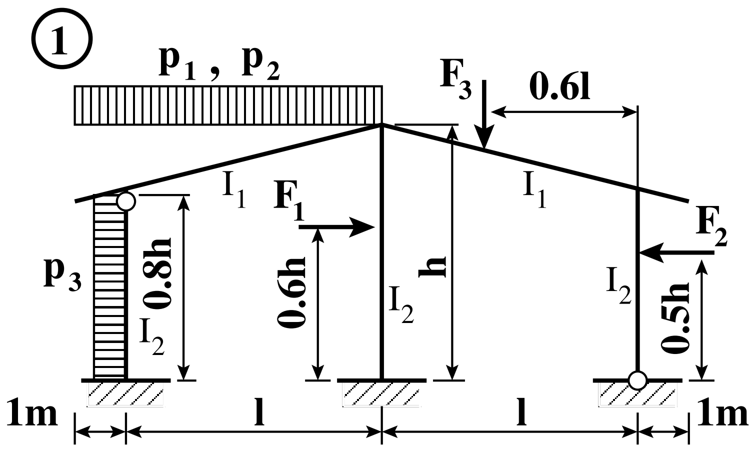
TALLINN UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF MECHANICS

Andres Lahe

The EST Method

Examples in structural analysis



Tallinn 2014

l\vspace*1mm

Mente et manu

The EST method[1] is a method for solving *boundary value problems* for the structural analysis of frames, beams and trusses. Differential equations are considered here together with a set of boundary conditions:

- compatibility equations of the displacements at nodes,
- joint equilibrium equations,
- side conditions (hinges),
- restrictions on support displacements.

The EST method programs written in GNU Octave language assemble and solve sparse systems of equations with unknown member-end displacements, member-end forces and support reactions. The analysis technique is illustrated with numerous examples accompanied with GNU Octave programs.

Andres Lahe

[1] `./ESTmethod.pdf`.

Table of Contents

List of Figures

List of Tables

1. First-order structural analysis

1.1 Computation of frames with the EST method

Example 1.1. ***A two-span frame*** *(Fig. 1.1). Computation of the displacements, internal forces M, Q, N and support reactions.*

Initial data *are given in Table 1.1 (**B** denotes load case numbers), and free-body diagrams are shown in Figs. 1.1 and 1.2.*

*The free-body diagram number **N** (circled numbers 1, ..., 0 shown in Figs. 1.1 and 1.2) conforms with the numbers of GNU Octave programs for the EST method. The programs can be downloaded from*

*spESTframe**N**LaheDefWFI.m.zip*[1]

1. spESTframe1LaheDefWFI.m
2. spESTframe2LaheDefWFI.m
3. spESTframe3LaheDefWFI.m
4. spESTframe4LaheDefWFI.m
5. spESTframe5LaheDefWFI.m
6. spESTframe6LaheDefWFI.m
7. spESTframe7LaheDefWFI.m
8. spESTframe8LaheDefWFI.m
9. spESTframe9LaheDefWFI.m
0. spESTframe10LaheDefWFI.m

In Fig. 1.3, the elements, and in Fig. 1.4, the sparsity pattern of matrix spA of a two-span frame are shown.

Table 1.1. Dimensions and loads of a two-span frame[2]

B	1	2	3	4	5	6	7	8	9	10
$l\,[m]$	6	8	10	6	8	10	6	8	10	12
$h\,[m]$	3	4	4	4	5	5	3	4	4	5
I_1/I_2	2	2	2	2	2	3	3	3	3	3
$p_1\,[kN/m]$	12	0	0	14	0	0	8	0	0	10
$p_2\,[kN/m]$	0	14	0	0	8	0	0	16	0	0
$p_3\,[kN/m]$	0	0	8	0	0	12	0	0	14	0
$F_1\,[kN]$	50	0	0	40	0	0	30	0	0	20
$F_2\,[kN]$	0	50	0	0	40	0	0	30	0	0
$F_3\,[kN]$	0	0	50	0	0	40	0	0	30	0

[1] ./octavePrograms/spESTframeNLaheDefWFI.m.zip.
[2] http://digi.lib.ttu.ee/opik_eme/ylesanded.pdf#page=39. Web. 06 Jan. 2014.

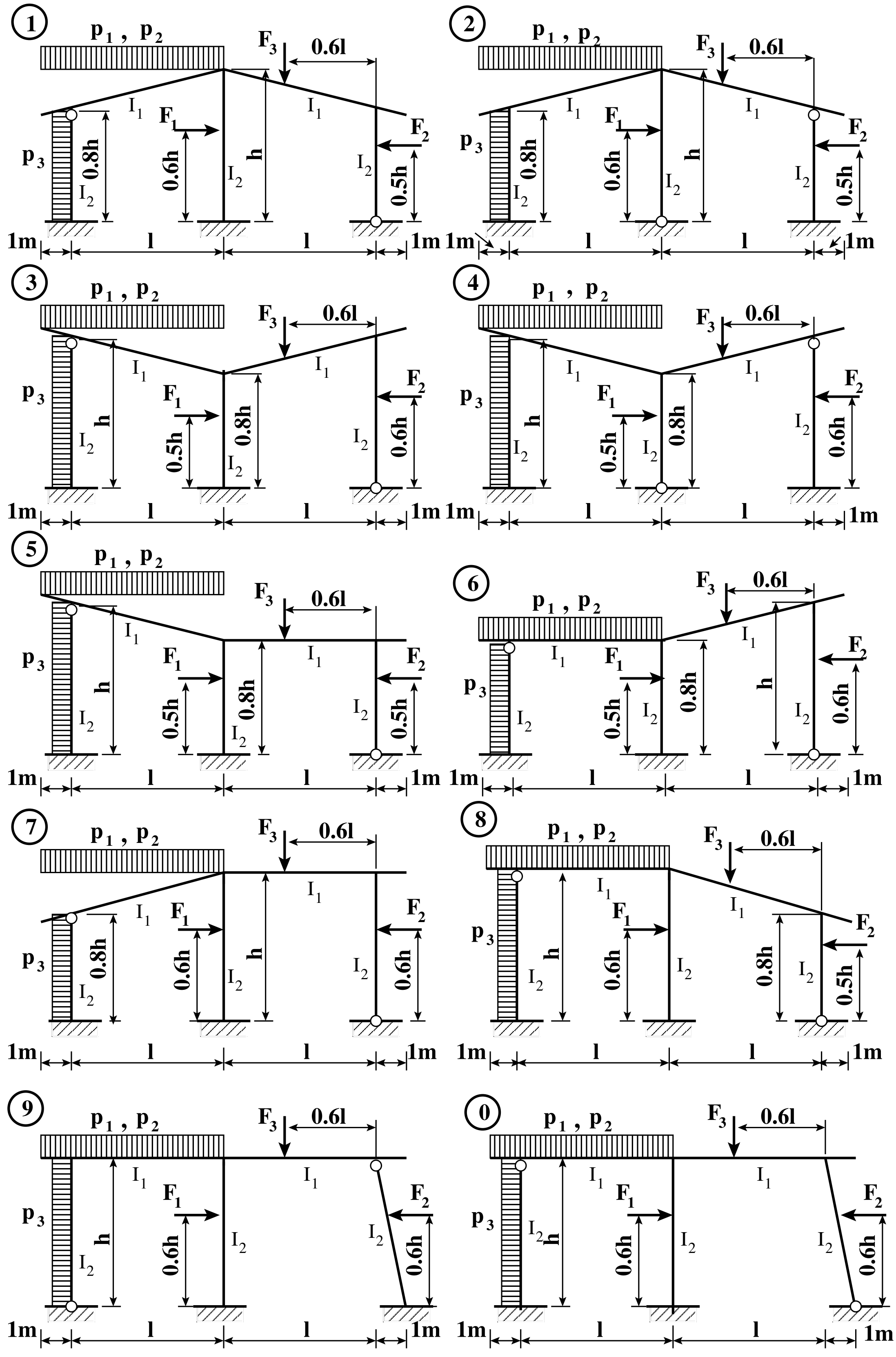


Figure 1.1. Structural systems of a two-span frame

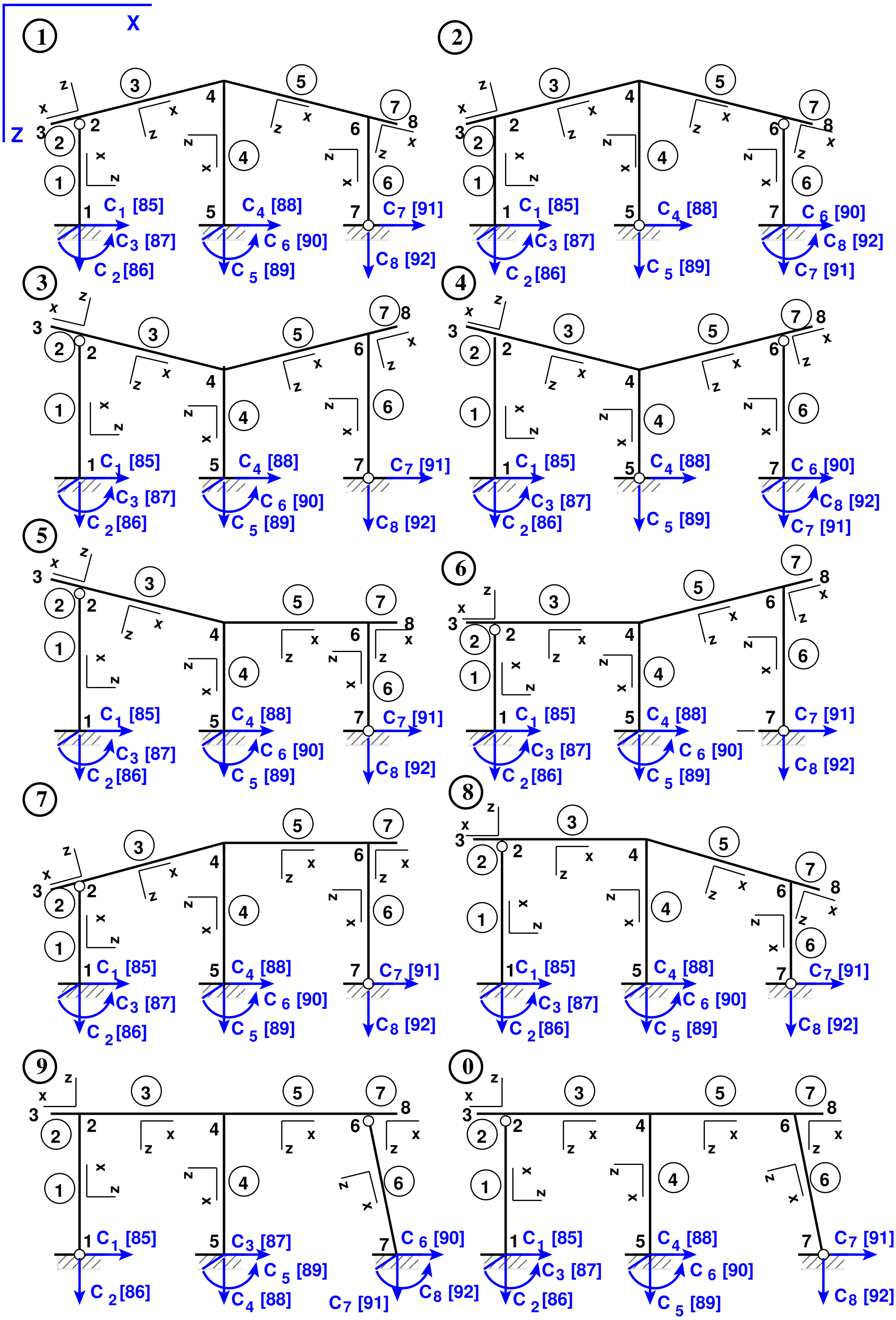


Figure 1.2. Numeration of nodes and members of a two-span frame

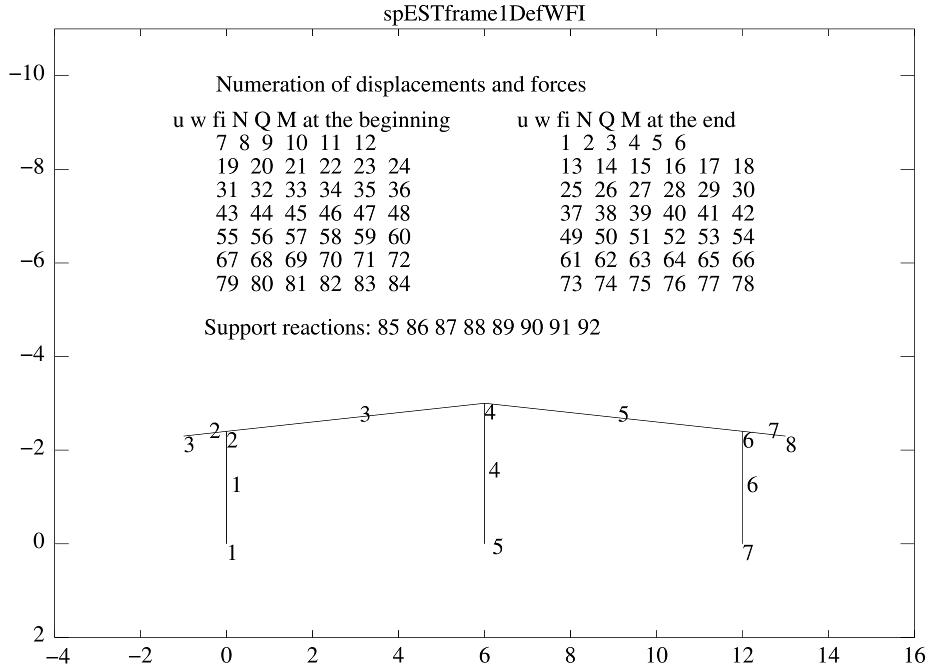


Figure 1.3. Elements of a two-span frame

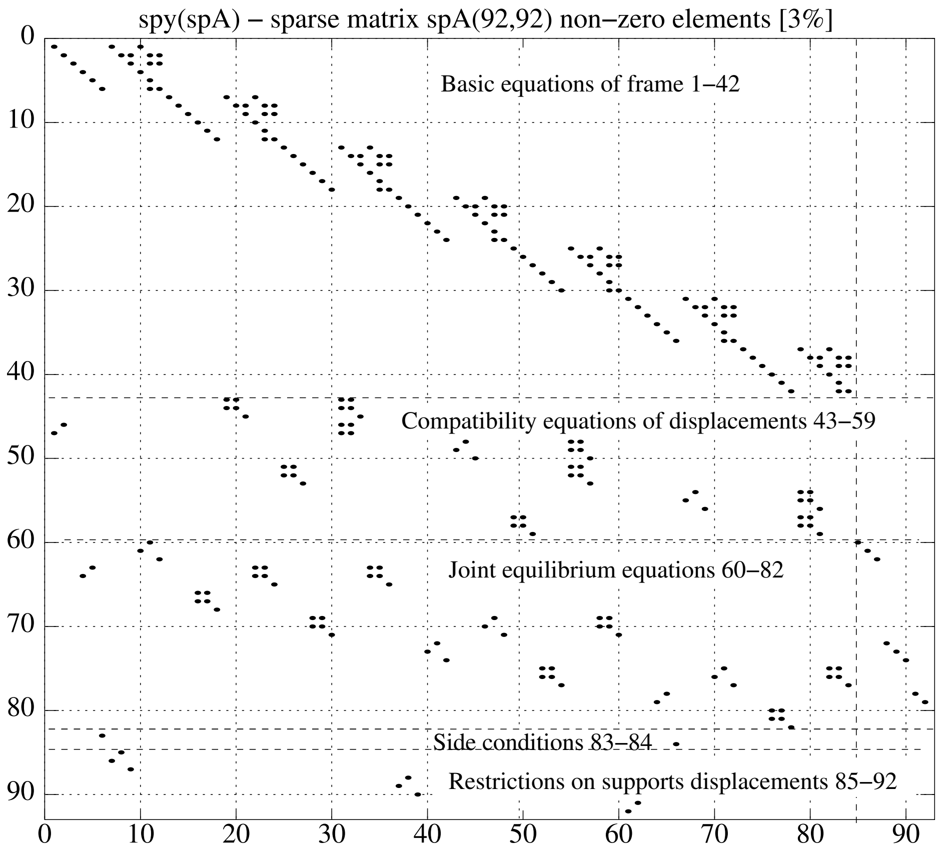


Figure 1.4. Sparsity pattern of matrix spA of a two-span frame

Example 1.2. ***A frame with shear force hinge*** *(Fig. 1.5). Computation of the displacements, internal forces M, Q, N and support reactions.*

Initial data *are given in Table 1.2 (**B** denotes load case numbers), and free-body diagrams are shown in Figs. 1.5 and 1.6.*

*The free-body diagram number **NM** (circled numbers 16, 27, 38, 49, and 50 shown in Figs. 1.5 and 1.6) conforms with the numbers of GNU Octave programs for the EST method. The programs can be downloaded from*

*spESTframe**NM**WFI.m.zip* [3]

16. spESTframe16WFI.m
27. spESTframe27WFI.m
38. spESTframe38WFI.m
49. spESTframe49WFI.m
50. spESTframe50WFI.m

Table 1.2. Dimensions and loads of a frame with shear force hinge[4]

B	1	2	3	4	5	6	7	8	9	0
l [m]	5	6	8	9	10	5	6	8	9	10
h [m]	4	5	6	7	8	4	5	6	7	8
I_1/I_2	2	3	2	3	2	3	2	3	2	3
p_1 [kN/m]	8	0	10	0	12	0	8	0	10	0
p_2 [kN/m]	0	10	0	12	0	14	0	16	0	10
F_1 [kN]	10	0	15	0	20	0	12	0	14	0
F_2 [kN]	0	14	0	16	0	18	0	20	0	16

[3] ./octavePrograms/spESTframeNMWFI.m.zip.
[4] Compiled by Andrus Räämet, PhD: http://staff.ttu.ee/~raamet/Failid/Joumkodutoo.pdf. Web. 06 Jan. 2014.

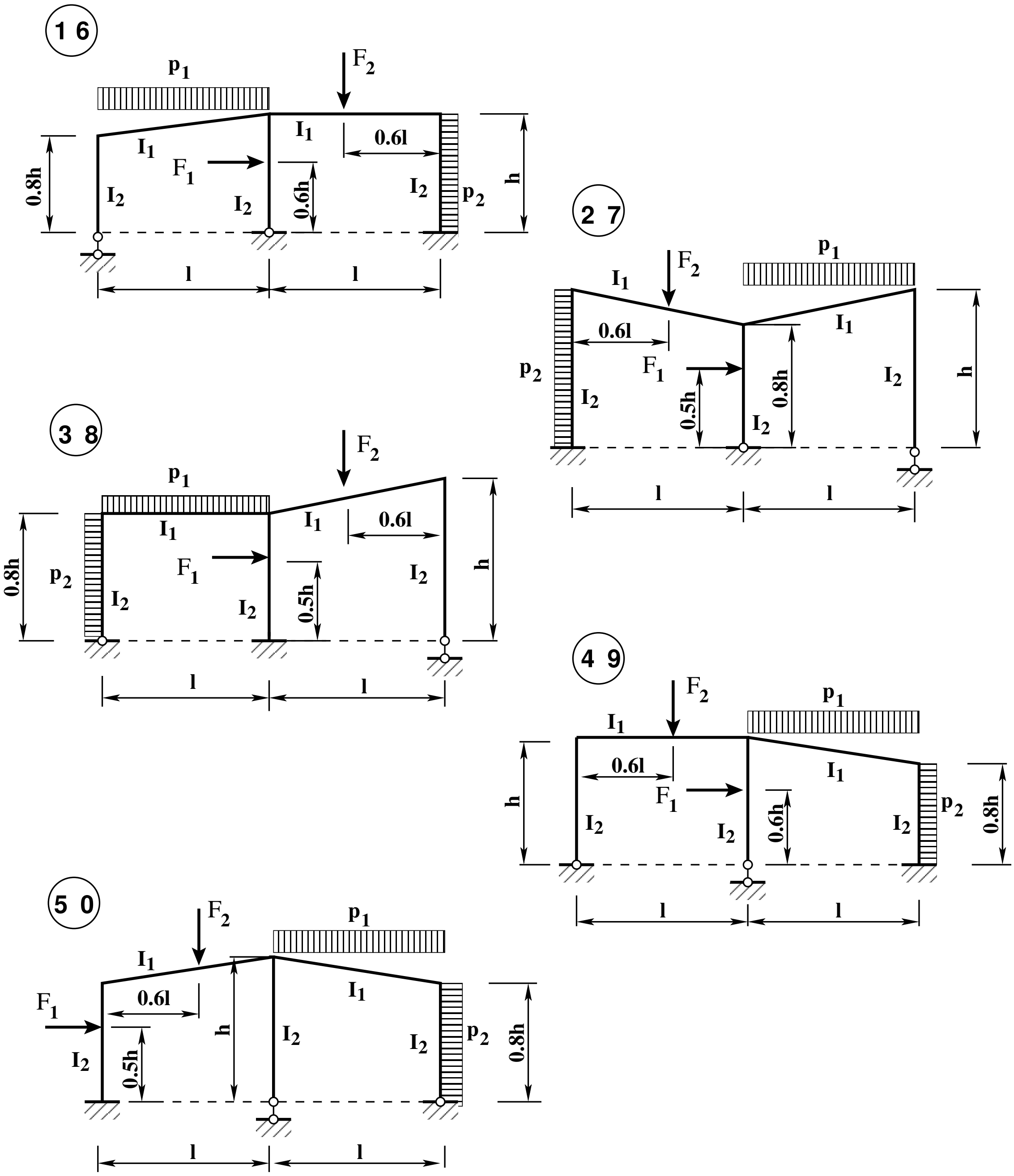


Figure 1.5. Frames with shear force hinge[4]

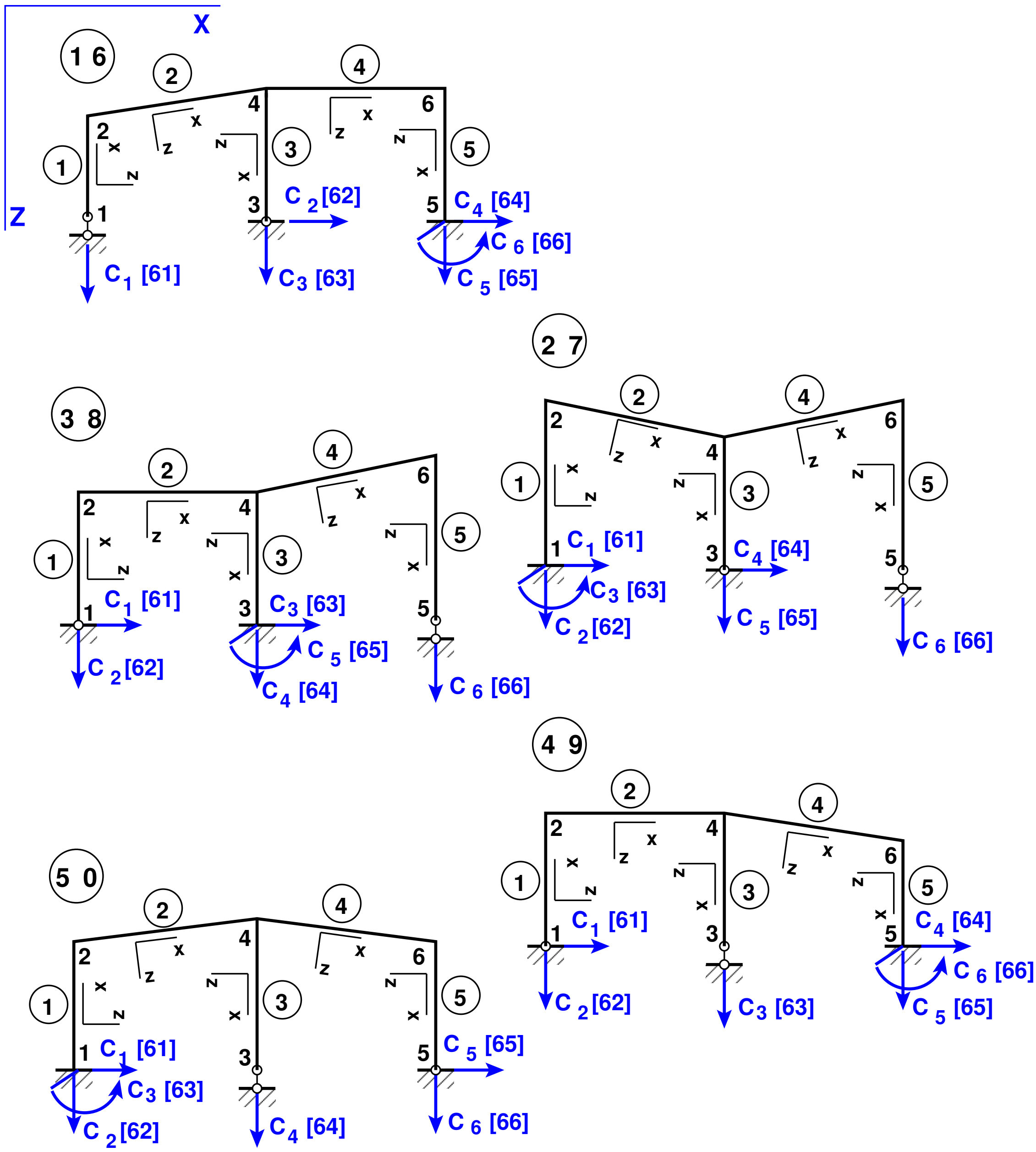


Figure 1.6. Numeration of nodes and members of frames with shear force hinge

In Fig. 1.7, the elements, and in Fig. 1.8, the sparsity pattern of matrix spA of a frame with shear force hinge are represented.

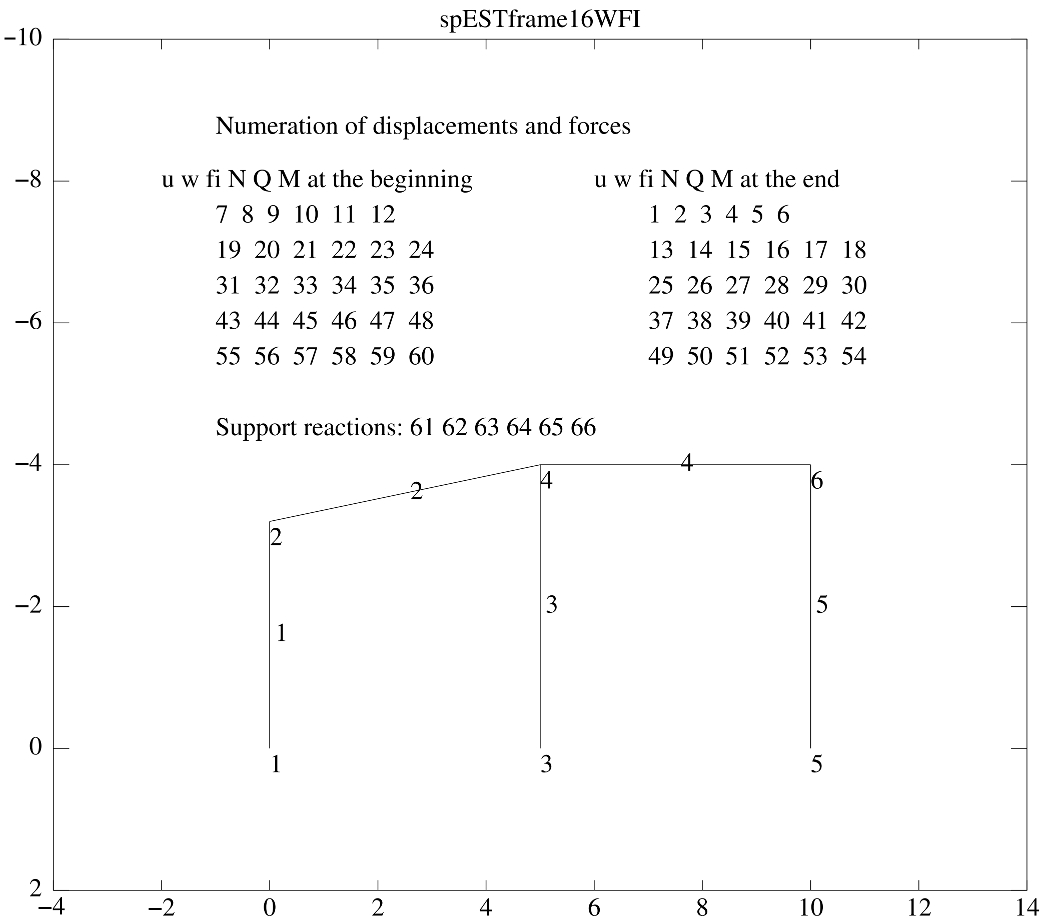


Figure 1.7. Elements of a frame with shear force hinge

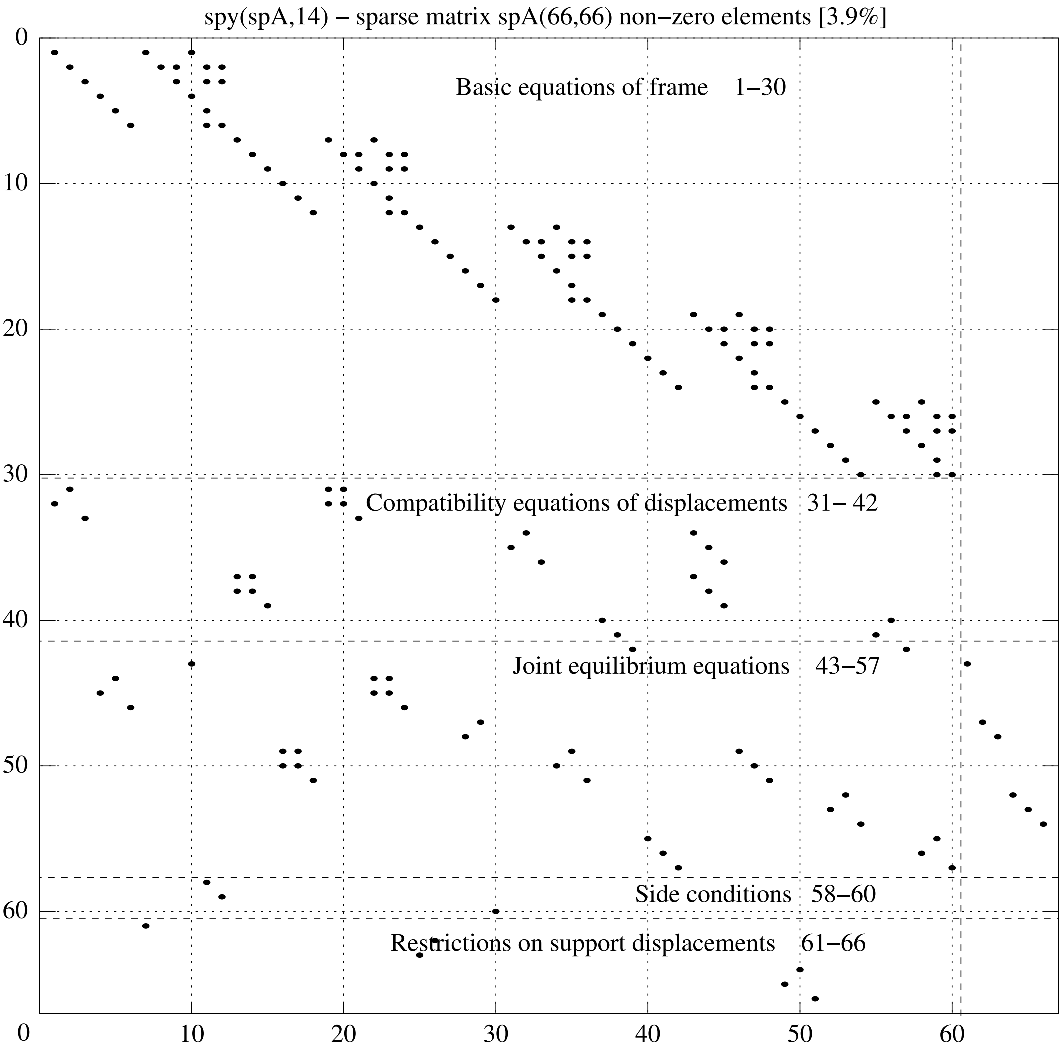


Figure 1.8. Sparsity pattern of matrix spA of a frame with shear force hinge

Example 1.3. ***A three-hinged frame*** *(Fig. 1.9). Computation of the displacements, internal forces M, Q, N and support reactions.*

Initial data *are given in Table 1.3 (**B** denotes load case numbers), and free-body diagrams are shown in Figs 1.9 and 1.10.*

*The free-body diagram number **K** (circled numbers 1, ..., 0 shown in Figs. 1.9 and 1.10) conforms with the numbers of GNU Octave programs for the EST method. The programs can be downloaded from*

*spESTframe3hinge**K**NQM.m.zip*[5]

1. *spESTframe3hinge1NQM.m*
2. *spESTframe3hinge2NQM.m*
3. *spESTframe3hinge3NQM.m*
4. *spESTframe3hinge4NQM.m*
5. *spESTframe3hinge5NQM.m*
6. *spESTframe3hinge6NQM.m*
7. *spESTframe3hinge7NQM.m*
8. *spESTframe3hinge8NQM.m*
9. *spESTframe3hinge9NQM.m*
0. *spESTframe3hinge10NQM.m*

In Fig. 1.11, the elements, and in 1.12, the sparsity pattern of matrix spA of a three-hinged frame are shown.

Table 1.3. Dimensions and loads of a three-hinged frame[6]

B	1	2	3	4	5	6	7	8	9	0
l [m]	6	8	10	6	8	10	6	8	10	6
h [m]	4	5	6	4	5	6	4	5	6	4
ξ	0.4	0.4	0.4	0.5	0.5	0.5	0.6	0.6	0.6	0.75
F [kN]	35	30	25	30	25	20	25	20	15	20
p [kN/m]	24	22	20	22	20	18	20	18	16	26

[5] `./octavePrograms/spESTframe3hingeKNQM.m.zip`.

[6] Compiled by Andrus Räämet, PhD: `http://staff.ttu.ee/~raamet/Failid/Raamikodutoo.pdf`. Web. 06 Jan. 2014.

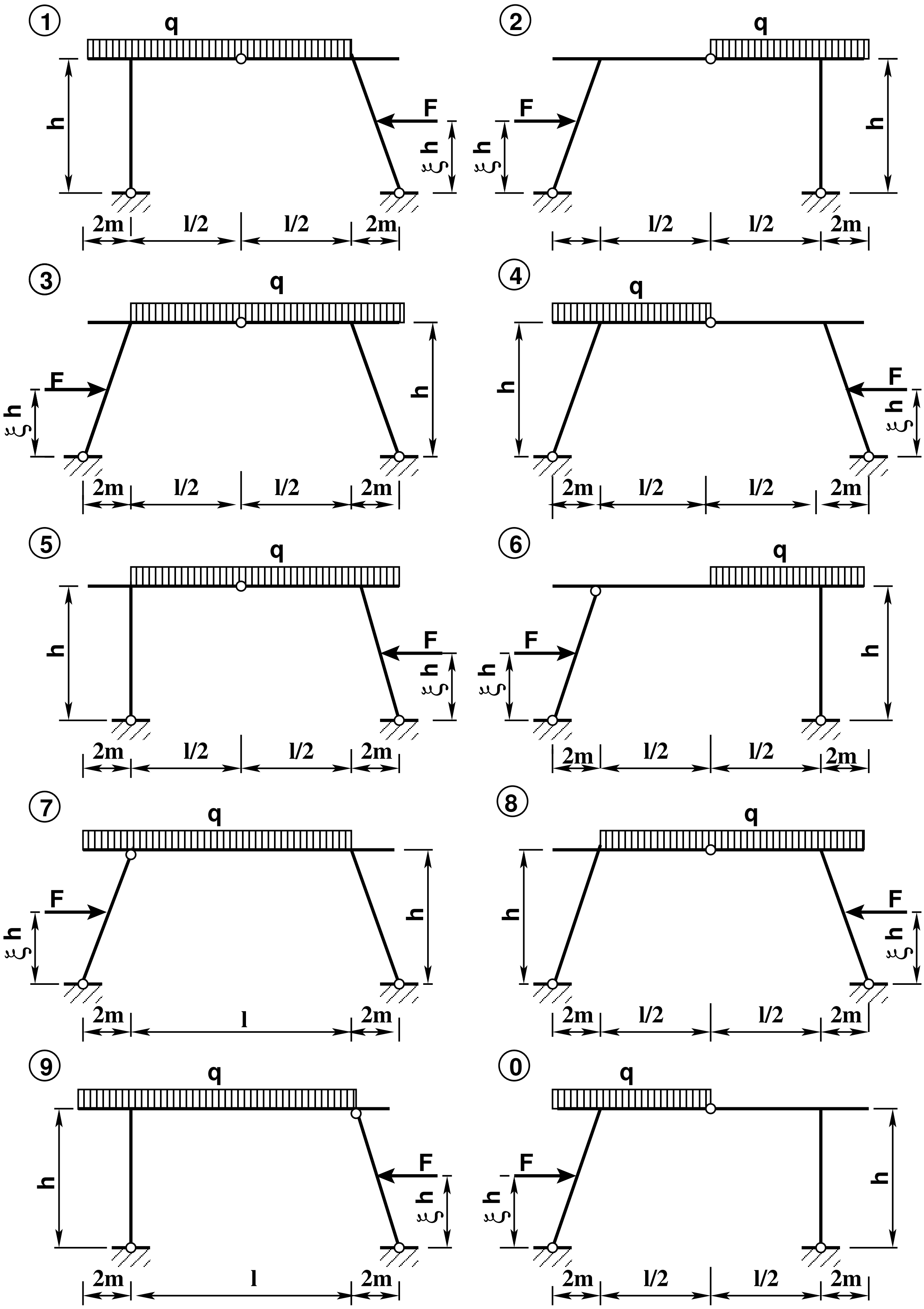


Figure 1.9. Three-hinged frames[6]

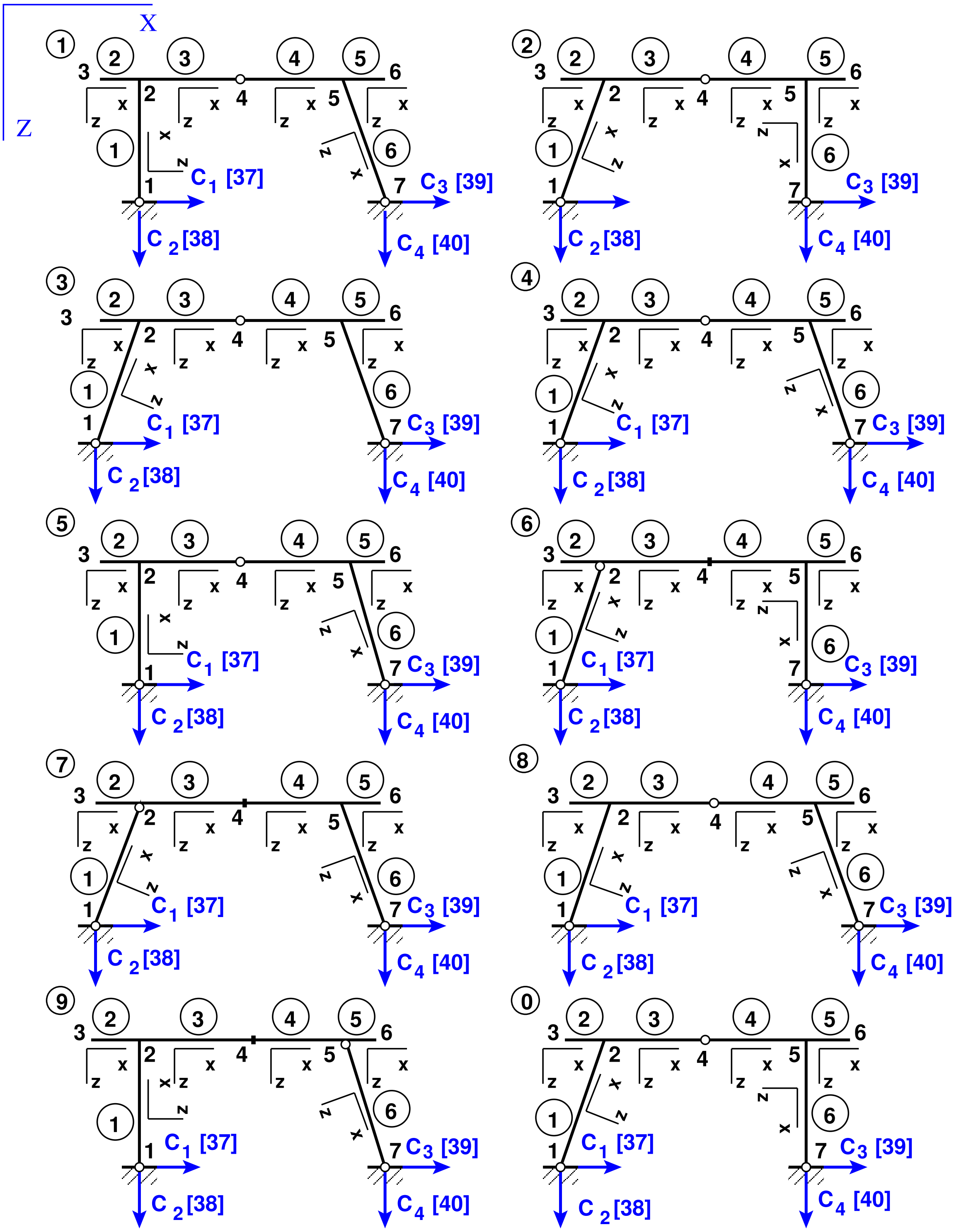


Figure 1.10. Numeration of nodes and members of a three-hinged frame

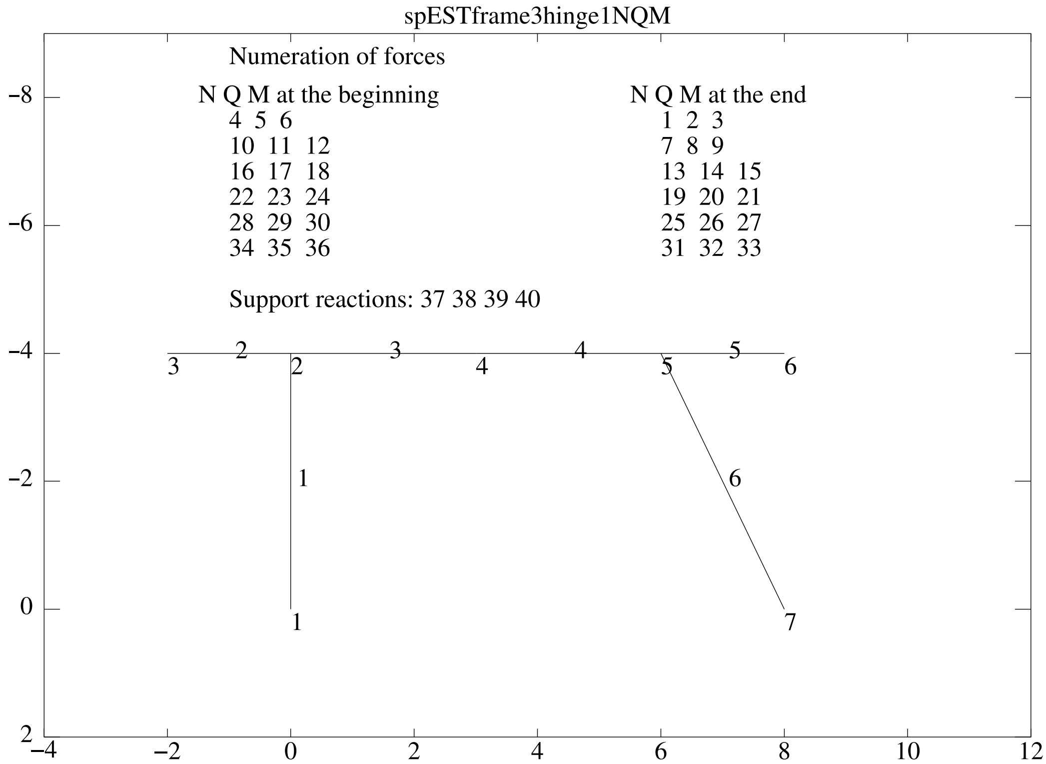


Figure 1.11. Elements of a three-hinged frame

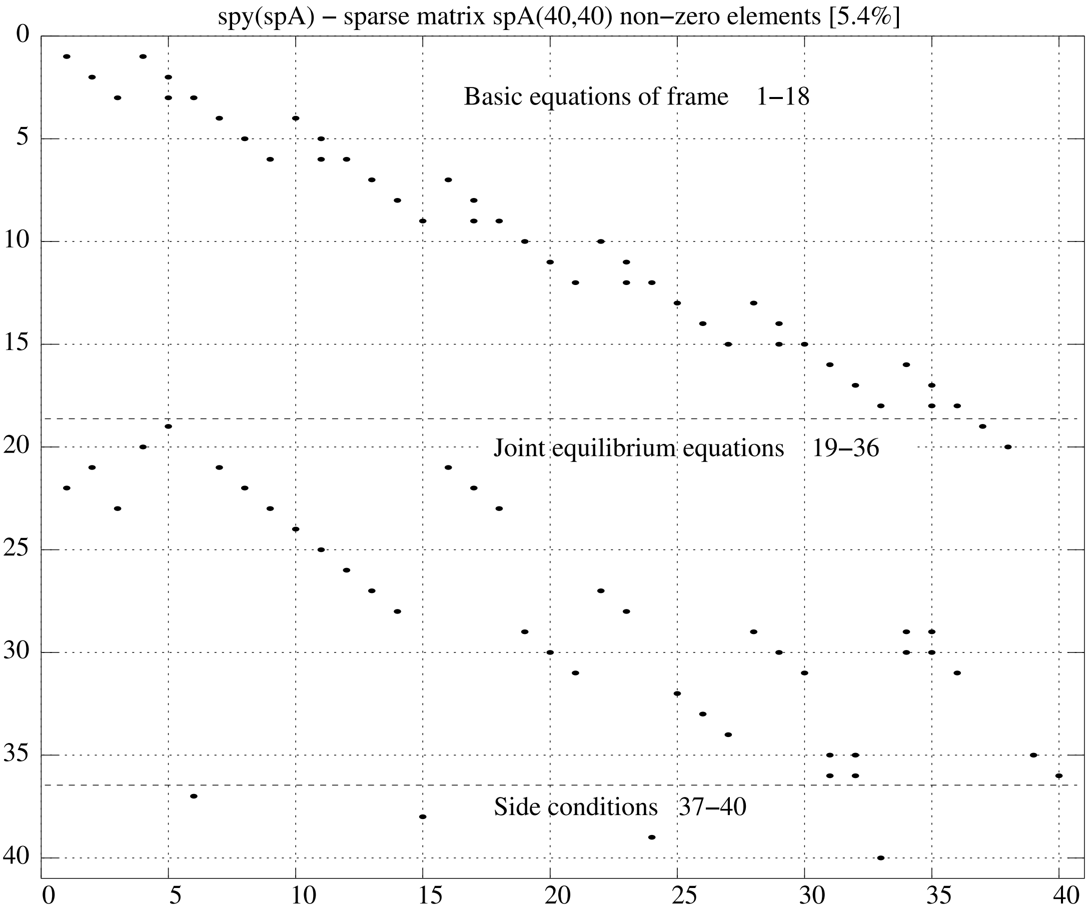


Figure 1.12. Sparsity pattern of matrix spA of a three-hinged frame

1.2 Computation of beams with the EST method

Example 1.4. ***A continuous beam*** *(Fig. 1.13). Computation of the displacements, internal forces M, Q and support reactions.*

Initial data. *The dead load $\boldsymbol{g}$ is given in Table 1.5, where $\boldsymbol{B}$ denotes load case numbers. The forces $F_a = 80\,kN$ and simultaneously acting $F_b = 60\,kN$ and $F_c = 40\,kN$ are live loads. Location of the points (see Fig. 1.13) a, b, c at which concentrated loads F_a, F_b, and F_c act is indicated by numbers in Table 1.5. Versions ($\boldsymbol{A}$) of beam dimensions are given in Table 1.4. The flexural rigidity EI is assumed to be constant along the beam.*

In Figs. 1.13 and 1.14, free-body diagrams are shown. The GNU Octave program for continuous beam II spESTbeam32LaheWFI.m can be downloaded from spESTbeam32LaheWFI.m.zip[7]*.*

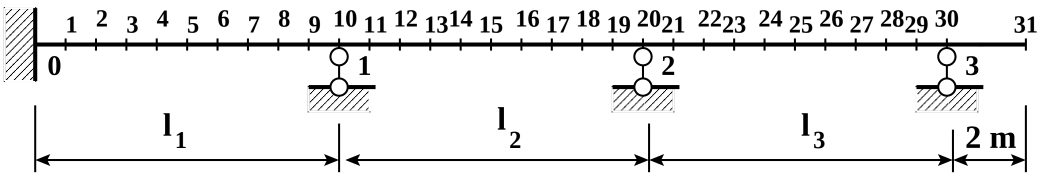


Figure 1.13. Continuous beam II[8]

Table 1.4. Dimensions of continuous beam II[8]

A	1	2	3	4	5	6	7	8	9	0
$l_1\ [m]$	8	10	8	10	8	10	6	8	8	6
$l_2\ [m]$	10	10	8	8	10	8	8	6	8	6
$l_3\ [m]$	8	8	10	8	10	10	8	8	6	8
i	0	0	0	1	1	1	1	2	2	2

Table 1.5. Loads of continuous beam II[8]

B	1	2	3	4	5	6	7	8	9	0
$g\ [kN/m]$	12	14	16	18	12	14	16	12	14	16
a	3	4	2	26	27	18	17	26	24	23
b	12	12	13	14	14	2	2	3	4	4
c	17	16	18	18	17	7	6	8	8	7
k	24	22	26	8	4	6	28	16	14	12

In Fig. 1.14, nodes and elements of the continuous beam II are shown.

[7] `./octavePrograms/spESTbeam32LaheWFI.m.zip`.
[8] `http://digi.lib.ttu.ee/opik_eme/ylesanded.pdf#page=39`.

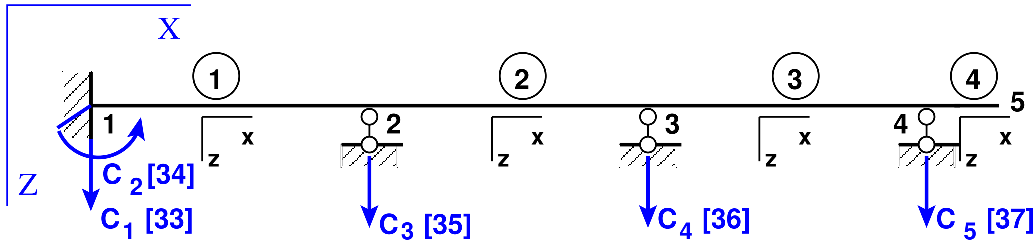


Figure 1.14. Elements of continuous beam II

In Fig. 1.15, the sparsity pattern of matrix spA, and in Fig. 1.16, the node and member numbers of the continuous beam II are shown.

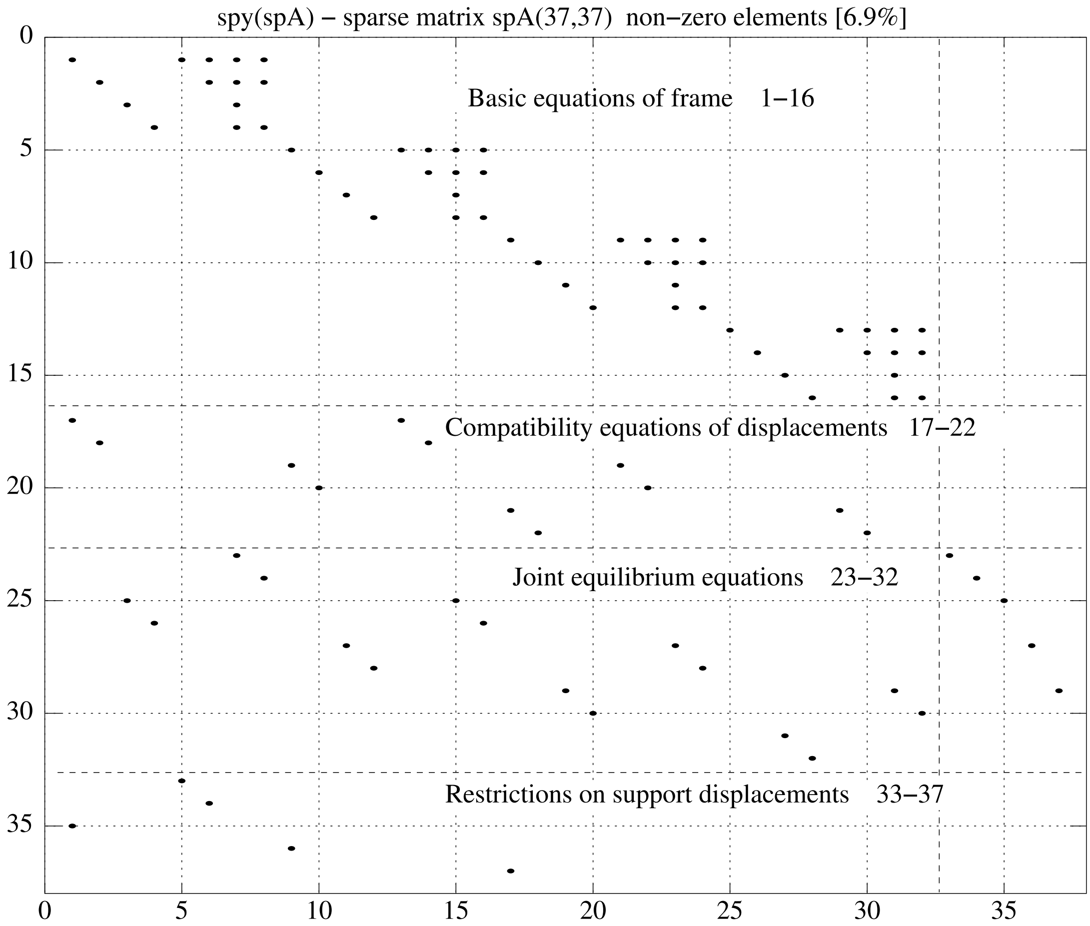


Figure 1.15. Sparsity pattern of matrix spA of continuous beam II

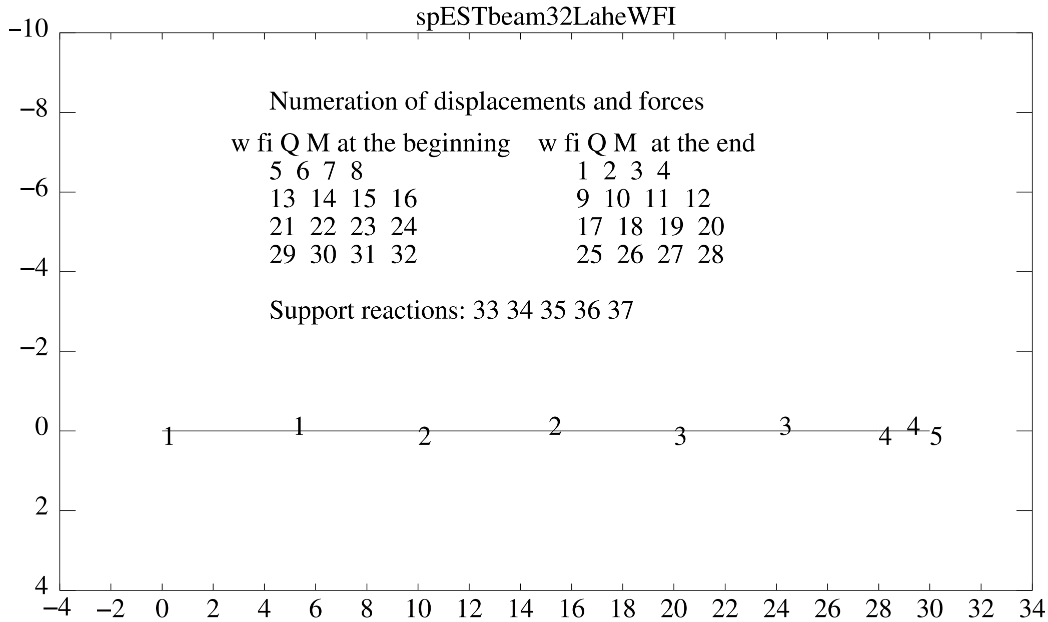


Figure 1.16. Numeration of nodes and members of continuous beam II

Example 1.5. ***A multispan hinged beam*** *(Fig. 1.17). Computation of the internal forces M, Q and support reactions.*

Initial data. *The free-body diagram numbers **A** (circled numbers 1, ..., 0 shown in Figs. 1.17 and 1.19) conform with the numbers of GNU Octave programs for the EST method. Beam dimensions for loading variant **B** are given in Table 1.6. The dead load: uniform load $g = 16\,kN/m$, forces $F_k = 60\,kN$ (section **k** in Table 1.7), $F_i = 60\,kN$ (section **i** in Table 1.7), and $F_{jr} = 40\,kN$ (in moment hinge near the support **r**, Table 1.6).*

Table 1.6. Beam dimensions[9]

B	1	2	3	4	5	6	7	8	9	0
l_1 [m]	10	16	18	10	15	18	12	16	15	10
l_2 [m]	15	15	12	12	12	16	16	12	16	10
l_3 [m]	12	12	16	15	16	12	12	18	12	12
l_4 [m]	12	16	12	12	12	12	10	16	12	16
r	b	c	d	b	c	d	b	c	d	c

[9] http://digi.lib.ttu.ee/opik_eme/ylesanded.pdf#page=19.

The uniform load g and forces F_k, F_i are element loads. The force F_{jr} is a nodal load.

The free-body diagram numbers ***A*** *conform with the numbers of GNU Octave programs for a multispan hinged beam. The programs can be downloaded from spESTGerberBeamNQM.m.zip*[10]

1. *spESTGerberBeam1QM.m*
2. *spESTGerberBeam2QM.m*
3. *spESTGerberBeam3QM.m*
4. *spESTGerberBeam4QM.m*
5. *spESTGerberBeam5QM.m*
6. *spESTGerberBeam6QM.m*
7. *spESTGerberBeam7QM.m*
8. *spESTGerberBeam8QM.m*
9. *spESTGerberBeam9QM.m*

0\. *spESTGerberBeam10QM.m*

Table 1.7. Beam loads. Sections k and i[9]

$B_\Downarrow \setminus A_\Rightarrow$		1	2	3	4	5	6	7	8	9	0
k	1	6	12	16	11	7	11	7	16	11	18
	2	16	14	11	6	12	3	1	17	2	16
	3	7	6	6	8	2	12	2	11	12	13
	4	8	12	17	12	9	13	9	18	13	18
	5	17	14	13	6	14	2	3	19	3	16
	6	9	8	8	8	3	14	4	11	14	11
	7	6	12	18	13	7	12	7	16	12	18
	8	18	14	11	6	12	3	1	17	2	16
	9	8	7	6	12	4	13	2	11	13	13
	0	19	12	13	8	14	2	3	18	3	18
i	1	12	7	11	17	12	17	16	11	17	11
	2	1	6	16	1	2	7	17	13	18	12
	3	3	12	12	11	13	8	7	17	7	17
	4	13	9	13	18	14	18	18	12	17	13
	5	2	8	17	2	3	7	16	11	18	12
	6	1	13	11	13	12	8	8	18	8	17
	7	12	7	12	19	13	19	17	13	17	11
	8	3	8	18	3	2	7	18	12	18	12
	9	1	14	17	7	14	8	7	19	7	18
	0	2	7	19	2	3	7	17	13	17	13

10 ./octavePrograms/spESTGerberBeamNQM.m.zip.

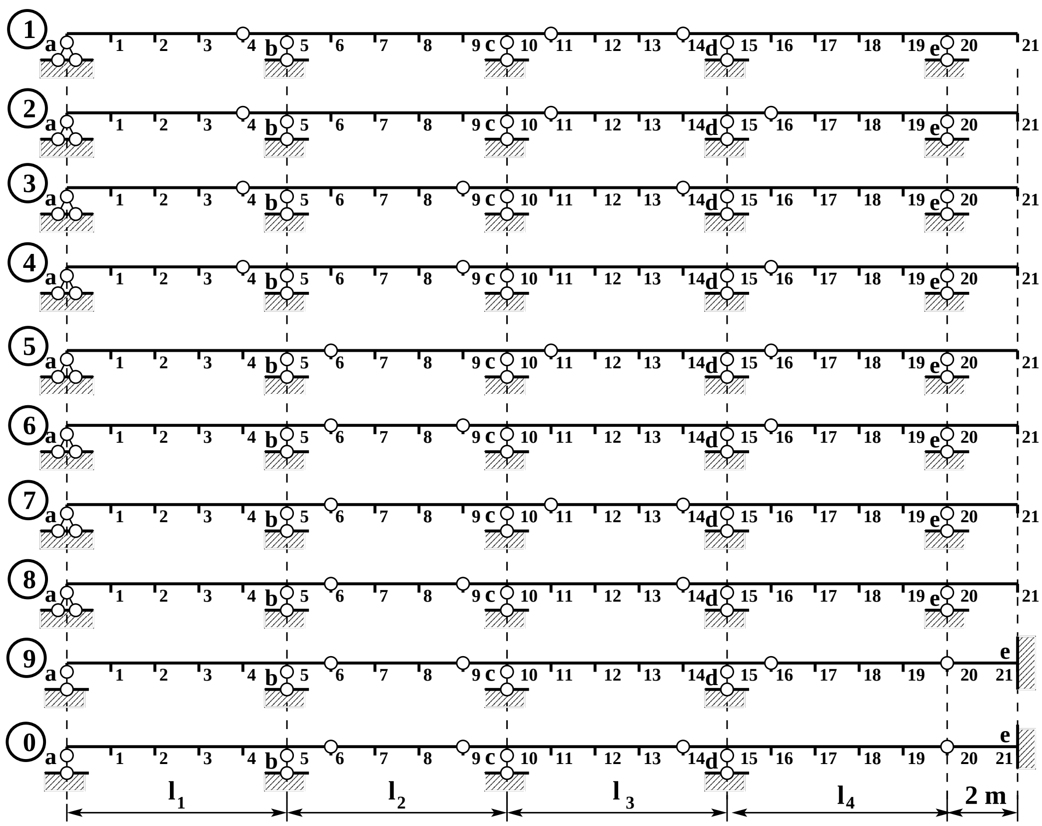


Figure 1.17. Multispan hinged beams

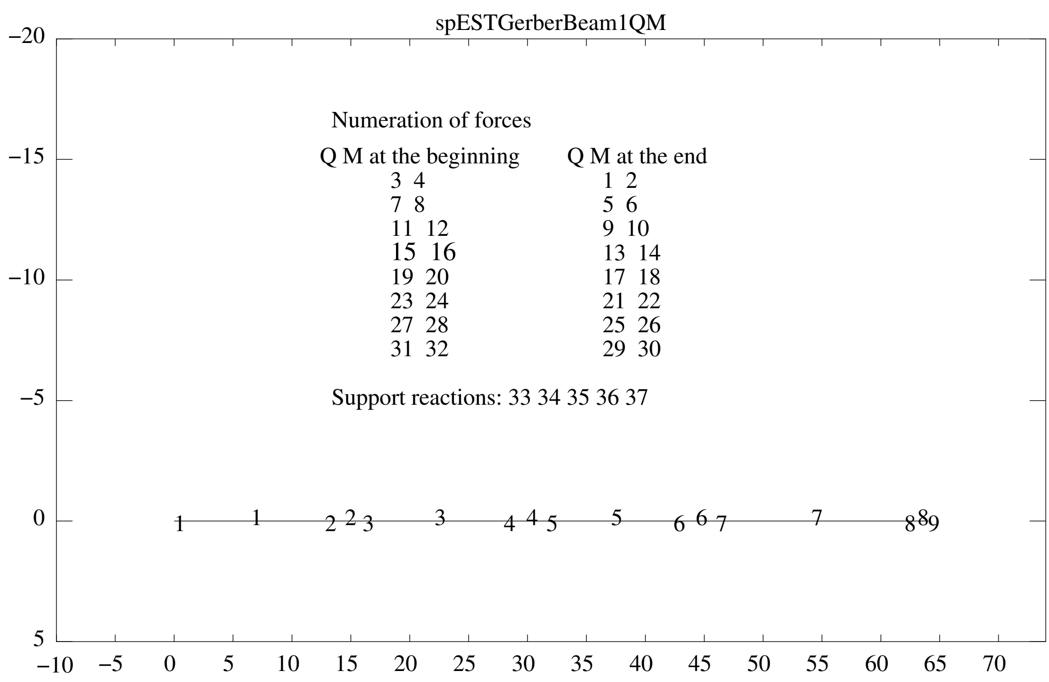


Figure 1.18. Elements of a Gerber beam

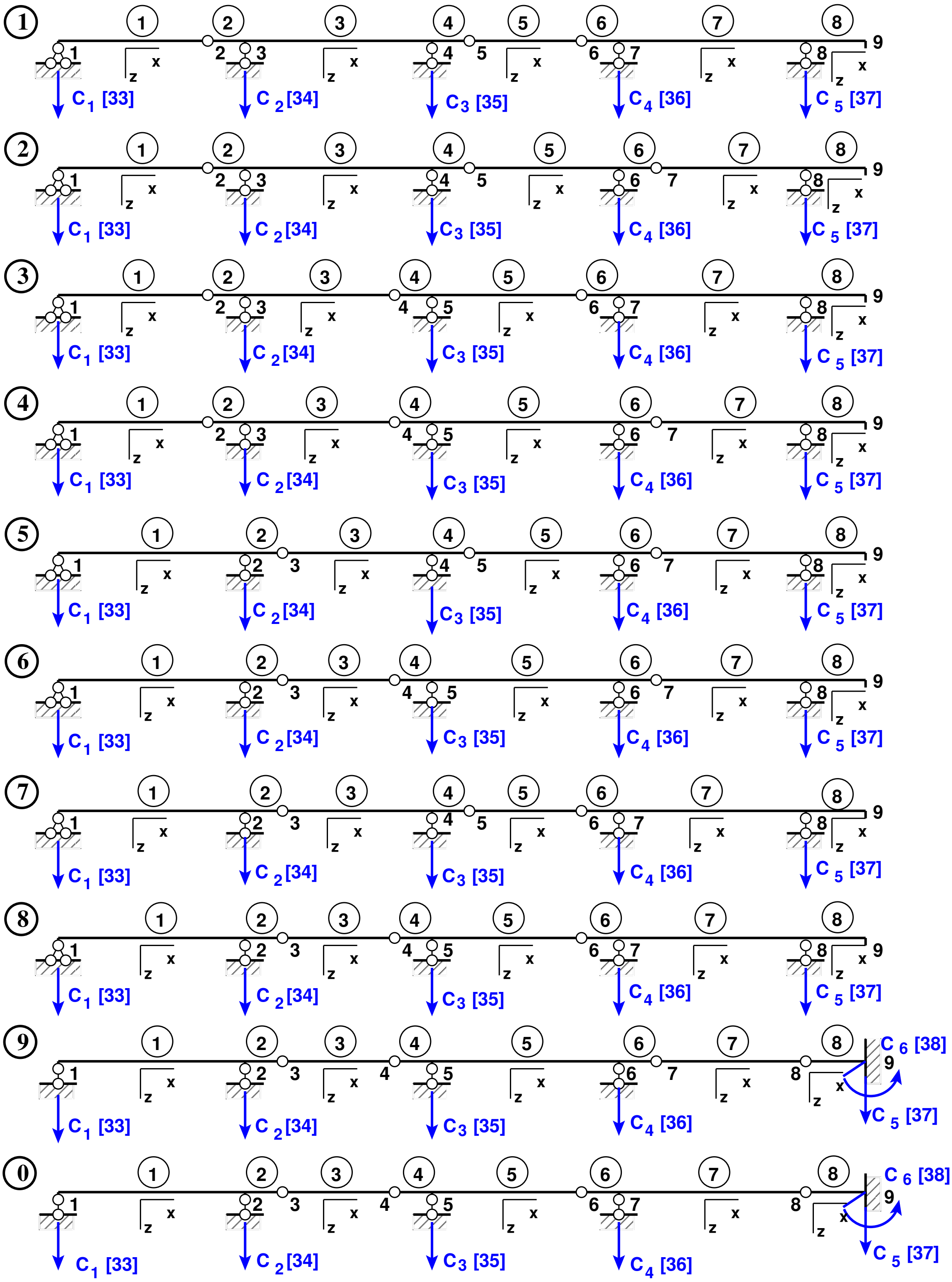


Figure 1.19. Numeration of nodes and members of Gerber beams

In Fig. 1.20, the sparsity pattern of matrix spA of the Gerber beam is shown.

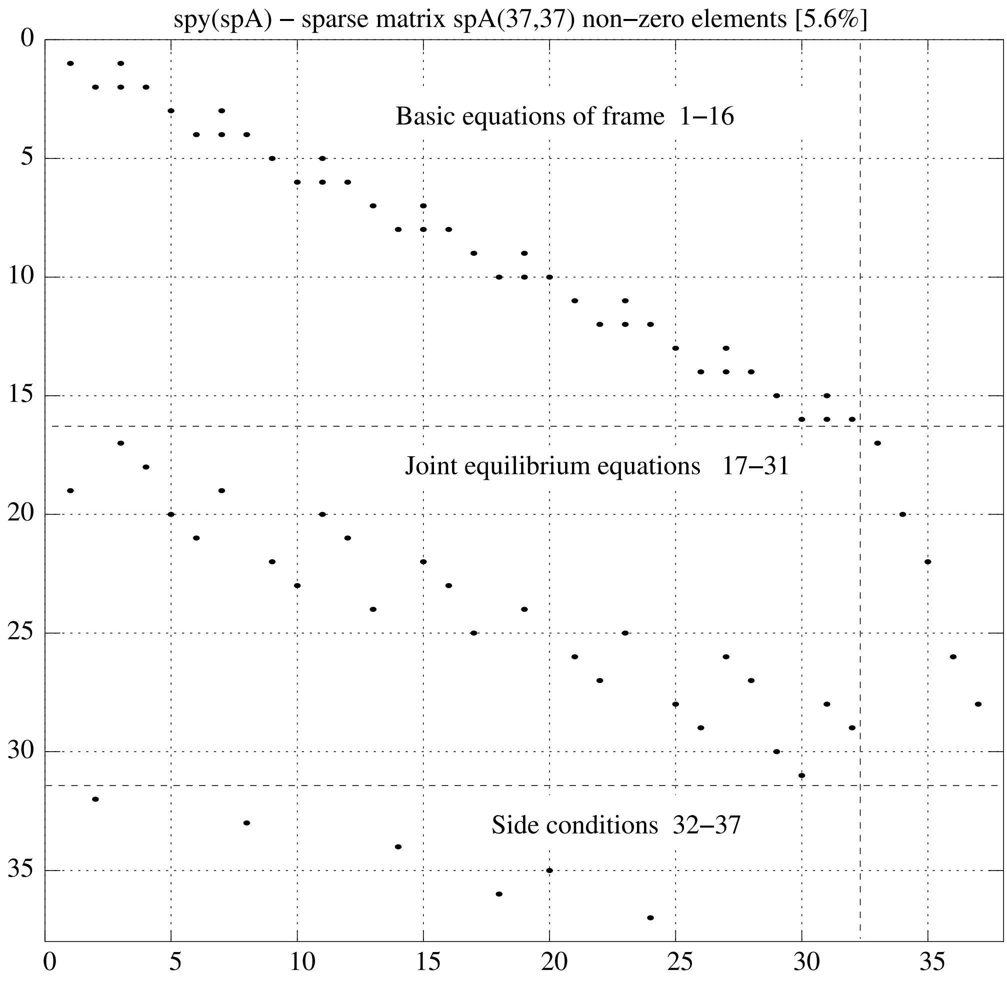


Figure 1.20. Sparsity pattern of matrix spA of the Gerber beam

1.3 Computation of trusses with the EST method

Example 1.6. ***Statically indeterminate planar trusses*** *(Fig. 1.21). Computation of the displacements and internal forces N.*

Initial data. *The trusses depicted in Fig. 1.21 are subjected to loads F_1, F_2, and F_3. Load values and dimensions of the truss are given in Table 1.8, where* ***B*** *denotes load case numbers. The span is of length $l = 4d$ (4 equal panels, each of length d) and of height $h = d$ [REL83].*

The free-body diagram numbers ***A*** *are shown in Fig. 1.22. The programs can be downloaded from*
spESTtrussN15WFI.m.zip[11]
spESTtruss1N15WFI.m
spESTtruss2N15WFI.m
spESTtruss3N15WFI.m
spESTtruss4N15WFI.m
spESTtruss5N15WFI.m
spESTtruss6N15WFI.m
spESTtruss7N15WFI.m
spESTtruss8N15WFI.m
spESTtruss9N15WFI.m
spESTtruss10N15WFI.m

Table 1.8. Loads and dimensions of trusses

B	1	2	3	4	5	6	7	8	9	10
$d\,[m]$	3.0	3.2	3.4	3.6	4.0	3.0	3.2	3.4	3.6	4.0
$F_1\,[kN]$	60	60	60	60	60	80	80	80	80	80
$F_2\,[kN]$	50	0	50	0	50	0	50	0	50	0
$F_3\,[kN]$	0	40	0	40	0	40	0	40	0	40
A_1/A_2	1.2	1.3	1.4	1.5	1.2	1.3	1.4	1.2	1.3	1.4
A_1/A_3	1.5	1.8	2.0	2.2	2.4	2.2	1.8	2.0	1.5	2.2

[11] `./octavePrograms/spESTtrussN15WFI.m.zip`.

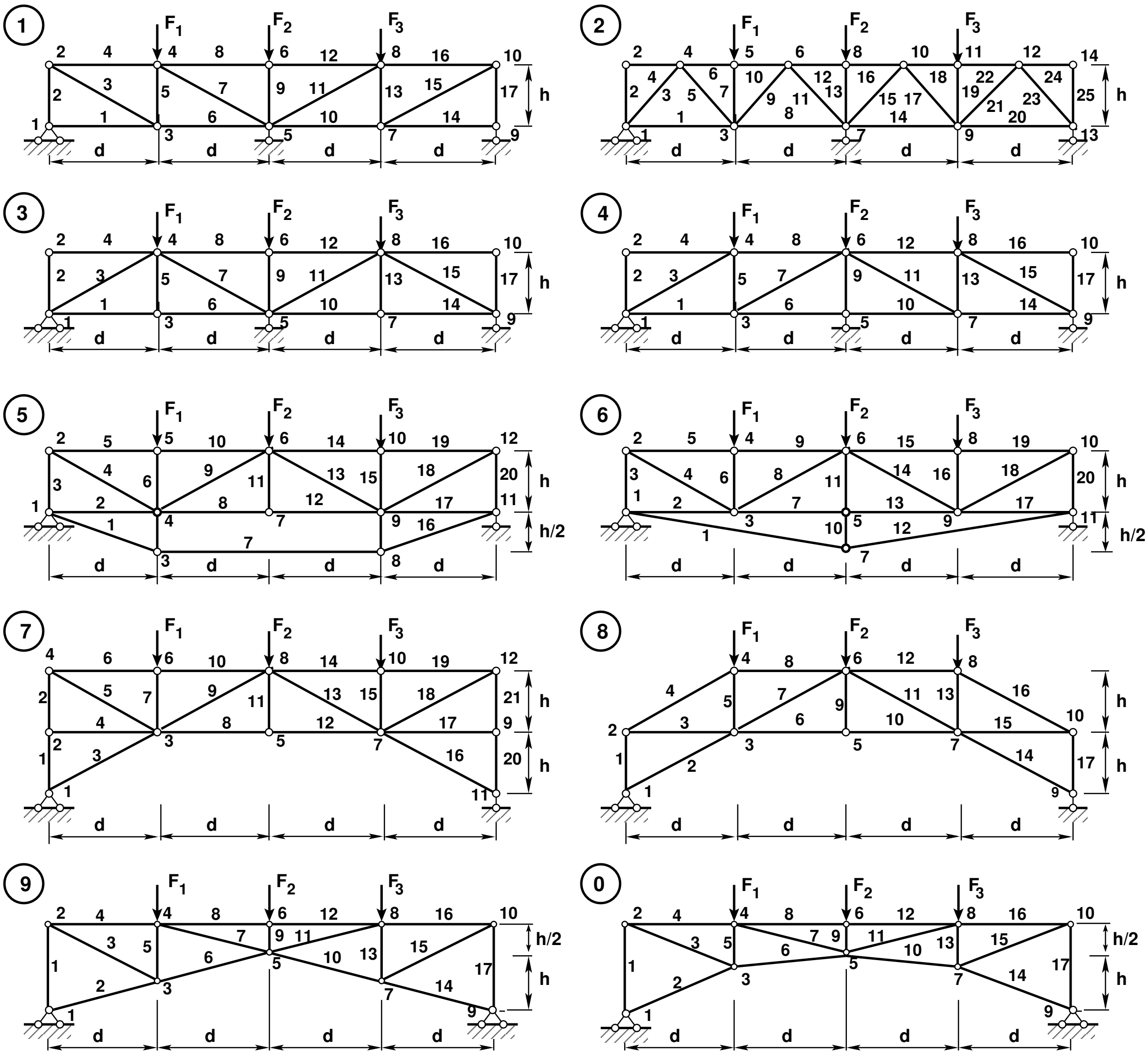


Figure 1.21. The trusses EST

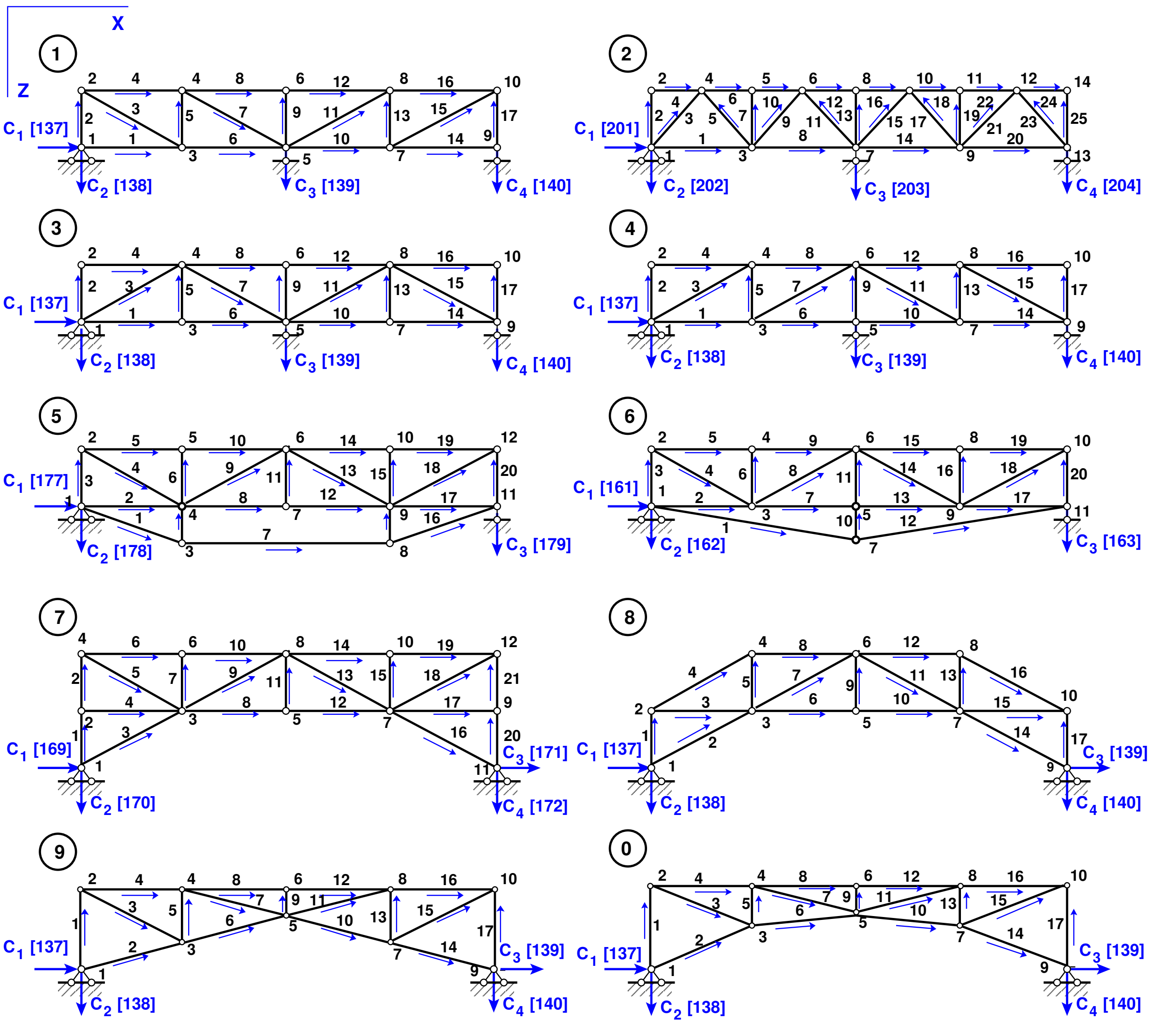


Figure 1.22. Free-body diagrams of the trusses with joint numbers

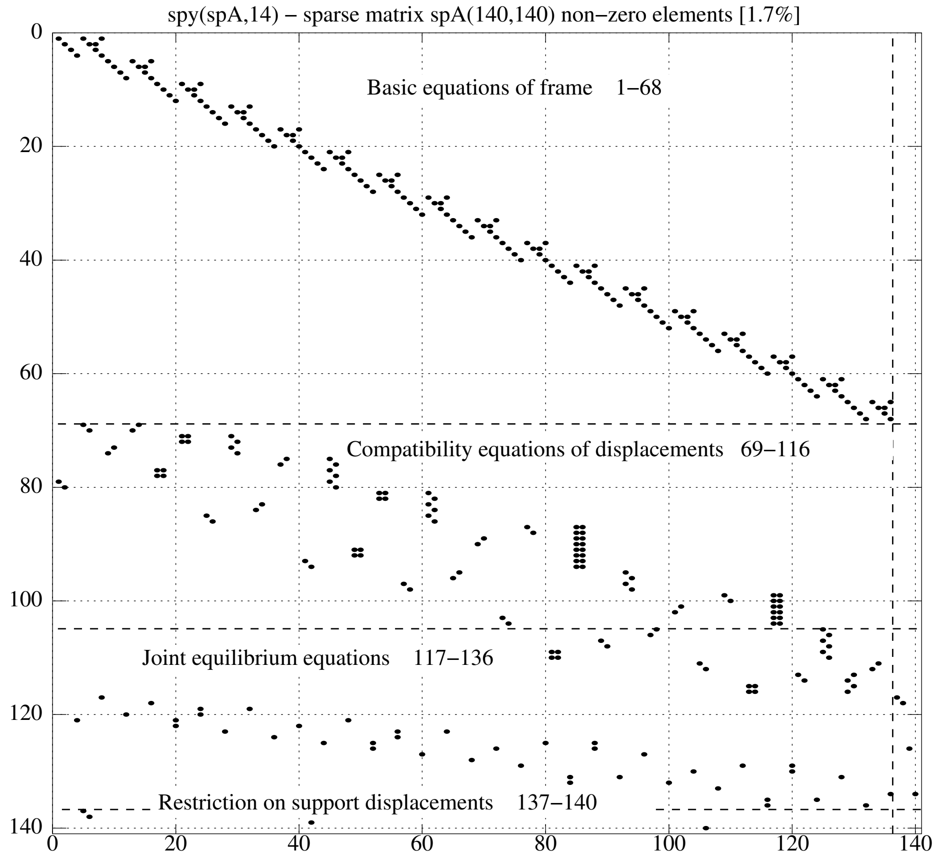


Figure 1.23. Sparsity pattern of matrix spA of truss 1N15WFI

Example 1.7. ***Planar trusses*** *(Fig. 1.24). Computation of the internal forces N, influence line ordinates and support reactions. Computation of the maximum and minimum internal forces due to dead and live loads. Draw the influence line for the truss member of panel **k** shown in Table 1.10.*

Initial data. *The simply supported trusses shown in Fig. 1.24 are subjected to uniform distributed dead load g, uniform distributed live load p (Table 1.9) and live load $F_i = 100\,\mathrm{kN}$ at node **i** shown in Table 1.10. The span is of length $l = 8d$ (8 equal panels, each of length d) and of height h, the distance between rafters is marked by the letter **a** (Table 1.10).*

*In Figs. 1.24 and 1.25, free-body diagram numbers **A** are shown. The programs can be downloaded from*
spESTtrussN27.m.zip[12]
spESTtruss1N27.m
spESTtruss2N27.m
spESTtruss3N27.m
spESTtruss4N27.m
spESTtruss5N27.m
spESTtruss6N27.m
spESTtruss7N27.m
spESTtruss8N27.m
spESTtruss9N27.m
spESTtruss10N27.m

Table 1.9. Distributed dead and live loads of trusses II[13]

A	1	2	3	4	5	6	7	8	9	0
$g\,[kPa]$	3.0	3.0	3.0	3.0	3.0	4.0	4.0	4.0	4.0	4.0
$p\,[kPa]$	0.75	0.75	0.75	0.75	0.75	1.0	1.0	1.0	1.0	1.0

Table 1.10. Dimensions of trusses. Nodes i and k[13]

B	1	2	3	4	5	6	7	8	9	0
$d\,[m]$	1.5	2.0	2.5	3.0	3.5	1.5	2.0	2.5	3.0	3.5
$h\,[m]$	2.0	2.6	3.5	4.0	5.0	2.4	3.0	4.0	5.0	5.5
$a\,[m]$	6	6	6	6	6	5	5	5	5	5
i	7	5	11	3	7	5	11	13	7	11
k	3	4	5	6	7	2	3	4	5	6

[12] ./octavePrograms/spESTtrussN27.m.zip.
[13] http://digi.lib.ttu.ee/opik_eme/ylesanded.pdf#page=29.

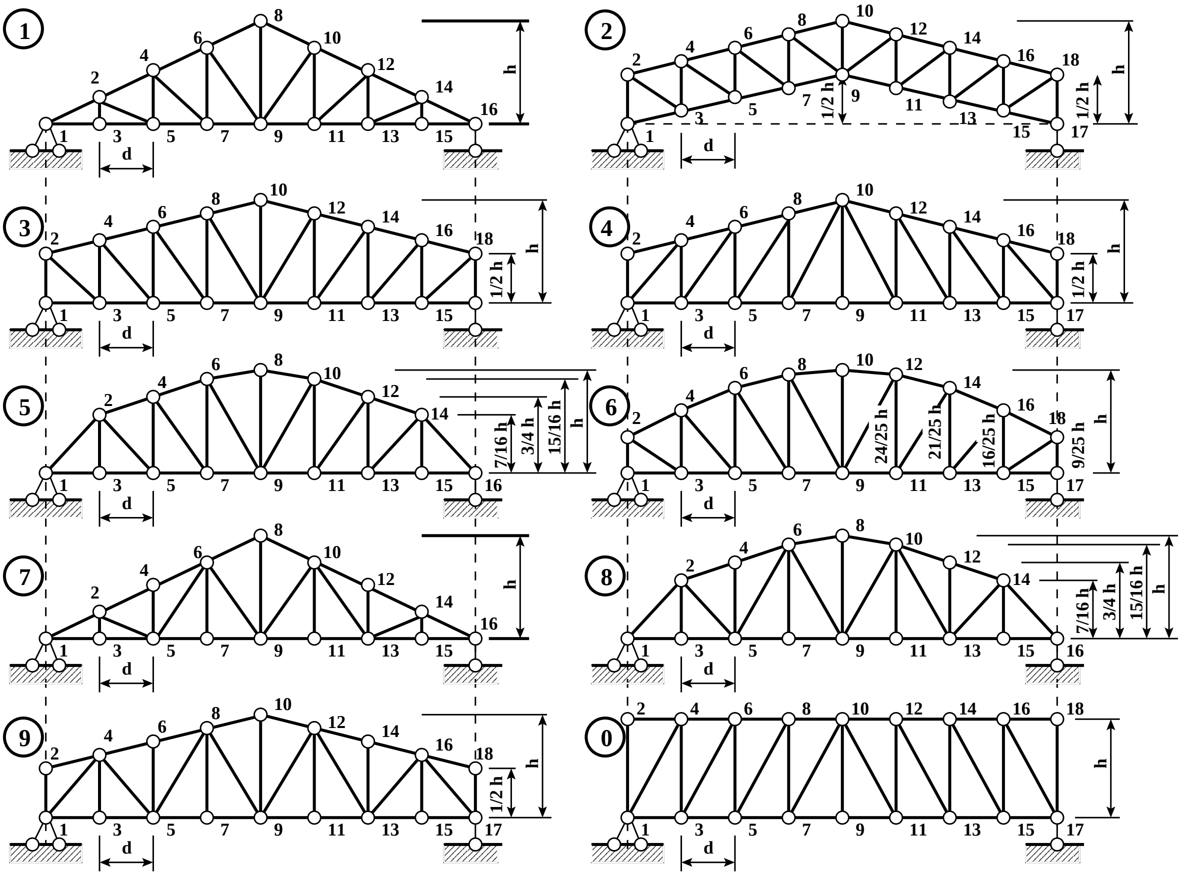


Figure 1.24. Planar trusses II[13]

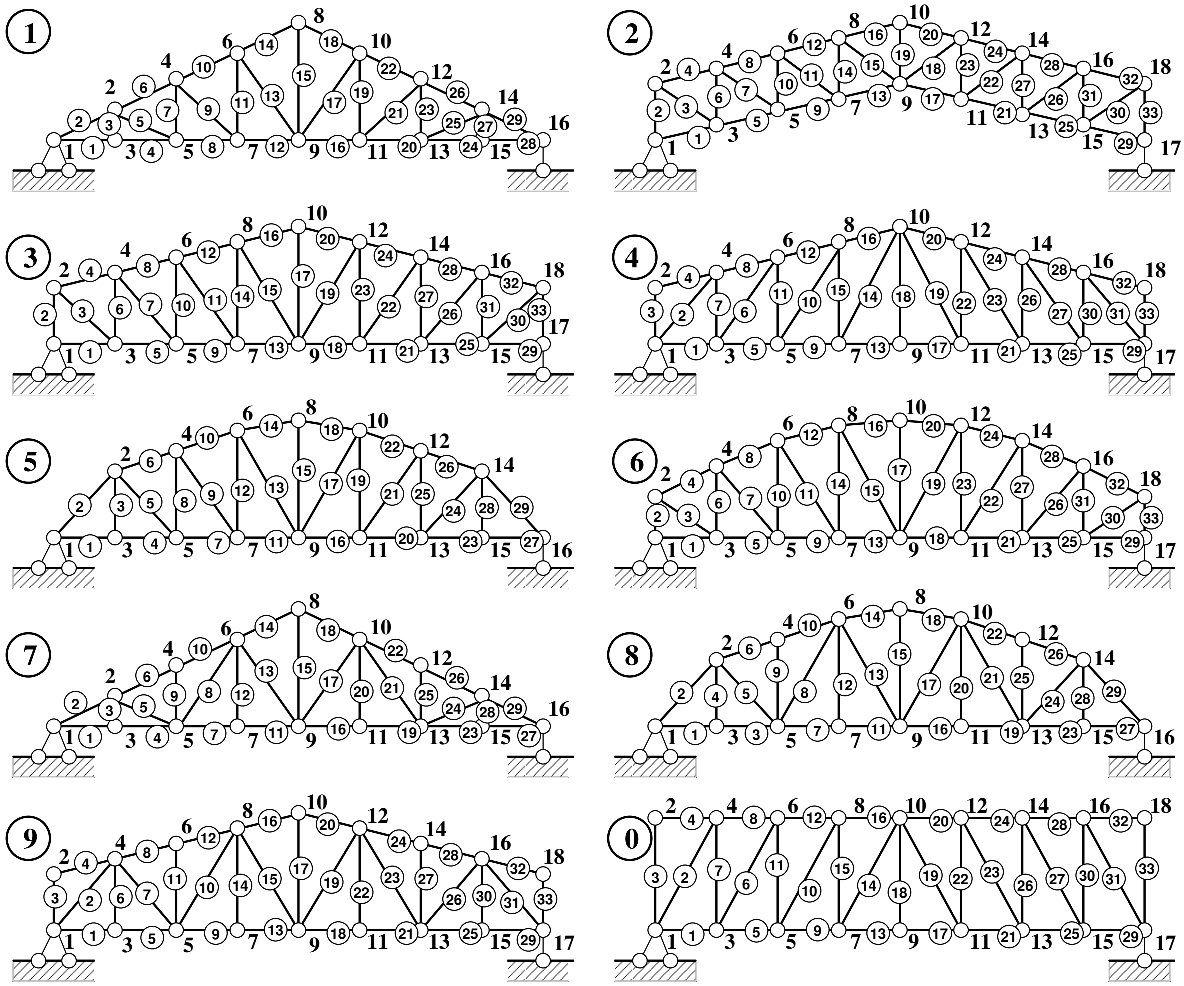


Figure 1.25. Numeration of nodes and members of trusses II

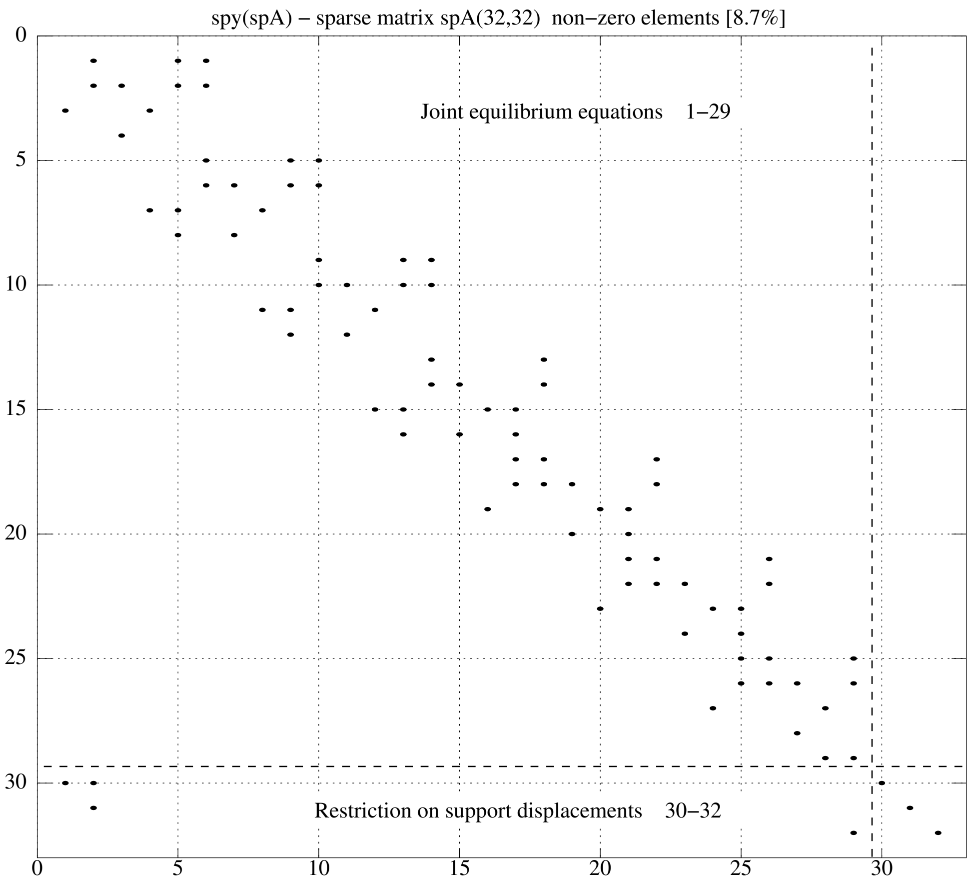


Figure 1.26. Sparsity pattern of matrix spA of truss 1N27

A. Matrices

A matrix type that stores only the values of non-zero elements and their row and column indexes is generally called sparse[1] [2]. For the storage and creation of sparse matrices we use *GNU Octave*[3] [4].

A.1 Sparse matrices and GNU Octave

A.1.1 Introduction to sparse matrices

For calculating support reactions and interaction forces on statically determinate hinged beams, also known as Gerber[5] [6] beams (see Fig. A.2), we have a system of equilibrium equations where the coefficient matrix is sparse. The sparsity pattern of this matrix spA is shown in Fig. A.1.

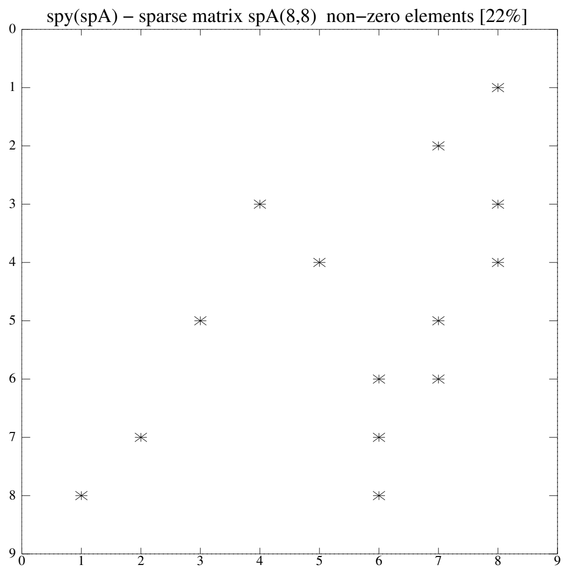


Figure A.1. Sparsity pattern of matrix spA

[1] http://www.gnu.org/software/octave/doc/interpreter/Sparse-Matrices.html. Web. 08 August 2013.

[2] http://en.wikipedia.org/wiki/Sparse_matrix. Web. 08 August 2013.

[3] http://www.obihiro.ac.jp/~suzukim/masuda/octave/html3/octave_112.html#SEC216. Web. 08 August 2013.

[4] http://www.network-theory.co.uk/docs/octave3/octave_205.html. Web. 08 August 2013.

[5] Heinrich Gerber (1832–1912), a German civil engineer and inventor of multispan hinged beams.

[6] http://de.wikipedia.org/wiki/Heinrich_Gottfried_Gerber. Web. 08 August 2013.

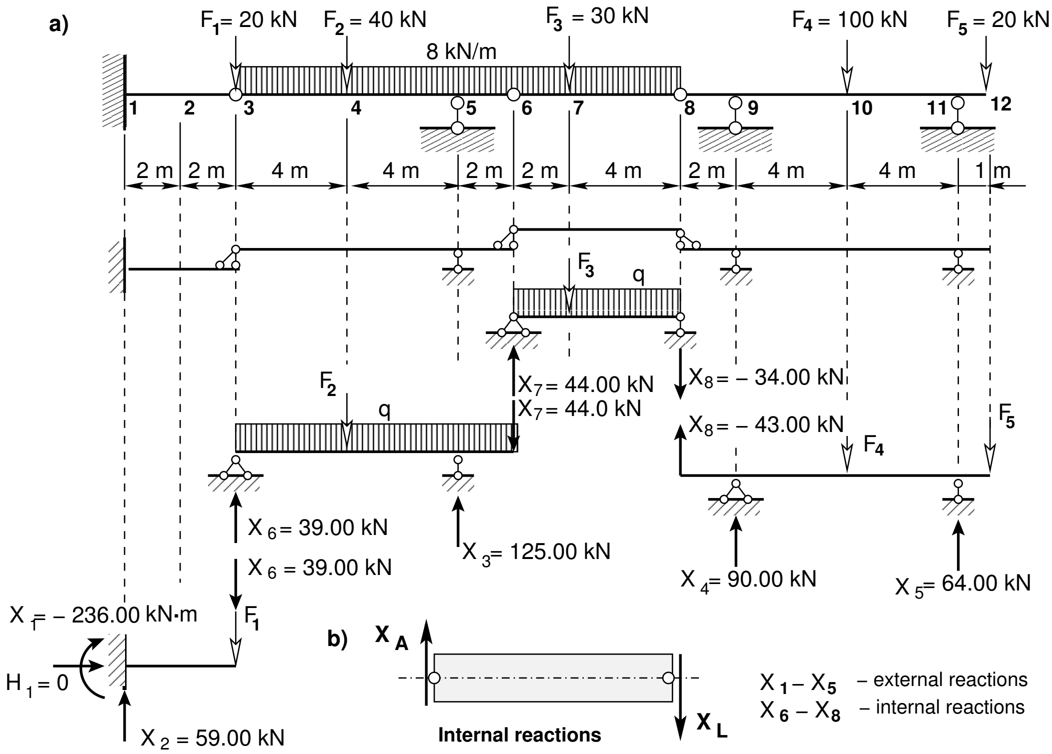


Figure A.2. Assembly sequence of a Gerber beam

To prevent a large number of calculations, this system is decomposed in such a way that an unknown force could be calculated directly with each equilibrium equation.

Consider the equilibrium equations for the beams in Fig. A.2:

beam 6–8

$$\Sigma M_6 = 0; \qquad X_8 \cdot 6 + F_3 \cdot 2 + q \cdot 6 \cdot 3 = 0 \tag{A.1}$$

$$\Sigma M_8 = 0; \qquad -X_7 \cdot 6 + F_3 \cdot 4 + q \cdot 6 \cdot 3 = 0 \tag{A.2}$$

beam 8–12

$$\Sigma M_{11} = 0; \qquad -X_8 \cdot 10 - X_4 \cdot 8 + F_4 \cdot 4 - F_5 \cdot 1 = 0 \tag{A.3}$$

$$\Sigma M_9 = 0; \qquad -X_8 \cdot 2 + X_5 \cdot 8 - F_4 \cdot 4 - F_5 \cdot 9 = 0 \tag{A.4}$$

beam 3–6

$$\Sigma M_3 = 0; \qquad -X_7 \cdot 10 + X_3 \cdot 8 - F_2 \cdot 4 - q \cdot 10 \cdot 5 = 0 \tag{A.5}$$

$$\Sigma M_5 = 0; \qquad -X_7 \cdot 2 - X_6 \cdot 8 + F_2 \cdot 4 + q \cdot 10 \cdot (5 - 2) = 0 \tag{A.6}$$

beam 1–3

$$\Sigma Z = 0; \qquad X_6 - X_2 + F_1 = 0 \tag{A.7}$$

$$\Sigma M_1 = 0; \qquad X_1 + X_6 \cdot 4 + F_1 \cdot 4 = 0 \tag{A.8}$$

Now rewrite the systems of equations (A.1)–(A.8) in matrix form:

$$\begin{array}{l} 1.\ \Sigma M_6 = 0; \\ 2.\ \Sigma M_8 = 0; \\ 3.\ \Sigma M_{11} = 0; \\ 4.\ \Sigma M_9 = 0; \\ 5.\ \Sigma M_3 = 0; \\ 6.\ \Sigma M_5 = 0; \\ 7.\ \Sigma Z = 0; \\ 8.\ \Sigma M_1 = 0 \end{array} \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & 6 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -6 & \cdot \\ \cdot & \cdot & \cdot & -8 & \cdot & \cdot & \cdot & -10 \\ \cdot & \cdot & \cdot & \cdot & 8 & \cdot & \cdot & -2 \\ \cdot & \cdot & 8 & \cdot & \cdot & \cdot & -10 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & -8 & -2 & \cdot \\ \cdot & -1 & \cdot & \cdot & \cdot & 1 & \cdot & \cdot \\ 1 & \cdot & \cdot & \cdot & \cdot & 4 & \cdot & \cdot \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \\ X_7 \\ X_8 \end{bmatrix} =$$

$$= - \begin{bmatrix} F_3 \cdot 2 + q \cdot 6 \cdot 3 \\ F_3 \cdot 4 + q \cdot 6 \cdot 3 \\ F_4 \cdot 4 - F_5 \cdot 1 \\ -F_4 \cdot 4 - F_5 \cdot 9 \\ -F_2 \cdot 4 - q \cdot 10 \cdot 5 \\ F_2 \cdot 4 + q \cdot 10 \cdot (5-2) \\ F_1 \\ F_1 \cdot 4 \end{bmatrix} \tag{A.9}$$

We have obtained the sparse system (8×8) of equilibrium equations

$$\mathbf{A} \cdot \mathbf{X} = -\mathbf{B} \tag{A.10}$$

where

$$\mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6 \\ 0 & 0 & 0 & 0 & 0 & 0 & -6 & 0 \\ 0 & 0 & 0 & -8 & 0 & 0 & 0 & -10 \\ 0 & 0 & 0 & 0 & 8 & 0 & 0 & -2 \\ 0 & 0 & 8 & 0 & 0 & 0 & -10 & 0 \\ 0 & 0 & 0 & 0 & 0 & -8 & -2 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 4 & 0 & 0 \end{bmatrix}, \qquad \mathbf{B} = \begin{bmatrix} -204 \\ -264 \\ -380 \\ 580 \\ 560 \\ -400 \\ -20 \\ -80 \end{bmatrix} \tag{A.11}$$

The Gerber beam support reactions are calculated by hand in the reverse order to that of the assembly sequence (see calculating order in Eqs. (A.1)–(A.9), and Fig. A.2). The sparsity pattern of the matrix $\mathbf{A}$ of Eq. (A.11) is shown in Fig. A.1.

The non-zero elements of the matrix $\mathbf{A}$ can be represented as the row (**iv**), column (**jv**) and data (**sv**) vectors in Eq. (A.12).

$$\begin{aligned} \mathbf{iv} &= [1 \quad 2 \quad 3 \quad 3 \ 4 \quad 4 \ 5 \quad 5 \quad 6 \quad 6 \quad 7 \ 7 \ 8 \ 8] \\ \mathbf{jv} &= [8 \quad 7 \quad 4 \quad 8 \ 5 \quad 8 \ 3 \quad 7 \quad 6 \quad 7 \quad 2 \ 6 \ 1 \ 6] \\ \mathbf{sv} &= [6 \ -6 \ -8 \ -10 \ 8 \ -2 \ 8 \ -10 \ -8 \ -2 \ -1 \ 1 \ 1 \ 4] \end{aligned} \tag{A.12}$$

A.1.2 Creating sparse matrices

GNU Octave uses the compressed column format *storage technique*.[7] There are several modes to create a sparse matrix. Sparse matrices can be constructed from matrices or vectors.

The function **sparse**(**iv**, **jv**, **sv**) has constructed a sparse matrix **spA** (shown in computing diary A.1) from three vectors representing the row (**iv**), column (**jv**) and data (**sv**) (see Eq. (A.12)). In this diary, the conversion of the sparse matrix **spA** to a full matrix **A** (**full**(**spA**)) is shown.

Computing diary A.1

```
octave:1> iv = [1 2 3 3 4 4 5 5 6 6 7 7 8 8]
iv =

   1   2   3   3   4   4   5   5   6   6   7   7   8   8

octave:2> jv = [8 7 4 8 5 8 3 7 6 7 2 6 1 6]
jv =

   8   7   4   8   5   8   3   7   6   7   2   6   1   6

octave:3> sv = [6 -6 -8 -10 8 -2 8 -10 -8 -2 -1 1 1 4]
sv =

    6   -6   -8  -10    8   -2    8  -10   -8   -2   -1    1    1    4

octave:4> spA=sparse(iv, jv, sv)
spA =
Compressed Column Sparse (rows = 8, cols = 8, nnz = 14 [22%])

  (8, 1) ->  1
  (7, 2) -> -1
  (5, 3) ->  8
  (3, 4) -> -8
  (4, 5) ->  8
  (6, 6) -> -8
  (7, 6) ->  1
  (8, 6) ->  4
  (2, 7) -> -6
  (5, 7) -> -10
  (6, 7) -> -2
  (1, 8) ->  6
  (3, 8) -> -10
  (4, 8) -> -2

octave:5> A=full(spA)
A =
    0    0    0    0    0    0    0    6
    0    0    0    0    0    0   -6    0
    0    0    0   -8    0    0    0  -10
    0    0    0    0    8    0    0   -2
    0    0    8    0    0    0  -10    0
    0    0    0    0    0   -8   -2    0
    0   -1    0    0    0    1    0    0
    1    0    0    0    0    4    0    0
octave:6>
```

[7] http://www.obihiro.ac.jp/~suzukim/masuda/octave/html3/octave_113.html#SEC219. Web. 08 August 2013.

The non-zero elements of the matrix **A** of Eq. (A.11) can be represented as the matrix **A1** with columns **iv**, **jv** and **sv**, see Eq. (A.12):

$$\mathbf{A1} = \begin{bmatrix} 1 & 8 & 6 \\ 2 & 7 & -6 \\ 3 & 4 & -8 \\ 3 & 8 & -10 \\ 4 & 5 & 8 \\ 4 & 8 & -2 \\ 5 & 3 & 8 \\ 5 & 7 & -10 \\ 6 & 6 & -8 \\ 6 & 7 & -2 \\ 7 & 2 & -1 \\ 7 & 6 & 1 \\ 8 & 1 & 1 \\ 8 & 6 & 4 \end{bmatrix} \tag{A.13}$$

The function **spconvert**(**A1**) (given in computing diary A.2) constructs a sparse matrix **spA** from the matrix **A1**.

Computing diary A.2

```
octave:1>  A1=[1    8    6;
    2    7    -6;
    3    4    -8;
    3    8   -10;
    4    5     8;
    4    8    -2;
    5    3     8;
    5    7   -10;
    6    6    -8;
    6    7    -2;
    7    2    -1;
    7    6     1;
    8    1     1;
    8    6     4]
```

```
A1 =
    1    8    6
    2    7   -6
    3    4   -8
    3    8  -10
    4    5    8
    4    8   -2
    5    3    8
    5    7  -10
    6    6   -8
    6    7   -2
    7    2   -1
    7    6    1
    8    1    1
    8    6    4
```

```
octave:2> spA=spconvert(A1)
spA =
Compressed Column Sparse (rows = 8, cols = 8, nnz = 14 [22%])

  (8, 1) ->  1
  (7, 2) -> -1
  (5, 3) ->  8
  (3, 4) -> -8
  (4, 5) ->  8
  (6, 6) -> -8
  (7, 6) ->  1
  (8, 6) ->  4
  (2, 7) -> -6
  (5, 7) -> -10
  (6, 7) -> -2
  (1, 8) ->  6
  (3, 8) -> -10
  (4, 8) -> -2
```

The function **sparse**(**A**) converts the full matrix **A** to a sparse matrix (see computing diary A.3).

Computing diary A.3

```
octave:6> A
A =
    0    0    0    0    0    0    0    6
    0    0    0    0    0    0   -6    0
    0    0    0   -8    0    0    0  -10
    0    0    0    0    8    0    0   -2
    0    0    8    0    0    0  -10    0
    0    0    0    0    0   -8   -2    0
    0   -1    0    0    0    1    0    0
    1    0    0    0    0    4    0    0

octave:7> spA=sparse(A)
spA =
Compressed Column Sparse (rows = 8, cols = 8, nnz = 14 [22%])

  (8, 1) -> 1
  (7, 2) -> -1
  (5, 3) -> 8
  (3, 4) -> -8
  (4, 5) -> 8
  (6, 6) -> -8
  (7, 6) -> 1
  (8, 6) -> 4
  (2, 7) -> -6
  (5, 7) -> -10
  (6, 7) -> -2
  (1, 8) -> 6
  (3, 8) -> -10
  (4, 8) -> -2

octave:8>
```

A.1.3 Sparse matrix functions in the EST method

There are several functions that manipulate sparse matrices: full, sparse, spconvert, spfind, sprank, spy, speye, etc. [8]

We shall introduce the GNU Octave function spA=spInsertBtoA(spA,M,N,spB) (p. 69) written for the EST method. This function inserts a sparse matrix spB into the sparse matrix spA, starting at row index M and column index N. The overlapping elements of the matrices spA and spB are added together.

The insertion of the matrix **B** (Eq. (A.14)) into the sparse matrix **spC** is described in computing diary A.4. There, the elements of the matrix C of value *2* (C(5,5) and C(6,6)) have been obtained as the sum of overlapped elements of matrices spB and spB1.

$$\mathbf{B} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.14}$$

[8] http://www.obihiro.ac.jp/~suzukim/masuda/octave/html3/octave_113.html#SEC222. Web. 08 August 2013.

Computing diary A.4

```
octave:1> iv = [1 2 3 4 5 6]
iv =

   1   2   3   4   5   6

octave:2> jv = [1 2 4 3 5 6]
jv =

   1   2   4   3   5   6

octave:3> sv = [1 1 1 1 1 1]
sv =sparse matrix spB

   1   1   1   1   1   1

octave:4> spB=sparse(iv, jv, sv)
spB =

Compressed Column Sparse (rows = 6, cols = 6, nnz = 6 [17%])

  (1, 1) ->  1
  (2, 2) ->  1
  (4, 3) ->  1
  (3, 4) ->  1
  (5, 5) ->  1
  (6, 6) ->  1

octave:5> spB1=spB;
octave:6> spC=sparse(10,10)
spC =

Compressed Column Sparse (rows = 10, cols = 10, nnz = 0 [0%])


octave:7> spC=spInsertBtoA(spC,1,1,spB);
octave:8> spC=spInsertBtoA(spC,5,5,spB1)
spC =

Compressed Column Sparse (rows = 10, cols = 10, nnz = 10 [10%])

  (1, 1) ->  1
  (2, 2) ->  1
  (4, 3) ->  1
  (3, 4) ->  1
  (5, 5) ->  2
  (6, 6) ->  2
  (8, 7) ->  1
  (7, 8) ->  1
  (9, 9) ->  1
  (10, 10) ->  1
```

```
octave:9> C=full(spC)
C =

   1   0   0   0   0   0   0   0   0   0
   0   1   0   0   0   0   0   0   0   0
   0   0   0   1   0   0   0   0   0   0
   0   0   1   0   0   0   0   0   0   0
   0   0   0   0   2   0   0   0   0   0
   0   0   0   0   0   2   0   0   0   0
   0   0   0   0   0   0   0   1   0   0
   0   0   0   0   0   0   1   0   0   0
   0   0   0   0   0   0   0   0   1   0
   0   0   0   0   0   0   0   0   0   1

octave:10>
```

Next we introduce the GNU Octave function spA=spSisestaArv(spA,M,N,V) (p. 69) written for the EST method. This function inserts a numeric value **V** into the sparse matrix spA at row index M and column index N.

A.2 Transformation matrices

Consider the two right-handed coordinate systems of Fig. A.3, defined by orthogonal unit vectors $\mathbf{i}, \mathbf{j}, \mathbf{k}$, and $\mathbf{i}^*, \mathbf{j}^*, \mathbf{k}^*$. Let xyz be global coordinates and $x^*y^*z^*$ a local coordinate system.

The vector $\vec{\mathbf{F}}$ in Fig. A.3 can be written as the sum of two vectors along the coordinate axes $\mathbf{i}, \mathbf{k}$ with magnitude F_x, F_z and along the coordinate axes $\mathbf{i}^*, \mathbf{k}^*$ with magnitude F_x^*, F_z^*.

$$\vec{\mathbf{F}} = F_x \cdot \vec{\mathbf{i}} + F_z \cdot \vec{\mathbf{k}} = F_x^* \vec{\mathbf{i}}^* + F_z^* \vec{\mathbf{k}}^*, \qquad \left\{ \begin{array}{l} \cdot \vec{\mathbf{i}}^* \\ \cdot \vec{\mathbf{k}}^* \end{array} \right. , \quad \left\{ \begin{array}{l} \cdot \vec{\mathbf{i}} \\ \cdot \vec{\mathbf{k}} \end{array} \right. \tag{A.15}$$

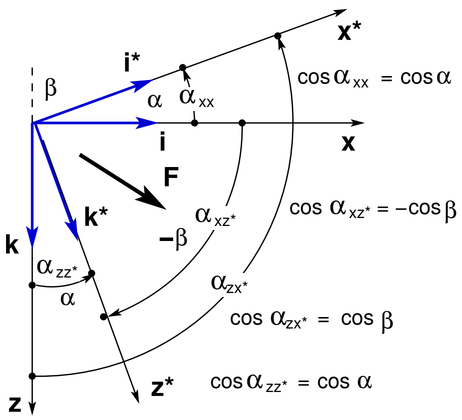


Figure A.3. Coordinate transformation

To find the components of the vector $\vec{\mathbf{F}}$ of Eq. (A.15), we multiply this equation by $\vec{\mathbf{i}}^*$ and $\vec{\mathbf{k}}^*$. The scalar products are:

$$\begin{aligned} \vec{\mathbf{F}} \cdot \vec{\mathbf{i}}^* &= F_x^* &= F_x \cdot \vec{\mathbf{i}} \cdot \vec{\mathbf{i}}^* &+ F_z \cdot \vec{\mathbf{k}} \cdot \vec{\mathbf{i}}^* \\ \vec{\mathbf{F}} \cdot \vec{\mathbf{k}}^* &= F_z^* &= F_x \cdot \vec{\mathbf{i}} \cdot \vec{\mathbf{k}}^* &+ F_z \cdot \vec{\mathbf{k}} \cdot \vec{\mathbf{k}}^* \end{aligned} \tag{A.16}$$

where the scalar product of the two orthogonal vectors is zero.

To find the inverse transformation, we multiply Eq. (A.15) by $\vec{\mathbf{i}}$ and $\vec{\mathbf{k}}$. The scalar products are:

$$\begin{aligned} \vec{\mathbf{F}} \cdot \vec{\mathbf{i}} &= F_x &= F_x^* \cdot \vec{\mathbf{i}}^* \cdot \vec{\mathbf{i}} &+ F_z^* \cdot \vec{\mathbf{k}}^* \cdot \vec{\mathbf{i}} \\ \vec{\mathbf{F}} \cdot \vec{\mathbf{k}} &= F_z^* &= F_x^* \cdot \vec{\mathbf{i}}^* \cdot \vec{\mathbf{k}} &+ F_z^* \cdot \vec{\mathbf{k}}^* \cdot \vec{\mathbf{k}} \end{aligned} \tag{A.17}$$

The scalar product of the two unit vectors is related to the cosine of the angle between these vectors (Fig. A.3).

$$\begin{aligned} \vec{\mathbf{i}} \cdot \vec{\mathbf{i}}^* &= \vec{\mathbf{i}}^* \cdot \vec{\mathbf{i}} = \cos\alpha_{xx^*}, & \vec{\mathbf{i}} \cdot \vec{\mathbf{k}}^* &= \cos\alpha_{xz^*} \\ \vec{\mathbf{k}} \cdot \vec{\mathbf{k}}^* &= \vec{\mathbf{k}}^* \cdot \vec{\mathbf{k}} = \cos\alpha_{zz^*}, & \vec{\mathbf{i}}^* \cdot \vec{\mathbf{k}} &= \cos\alpha_{zx^*} \end{aligned} \tag{A.18}$$

In Fig. A.3, we show the direction cosines of the vector $\vec{\mathbf{F}}$:

$$\begin{aligned} \cos\alpha_{xx^*} &= \cos\alpha, & \cos\alpha_{zx^*} &= \cos\beta \\ \cos\alpha_{zz^*} &= \cos\alpha, & \cos\alpha_{xz^*} &= -\cos\beta \end{aligned} \tag{A.19}$$

Be careful using cosine and sine angles associated with coordinate system transformation: $\cos\alpha_{xx^*} = \cos\alpha$ and $\cos\alpha_{zx^*} = \cos\beta$ $(\cos\beta = \cos(90^\circ + \alpha) = -\sin\alpha)$.

The length l and the direction cosines of an element can be calculated using coordinates x_A, z_A of the node at the beginning and coordinates x_L, z_L of the node at the end of the element (Fig. A.4):

$$l = \sqrt{(z_L - z_A)^2 + (x_L - x_A)^2} \tag{A.20}$$

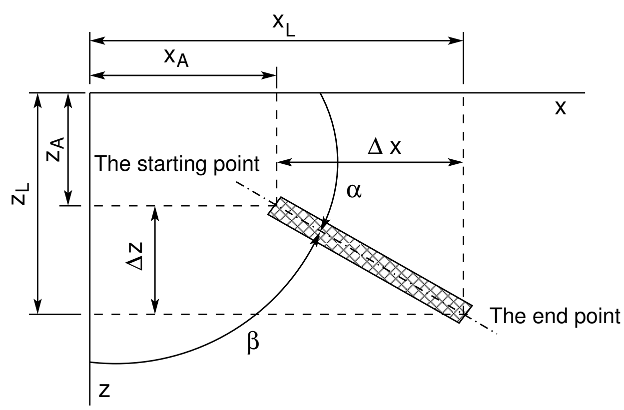


Figure A.4. Direction cosines of an element

$$\cos\alpha = \frac{x_L - x_A}{l} \tag{A.21}$$

$$\cos\beta = \frac{z_L - z_A}{l} \tag{A.22}$$

We now consider the transformation of the vector $\vec{\mathbf{F}}$ components F_x, F_z of Eq. (A.15) from the global xy coordinate system to the components F_x^*, F_z^* in the local x^*y^* coordinate system.

$$\begin{bmatrix} F_x^* \\ F_z^* \end{bmatrix} = \begin{bmatrix} \cos\alpha & \cos\beta \\ -\cos\beta & \cos\alpha \end{bmatrix} \begin{bmatrix} F_x \\ F_z \end{bmatrix} \tag{A.23}$$

Taking into account that $\cos\beta = \cos(90\,^\circ + \alpha) = -\sin\alpha$, we can write the above equation as

$$\begin{bmatrix} F_x^* \\ F_z^* \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} F_x \\ F_z \end{bmatrix} \tag{A.24}$$

The inverse transformation of the vector $\vec{\mathbf{F}}$ components F_x^*, F_z^* of Eq. (A.15) from the local x^*y^* coordinate system to the components F_x, F_z of the global xy coordinate system:

$$\begin{bmatrix} F_x \\ F_z \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\cos\beta \\ \cos\beta & \cos\alpha \end{bmatrix} \begin{bmatrix} F_x^* \\ F_z^* \end{bmatrix} \tag{A.25}$$

Taking into account that $\cos\beta = \cos(90\,^\circ + \alpha) = -\sin\alpha$, we can write Eq. (A.25) in the form

$$\begin{bmatrix} F_x \\ F_z \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} F_x^* \\ F_z^* \end{bmatrix} \tag{A.26}$$

Comparing the transformation matrix (A.23) with that of (A.25), we can see that they are transposed (rows and columns reversed). The multiplication of the matrix (A.23) by a transpose of itself of Eq. (A.25) gives the identity matrix

$$\begin{bmatrix} \cos\alpha & \cos\beta \\ -\cos\beta & \cos\alpha \end{bmatrix} \begin{bmatrix} \cos\alpha & -\cos\beta \\ \cos\beta & \cos\alpha \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{A.27}$$

We have thus proved that the matrices are *orthogonal* – an orthogonal matrix has the property that its transpose equals the inverse.

B. Work and work-energy theorem

B.1 Work done by internal and external forces

The work-energy theorem in structural analysis: the sum of the work done by internal and external forces is zero:

$$W_i + W_e = 0 \tag{B.1}$$

where W_i is the work done by internal forces and W_e is the work done by external forces. The Green's functional for a frame element is [KHMW10]

$$\begin{aligned} &\underbrace{-\int_a^b N_x \hat{\lambda} dx - \int_a^b M_y \hat{\psi}_y dx}_{W_i \; - \; work \; of \; internal \; forces} + \\ &+ \underbrace{[N_x \hat{u}]_a^b + [Q_y \hat{w} + M_y \hat{\varphi}_y]_a^b + \int_a^b q_x(x) \hat{u} dx + F_{xi} \hat{u}_i + \int_a^b q_z(x) \hat{w} dx + F_{zi} \hat{w}_i}_{W_e \; - \; work \; of \; external \; forces} = 0 \end{aligned} \tag{B.2}$$

where we consider two systems (states) of forces associated with respective deformations and displacements [BP13]:

N_x, M_y are the internal axial force and bending moment of the first load state;

$\hat{\lambda}$, $\hat{\psi}_y$ – axial and bending deformations of the second load state;

$N_x|_a^b$, $Q_y|_a^b$, $M_y|_a^b$ – axial force, shear force and bending moment of the first load state at boundaries a and b;

$\hat{u}|_a^b$, $\hat{w}|_a^b$, $\hat{\varphi}_y|_a^b$ – longitudinal and transverse displacements, and the rotation of the cross section of the second load state at boundaries a and b;

$q_x(x)$, $q_z(x)$ – distributed loads of the first load state;

$\hat{u}(x)$, $\hat{w}(x)$ – longitudinal and transverse displacements of the second load state;

F_{xi}, F_{zi} – force components of the first load state, applied at point i in the x and z directions, respectively;

$\hat{u}_i$, $\hat{w}_i$ – longitudinal and transverse displacements of point i of the second load state.

The first and second elements of Eq. (B.2) describe the work W_i of internal forces:

$$W_i = -\int_a^b N_x \hat{\lambda} dx - \int_a^b M_y \hat{\psi}_y dx \tag{B.3}$$

The final four elements of Eq. (B.2) describe the work W_f done by active forces:

$$W_f = \int_a^b q_x(x)\hat{u}dx + F_{xi}\hat{u}_i + \int_a^b q_z(x)\hat{w}dx + F_{zi}\hat{w}_i \tag{B.4}$$

The third and fourth elements of Eq. (B.2) describe the work W_b done by boundary forces (fixed-end forces and moments at joints[1], support reactions):

$$W_b = [N_x\hat{u}]_a^b + [Q_y\hat{w} + M_y\hat{\varphi}_y]_a^b \tag{B.5}$$

The external work W_e can be divided into two parts:

- W_f – work done by active forces, e.g. concentrated loads, uniformly distributed loads;
- W_b – work done by reaction forces, e.g. support reactions (Fig. **??**), internal reactions [WP960] (contact forces [2]) (Fig. **??**).

$$W_e = W_b + W_f \tag{B.6}$$

Applying now Eq. (B.6) to (B.1), we obtain

$$W_i + W_b + W_f = 0 \tag{B.7}$$

Equation (B.7) is a shortened form of Eq. (B.2).

With the elastic energy $U_{elastic\ energy}$ and dissipation energy D existing, the work W_i done by internal forces is in relation to the internal energy $U_{elastic} + D$:

$$\underbrace{W_i}_{internal\ work} = - \underbrace{U}_{elastic\ energy} - \underbrace{D}_{dissipation\ energy} \tag{B.8}$$

Equations (B.2) and (B.7) are the basic methodical tools of structural analysis.

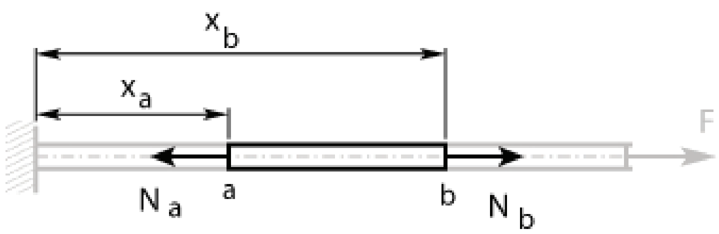


Figure B.1. Bar member a–b

[1] The fixed-end forces and moments at joints are called the internal reactions [WP960] or the joint contact forces [GN12].

[2] *A contact force* is a force that acts at the points of contact between two objects [Rand07].

Example B.1 ***(conservation of mechanical energy).*** *We consider the bar from Fig. B.1, subjected to the load F. The bar has been split into sections where the normal forces N_a and N_b exerted on the member at cross-sections a and b are treated as external loads [VrCty] or, to be more precise, as boundary forces.*

No loads act on the bar member a–b, and so $W_f = 0$ and $W_i \neq 0$ in Eq. (B.7). The expression for energy conservation of the bar member a–b is

$$W_i + W_b = 0 \tag{B.9}$$

or

$$\underbrace{-\int_a^b N_x \hat{\lambda} dx}_{W_i \,-\, work\ of\ internal\ forces} \quad + \quad \underbrace{[N_x \hat{u}]_a^b}_{W_b \,-\, work\ of\ internal\ reactions\ [WP960]} \quad = 0 \tag{B.10}$$

C. Computer programs for the EST method

The basic URLs (folder paths for programs): D:/, E:/, F:/, S:/, Z:/

C.1 Programs for first-order analysis

Program C.1 (spESTframe1LaheDefWFI.m)[1] *11 – is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.2 (spESTframe2LaheDefWFI.m)[2] *11 – is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.3 (spESTframe3LaheDefWFI.m)[3] *11 – is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.4 (spESTframe4LaheDefWFI.m)[4] *11 – is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

[1] ./octavePrograms/spESTframe1LaheDefWFI.m
[2] ./octavePrograms/spESTframe2LaheDefWFI.m
[3] ./octavePrograms/spESTframe3LaheDefWFI.m
[4] ./octavePrograms/spESTframe4LaheDefWFI.m

Program C.5 (spESTframe5LaheDefWFI.m)[5] *11* – *is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.6 (spESTframe6LaheDefWFI.m)[6] *11* – *is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.7 (spESTframe7LaheDefWFI.m)[7] *11* – *is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.8 (spESTframe8LaheDefWFI.m)[8] *11* – *is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.9 (spESTframe9LaheDefWFI.m)[9] *11* – *is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.10 (spESTframe10LaheDefWFI.m)[10] *11* – *is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Function C.1 (LaheFrameDFIm(baasi0,Ntoerkts,esQkoormus,esFjoud, sSolmF,tsolm,tSiire,krdn,selem)) [11] *55, 56, 58* – *is used to assemble and solve*

[5] ./octavePrograms/spESTframe5LaheDefWFI.m
[6] ./octavePrograms/spESTframe6LaheDefWFI.m
[7] ./octavePrograms/spESTframe7LaheDefWFI.m
[8] ./octavePrograms/spESTframe8LaheDefWFI.m
[9] ./octavePrograms/spESTframe9LaheDefWFI.m
[10] ./octavePrograms/spESTframe10LaheDefWFI.m
[11] ./octavePrograms/LaheFrameDFIm.m

the boundary problem equations of a plane frame.
Called functions:
yzhqzm(baasi0,x,a,qx,qz,EA,EJ) [12],
InsertBtoA(A,I,J,IM,JN,B,M,N) [13],
spInsertBtoA(spA,IIv,IJv,spvF) [14],
spSisestaArv(spA,iv,jv,sv) [15],
SpTeisendusMaatriks2x2(NSARV,NEARV,VarrasN,krdn,selem) [16],
SpTeisendusMaatriks(NSARV,NEARV,VarrasN,krdn,selem) [17],
SpTeisendusUhikMaatriks2x2(VarrasN) [18],
SpTeisendusUhikMaatriks0x1v(VarrasN) [19],
SpTeisendusUhikMaatriks(VarrasN) [20],
SpToeReaktsioonZvektor(NSARV,NEARV,VarrasN,krdn,selem) [21],
SpToeReaktsioonXvektor(NSARV,NEARV,VarrasN,krdn,selem) [22],
SpToeSiirdeFiVektor(VarrasN) [23],
SpToeSiirdeUvektor(NSARV,NEARV,VarrasN,krdn,selem) [24],
SpToeSiirdeWvektor(NSARV,NEARV,VarrasN,krdn,selem) [25],
VardadSolmes(NSARV,NEARV,Solm,AB,ABB) [26],
VardaPikkus(NSARV,NEARV,krdn,selem) [27],
ylfhlin(baasi0,x,EA,GAr,EJ) [28],
ysplfhlin(baasi0,x,EA,GAr,EJ)) [29],
ysplvfmhvI(baasi0,x,l,EA,GAr,EJ) [30],
yzfzv(baasi0,x,a,Fx,Fz,EA,EJ) [31],
yzhqzm(baasi0,x,a,qx,qz,EA,EJ) [32].

[12] `./octavePrograms/ESTFrKrmus.m`
[13] `./octavePrograms/InsertBtoA.m`
[14] `./octavePrograms/spInsertBtoA.m`
[15] `./octavePrograms/spSisestaArv.m`
[16] `./octavePrograms/SpTeisendusMaatriks2x2.m`
[17] `./octavePrograms/SpTeisendusMaatriks.m`
[18] `./octavePrograms/SpTeisendusUhikMaatriks2x2.m`
[19] `./octavePrograms/SpTeisendusUhikMaatriks0x1v.m`
[20] `./octavePrograms/SpTeisendusUhikMaatriks.m`
[21] `./octavePrograms/SpToeReaktsioonZvektor.m`
[22] `./octavePrograms/SpToeReaktsioonXvektor.m`
[23] `./octavePrograms/SpToeSiirdeFiVektor.m`
[24] `./octavePrograms/SpToeSiirdeUvektor.m`
[25] `./octavePrograms/SpToeSiirdeWvektor.m`
[26] `./octavePrograms/VardadSolmes.m`
[27] `./octavePrograms/VardaPikkus.m`
[28] `./octavePrograms/ylfhlin.m`
[29] `./octavePrograms/ysplfhlin.m`
[30] `./octavePrograms/ysplvfmhvI.m`
[31] `./octavePrograms/yzfzv.m`
[32] `./octavePrograms/yzhqzm.m`

Function C.2 (SisejoudPunktism(VardaNr,X,AlgPar,lvarras,selem, esQkoormus,esFjoud,suurused)) [33] *55, 56, 58* – *is used to compute the displacements and forces of the element 'VardaNr' at $x = X$.*
Called function:
ESTFrKrmus(baasi0,xx,Li,Fjoud,qkoormus,EA,EI) [34].

Program C.11 (spESTframe16WFI.m)[35] *15* – *is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.12 (spESTframe27WFI.m)[36] *15* – *is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.13 (spESTframe38WFI.m)[37] *15* – *is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.14 (spESTframe49WFI.m)[38] *15* – *is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.15 (spESTframe50WFI.m)[39] *15* – *is used to compute the displacements and internal forces of a plane frame.*
Called functions:
LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.16 (spESTframe3hinge1WFI.m)[40] – *is used to compute the displacements and internal forces of a three-hinged frame.*

[33] ./octavePrograms/SisejoudPunktism.m
[34] ./octavePrograms/ESTFrKrmus.m
[35] ./octavePrograms/spESTframe16WFI.m
[36] ./octavePrograms/spESTframe27WFI.m
[37] ./octavePrograms/spESTframe38WFI.m
[38] ./octavePrograms/spESTframe49WFI.m
[39] ./octavePrograms/spESTframe50WFI.m
[40] ./octavePrograms/spESTframe3hinge1WFI.m

Called functions:

LaheFrameDFIm.m,
SisejoudPunktism.m.

Program C.17 (spESTframe3hinge1NQM.m)[41] *20* – *is used to compute the internal forces of a three-hinged frame.*
Called functions:

LaheFrame3hingeNQM.m
Sisejoud3LraamiPnktism.m.

Program C.18 (spESTframe3hinge2NQM.m)[42] *20* – *is used to compute the internal forces of a three-hinged frame.*
Called functions:

LaheFrame3hingeNQM.m
Sisejoud3LraamiPnktism.m.

Program C.19 (spESTframe3hinge3NQM.m)[43] *20* – *is used to compute the internal forces of a three-hinged frame.*
Called functions:

LaheFrame3hingeNQM.m
Sisejoud3LraamiPnktism.m.

Program C.20 (spESTframe3hinge4NQM.m)[44] *20* – *is used to compute the internal forces of a three-hinged frame.*
Called functions:

LaheFrame3hingeNQM.m
Sisejoud3LraamiPnktism.m.

Program C.21 (spESTframe3hinge5NQM.m)[45] *20* – *is used to compute the internal forces of a three-hinged frame.*
Called functions:

LaheFrame3hingeNQM.m
Sisejoud3LraamiPnktism.m.

Program C.22 (spESTframe3hinge6NQM.m)[46] *20* – *is used to compute the internal forces of a three-hinged frame.*
Called functions:

LaheFrame3hingeNQM.m
Sisejoud3LraamiPnktism.m.

---

41 `./octavePrograms/spESTframe3hinge1NQM.m`
42 `./octavePrograms/spESTframe3hinge2NQM.m`
43 `./octavePrograms/spESTframe3hinge3NQM.m`
44 `./octavePrograms/spESTframe3hinge4NQM.m`
45 `./octavePrograms/spESTframe3hinge5NQM.m`
46 `./octavePrograms/spESTframe3hinge6NQM.m`

Program C.23 (spESTframe3hinge7NQM.m)[47] *20* – *is used to compute the internal forces of a three-hinged frame.*
Called functions:
LaheFrame3hingeNQM.m
Sisejoud3LraamiPnktism.m.

Program C.24 (spESTframe3hinge8NQM.m)[48] *20* – *is used to compute the internal forces of a three-hinged frame.*
Called functions:
LaheFrame3hingeNQM.m
Sisejoud3LraamiPnktism.m.

Program C.25 (spESTframe3hinge9NQM.m)[49] *20* – *is used to compute the internal forces of a three-hinged frame.*
Called functions:
LaheFrame3hingeNQM.m
Sisejoud3LraamiPnktism.m.

Program C.26 (spESTframe3hinge10NQM.m)[50] *20* – *is used to compute the internal forces of a three-hinged frame.*
Called functions:
LaheFrame3hingeNQM.m
Sisejoud3LraamiPnktism.m.

Function C.3 (LaheFrame3hingeNQM(Ntoerkts,esQkoormus,esFjoud, sSolmF,tsolm,krdn,selem)) [51] *59*, *60* – *is used to assemble and solve the boundary problem equations of a statically determinate plane frame.*
Called functions:
ESTSKrmus(xx,Li,Fjoud,qkoormus) [52],
InsertBtoA(A,I,J,IM,JN,B,M,N) [53],
spInsertBtoA(spA,IIv,IJv,spvF) [54],
spSisestaArv(spA,iv,jv,sv) [55],
SpTeisendusMaatriks2x2D(NSARV,NEARV,VarrasN,krdn,selem) [56],
SpTeisendusMaatriksD(NSARV,NEARV,VarrasN,krdn,selem) [57],

---
[47] ./octavePrograms/spESTframe3hinge7NQM.m
[48] ./octavePrograms/spESTframe3hinge8NQM.m
[49] ./octavePrograms/spESTframe3hinge9NQM.m
[50] ./octavePrograms/spESTframe3hinge10NQM.m
[51] ./octavePrograms/LaheFrame3hingeNQM.m
[52] ./octavePrograms/ESTSKrmus.m
[53] ./octavePrograms/InsertBtoA.m
[54] ./octavePrograms/spInsertBtoA.m
[55] ./octavePrograms/spSisestaArv.m
[56] ./octavePrograms/SpTeisendusMaatriks2x2D.m
[57] ./octavePrograms/SpTeisendusMaatriksD.m

SpTeisendusUhikMaatriks0x1v(VarrasN) [58],
SpTeisendusUhikMaatriks2x2(VarrasN) [59],
SpTeisendusUhikMaatriks(VarrasN) [60],
VardadSolmesD(NSARV,NEARV,Solm,AB,ABB) [61],
VardaPikkusD(NSARV,NEARV,krdn,selem) [62],
ylSfhlin(x) [63],
yspSlfhlin(x) [64],
yspSlvfmhvI(x) [65],
yzSfzv(x,a,Fx,Fz) [66],
yzShqz(x,qx,qz) [67].

Function C.4 (Sisejoud3LraamiPnktism(VardaNr,X,AlgPar,lvarras, esFjoud,esQkoormus,suurused)) [68] *59, 60* – *is used to compute the displacements and internal forces of the element 'VardaNr' at* $x = X$.
Called function:
ESTSKrmu(xx,Li,Fjoud,qkoormus) [69].

Program C.27 (spESTbeam32LaheWFI.m)[70] *25* – *is used to compute the displacements and internal forces of a beam.*
Called functions:
LaheBeamDFI.m
SisejoudTalaPunktis.m.

Function C.5 (LaheBeamDFI(baasi0,Ntoerkts,esQkoormus,esFjoud, sSolmF,tsolm,tSiire,krdn,selem)) [71] *61* – *is used to assemble and solve the boundary problem equations of a beam.*
Called functions:
VardaPikkusT(NSARV,NEARV,krdn,selem) [72],
yspTlvfmhvI(baasi0,x,l,GAr,EJ) [73],
yspTlfhlin(baasi0,x,GAr,EJ) [74],

---

[58] ./octavePrograms/SpTeisendusUhikMaatriks0x1v.m
[59] ./octavePrograms/SpTeisendusUhikMaatriks2x2.m
[60] ./octavePrograms/SpTeisendusUhikMaatriks.m
[61] ./octavePrograms/VardadSolmesD.m
[62] ./octavePrograms/VardaPikkusD.m
[63] ./octavePrograms/ylSfhlin.m
[64] ./octavePrograms/yspSlfhlin.m
[65] ./octavePrograms/yspSlvfmhvI.m
[66] ./octavePrograms/yzSfzv.m
[67] ./octavePrograms/yzShqz.m
[68] ./octavePrograms/Sisejoud3LraamiPnktism.m
[69] ./octavePrograms/ESTFrKrmus.m
[70] ./octavePrograms/spESTbeam32LaheWFI.m
[71] ./octavePrograms/LaheBeamDFI.m
[72] ./octavePrograms/VardaPikkusT.m
[73] ./octavePrograms/yspTlvfmhvI.m
[74] ./octavePrograms/yspTlfhlin.m

ESTtalaKrmus(baasi0,xx,Li,Fjoud,qkoormus,EI) [75],
yzTfzv(baasi0,x,a,Fz,EJ) [76],
yzThqz(baasi0,x,qz,EJ) [77],
VardadSolmesT(NSARV,NEARV,Solm,AB,ABB) [78],
SpTeisendusUhikMaatriks1x0(VarrasN) [79],
SpTeisendusUhikMaatriks2x2(VarrasN) [80],
SpToeSiirdeWvektorT(VarrasN) [81],
SpToeSiirdeFiVektorT(VarrasN) [82],
ylTfhlin(baasi0,x,GAr,EJ) [83],
spInsertBtoA(spA,IIv,IJv,spvF) [84],
spSisestaArv(spA,iv,jv,sv) [85],
InsertBtoA(A,I,J,IM,JN,B,M,N) [86].

Function C.6 (SisejoudTalaPunktis(VardaNr,X,AlgPar,lvarras,selem, esFjoud,esQkoormus,suurused)) [87] *61* – *is used to compute the displacements and internal forces of the element 'VardaNr' at $x = X$.*
Called function:
ESTtalaKrmus.m

.

Function C.7 (ESTtalaKrmus(baasi0,xx,Li,Fjoud,qkoormus,EI)) [88] *62* – *is used to compute the loading vector $(q + F)$ for a continuous beam.*
Called functions:
yzThqz(baasi0,x,qz,EJ) [89],
yzTfzv(baasi0,x,a,Fz,EJ) [90].

Program C.28 (spESTGerberBeam1QM.m) [91] *28* – *is used to compute the internal forces of a Gerber beam.*
Called functions:

75 `./octavePrograms/ESTtalaKrmus.m`
76 `./octavePrograms/yzTfzv.m`
77 `./octavePrograms/yzThqz.m`
78 `./octavePrograms/VardadSolmesT.m`
79 `./octavePrograms/SpTeisendusUhikMaatriks1x0.m`
80 `./octavePrograms/SpTeisendusUhikMaatriks2x2.m`
81 `./octavePrograms/SpToeSiirdeWvektorT.m`
82 `./octavePrograms/SpToeSiirdeFiVektorT.m`
83 `./octavePrograms/ylTfhlin.m`
84 `./octavePrograms/spInsertBtoA.m`
85 `./octavePrograms/spSisestaArv.m`
86 `./octavePrograms/InsertBtoA.m`
87 `./octavePrograms/SisejoudTalaPunktis.m`
88 `./octavePrograms/ESTtalaKrmus.m`
89 `./octavePrograms/yzThqz.m`
90 `./octavePrograms/yzTfzv.m`
91 `./octavePrograms/spESTGerberBeam1QM.m`

LaheGerberBeamQM.m
SsjoudGrbrTalaPnktis.m.

Program C.29 (spESTGerberBeam2QM.m)[92] *28* *– is used to compute the internal forces of a Gerber beam.*
Called functions:
LaheGerberBeamQM.m
SsjoudGrbrTalaPnktis.m.

Program C.30 (spESTGerberBeam3QM.m)[93] *28* *– is used to compute the internal forces of a Gerber beam.*
Called functions:
LaheGerberBeamQM.m
SsjoudGrbrTalaPnktis.m.

Program C.31 (spESTGerberBeam4QM.m)[94] *28* *– is used to compute the internal forces of a Gerber beam.*
Called functions:
LaheGerberBeamQM.m
SsjoudGrbrTalaPnktis.m.

Program C.32 (spESTGerberBeam5QM.m)[95] *28* *– is used to compute the internal forces of a Gerber beam.*
Called functions:
LaheGerberBeamQM.m
SsjoudGrbrTalaPnktis.m.

Program C.33 (spESTGerberBeam6QM.m)[96] *28* *– is used to compute the internal forces of a Gerber beam.*
Called functions:
LaheGerberBeamQM.m
SsjoudGrbrTalaPnktis.m.

Program C.34 (spESTGerberBeam7QM.m)[97] *28* *– is used to compute the internal forces of a Gerber beam.*
Called functions:
LaheGerberBeamQM.m
SsjoudGrbrTalaPnktis.m.

[92] `./octavePrograms/spESTGerberBeam2QM.m`
[93] `./octavePrograms/spESTGerberBeam3QM.m`
[94] `./octavePrograms/spESTGerberBeam4QM.m`
[95] `./octavePrograms/spESTGerberBeam5QM.m`
[96] `./octavePrograms/spESTGerberBeam6QM.m`
[97] `./octavePrograms/spESTGerberBeam7QM.m`

Program C.35 (spESTGerberBeam8QM.m)[98] *28 – is used to compute the internal forces of a Gerber beam.*
Called functions:
LaheGerberBeamQM.m
SsjoudGrbrTalaPnktis.m.

Program C.36 (spESTGerberBeam9QM.m)[99] *28 – is used to compute the internal forces of a Gerber beam.*
Called functions:
LaheGerberBeamQM.m
SsjoudGrbrTalaPnktis.m.

Program C.37 (spESTGerberBeam10QM.m)[100] *28 – is used to compute the internal forces of a Gerber beam.*
Called functions:
LaheGerberBeamQM.m
SsjoudGrbrTalaPnktis.m.

Function C.8 (LaheGerberBeamQM(Ntoerkts,esQkoormus,esFjoud, sSolmF,tsolm,krdn,selem)) [101] *63, 63 – is used to assemble and solve the boundary problem equations of a statically determinate beam.*
Called functions:
InsertBtoA(A,I,J,IM,JN,B,M,N) [102],
spInsertBtoA(spA,IIv,IJv,spvF) [103],
spSisestaArv(spA,iv,jv,sv) [104],
SpTeisendusUhikMaatriks1x0(VarrasN) [105],
SpTeisendusUhikMaatriks2x2(VarrasN) [106],
VardadSolmesDT(NSARV,NEARV,Solm,AB,ABB [107],
VardaPikkusDT(NSARV,NEARV,krdn,selem) [108],
ylSTfhlin(x) [109],
yspSTlfhlin(x) [110],
yspSTlvfmhvI(x) [111],

[98] ./octavePrograms/spESTGerberBeam8QM.m
[99] ./octavePrograms/spESTGerberBeam9QM.m
[100] ./octavePrograms/spESTGerberBeam10QM.m
[101] ./octavePrograms/LaheGerberBeamQM.m
[102] ./octavePrograms/InsertBtoA.m
[103] ./octavePrograms/spInsertBtoA.m
[104] ./octavePrograms/spSisestaArv.m
[105] ./octavePrograms/SpTeisendusUhikMaatriks1x0.m
[106] ./octavePrograms/SpTeisendusUhikMaatriks2x2.m
[107] ./octavePrograms/VardadSolmesDT.m
[108] ./octavePrograms/VardaPikkusDT.m
[109] ./octavePrograms/ylSTfhlin.m
[110] ./octavePrograms/yspSTlfhlin.m
[111] ./octavePrograms/yspSTlvfmhvI.m

yzSTfzv(x,a,Fz) [112],
yzSThqz(x,qz) [113].

Function C.9 (SsjoudGrbrTalaPnktis(VardaNr,X,AlgPar,lvarras, esFjoud,esQkoormus,suurused)) [114] *63, 63* – *is used to compute the displacements and internal forces of the element 'VardaNr' at $x = X$.*
Called function:
ESTSTKrmus.m

.

Function C.10 (ESTSTKrmus(xx,Li,Fjoud,qkoormus))[115] *65* – *is used to compute the loading vector $(q + F)$ for a Gerber beam.*
Called functions:
yzSThqz(x,qz) [116],
yzSTfzv(x,a,Fz) [117].

Program C.38 (spESTtruss1N15WFI.m)[118] *32* – *is used to compute the displacements and internal forces of a plane truss.*
Called functions:
LaheTrussDFI.m

Program C.39 (spESTtruss2N15WFI.m)[119] *32* – *is used to compute the displacements and internal forces of a plane truss.*
Called function:
LaheTrussDFI.m

Program C.40 (spESTtruss3N15WFI.m)[120] *32* – *is used to compute the displacements and internal forces of a plane truss.*
Called function:
LaheTrussDFI.m

Program C.41 (spESTtruss4N15WFI.m)[121] *32* – *is used to compute the displacements and internal forces of a plane truss.*
Called function:
LaheTrussDFI.m

[112] ./octavePrograms/yzSTfzv.m
[113] ./octavePrograms/yzSThqz.m
[114] ./octavePrograms/SsjoudGrbrTalaPnktis.m
[115] ./octavePrograms/ESTSTKrmus.m
[116] ./octavePrograms/yzSThqz.m
[117] ./octavePrograms/yzSTfzv.m
[118] ./octavePrograms/spESTtruss1N15WFI.m
[119] ./octavePrograms/spESTtruss2N15WFI.m
[120] ./octavePrograms/spESTtruss3N15WFI.m
[121] ./octavePrograms/spESTtruss4N15WFI.m

Program C.42 (spESTtruss5N15WFI.m)[122] *32* – *is used to compute the displacements and internal forces of a plane truss.*
Called function:
LaheTrussDFI.m

Program C.43 (spESTtruss6N15WFI.m)[123] *32* – *is used to compute the displacements and internal forces of a plane truss.*
Called function:
LaheTrussDFI.m

Program C.44 (spESTtruss7N15WFI.m)[124] *32* – *is used to compute the displacements and internal forces of a plane truss.*
Called function:
LaheTrussDFI.m

Program C.45 (spESTtruss8N15WFI.m)[125] *32* – *is used to compute the displacements and internal forces of a plane truss.*
Called function:
LaheTrussDFI.m

Program C.46 (spESTtruss9N15WFI.m)[126] *32* – *is used to compute the displacements and internal forces of a plane truss.*
Called function:
LaheTrussDFI.m

Program C.47 (spESTtruss10N15WFI.m)[127] *32* – *is used to compute the displacements and internal forces of a plane truss.*
Called function:
LaheTrussDFI.m

Function C.11
(LaheTrussDFI(baasi0,Ntoerkts,sSolmF,tsolm, tSiire,krdn,selem)) [128] *65*, *65*
– *is used to assemble and solve the boundary problem equations for a truss.*
Called functions:
VardaPikkusTr(NSARV,NEARV,krdn,selem) [129],
yspSRmhvI(baasi0,x,EA) [130],

[122] ./octavePrograms/spESTtruss5N15WFI.m
[123] ./octavePrograms/spESTtruss6N15WFI.m
[124] ./octavePrograms/spESTtruss7N15WFI.m
[125] ./octavePrograms/spESTtruss8N15WFI.m
[126] ./octavePrograms/spESTtruss9N15WFI.m
[127] ./octavePrograms/spESTtruss10N15WFI.m
[128] ./octavePrograms/LaheTrussDFI.m
[129] ./octavePrograms/VardaPikkusTr.m
[130] ./octavePrograms/yspSRmhvI.m

yspSRhlin(baasi0,x,EA) [131],
VardadSolmesTr(NSARV,NEARV,Solm,AB,ABB) [132],
SpTeisendusMaatriksTr2x2(NSARV,NEARV,VarrasN,krdn,selem) [133],
SpTeisendusMaatriksTr2x1(NSARV,NEARV,VarrasN,krdn,selem) [134],
SpTeisendusUhikMaatriks0x1v(VarrasN) [135],
SpTeisendusUhikMaatriks2x2(VarrasN) [136],
SpToeSiirdeUvektorTr(NSARV,NEARV,VarrasN,krdn,selem) [137],
SpToeSiirdeWvektorTr(NSARV,NEARV,VarrasN,krdn,selem) [138],
spInsertBtoAvect(spA,IM,JN,spB) [139],
spInsertBtoA(spA,IIv,IJv,spvF) [140],
spSisestaArv(spA,iv,jv,sv) [141],
InsertBtoA(A,I,J,IM,JN,B,M,N) [142].

Program C.48 (spESTtruss1N27.m)[143] *36* – *is used to compute the internal forces of a plane truss.*
Called functions:
spSisestaArv.m
spInsertBtoA.m.

Program C.49 (spESTtruss2N27.m)[144] *36* – *is used to compute the internal forces of a plane truss.*
Called functions:
spSisestaArv.m
spInsertBtoA.m.

Program C.50 (spESTtruss3N27.m)[145] *36* – *is used to compute the internal forces of a plane truss.*
Called functions:
spSisestaArv.m
spInsertBtoA.m.

---

131 ./octavePrograms/yspSRhlin.m
132 ./octavePrograms/VardadSolmesTr.m
133 ./octavePrograms/SpTeisendusMaatriksTr2x2.m
134 ./octavePrograms/SpTeisendusMaatriksTr2x1.m
135 ./octavePrograms/SpTeisendusUhikMaatriks0x1v.m
136 ./octavePrograms/SpTeisendusUhikMaatriks2x2.m
137 ./octavePrograms/SpToeSiirdeUvektorTr.m
138 ./octavePrograms/SpToeSiirdeWvektorTr.m
139 ./octavePrograms/spInsertBtoAvect.m
140 ./octavePrograms/spInsertBtoA.m
141 ./octavePrograms/spSisestaArv.m
142 ./octavePrograms/InsertBtoA.m
143 ./octavePrograms/spESTtruss1N27.m
144 ./octavePrograms/spESTtruss2N27.m
145 ./octavePrograms/spESTtruss3N27.m

Program C.51 (spESTtruss4N27.m)[146] *36 – is used to compute the internal forces of a plane truss.*
Called functions:
spSisestaArv.m
spInsertBtoA.m.

Program C.52 (spESTtruss5N27.m)[147] *36 – is used to compute the internal forces of a plane truss.*
Called functions:
spSisestaArv.m
spInsertBtoA.m.

Program C.53 (spESTtruss6N27.m)[148] *36 – is used to compute the internal forces of a plane truss.*
Called functions:
spSisestaArv.m
spInsertBtoA.m.

Program C.54 (spESTtruss7N27.m)[149] *36 – is used to compute the internal forces of a plane truss.*
Called functions:
spSisestaArv.m
spInsertBtoA.m.

Program C.55 (spESTtruss8N27.m)[150] *36 – is used to compute the internal forces of a plane truss.*
Called functions:
spSisestaArv.m
spInsertBtoA.m.

Program C.56 (spESTtruss9N27.m)[151] *36 – is used to compute the internal forces of a plane truss.*
Called functions:
spSisestaArv.m
spInsertBtoA.m.

Program C.57 (spESTtruss10N27.m)[152] *36 – is used to compute the internal forces of a plane truss.*

[146] ./octavePrograms/spESTtruss4N27.m
[147] ./octavePrograms/spESTtruss5N27.m
[148] ./octavePrograms/spESTtruss6N27.m
[149] ./octavePrograms/spESTtruss7N27.m
[150] ./octavePrograms/spESTtruss8N27.m
[151] ./octavePrograms/spESTtruss9N27.m
[152] ./octavePrograms/spESTtruss10N27.m

Called functions:

spSisestaArv.m

spInsertBtoA.m.

Program C.58 (spSisestaArv(spA,iv,jv,sv))[153] *68* – *inserts the number sv into sparse matrix spA, starting at row index iv and column index jv.*

Program C.59 (spInsertBtoA(spA,IM,JN,spB))[154] *68* – *inserts sparse matrix spB into sparse matrix spA, starting at row index IM and column index JN. Overlapping elements of matrices spA and spB were added together.*

Program C.60 (InsertBtoA(A,I,J,IM,JN,B,M,N))[155] *57* – *inserts matrix B (dimensions M, N) into matrix A (dimensions I, J) starting at row index IM and column index JN.*

[153] `./octavePrograms/spSisestaArv.m`
[154] `./octavePrograms/spInsertBtoA.m`
[155] `./octavePrograms/InsertBtoA.m`

Bibliography

[BP13] R. M. Barker, J. A. Puckett. *Design of Highway Bridges:* An LRFD Approach, 3. edition.[1] Hoboken, New Jersey: John Wiley & Sons, 2013. Web. 08 August 2013. 51

[GN12] M. K. Gíslason, D. H. Nash. Finite Element Modelling of a Multi-Bone Joint: The Human Wrist.[2] In: Dr. Farzad Ebrahimi (ed.), *Finite Element Analysis – New Trends and Developments.* InTech, 2012. DOI: 10.5772/50560. Web. 08 August 2013. 52

[KHMW10] W. B. Krätzig, R. Harte, K. Meskouris und U. Wittek. *Tragwerke 1. Theorie und Berechnungsmethoden statisch bestimmter Stabtragwerke.* 5., überarbeitete Auflage.[3] Springer-Verlag, 2010. Web. 08 August 2013. 51

[KHMW05] W. B. Krätzig, R. Harte, K. Meskouris und U. Wittek. *Tragwerke 2. Theorie und Berechnungsmethoden statisch unbestimmter Stabtragwerke.* 4., überarbeitete und erweiterte Auflage.[4] Springer-Verlag, 2004, 1994, 1998, 2005. Web. 08 August 2013.

[1] 9.4.1 Betti's Theorem – http://books.google.ee/books?id=ytKw4IZD_ZMC&pg=PA137&dq=betti%27s+theorem+forces+of+first+load+state+1+to+state+2&hl=en&sa=X&ei=j3wDUsSKAsrQsgaUp4GIDA&ved=0CDgQ6AEwAg#v=onepage&q=betti%27s%20theorem%20forces%20of%20first%20load%20state%201%20to%20state%202&f=false

[2] http://www.intechopen.com/books/finite-element-analysis-new-trends-and-...developments/finite-element-modelling-of-a-multi-bone-joint-the-human-wrist

[3] 2.4.1 Zustandsgrössen – http://books.google.ee/books?id=itFS-x8fv7oC&pg=PA53&dq=Zustandsgr%C3%B6ssen+Kr%C3%A4tzig&hl=en&sa=X&ei=LSsDUsjfAZH2sgatiIHgCQ&ved=0CCsQ6AEwAA#v=onepage&q=Zustandsgr%C3%B6ssen%20Kr%C3%A4tzig&f=false
2.4.4 Formänderungsarbeits-Funktionale – http://books.google.ee/books?id=itFS-x8fv7oC&pg=PA58&dq=Form%C3%A4nderungsarbeits-Funktionale+Kr%C3%A4tzig&hl=en&sa=X\string&ei=wVsDUoWDGsrDswa204HAAw&ved=0CCsQ6AEwAA#v=onepage&q=Form%C3%A4nderungsarbeits-Funktionale%20Kr%C3%A4tzig&f=false

[4] http://books.google.ee/books?id=6tFZQT71Ff8C&pg=PA289&dq=Deformationsspr%C3%BCnge+Kr%C3%A4tzig&hl=en&sa=X&ei=TEEDUtuHKZHdsgbbv4D4BA&ved=0CC0Q6AEwAA#v=onepage&q=Deformationsspr%C3%BCnge%20Kr%C3%A4tzig&f=false

[Lah97a] A. Lahe. The transfer matrix and the boundary element method.[5] *Proc. Estonian Acad. Sci. Engng.*, 1997, 3, 1, pp. 3–12. Web. 08 August 2013.

[Lah97b] A. Lahe. The EST method for the frame analysis. In: *Proc. Tenth Nordic Seminar on Computational Mechanics,* October 24–25, 1997. Tallinn: Tallinn Technical University, 1997, pp. 202–205.

[Lah98a] A. Lahe. The EST method for the frame analysis in second order theory, In: *Proc. of the NSCM–11: Nordic Seminar on Computational Mechanics,* October 16–17, 1997. Stockholm: Royal Institute of Technology, Department of Structural Engineering, TRITA-BKN. Bulletin 39, 1998, pp.167–170.

[Lah12] A. Lahe. *Ehitusmehaanika.*[6] Tallinn: Tallinna Tehnikaülikooli Kirjastus, 2012. 764 lk. Web. 08 August 2013.

[LT80] M. Lawo, O. Tierauf. *Matrizenmethoden der Statik und Dynamik, I: Statik.* Braunschweig-Wiesbaden: Fried. Vieweg & Sohn, 1980.

[PL63] E. C. Pestel, F. A. Leckie. *Matrix Method in Elastomechanics.* New York, San Francisco, Toronto, London: McGraw-Hill, 1963.

[PW94] W. D. Pilkey, W. Wunderlich. *Mechanics of Structures. Variational and Computational Methods.* Boca Raton, Ann Arbor, London, Tokyo: CRC Press, 1994.

[Rand07] K.D. Randall. *Physics for Scientists and Engineers. A Strategic Approach.* 2nd Edition.[7] Pearson Addison-Wesley, 2007. Web. 08 August 2013. 52

[REL83] *Р. Ряэмет, Р. Еек, А. Лахе. Строительная механика. Программа и задачи.* Таллинн: ТПИ, Кафедра строительной механики, 1983, 70 стр. 32

[Str89] M. Stern. Static analysis of beams, plates and shells. In: D. E. Beskos (ed.), *Boundary Element Methods in Structural Analysis.* New York: Am. Soc. Civil Engineers, 1989, pp. 41–64.

[VrCty] Method of Sections. *Truss Structures.*[8] VirtualCity.[9] Web. 09 August 2013. 53

[5]http://books.google.ee/books?id=ghco7svk5T4C&pg=PA3&lpg=PA3&dq=Andres+Lahe&source=bl&ots=3SFfo4UCES&sig=_XLUez-SfW2FVYGRx8v2LVm16V8&hl=et&ei=YQaFTMeIEoWcOOyCyNwP&sa=X&oi=book_result&ct=result&resnum=5&ved=0CB0Q6AEwBDgK#v=onepage&q=Andres%20Lahe&f=false

[6]http://digi.lib.ttu.ee/opik_eme/Ehitusmehaanika.pdf

[7]http://www.goodreads.com/author/show/10633.Randall_D_Knight

[8]http://vcity.ou.edu/demoModules/analysis/truss/truss.htm

[9]http://vcity.ou.edu/

[WP960] Beam on 2 Supports: Shear Force & Bending Moment Diagrams. Technical Description.[10] GUNT Hamburg.[11] Web. 08 August 2013. 52, 53

[Yaw09] L. L. Yaw. *2D Corotational Beam Formulation.*[12] Walla Walla University, November 30, 2009. Web. 27 August 2013.

[10] `http://www.gunt.de/static/s3117_1.php?p1=&p2=&pN=;;`
[11] `http://www.gunt.de/static/s1_1.php?p1=&p2=&pN=`
[12] http://people.wallawalla.edu/~louie.yaw/Co-rotational_docs/2Dcorot_beam.pdf

Index