The dimension is then:

> (number of degrees-of-freedom) × (number of requested times + 1)

If the history is to be stored in `ahist` and `dahist` for some degrees of freedom, the parameter `dofs` specifies for which degrees of freedom the history is to be stored. The computed time histories are stored in `ahist` and `dahist`, respectively, with one row for each requested degree of freedom. The dimension is:

> (number of specified degrees of freedom) × (number of timesteps + 1)

The time history functions can be generated using the command `gfunc`. If all the values of the time histories of `f` or `bc` are kept constant, these values need to be stated only once. In this case, the number of columns in `f` is one and in `bc` two.

In most cases, only a few degrees-of-freedom are affected by the exterior load, and hence the matrix contains only a few non-zero entries. In such cases, it is possible to save space by defining `f` as `sparse` (a MATLAB built-in function).

Note

Reference: Bathe, K.J.: *Finite Element Procedures in Engineering Analysis*, Prentice-Hall, Englewood Cliffs, New Jersey, pp. 511-514, 1982.

12.3.5 step2

Purpose

Compute the dynamic solution to a set of second order differential equations.

Syntax

```
[a, da, d2a] = step2(K, C, M, f, a0, da0, bc, ip)
[a, da, d2a] = step2(K, C, M, f, a0, da0, bc, ip,␣
↪times)
[a, da, d2a, ahist, dahist, d2ahist] = step2(K, C,
↪ M, f, a0, da0, bc, ip, times, dofs)
```

Description

`step2` computes at equal time steps the solution to a set of second order differential equations of the form:

$$\begin{aligned} \mathbf{M}\ddot{\mathbf{a}} + \mathbf{C}\dot{\mathbf{a}} + \mathbf{K}\mathbf{a} &= \mathbf{f}(x,t), \\ \mathbf{a}(0) &= \mathbf{a}_0, \\ \dot{\mathbf{a}}(0) &= \mathbf{v}_0. \end{aligned}$$

In structural mechanics problems, `K`, `C` and `M` represent the $n \times n$ stiffness, damping and mass matrices, respectively. `a` is the displacement, `da` (= $\dot{\mathbf{a}}$) is the velocity and `d2a` (= $\ddot{\mathbf{a}}$) is the acceleration.

The matrix `f` contains the time-dependent load vectors. If no external loads are active, use `[]` for `f`. The matrix `f` is organized as:

$$f = \begin{bmatrix} \text{time history of the load at } dof_1 \\ \text{time history of the load at } dof_2 \\ \vdots \\ \text{time history of the load at } dof_n \end{bmatrix}$$

The dimension of `f` is:

> (number of degrees-of-freedom) × (number of timesteps + 1)

The initial conditions are given by the vectors `a0` and `da0`, containing initial displacements and initial velocities.

The matrix `bc` contains the time-dependent prescribed displacement. If no displacements are prescribed, use `[]` for `bc`. The matrix `bc` is organized as:

$$bc = \begin{bmatrix} dof_1 & \text{time history of the displacement} \\ dof_2 & \text{time history of the displacement} \\ \vdots & \vdots \\ dof_{m_2} & \text{time history of the displacement} \end{bmatrix}$$

The dimension of `bc` is:

> (number of dofs with prescribed displacement) × (number of timesteps + 2)

The time integration procedure is governed by the parameters given in the vector `ip` defined as:

$$ip = [dt, T, \alpha, \delta]$$

where `dt` specifies the time increment, `T` the total time, and `alpha` and `delta` are time integration constants for the Newmark family of methods.

Frequently used values:

α	δ	Method
$\frac{1}{4}$	$\frac{1}{2}$	Average acceleration (trapezoidal) rule
$\frac{1}{6}$	$\frac{1}{2}$	Linear acceleration
0	$\frac{1}{2}$	Central difference

The computed values of $\mathbf{a}$, $\dot{\mathbf{a}}$ and $\ddot{\mathbf{a}}$ are stored in `a`, `da` and `d2a`, respectively. The first column contains the initial values and the following columns contain the values for each time step.

The dimension of `a`, `da` and `d2a` is:

(number of degrees-of-freedom) × (number of time steps + 1)

If the values are to be stored only for specific times, the parameter `times` specifies at which times the solution will be stored. The values are stored in `a`, `da` and `d2a`, one column for each requested time according to `times`. The dimension is then:

(number of degrees-of-freedom) × (number of requested times + 1)

If the history is to be stored in `ahist`, `dahist` and `d2ahist` for some degrees of freedom, the parameter `dofs` specifies for which degrees of freedom the history is to be stored. The computed time histories are stored in `ahist`, `dahist` and `d2ahist`, one row for each requested degree of freedom according to `dofs`. The dimension is:

(number of specified degrees of freedom) × (number of timesteps + 1)

In most cases only a few degrees-of-freedom are affected by the exterior load, and hence the matrix contains only few non-zero entries. In such cases it is possible to save space by defining `f` as sparse (a MATLAB built-in function).

12.3.6 sweep

Purpose

Compute complex frequency response functions.

Syntax

```
Y = sweep(K, C, M, p, w)
```

Description

`sweep` computes the complex frequency response function for a system of the form:

$$[\mathbf{K} + i\omega\mathbf{C} - \omega^2\mathbf{M}]\mathbf{y}(\omega) = \mathbf{p}$$

Here, `K`, `C`, and `M` represent the *m*-by-*m* stiffness, damping, and mass matrices, respectively. The vector `p` defines the amplitude of the

force. The frequency response function is computed for the values of ω given by the vector `w`.

The complex frequency response function is stored in the matrix `Y` with dimension m-by-n, where n is equal to the number of circular frequencies defined in `w`.

Example

The steady-state response can be computed by:

```
X = real(Y * exp(i * w * t));
```

and the amplitude by:

```
Z = abs(Y)
```

CHAPTER

THIRTEEN

STATEMENTS

CHAPTER

FOURTEEN

GRAPHICS FUNCTIONS

The group of graphics functions comprises functions for element based graphics. Mesh plots, displacements, section forces, flows, iso lines and principal stresses can be displayed. The functions are divided into two dimensional, and general graphics functions.

14.1 Two dimensional graphics functions

14.1.1 dispbeam2

Purpose

Draw the displacements for a two dimensional beam element.

Syntax

```
[sfac] = dispbeam2(ex, ey, edi)
[sfac] = dispbeam2(ex, ey, edi, plotpar)
dispbeam2(ex, ey, edi, plotpar, sfac)
```

Description

Input variables are the coordinate matrices ex and ey, see e.g. $beam2e$, and the element displacements edi obtained by e.g. $beam2s$.

The variable $plotpar$ sets plot parameters for linetype, linecolour and node marker:

$$plotpar = [\,linetype \;\; linecolor \;\; nodemark\,]$$

where

$linetype =$	solid line	$linecolor =$	black
1		1	
2	dashed line	2	blue
3	dotted line	3	magenta
		4	red

$nodemark = 1$	circle
2	star
0	no mark

Default is dashed black lines with circles at nodes.

The scale factor $sfac$ is a scalar that the element displacements are multiplied with to get a suitable geometrical representation. If $sfac$ is omitted in the input list, the scale factor is set automatically.

14.1.2 eldraw2

Purpose

Draw the undeformed mesh for a two dimensional structure.

Syntax

```
eldraw2(Ex, Ey)
eldraw2(Ex, Ey, plotpar)
eldraw2(Ex, Ey, plotpar, elnum)
```

Description

`eldraw2` displays the undeformed mesh for a two dimensional structure.

Input variables are the coordinate matrices `Ex` and `Ey` formed by the function `coordxtr`.

The variable `plotpar` sets plot parameters for linetype, linecolor and node marker:

$$\texttt{plotpar} = [\,\text{linetype}\;\;\text{linecolor}\;\;\text{nodemark}\,]$$

linetype	meaning	linecolor	meaning
1	solid line	1	black
2	dashed line	2	blue
3	dotted line	3	magenta
		4	red

nodemark	meaning
1	circle
2	star
0	no mark

Default is solid black lines with circles at nodes.

Element numbers can be displayed at the center of the element if a column vector `elnum` with the element numbers is supplied. This column vector can be derived from the element topology matrix `Edof`:

$$\texttt{elnum} = \texttt{Edof}(:, 1)$$

i.e. the first column of the topology matrix.

Limitations

Supported elements are bar, beam, triangular three node, and quadrilateral four node elements.

14.1.3 eldisp2

Purpose

Draw the deformed mesh for a two dimensional structure.

Syntax

```
[sfac] = eldisp2(Ex, Ey, Ed)
[sfac] = eldisp2(Ex, Ey, Ed, plotpar)
eldisp2(Ex, Ey, Ed, plotpar, sfac)
```

Description

eldisp2 displays the deformed mesh for a two dimensional structure.

Input variables are the coordinate matrices Ex and Ey formed by the function coordxtr, and the element displacements Ed formed by the function extract.

The variable plotpar sets plot parameters for linetype, linecolor and node marker:

$$\text{plotpar} = [\,\text{linetype} \quad \text{linecolor} \quad \text{nodemark}\,]$$

where

linetype	line style	linecolor	color
1	solid line	1	black
2	dashed line	2	blue
3	dotted line	3	magenta
		4	red

nodemark	marker
1	circle
2	star
0	no mark

Default is dashed black lines with circles at nodes.

The scale factor sfac is a scalar that the element displacements are multiplied with to get a suitable geometrical representation. The scale factor is set automatically if it is omitted in the input list.

Limitations

Supported elements are bar, beam, triangular three node, and quadrilateral four node elements.

14.1.4 elflux2

Purpose

Draw element flow arrows for two dimensional elements.

Syntax

```
[sfac] = elflux2(Ex, Ey, Es)
[sfac] = elflux2(Ex, Ey, Es, plotpar)
elflux2(Ex, Ey, Es, plotpar, sfac)
```

Description

elflux2 displays element heat flux vectors (or corresponding quantities) for a number of elements of the same type. The flux vectors are displayed as arrows at the element centroids. Note that only the flux vectors are displayed. To display the element mesh, use eldraw2.

Input variables are the coordinate matrices Ex and Ey, and the element flux matrix Es defined in flw2ts or flw2qs.

The variable plotpar sets plot parameters for the flux arrows:

plotpar = [arrowtype arrowcolor]

arrowtype	Line style	*arrowcolor*	Color
1	solid	1	black
2	dashed	2	blue
3	dotted	3	magenta
		4	red

Default, if plotpar is omitted, is solid black arrows.

The scale factor sfac is a scalar that the values are multiplied with to get a suitable arrow size in relation to the element size. The scale factor is set automatically if it is omitted in the input list.

Limitations

Supported elements are triangular 3 node and quadrilateral 4 node elements.

14.1.5 eliso2

Purpose

Display element iso lines for two dimensional elements.

Syntax

```
eliso2(Ex, Ey, Ed, isov)
eliso2(Ex, Ey, Ed, isov, plotpar)
```

Description

`eliso2` displays element iso lines for a number of elements of the same type. Note that only the iso lines are displayed. To display the element mesh, use `eldraw2`.

Input variables are the coordinate matrices `Ex` and `Ey` formed by the function `coordxtr`, and the element nodal quantities (e.g., displacement or energy potential) matrix `Ed` defined in `extract`.

If `isov` is a scalar, it determines the number of iso lines to be displayed. If `isov` is a vector, it determines the values of the iso lines to be displayed (number of iso lines equal to the length of vector `isov`):

`isov` $= [\,\text{iso lines}\,]$

`isov` $= [\,\text{isovalue}(1)\ \ldots\ \text{isovalue}(n)\,]$

The variable `plotpar` sets plot parameters for the iso lines:

`plotpar` $= [\,\text{linetype}\ \ \text{linecolor}\ \ \text{textfcn}\,]$

- linetype: 1 = solid, 2 = dashed, 3 = dotted
- linecolor: 1 = black, 2 = blue, 3 = magenta, 4 = red
- textfcn: 0 = iso values not printed, 1 = iso values printed at the iso lines, 2 = iso values printed where the cursor indicates

Default is solid, black lines and no iso values printed.

Limitations

Supported elements are triangular 3 node and quadrilateral 4 node elements.

14.1.6 elprinc2

Purpose

Draw element principal stresses as arrows for two dimensional elements.

Syntax

```
[sfac] = elprinc2(Ex, Ey, Es)
[sfac] = elprinc2(Ex, Ey, Es, plotpar)
elprinc2(Ex, Ey, Es, plotpar, sfac)
```

Description

elprinc2 displays element principal stresses for a number of elements of the same type. The principal stresses are displayed as arrows at the element centroids. Note that only the principal stresses are displayed. To display the element mesh, use eldraw2.

Input variables are the coordinate matrices Ex and Ey, and the element stresses matrix Es defined in plants or planqs.

The variable plotpar sets plot parameters for the principal stress arrows:

$$\text{plotpar} = [\,\text{arrowtype} \;\; \text{arrowcolor}\,]$$

where

arrowtype =	solid	arrowcolor	black
1		1	
2	dashed	2	blue
3	dotted	3	magenta
		4	red

Default, if plotpar is omitted, is solid black arrows.

The scale factor sfac is a scalar that values are multiplied with to get a suitable arrow size in relation to the element size. The scale factor is set automatically if it is omitted in the input list.

Limitations

Supported elements are triangular 3 node and quadrilateral 4 node elements.

14.1.7 scalfact2

Purpose

Determine scale factor for drawing computational results.

Syntax

```
[sfac] = scalfact2(ex, ey, ed)
[sfac] = scalfact2(ex, ey, ed, rat)
```

Description

`scalfact2()` determines a scale factor $sfac$ for drawing computational results, such as displacements, section forces, or flux.

Input variables are the coordinate matrices ex and ey, and the matrix ed containing the quantity to be displayed. The scalar rat defines the ratio between the geometric representation of the largest quantity to be displayed and the element size. If rat is not specified, 0.2 is used.

Theory

The scale factor $sfac$ is computed so that the largest value in ed is represented as a fraction rat of the element size, ensuring a visually appropriate scaling of computational results.

14.1.8 scalgraph2

Purpose

Draw a graphic scale.

Syntax

```
scalgraph2(sfac, magnitude)
scalgraph2(sfac, magnitude, plotpar)
```

Description

`scalgraph2` draws a graphic scale to visualize the magnitude of displayed computational results. The input variable $sfac$ is a scale factor determined by the function `scalfact2`. The variable $magnitude$ is defined as $[S \quad x \quad y]$, where S specifies the value corresponding to the length of the graphic scale, and (x, y) are the coordinates of the starting point. If no coordinates are given, the starting point will be $(0, -0.5)$.

Theory

The variable `plotpar` sets the graphic scale color:

$$\text{plotpar} = [\text{color}]$$

where

color = 1	black
2	blue
3	magenta
4	red

14.1.9 secforce2

Purpose

Draw the section force diagrams of a two dimensional bar or beam element in its global position.

Syntax

```
secforce2(ex, ey, es, plotpar, sfac)
secforce2(ex, ey, es, plotpar, sfac, eci)
[sfac] = secforce2(ex, ey, es)
[sfac] = secforce2(ex, ey, es, plotpar)
```

Description

The input variables `ex` and `ey` are defined in `bar2e` or `beam2e`. The input variable

$$\texttt{es} = \begin{bmatrix} S_1 \\ S_2 \\ \vdots \\ S_n \end{bmatrix}$$

consists of a column matrix that contains section forces. The values in `es` are computed in, e.g., `bar2s` or `beam2s`.

The variable `plotpar` sets plot parameters for the diagram:

$$\texttt{plotpar} = [\,\text{linecolor} \;\; \text{elementcolor}\,]$$

where

linecolor	color	element-color	color
1	black	1	black
2	blue	2	blue
3	magenta	3	magenta
4	red	4	red

The scale factor `sfac` is a scalar that the section forces are multiplied with to get a suitable graphical representation. If `sfac` is omitted in the input list, the scale factor is set automatically.

The input variable

$$\texttt{eci} = \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_n \end{bmatrix}$$

specifies the local $\bar{x}$-coordinates of the quantities in `es`. If `eci` is not given, uniform distance is assumed.

CHAPTER

FIFTEEN

EXAMPLES

INDEX

B

C