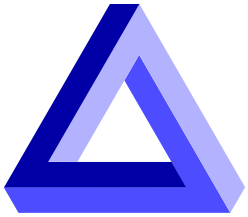
Lecture Notes

Finite Element Method: Foundations

Prof. Dr.-Ing. Jörg Schröder
Dr.-Ing. Daniel Balzani
University Duisburg-Essen
Faculty of Engineering Sciences
Department of Civil Engineering
Institute of Mechanics



Summer term 2008
(Translation of the script of 2003)

Contents

1 Introduction **1**

2 Classification of partial differential equations of second order **2**

3 Finite-Difference-Method **5**

3.1 One-dimensional Problem 5

3.1.1 Discretization 1D 5

3.1.2 Differential quotients 1D 6

3.1.3 Example: Beam with constant cross section. 8

3.1.4 Parabolic problems: equation of heat conduction 12

3.2 Two-dimensional problem 15

3.2.1 Discretization 2D 15

3.2.2 Difference quotients 2D 16

4 Foundations of the Calculus of Variations **21**

4.1 Preliminaries 21

4.2 Classical Method, Euler-Lagrange Equation 23

4.2.1 Boundary Conditions 27

4.2.2 Constraint Conditions 29

4.3 Direct Methods 31

4.3.1 Ritz's Method 31

4.3.2 Weighted Residual Methods 33

5 Introduction to Matlab **39**

5.1 Introduction 39

5.2 Basics 40

5.2.1 Graphical User Interface of MATLAB 40

5.2.2 Additional Environments 41

5.2.3 General Commands 42

5.2.4 Variables 42

5.2.5 Mathematical Functions 43

5.2.6 Vectors and Matrices 43

5.2.7 Operations on Vectors and Matrices 44

5.2.8 Scripts and Functions 45

5.2.9 Relational and Logical Operators 46

5.2.10 IF-/CASE Statements and Loops 46
5.2.11 Plotting Figures 46
5.3 Exercises 48
5.3.1 Animation of a rotating cantilever 48
5.3.2 Animation of a rotating cantilever using a subroutine 48
5.3.3 Source code 49

6 Linear Finite Element Method 51
6.1 Illustration of the Finite Element Concept 51
6.1.1 Idealization of the Physical System 52
6.1.2 The Assembly Procedure 54
6.1.3 Remarks Concerning the Implementation 57
6.2 Generalizations 71
6.3 Triangular and Tetrahedral Elements 73
6.3.1 Triangular Elements 73
6.3.2 Tetrahedral Elements 80
6.4 Isoparametric Concept 82
6.5 Numerical Integration 84
6.6 Quadrilateral Element 90
6.6.1 Variational Formulation 90
6.6.2 Finite-Element-Discretization 92
6.7 Beam Elements 96
6.7.1 Beam Theory according to Euler-Bernoulli for simple bending . . . 96
6.8 Plate Elements 110
6.8.1 Kinematics 110
6.8.2 Kirchhoff Theory 111
6.8.3 16 Parameter Element by Kirchhoff- Theory 112
6.8.4 Reissner-Mindlin Theory 114
6.9 Isoparametric Four-Node-Element of the Shear-Elastic Theory 115
6.10 Axis-Symmetric Shell Elements 116
6.10.1 Two-Node Axis-Symmetric Shell Element 120

1 Introduction

Due to the rapid developments in the field of computer technology and the methods of calculation, respectively, the application of numerical simulations becomes more and more significant in all areas of engineering and research. Generally, typical mechanical problems are examined as follows:

1. An appropriate mathematical model of the problem is derived. Usually, this results in a continuous problem, which is for example formulated in the framework of continuum mechanics. The problem is then described by a system of (partial) differential equations.

2. The continuous system is approximated by an appropriate discrete problem. There the field variables, which have to be computed, are approximated by a finite number of values. This process is referred to as disretization; in this context we distinguish between disretization of the physical space and disretization of the system of differential equations.

3. As a result of the discretization process we obtain algebraic systems of equations, which are solved by direct or iterative algorithms.

4. After the solution of the discrete problem often several millions of computed quantities are available, which have to be visualized for interpretation.

Well-known discretization methods are:

- Finite-Difference-Method
- Finite-Element-Method
- Finite-Volume-Method
- Boundary-Element-Method.

In the lecture “Finite Element Method: Foundations” we are dealing with the Finite-Difference-Method and the Finite-Element-Method for linear problems. In the field of solid mechanics the Finite-Element-Method is of particular significance due to its flexibility enabling the handling of complex problems for the numerical solution of elliptic and parabolic problems. Compared to the Finite-Difference-Method and the Finite-Volume-Method the Finite-Element-Method is specific to the variational formulation of differential equations.

2 Classification of partial differential equations of second order

In this chapter some foundations and definitions are repeated in the context of partial differential equations with view to mechanical engineering problems. In the sequel we classify partial differential equations of second order and present prototypes of the particular classes.

The linear partial differential equation in the two variables $\boldsymbol{x} := (x_1, x_2)^T$ and the unknown function $u(\boldsymbol{x})$ is given by

$$A(\boldsymbol{x})u_{,11} + 2B(\boldsymbol{x})u_{,12} + C(\boldsymbol{x})u_{,22} + \ldots = F(\boldsymbol{x}) \tag{1}$$

for all $\boldsymbol{x}$ in the considered domain $\mathcal{B}$. Herein, we have used the abbreviations

$$u_{,1} := \frac{\partial u}{\partial x_1}\,; \quad u_{,11} := \frac{\partial^2 u}{\partial x_1 \partial x_1}\,; \quad u_{,12} := \frac{\partial^2 u}{\partial x_1 \partial x_2}\,; \qquad \ldots \tag{2}$$

The type of partial differential equation depends on the sign of the discriminant

$$\delta := AC - B^2 \tag{3}$$

in the considered domain. We distinguish between the following four cases:

$$\left.\begin{array}{l} 1.\ \delta < 0 : \text{hyperbolic DE} \\ 2.\ \delta = 0 : \text{parabolic DE} \\ 3.\ \delta > 0 : \text{elliptic DE} \\ 4.\ \delta \text{ modifies its sign: mixed type} \end{array}\right\}\,. \tag{4}$$

The sign of the discriminant is invariant to any transformations of the independent variables. Hence, the type of the differential equation is an invariant with respect to the independent variables.

In case of the classification of partial differential equations with more than two independent variables we consider the description

$$\sum_{i,k} a_{ik} \frac{\partial^2 u(\boldsymbol{x})}{\partial x_i \partial x_k} + \ldots = 0 \tag{5}$$

with the symmetric real matrix $\boldsymbol{A} = (a_{ik})$. If the coefficients a_{ik} are no functions of the $\boldsymbol{x}$, we refer to the differential equation as linear differential equation of second order with constant coefficients. By the use of a linear transformation, Equation (5) can be reformulated by

$$\sum_{i} \kappa_i \frac{\partial^2 u(\boldsymbol{x})}{\partial x_i^2} + \ldots = 0\,, \tag{6}$$

in such a way that all coefficients κ_i take the values $+1, -1$ or 0.

Depending on the signs we consider the following cases:

$$\left.\begin{array}{ll} 1. & \text{The DE is hyperbolic, if all coefficients } \kappa_i \text{ differ from zero} \\ & \text{and one } \kappa_i \text{ has a converse sign with respect to the others.} \\ 2. & \text{The DE is parabolic, if one of the coefficients } \kappa_i \text{ vanishes} \\ & \text{and the others differ from zero and have the same sign.} \\ 3. & \text{The DE is elliptic, if all coefficients differ from zero} \\ & \text{and have the same sign.} \\ 4. & \text{An equation is called parabolic, hyperbolic or elliptic, if it} \\ & \text{exhibits the corresponding characteristics for all points of the domain.} \end{array}\right\} \tag{7}$$

Prototype of a hyperbolic differential equation: wave equation

The motion of a string $\mathcal{B}$ with $\mathcal{B} \in [0, l]$, with given (zero) displacements at both ends, is described by

$$\left.\begin{array}{lll} \text{PDE :} & \frac{1}{c^2} u_{,tt} - u_{,11} = 0 & x \in \mathcal{B}\,, t \geq 0 \\ \text{Boundary conditions:} & u(0,t) = u(l,t) = 0 & t \geq 0 \\ \text{Initial state:} & u(x,0) = u_0(x) & x \in \mathcal{B} \\ \text{Initial velocity} & u_{,t}\,(x,0) = v_0(x) & x \in \mathcal{B} \end{array}\right\} \tag{8}$$

Herein, the parameter $c > 0$ characterizes the velocity of wave propagation. We consider the tranformed partial differential equation

$$c^2 u_{,11} - u_{,tt} = 0 \tag{9}$$

and identify the values of the coefficients $A = c^2$, $B = 0$ and $C = -1$. The discriminant

$$\delta = -c^2 < 0\,, \tag{10}$$

shows that the wave equation is hyperbolic.

Prototype of a parabolic differential equation: equation of heat conduction

Parabolic problems describe equalizing processes like diffusion or heat conduction. With proceeding time these processes exhibit a smoothing effect. Let $\vartheta(\boldsymbol{x}, t)$ be the distribution of temperature in points $\boldsymbol{x} = (x_1, x_2)^T$ of the body $\mathcal{B} \subset \mathbb{R}^2$ with the boundary $\partial\mathcal{B}$ at a time t. The heat flux $\boldsymbol{h}$ results from the Fourier law

$$\boldsymbol{h} = -\kappa\, \text{grad}\vartheta(\boldsymbol{x}, t) \qquad \text{bzw.} \qquad \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = -\kappa \begin{bmatrix} \vartheta_{,1} \\ \vartheta_{,2} \end{bmatrix}, \tag{11}$$

where κ is the heat conductivity constant. From the law of conservation of energy we obtain the classical equation of the heat flux (without thermomechanical coupling)

$$\rho c_p \vartheta_{,t} = \kappa \triangle\vartheta + \rho r \tag{12}$$

with a constant density ρ, a constant specific heat capacity c_p, the available heat source r and the Laplace operator $\triangle\vartheta = \sum_j \dfrac{\partial^2 \vartheta}{\partial x_j^2}$. In case of parabolic problems we deal with initial boundary value problems. The system is described by

$$\left.\begin{array}{lll} \text{PDE :} & \rho c_p \vartheta_{,t} = \kappa \triangle\vartheta + \rho r & \boldsymbol{x} \in \mathcal{B}\,, t > 0 \\ \text{Boundary conditions:} & \vartheta(\boldsymbol{x}, t) = \vartheta_0(\boldsymbol{x}) & \boldsymbol{x} \in \partial\mathcal{B}\,, t \geq 0 \\ \text{Initial temperature:} & \vartheta(\boldsymbol{x}, 0) = \vartheta_1(\boldsymbol{x}) & \boldsymbol{x} \in \mathcal{B}\,, t = 0 \end{array}\right\} \tag{13}$$

Let us consider the one-dimensional case of equation $(13)_1$ we obtain

$$\rho c_p \vartheta_{,t} = \kappa \vartheta_{,11} + \rho r\,. \tag{14}$$

With $A = \kappa$, $B = 0$ and $C = 0$ we obtain from Equation (3) the discriminant

$$\delta = 0\,, \tag{15}$$

i.e. the diffusion equation is parabolic.

Prototype of an elliptic differential equation: potential equation

A classical example of an elliptic differential equation is the potential equation. We consider a domain $\mathcal{B} \subset \mathbb{R}^2$, in which a function $u(\boldsymbol{x})$ is searched for satisfying the differential equation of 2nd Order

$$u_{,11} + u_{,22} = 0\,. \tag{16}$$

In order to solve such problems corresponding boundaries u have to be formulated, for example

$$u = 0 \qquad \text{auf} \qquad \partial\mathcal{B}\,. \tag{17}$$

Identifying the coefficients in equation (1), i.e. $A = C = 1$ and $B = 0$, the discriminant of equation (3) yields

$$\delta = 1 > 0\,, \tag{18}$$

and we find that Equation (16) is elliptic.

Remark: The potential equation $\triangle u = 0$ is a special case of the Poisson equation

$$\triangle u(\boldsymbol{x}) = -f(\boldsymbol{x})\,. \tag{19}$$

Summary: Classification in n variables in case of linear differential equations of second order

The general partial differential equation of 2nd order has the form

$$-\sum_{i,k=1}^{n} a_{ik}(\boldsymbol{x}) u_{,ik} + \sum_{i=1}^{n} b_i(\boldsymbol{x}) u_{,i} + c(\boldsymbol{x}) u = f(\boldsymbol{x})\,. \tag{20}$$

If $u(\boldsymbol{x})$ is two times consistently differentiable, we are able to conclude from $u_{,ik} = u_{,ki}$ the symmetry $a_{ik} = a_{ki}$. The corresponding coefficient matrix $\boldsymbol{A}$ is symmetric.

Following classifications are valid:

$$\left.\begin{array}{l}
\text{1. The DE in equation (20) is hyperbolic in a point } \boldsymbol{x}\text{, if } \boldsymbol{A}(\boldsymbol{x}) \\
\quad \text{has one negative eigenvalue and } n-1 \text{ positive eigenvalues.} \\
\text{2. The DE in equation (20) is parabolic in a point } \boldsymbol{x}\text{, if } \boldsymbol{A}(\boldsymbol{x}) \\
\quad \text{is positive semidefinite.} \\
\text{3. The DE in equation (20) is elliptic in a point } \boldsymbol{x}\text{, if } \boldsymbol{A}(\boldsymbol{x}) \\
\quad \text{is positive definite.} \\
\text{4. The DE has the property 1., 2. or 3., if it has the associated properties} \\
\quad \text{for all points of the area.}
\end{array}\right\} \tag{21}$$

In general, the partial differential equation can be transformed to the representation

$$\left.\begin{array}{rcll}
\ddot{\boldsymbol{u}} + \boldsymbol{L}\boldsymbol{u} & = & \boldsymbol{f} & \text{hyperbolic case} \\
\dot{\boldsymbol{u}} + \boldsymbol{L}\boldsymbol{u} & = & \boldsymbol{f} & \text{parabolic case} \\
\boldsymbol{L}\boldsymbol{u} & = & \boldsymbol{f} & \text{elliptic case}
\end{array}\right\}\,. \tag{22}$$

Herein, $\boldsymbol{L}$ is an elliptic differential operator, see Braess [1].

3 Finite-Difference-Method

In this chapter we consider the numerical solution of common and partial differential equations by the Finite-Difference-Method. There, we compute approximated values of the solution at grid points by replacing the derivatives in the differential equations with difference quotients. This leads to a discrete problem resulting in an algebraic system of equations of the form $\boldsymbol{A}\boldsymbol{x} = \mathbf{f}$. The solution $\boldsymbol{x}$ are the approximated values of the continuous problem at the grid points.

3.1 One-dimensional Problem

In this section we want to figure out the way of solving ordinary differential equations of 2nd order with the Finite-Difference-Method in one-dimensional problems. We are looking for a scalar-valued function

$$u : \mathcal{B} \cup \partial\mathcal{B} \to \mathbb{R} \tag{23}$$

in the domain $\mathcal{B} \subset \mathbb{R}$ with the boundary $\partial\mathcal{B}$, which satisfies for given

$$f : \mathcal{B} \to \mathbb{R} \text{ and } g : \partial\mathcal{B} \to \mathbb{R} \tag{24}$$

the differential equation in the domain $\mathcal{B}$ with the boundary conditions

$$\left.\begin{array}{rcll} u''(x) & = & -f(x) & \text{in } \mathcal{B} \\ u & = & g & \text{on } \partial\mathcal{B} \end{array}\right\} . \tag{25}$$

3.1.1 Discretization 1D At first we consider one-dimensional problems, in which the considered area $\mathcal{B} \subset \mathbb{R}$ is defined by

$$\mathcal{B} := [0, l] \qquad \text{with} \qquad l > 0 . \tag{26}$$

For the discretization we choose the constant increment $h > 0$ in order to satisfy

$$l = n \cdot h \qquad \text{with} \qquad n \in \mathbb{N} . \tag{27}$$

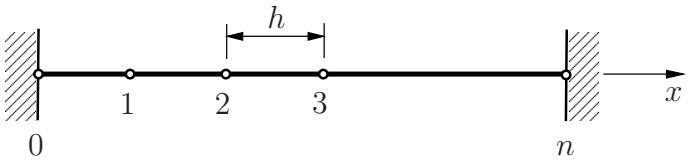


Figure 1: Spatial discretization of the one-dimensional problem with constant increment h.

The grid points of the area $\mathcal{B}$ are defined by

$$\mathcal{B}_h := \{ih \mid i = 1, \ldots, n-1\} . \tag{28}$$

The points of the boundary $\partial\mathcal{B}$ result from

$$\partial\mathcal{B}_h := \{ih \mid i = 0 \quad \text{and} \quad i = n\} . \tag{29}$$

3.1.2 Differential quotients 1D First of all we consider the Taylor series expansion of an one-dimensional function $u(x+h)$ in the point x

$$u(x+h) = u(x) + \frac{1}{1!}u'(x)h + \frac{1}{2!}u''(x)h^2 + \frac{1}{3!}u'''(x)h^3 + \dots . \tag{30}$$

Then the first derivative is computed from solving Equation (30) with respect to $u'(x)$

$$\left.\begin{aligned} u'(x) &= \frac{u(x+h)-u(x)}{h} - \frac{1}{2!}u''(x)h - \frac{1}{3!}u'''(x)h^2 - \dots \\ u'(x) &= \frac{u(x+h)-u(x)}{h} + O(h) \end{aligned}\right\} . \tag{31}$$

Equation $(31)_2$ is referred to as difference quotient.

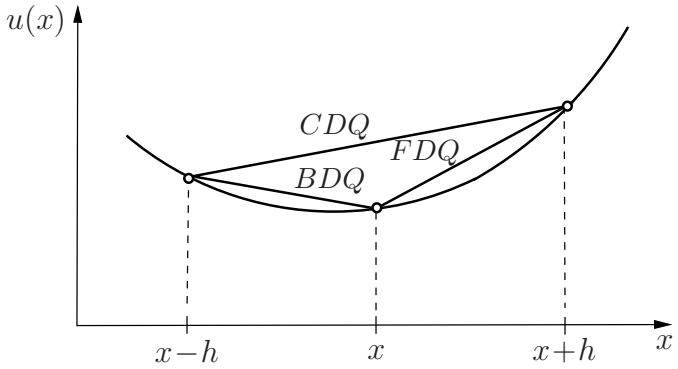


Figure 2: Visualization of the central difference quotient (CDQ), the backward difference quotient (BDQ) and the forward difference quotient (FDQ) of a function $u(x)$.

From the development of $u(x-h)$ at point x we receive analogously

$$u(x-h) = u(x) - \frac{1}{1!}u'(x)h + \frac{1}{2!}u''(x)h^2 - \frac{1}{3!}u'''(x)h^3 + \dots . \tag{32}$$

Then, the first derivative yields

$$\left.\begin{aligned} u'(x) &= \frac{u(x)-u(x-h)}{h} + \frac{1}{2!}u''(x)h - \frac{1}{3!}u'''(x)h^2 + \dots \\ u'(x) &= \frac{u(x)-u(x-h)}{h} + O(h) \end{aligned}\right\} , \tag{33}$$

Here, Equation $(33)_2$ is referred to as backward difference quotient.

By summing up Equations (31) and (33) yields the central difference quotient being the approximation of the first derivative of $u(x)$

$$\left.\begin{aligned} u'(x) &= \frac{u(x+h)-u(x-h)}{2h} - \frac{1}{3!}u'''(x)h^2 + \dots \\ u'(x) &= \frac{u(x+h)-u(x-h)}{2h} + O(h^2) \end{aligned}\right\} . \tag{34}$$

For the approximaion of the second derivative we sum the Equations (30) and (32)

$$u(x+h) + u(x-h) = 2u(x) + u''(x)h^2 + O(h^4) \tag{35}$$

and obtain directly

$$u''(x) = \frac{u(x+h) - 2u(x) + u(x-h)}{h^2} + O(h^2) . \tag{36}$$

Table 1: Differential quotients for an one-dimensional function

1.	forward DQ	$u'(x) = \dfrac{u(x+h) - u(x)}{h}$
2.	backward DQ	$u'(x) = \dfrac{u(x) - u(x-h)}{h}$
3.	central DQ	$u'(x) = \dfrac{u(x+h) - u(x-h)}{2h}$
4.	2nd derivative	$u''(x) = \dfrac{u(x+h) - 2u(x) + u(x-h)}{h^2}$

Example: We consider the function

$$u(x) = e^x \tag{37}$$

and compute the first derivative on the point $x = 0$ with the increment $h = 0.1$. The approximations yield

$$\left.\begin{array}{lcll} u'(x=0) & = & 1,05 & \text{forward dq} \\ u'(x=0) & = & 0,95 & \text{backward dq} \\ u'(x=0) & = & 1,0017 & \text{central dq} \\ u'(x=0) & = & 1,0 & \text{exact dq} \end{array}\right\} . \tag{38}$$

A graphical representation of the function (37) and its derivative is given in Fig. 3.

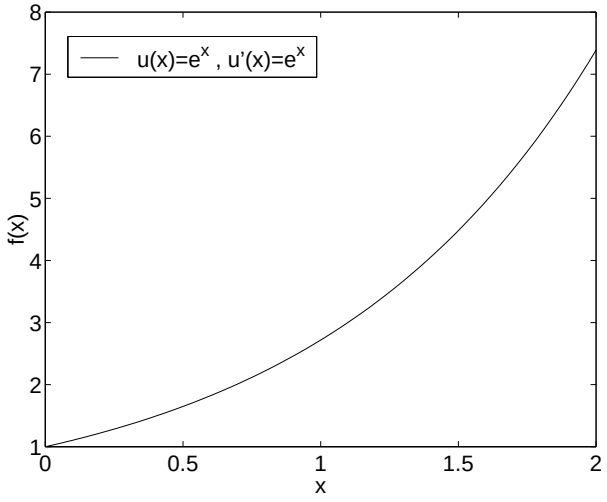


Figure 3: Graphical representation of the function $u(x) = e^x$ and its derivative.

3.1.3 Example: Beam with constant cross section. As a numerical example we consider a beam fixed at both sides, loaded by $p(x)$, wich has a constant cross-sectional area A and a constant Young's modulus E, see Fig. 4.

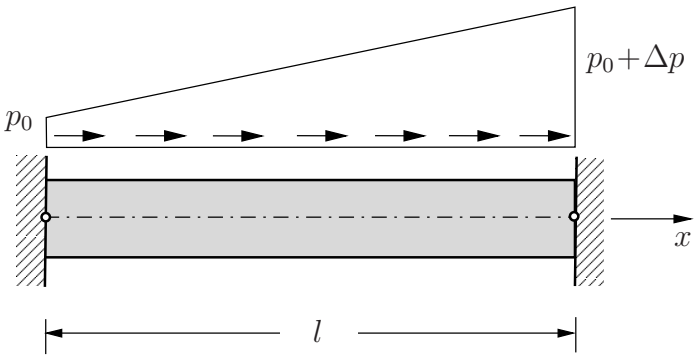


Figure 4: Model problem: beam with constant stiffness EA and distributed loading $p(x)$.

The model problem is described by an ordinary differential equation of 2nd order in the domain $\mathcal{B}$ and a given displacement on $\partial\mathcal{B}$

$$\left.\begin{array}{rcl} EAu''(x) & = & -p(x) \\ u(x=0) & = & 0 \\ u(x=l) & = & 0 \end{array}\right\} . \tag{39}$$

With $p(x) = p_0 + \Delta\, p \cdot x/l$ and with the integration of Equation $(39)_1$ we get the general solution

$$u(x) = -\frac{\Delta p}{6EAl}x^3 - \frac{p_0}{2EA}x^2 + c_1 x + c_2\,. \tag{40}$$

After incorporating the boundary conditions $(39)_{2,3}$ we obtain

$$u(x) = -\frac{\Delta p}{6EAl}x^3 - \frac{p_0}{2EA}x^2 + \left(\frac{p_0 l}{2EA} + \frac{\Delta p l}{6EA}\right) x\,. \tag{41}$$

In the framework of the Finite-Difference-Method a discretization of the problem is required. We consider a discretization with $n+1$ grid points of equidistant increments h. The this leads to the domain discretization $\mathcal{B}_h$ as shown in Fig. 5.

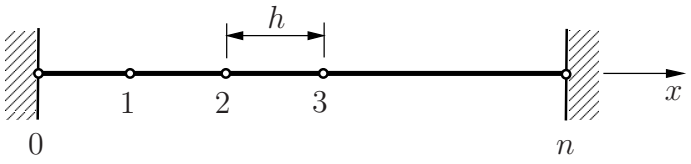


Figure 5: Spatial discretization of the beam.

We consider the case $p(x) = p_0$ and approximate the second derivative by using Table 1. Then we achieve the approximation of the field equations for the individual grid points by

$$EA\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} = -p_0 \qquad \text{for} \qquad i = 1, \ldots, n-1\,. \tag{42}$$

The discrete displacements on the boundary are

$$u_0 = 0 \qquad \text{und} \qquad u_n = 0\,. \tag{43}$$

We write Equation (42) for every grid point and obtain

$$\left.\begin{aligned} u_0 - 2u_1 + u_2 &= -\frac{p_0}{EA}h^2 \\ u_1 - 2u_2 + u_3 &= -\frac{p_0}{EA}h^2 \\ &\vdots \\ u_{n-2} - 2u_{n-1} + u_n &= -\frac{p_0}{EA}h^2 \end{aligned}\right\}\,. \tag{44}$$

Taking into account the boundary conditions of Equation (43) the linear system of equations can be reformulated in matrix notation by

$$\begin{bmatrix} 2 & -1 & 0 & 0 & 0 & \dots & 0 \\ -1 & 2 & -1 & 0 & 0 & \dots & 0 \\ 0 & -1 & 2 & -1 & 0 & \dots & 0 \\ 0 & 0 & -1 & 2 & -1 & \dots & 0 \\ \vdots & & & & & & \vdots \\ 0 & 0 & 0 & 0 & 0 & \dots & -1 \\ 0 & 0 & 0 & 0 & 0 & \dots & 2 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ \vdots \\ u_{n-2} \\ u_{n-1} \end{bmatrix} = \frac{h^2}{EA}p_0 \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{bmatrix} \tag{45}$$

or in absolute representation

$$\boldsymbol{A} \cdot \boldsymbol{u} = \mathbf{f}\,. \tag{46}$$

The solution of the problem with different discretizations is depicted in Fig. 6. Specific values of the system and the material are $l = 1$, $A = 1$, $p_0 = 1$, $E = 1$.

Exercise: Calculate the displacement $u(x)$ of the beam shown in Fig. 4 for the loading conditions $p_0 = 1$ and $\Delta p = 0$. For this purpose consider the discretizations $n = 2$, $n = 4$, $n = 11$ and compare the results with the exact analytical solution given in Equation (41).

Solution:

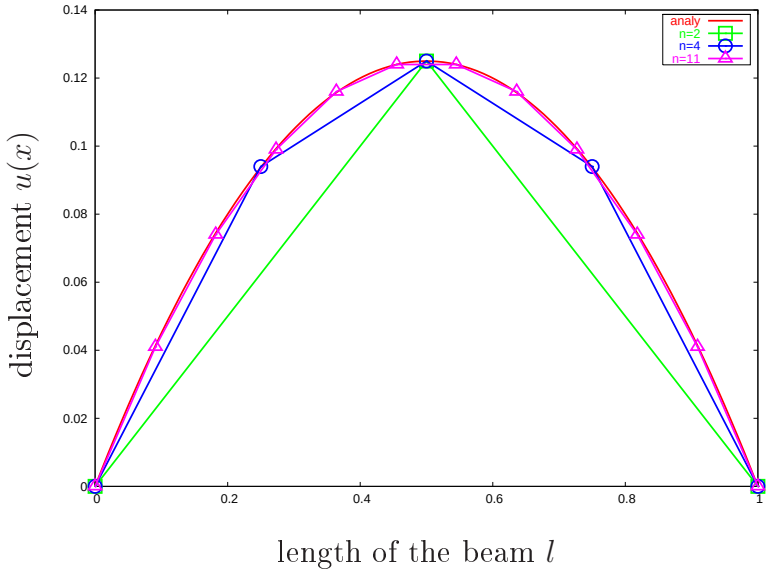


Figure 6: Displacement u(x) of the beam with constant cross section for the discretizations $n = 2$, $n = 4$, $n = 11$ and the analytical solution.

```
      program fdm1d1
c-----------------------------------------------------------------------72
c     Beispiel1: Beam with constant stiffness under constant load
c     Numerical solution with the Finite-Difference-Method
c
c     D. Balzani, April 14, 2008
c
c     Parameters:
c...  Input:
c       n,ll,yo,area,pp
c       n             : Number of grid points
c       ll            : beam length
c       yo            : E-Modulus
c       area          : cross-sectional area
c       pp            : constant load
c...  Output:
c       x             : physical location x
c       u             : displacement u(x)
c...  Local fields:
c       a(nmax,nmax) : coefficient matrix
c       b(nmax)       : right hand side vector
c-----------------------------------------------------------------------72
      implicit none
      integer nmax,i,j,n
      parameter (nmax=15)
      real*8 ll,yo,area,pp,hh,fakt
      real*8 a(nmax,nmax),b(nmax),x,u(nmax)
c
c...  Open input and output file
```

```
      open (1, file='beispein.dat')
      open (2, file='beispaus.dat')
c
c...  Read system values from file 'beispein.dat'
      read (1, *) n,ll,yo,area,pp
c
c...  Initialisize quantities
      do i=1,n-1
         b(i)=0.d0
 u(i)=0.d0
 do j=1,n-1
    a(i,j)=0.d0
 end do
      end do
c
c...  Compute coefficient matrix
      do i=1,n-2
            a(i,i) =  2.d0
  a(i,i+1) = -1.d0
  a(i+1,i) = -1.d0
      end do
      a(n-1,n-1) = 2.d0
c
c...  Compute right hand side vektor
      hh  = ll/dble(n)
      fakt = (hh*hh*pp)/(yo*area)
      do i=1,n-1
         b(i) = fakt
      end do
c
c...  Solve linear system of equations
      call gausspivot(a,b,u,nmax,n-1)
c
c...  Plot solution into file 'beispaus.dat'
      write (2,200) 0.d0, 0.d0
      do i=1,n-1
         x = dble(i)*hh
         write (2,200) x, u(i)
      end do
      write (2,200) ll, 0.d0
 200  format (f10.3,f10.3)
      end
```

3.1.4 Parabolic problems: equation of heat conduction In the previous sections an elliptic differential equation was considered. Using the equation of heat conduction as an example we now analyze the numerical solution of the parabolic differential equation

$$\vartheta_{,t}\,(x,t) = c^2 \vartheta_{,xx}\,(x,t) \quad \forall \quad x \in \mathcal{B} \quad \text{and} \quad t > 0 \tag{47}$$

in a one-dimensional domain

$$\mathcal{B} := (0,l) \quad \text{with} \quad l > 0\,. \tag{48}$$

The temperature boundary conditions are

$$\vartheta(0,t) = \vartheta(l,t) = 0 \quad \text{for} \quad t > 0 \tag{49}$$

and the initial temperature is given by

$$\vartheta(x,0) = \bar{\vartheta}(x) \quad \text{for} \quad x \in \mathcal{B} \quad \text{and} \quad t = 0\,. \tag{50}$$

The spatial discretization results analogously from Section 3.1.1, i.e. with the constant increment h

$$\mathcal{B}_h := \{ih \mid i = 1,\dots,n-1\} \quad \text{and} \quad \partial\mathcal{B}_h := \{ih \mid i = 0 \quad \text{and} \quad i = n\}\,. \tag{51}$$

In addition to a spatial discretization a discretization in time is needed. Therefore, we choose the grid-increment Δt and get the discrete times t_j by

$$t_j = j\Delta t \quad \text{for} \quad j = 0,1,2,\dots\,. \tag{52}$$

We calculate the Taylor polynom in t at a discrete time t_j according to the results from the previous section by using the forward difference quotient and obtain

$$\frac{\partial\vartheta(x_i,t_j)}{\partial t} = \frac{\vartheta(x_i,t_j+\Delta t) - \vartheta(x_i,t_j)}{\Delta t} - \frac{\Delta t}{2}\frac{\partial^2\vartheta(x_i,\mu_j)}{\partial t^2} - \dots\,, \tag{53}$$

where $\mu_j \in (t_j, t_{j+1})$ describes the continuous time in the discrete time-increment. By inserting the approximation for the second derivative of ϑ with respect to x at a point x_i yields the difference quotient

$$\frac{\partial^2\vartheta(x_i,t_j)}{\partial x^2} = \frac{\vartheta(x_{i+1},t_j) - 2\vartheta(x_i,t_j) + \vartheta(x_{i-1},t_j)}{h^2} - \frac{h^2}{12}\frac{\partial^4\vartheta(\xi_i,t_j)}{\partial x^4} - \dots \tag{54}$$

with $\xi_i \in (x_{i-1}, x_{i+1})$. We disregard the last terms in Equations (53) and (54) and insert the difference quotients in (47) in order to obtain the discretized equation

$$\frac{\vartheta(x_i,t_{j+1}) - \vartheta(x_i,t_j)}{\Delta t} - c^2\frac{\vartheta(x_{i+1},t_j) - 2\vartheta(x_i,t_j) + \vartheta(x_{i-1},t_j)}{h^2} = 0\,. \tag{55}$$

In the sequel we denote $\vartheta(x_i,t_j)$ with ϑ_{ij}. After solving Equation (55) with respect to $\vartheta_{i,j+1}$ we obtain

$$\vartheta_{i,j+1} = \left(1 - \frac{2c^2\Delta t}{h^2}\right)\vartheta_{ij} + \frac{c^2\Delta t}{h^2}(\vartheta_{i+1,j} + \vartheta_{i-1,j}) \tag{56}$$

for all $i = 1, \ldots, (n-1)$ and $j = 1, 2, \ldots$. For the solution of the discrete problem we proceed as follows; with given initial temperature $\vartheta(x, 0) = \vartheta_1(x)$ the values

$$\vartheta_{i0} = \bar{\vartheta}(x_i) \quad \forall \quad i = 0, 1, \ldots, n \tag{57}$$

are well-known. Furthermore, with given boundary conditions (49) we obtain

$$\vartheta_{0j} = \vartheta_{nj} = 0 \quad \forall \quad j = 1, 2, \ldots . \tag{58}$$

The abbreviation $\alpha = c^2 \Delta t / h^2$ leads to the discretized equation for $j = 0$

$$\vartheta_{i1} = (1 - 2\alpha)\vartheta_{i0} + \alpha(\vartheta_{i+1,0} + \vartheta_{i-1,0}) \, . \tag{59}$$

With the unknown temperature vector for the general discrete time t_j with $j = 1, 2, ...$

$$\boldsymbol{\vartheta}^{(j)} := [\; \vartheta_{1j} \quad \vartheta_{2j} \quad \ldots \quad \vartheta_{n-1,j} \;]^T \tag{60}$$

and the given vector of the initial temperature distribution ($j = 0$)

$$\boldsymbol{\vartheta}^{(0)} := [\; \bar{\vartheta}_1 \quad \bar{\vartheta}_2 \quad \ldots \quad \bar{\vartheta}_{n-1} \;]^T \tag{61}$$

the difference scheme (59) results for time t_1 and all positions x_i in the system of equations

$$\boldsymbol{\vartheta}^{(1)} = \boldsymbol{A} \boldsymbol{\vartheta}^{(0)} \, . \tag{62}$$

Here, $\boldsymbol{A} \in \mathbb{R}^{(n-1)\times(n-1)}$ is a tri-diagonal matrix of the form

$$\boldsymbol{A} = \begin{bmatrix} (1-2\alpha) & \alpha & 0 & \ldots & 0 \\ \alpha & (1-2\alpha) & \alpha & \ldots & 0 \\ 0 & \alpha & (1-2\alpha) & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \alpha & (1-2\alpha) \end{bmatrix} . \tag{63}$$

With Equation (62) we know the temperatures at all spatial grid-points at time t_1. Repeating this process for $j = 2$ yields the system of equations

$$\boldsymbol{\vartheta}^{(j)} = \boldsymbol{A} \boldsymbol{\vartheta}^{(j-1)} \quad \text{with} \quad j = 2 \, , \tag{64}$$

which can be computed directly. A repeated evaluation of (64) for $j = 3, 4, ...$ yields the unknown temperature $\boldsymbol{\vartheta}^{(j)}$ for all discrete points (x_i, t_j). The temperatures can be computed by the known values of the previous time steps. This *explicit* difference method is referred to as forward difference method.

Remark: The forward method is only stable, if the condition

$$c^2 \frac{\Delta t}{h^2} \leq \frac{1}{2} \tag{65}$$

holds. Hence, the time increment Δt is constrained with given increment h. More stable methods are the *implicit* difference methods. Such methods are e.g. obtained by inserting the backward difference quotient

$$\frac{\partial \vartheta(x_i, t_j)}{\partial t} \approx \frac{\vartheta(x_i, t_j) - \vartheta(x_i, t_{j-1})}{\Delta t} \, . \tag{66}$$

Introducing Equation (66) and (54) in the differential equation yields by disregarding terms of higher order the backward difference method

$$\frac{\vartheta_{ij} - \vartheta_{i,j-1}}{\Delta t} - c^2 \frac{\vartheta_{i+1,j} - 2\vartheta_{ij} + \vartheta_{i-1,j}}{h^2} = 0\,, \tag{67}$$

for all $i = 1, \ldots, n-1$ and $j = 1, 2, \ldots$. The repeated application of this procedure yields

$$\boldsymbol{A}\boldsymbol{\vartheta}^{(j)} = \boldsymbol{\vartheta}^{(j-1)} \tag{68}$$

with the tri-diagonal matrix

$$\boldsymbol{A} = \begin{bmatrix} (1+2\alpha) & -\alpha & 0 & \ldots & 0 \\ -\alpha & (1+2\alpha) & -\alpha & \ldots & 0 \\ 0 & -\alpha & (1+2\alpha) & \ldots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & -\alpha & (1+2\alpha) \end{bmatrix}. \tag{69}$$

This is an implicit method (backward difference method) since this linear system of equations has to be solved.

3.2 Two-dimensional problem

In the sequel the Finite-Difference-Method (FDM) is considered by means of the Poisson equation (elliptic equation) for a two-dimensional, rectangular domain $\mathcal{B}$, whose boundary is identified by $\partial\mathcal{B}$. The precise problem can be reformulated as follows:

Determine a function

$$u : \mathcal{B} \cup \partial\mathcal{B} \to \mathbb{R} \tag{70}$$

with given functions

$$g : \partial\mathcal{B} \to \mathbb{R} \quad \text{and} \quad f : \mathcal{B} \to \mathbb{R}\,, \tag{71}$$

such that

$$\left.\begin{aligned} -\sum_{i=1}^{d=2} \frac{\partial^2 u}{\partial x_i^2} &= f \text{ in } \mathcal{B} \\ u &= g \text{ on } \partial\mathcal{B} \end{aligned}\right\} \tag{72}$$

is fulfilled. The unknown function $u(x_1, x_2)$ can be interpreted as an electromagnetic potential, displacement of an elastic membrane or the planar equilibrium temperature distribution. An alternative notation for the right hand side of Equation $(72)_1$ is

$$\Delta u = u_{,11} + u_{,22} = \sum_{i=1}^{2} \frac{\partial^2 u}{\partial x_i^2}\,, \tag{73}$$

with the Laplace operator $\Delta(\cdot)$.

3.2.1 Discretization 2D In the framework of the FDM the numerical solution of the boundary value problem is computed based on a finite number of grid points in $\mathcal{B} \cup \partial\mathcal{B}$. For this purpose, the derivatives are approximated by difference quotients evaluated as discrete function values at the grid points. From this an algebraic system of equations follows for the approximate solution. This process is called discretization.

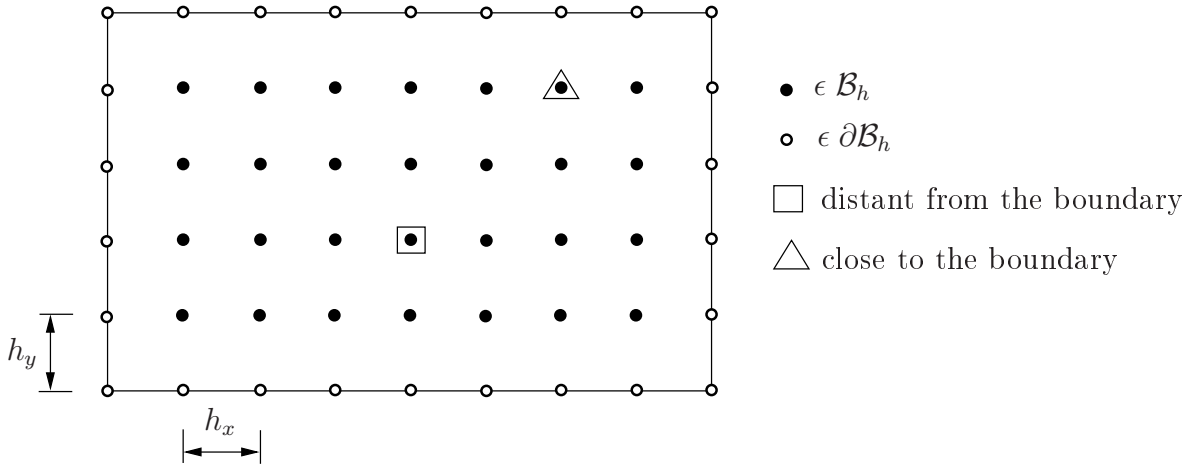


Figure 7: Discretization of the domain $\mathcal{B} = (0, l_x) \times (0, l_y)$ yields the discretized domain $\mathcal{B}_h$ with $(n_x + 1) \cdot (n_y + 1)$ grid points.

Here, we consider a two-dimensional problem ($d = 2$) and choose for the discretization the increments $h_x, h_y > 0$ in both directions. In Fig. 7 a rectangular domain

$$\mathcal{B} := (0, l_x) \times (0, l_y) \qquad \text{with} \quad l_x, l_y > 0 \tag{74}$$

is illustrated. For simplification an equidistant increment h_x is chosen in x-direction and h_y in y-direction, respectively, i.e.

$$l_x = n_x h_x \qquad \text{and} \qquad l_y = n_y h_y \qquad \text{with} \qquad n_x, n_y \in \mathbb{N}\,. \tag{75}$$

The grid points in the domain $\mathcal{B}$ are defined by

$$\mathcal{B}_h := \{(ih_x, jh_y) \mid i = 1, \dots, n_x - 1; j = 1, \dots, n_y - 1\} \tag{76}$$

and the grid points on the boundary $\partial\mathcal{B}$ by

$$\partial\mathcal{B}_h := \{(ih_x, jh_y) \mid i \in \{0, n_x\}, j \in \{0, \dots, n_y\} \text{ or } i \in \{0, \dots, n_x\}, j \in \{0, n_y\}\}\,. \tag{77}$$

Please note that here we focus on a boundary value problem which is rate-independent, thus, here no time discretization is required.

3.2.2 Difference quotients 2D The approximation formula for each point in the domain $\mathcal{B}_h$ is computed by

$$\frac{\partial^2 u(x_i, y_j)}{\partial x^2} = \frac{u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j)}{h_x^2} - \frac{h_x^2}{12}\frac{\partial^4 u(\xi_i, y_j)}{\partial x^4} - \dots \tag{78}$$

for $\xi_i \in [x_{i-1}, x_{i+1}]$. With the Taylor polynom for y at a point y_j we get analogously

$$\frac{\partial^2 u(x_i, y_j)}{\partial y^2} = \frac{u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j-1})}{h_y^2} - \frac{h_y^2}{12}\frac{\partial^4 u(x_i, \eta_j)}{\partial y^4} - \dots \tag{79}$$

for $\eta_j \in [y_{j-1}, y_{j+1}]$. Inserting Equation (78) and (79) in the Poisson equation yields

$$\left.\begin{aligned} &\frac{u(x_{i+1}, y_j) - 2u(x_i, y_j) + u(x_{i-1}, y_j)}{h_x^2} \\ &+\frac{u(x_i, y_{j+1}) - 2u(x_i, y_j) + u(x_i, y_{j-1})}{h_y^2} \\ &= f(x_i, y_j) + \frac{h_x^2}{12}\frac{\partial^4 u(\xi_i, y_j)}{x^4} + \frac{h_y^2}{12}\frac{\partial^4 u(x_i, \eta_j)}{y^4} \end{aligned}\right\} \quad \begin{aligned} \forall \quad & i = 1, 2, \dots, (n_x - 1) \quad \text{and} \\ & j = 1, 2, \dots, (n_y - 1). \end{aligned} \tag{80}$$

We set the boundary conditions

$$u(x_i, y_0) = g(x_i, y_0) \quad \text{and} \quad u(x_i, y_m) = g(x_i, y_m) \quad \forall\, i = 1, 2, \dots, (n_x - 1) \tag{81}$$

and

$$u(x_0, y_j) = g(x_0, y_j) \quad \text{and} \quad u(x_n, y_j) = g(x_n, y_j) \quad \forall\, j = 0, 1, \dots, n_y. \tag{82}$$

Let us denote the approximation of $u(x_i, y_j)$ as u_{ij}, then a straighforward transformation leads to the discrete form

$$\left.\begin{aligned} &2\left[\left(\frac{h_x}{h_y}\right)^2 + 1\right] u_{ij} - (u_{i+1,j} + u_{i-1,j}) - \left(\frac{h_x}{h_y}\right)^2 (u_{i,j+1} + u_{i,j-1}) = -h_x^2 f(x_i, y_j) \\ &\qquad \forall \quad i = 1, 2, \dots, (n_x - 1) \quad \text{and} \quad j = 1, 2, \dots, (n_y - 1) \\ &u_{0j} = g(x_0, y_j)\,,\ u_{nj} = g(x_n, y_j)\,,\ u_{i0} = g(x_i, y_0)\,,\ u_{im} = g(x_i, y_m) \\ &\qquad \forall \quad i = 1, 2, \dots, (n_x - 1)\,; \quad j = 1, 2, \dots, (n_y - 1) \end{aligned}\right\}. \tag{83}$$

For $h_x = h_y = h$ we get the standard five-point stencil

$$\frac{1}{h^2}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} . \tag{84}$$

Example: Computation of the stationary temperature distribution in a slim, quadratic metal plate with a length of $l_x = l_y = l = 0,5$ m. The given boundary temperatures are illustrated in Fig. 8.

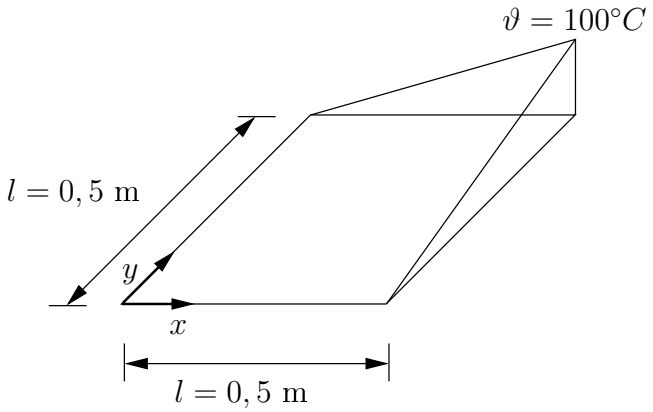


Figure 8: Quadratic plate with given boundary temperature.

The physical problem is described by the mathematical formulation

$$\frac{\partial^2 \vartheta}{\partial x^2}(x,y) + \frac{\partial^2 \vartheta}{\partial y^2}(x,y) = 0 \quad \forall \quad (x,y) \in \mathcal{B} , \tag{85}$$

which is referred to as Laplace equation. The domain $\mathcal{B}$ is defined by

$$\mathcal{B} := \{(x,y) | 0 < x < 0.5 \, , \, 0 < y < 0.5\} \tag{86}$$

and the boundary temperatures are

$$\vartheta(0,y) = 0 \, , \, \vartheta(x,0) = 0 \, , \, \vartheta(x,0,5) = 200x \, , \, \vartheta(0,5,y) = 200y \, . \tag{87}$$

In this example we are interested in the solution of the boundary value problem for the number of grid points given by $n_x = n_y = 4$, see Fig. 9.

Compare the numerical results with the exact analytical solution

$$\vartheta(x,y) = 400xy \, . \tag{88}$$

At the boundary nodes the temperature is known, whereas the temperature at the internal grid points has to be computed. These remaining 9 grid points are numbered according to Fig. 9.

Regarding equation $(83)_1$ we get for $h_x = h_y = h$ with $\vartheta \equiv u$ the expression

$$4\vartheta_{i,j} - (\vartheta_{i+1,j} + \vartheta_{i-1,j}) - (\vartheta_{i,j+1} + \vartheta_{i,j-1}) = -h^2 f(x_i, y_j) \, . \tag{89}$$

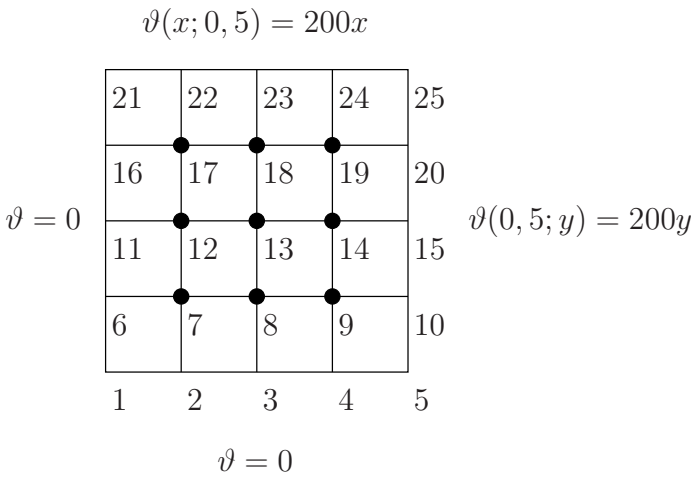


Figure 9: Discretization of the metal plate with 9 grid points in the domain and 16 boundary points.

Utilizing the Equation (89) for point 7 with $i = 1$ and $j = 1$ yields

$$4\vartheta_{1,1} - (\vartheta_{2,1} + \vartheta_{0,1}) - (\vartheta_{1,2} + \vartheta_{1,0}) = 0\,. \tag{90}$$

For simplification we use the transformation $\vartheta_{1,1} =: \vartheta_7, \vartheta_{1,2} =: \vartheta_{12}, \vartheta_{2,1} =: \vartheta_8$ and obtain

$$4\vartheta_7 - (\vartheta_8 + 0) - (\vartheta_{12} + 0) = 0\,, \tag{91}$$

respectively. The system of equations, which has to be solved, is given by application of the difference sheme for all inner grid points

$$\begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix} \begin{bmatrix} \vartheta_7 \\ \vartheta_8 \\ \vartheta_9 \\ \vartheta_{12} \\ \vartheta_{13} \\ \vartheta_{14} \\ \vartheta_{17} \\ \vartheta_{18} \\ \vartheta_{19} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 25 \\ 0 \\ 0 \\ 50 \\ 25 \\ 50 \\ 150 \end{bmatrix}. \tag{92}$$

The solution of this system of equations is

$$\boldsymbol{\vartheta} = \begin{bmatrix} 6,25 \\ 12,5 \\ 18,75 \\ 12,5 \\ 25,0 \\ 37,5 \\ 18,75 \\ 37,5 \\ 56,25 \end{bmatrix}$$

```
      program fdm
c-----------------------------------------------------------------------72
c     Example 2: Metal plate with stationary temperature distribution
c     Numerical solution with Finite-Difference-Method
c
c     D. Balzani, April 21, 2008
c
c     Parameters:
c     n        size of coefficient matrix a
c     a(n,n)   coefficient matrix
c     b(n)     right hand side vector
c...  output
c     x(n)     solution vector
c-----------------------------------------------------------------------
      implicit none
c
      integer n,nmax,i,k,j
      parameter (nmax=10)
      real*8 a(nmax,nmax),b(nmax),x(nmax)
c
c...  Number of inner grid points
      n = 9
c
c...  Open output file
      open (1, file='example2.dat')
c
c...  Initialize fields
      do i=1,n
        do j=1,n
          a(i,j) = 0.d0
        enddo
        b(i) = 0.d0
      enddo
c
c...  coefficient matrix a
      do i=1,n
        a(i,i) = 4.d0
      enddo
      a(1,2) = -1.d0
      a(1,4) = -1.d0
      a(2,3) = -1.d0
      a(2,5) = -1.d0
      a(3,6) = -1.d0
      a(4,5) = -1.d0
      a(4,7) = -1.d0
      a(5,6) = -1.d0
      a(5,8) = -1.d0
      a(6,9) = -1.d0
      a(7,8) = -1.d0
      a(8,9) = -1.d0
c     symmetrize
```

```
      do i=1,n
        do j=i+1,n
          a(j,i) = a(i,j)
        enddo
      enddo
c
c...  right hand side vector
      b(3) = 25.d0
      b(6) = 50.d0
      b(7) = 25.d0
      b(8) = 50.d0
      b(9) = 150.d0
c
c...  screen output
      write(*,*)
      write(*,*)'coefficient matrix:'
      do i=1,n
        write(*,100)(a(i,j),j=1,n)
      enddo
      write(*,*)
      write(*,*)'right hand side:'
      do i=1,n
        write(*,200)b(i)
      enddo
      write(*,*)
c
c...  solve system of equations
      call gausspivot(a,b,x,nmax,n)
c
c...  screen output of solution vector
      write(*,*)'solution vector:'
      do i=1,n
        write(*,200)x(i)
      enddo
      write(*,*)
c
c...  file output of solution vector
      do i=1,n
        write (1, 200) x(i)
      end do
c
 100  format (9f6.1)
 200  format (f10.3)
      end
```

4 Foundations of the Calculus of Variations

4.1 Preliminaries

Variational methods can be seen as the fundamental basis for the finite element method and other numerical techniques. Furthermore, they play a major role in physics, e.g., it is possible to derive the fundamental equations of Newtonian mechanics from a variational principle.

Variational methods can be distinguished between classical methods and direct methods. The classical ones lead to the so-called Euler-Lagrange differential equations (Euler differential equations) as a necessary condition for the solution of the variational problem. Exact solutions of the differential equations can only be constructed in some simple cases. In contrast to this, the direct methods start directly from the functional. Usually, ansatz-functions with initially unknown parameters are chosen for the approximation of the real solution. The parameters can be computed from some necesarry conditions for the solution of the variational problem. A popular technique is the well-known Ritz's method.

In the following we give a brief account of some classical parts of the calculus of variations. The main task is to find extreme values of a functional

$$\Pi(\mathrm{u}) = \int_{\mathrm{a}}^{\mathrm{b}} \mathrm{f}(\mathrm{x}, \mathrm{u}(\mathrm{x}), \mathrm{u}'(\mathrm{x}))\, \mathrm{dx}, \tag{93}$$

where the integrand of Π is governed by the unknown function $\mathrm{u}(\mathrm{x})$ and its derivative $\mathrm{u}'(\mathrm{x})$. For the function and its derivatives we have to define some boundary and side conditions. Possible boundary conditions for $\mathrm{u}(\mathrm{x})$ are

$$\mathrm{u}(\mathrm{a}) = \mathrm{u}_{\mathrm{a}} \quad \text{and} \quad \mathrm{u}(\mathrm{b}) = \mathrm{u}_{\mathrm{b}}. \tag{94}$$

Thus, the main exercise in the calculus of variations is to find a function $\mathrm{u}(\mathrm{x})$ for which the functional $\Pi(\mathrm{u})$ has an extremum, i.e.,

$$\Pi(\mathrm{u}) = \text{extreme!}. \tag{95}$$

Functions $\mathrm{u}(\mathrm{x})$ satisfying (95) are the so-called extremals. In the following we assume that all considered functions are sufficiently smooth, this means all operations like differentiation and integration by parts are possible and yield results having the needed properties.

Motivated by the Brachistochrone problem, formulated by Johann Bernoulli (1696), Euler developed a procedure for the solution of this kind of problems. So his fundamental idea was the construction of a comparable curve $\tilde{\mathrm{u}}(\mathrm{x})$ for the extremal $\mathrm{u}(\mathrm{x})$ with

$$\tilde{\mathrm{u}}(\mathrm{x}) = \mathrm{u}(\mathrm{x}) + \epsilon\eta(\mathrm{x}), \tag{96}$$

where ϵ is a small value and the arbitrary continuous function $\eta(x)$ vanishes at the boundaries, see Figure 10.

For all admissible functions $\tilde{\mathrm{u}}(\mathrm{x})$ the real function

$$\Pi_\epsilon(\epsilon) := \Pi(\tilde{\mathrm{u}}(\mathrm{x})) = \Pi(\mathrm{u}(\mathrm{x}) + \epsilon\eta(\mathrm{x})) \tag{97}$$

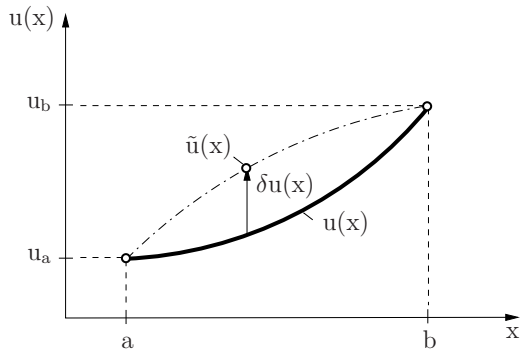


Figure 10: Boundary conditions u_a and u_b, extremal u(x), comparable curve ũ(x) and variation of the extremal $\delta u(x)$

has an extremum for $\epsilon = 0$, thus, the necessary condition is $\left.\frac{d}{d\epsilon}\Pi_\epsilon(\epsilon)\right|_{\epsilon=0} = 0$.
Let us now introduce the notation of the first variation or the so-called Gâteaux-derivative, defined by

$$\left.\frac{d}{d\epsilon}\Pi(u + \epsilon\eta)\right|_{\epsilon=0} = \lim_{\epsilon\to 0}\frac{1}{\epsilon}\left[\Pi(u + \epsilon\eta) - \Pi(u)\right] \tag{98}$$

A necessary condition for the solution of (95) is that for all admissible comparable functions ũ(x) the Gâteaux-variation vanishes,

$$\delta\Pi(u;\eta) = 0 \quad \text{with} \quad \delta\Pi(u;\eta) = \left.\frac{d}{d\epsilon}\Pi(u + \epsilon\eta)\right|_{\epsilon=0}\epsilon, \tag{99}$$

Remark: Formally, k-th order variations of the functional are denoted by

$$\delta^k\Pi(u;\eta) := \left.\frac{d^k}{d\epsilon^k}\Pi(u + \epsilon\eta)\right|_{\epsilon=0}\epsilon^k. \tag{100}$$

4.2 Classical Method, Euler-Lagrange Equation

As pointed above, we seek functions u(x) for which the functional $\Pi(u)$ has an extremum. In the following we follow the lines pointed out in standard text books on the claculus of variations, e.g., Bliss (1962), Weinstock (1974), Bufler (1991), van Brunt (2000), Lebedev et al. (2003). For the following analysis we define a comparable function (or test function)

$$\tilde{u}(x) = u(x) + \epsilon\eta(x),$$

see (96) with a small parameter ϵ and the arbitrary function $\eta(x)$ satisfying the homogenous boundary conditions

$$\eta(a) = 0 \quad \text{und} \quad \eta(b) = 0. \tag{101}$$

As the variation of the extremal we define

$$\delta u(x) := \tilde{u}(x) - u(x) = \epsilon\eta(x), \tag{102}$$

where $\delta u(x)$ could be interpreted as a virtual function, satisfying the homogenous boundary conditions

$$\delta u(a) = 0 \quad \text{und} \quad \delta u(b) = 0. \tag{103}$$

Inserting the comparable function $\tilde{u}(x)$ into Π leads to the dependency of $\Pi(\tilde{u}) = \Pi(\epsilon)$. It follows

$$\Pi(\tilde{u}) = \int_a^b f(x, \tilde{u}, \tilde{u}')dx = \int_a^b f(x, u + \epsilon\eta, u' + \epsilon\eta')dx =: \Pi_\epsilon(\epsilon)\,. \tag{104}$$

Obviously, we obtain for $\epsilon = 0$ the extremum $\Pi_\epsilon(0) = \Pi(u)$. A Taylor expansion of $\Pi_\epsilon(\epsilon)$ yields

$$\Pi_\epsilon(\epsilon) = \Pi_\epsilon(0) + \left.\frac{d\Pi_\epsilon}{d\epsilon}\right|_{\epsilon=0} \epsilon + \frac{1}{2!}\left.\frac{d^2\Pi_\epsilon}{d\epsilon^2}\right|_{\epsilon=0} \epsilon^2 + \ldots, \tag{105}$$

or in a more compact notation

$$\Pi_\epsilon(\epsilon) = \Pi(u, u') + \delta\Pi(u + \epsilon\eta) + \frac{1}{2!}\delta^2\Pi(u + \epsilon\eta) + \ldots. \tag{106}$$

Interchanging integration and differentiation yields for the first variation

$$\delta\Pi(u) \;=\; \epsilon \int_a^b \frac{d}{d\epsilon} f(x, \underbrace{u + \epsilon\eta}_{\tilde{u}}, \underbrace{u' + \epsilon\eta'}_{\tilde{u}'})\Big|_{\epsilon=0} dx. \tag{107}$$

In the latter equation we identify the second argument of f with $\tilde{u}$ and the third argument with the derivative of the comparable function

$$\tilde{u}'(x) = u'(x) + \epsilon\eta'(x), \tag{108}$$

so we write, applying the chain rule

$$\delta\Pi(u) \;=\; \int_a^b \left(\frac{\partial f}{\partial \tilde{u}}\frac{\partial \tilde{u}}{\partial \epsilon} + \frac{\partial f}{\partial \tilde{u}'}\frac{\partial \tilde{u}'}{\partial \epsilon}\right)\Big|_{\epsilon=0} dx\,\epsilon = \int_a^b \left(\frac{\partial f}{\partial \tilde{u}}\,\eta + \frac{\partial f}{\partial \tilde{u}'}\,\eta'\right)\Big|_{\epsilon=0} dx\,\epsilon.$$

Using (102) and the relation $\delta \mathrm{u}' = \epsilon \eta'$ leads to

$$\delta\Pi(\mathrm{u}) = \int_{\mathrm{a}}^{\mathrm{b}} \left(\frac{\partial \mathrm{f}}{\partial \mathrm{u}} \, \delta \mathrm{u} + \frac{\partial \mathrm{f}}{\partial \mathrm{u}'} \, \delta \mathrm{u}' \right) \mathrm{dx}. \tag{109}$$

Analogously, the second variation is given by the explicit expression

$$\begin{aligned} \delta^2\Pi(\mathrm{u}) &= \int_{\mathrm{a}}^{\mathrm{b}} \left(\frac{\partial^2 \mathrm{f}}{\partial \tilde{\mathrm{u}} \partial \tilde{\mathrm{u}}} \eta^2 + 2 \, \frac{\partial^2 \mathrm{f}}{\partial \tilde{\mathrm{u}} \partial \tilde{\mathrm{u}}'} \eta \eta' + \frac{\partial^2 \mathrm{f}}{\partial \tilde{\mathrm{u}}' \partial \tilde{\mathrm{u}}'} \eta'^2 \right) \Big|_{\epsilon=0} \mathrm{dx} \, \epsilon^2 \\ &= \int_{\mathrm{a}}^{\mathrm{b}} \left(\frac{\partial^2 \mathrm{f}}{\partial \mathrm{u} \partial \mathrm{u}} \delta \mathrm{u}^2 + 2 \, \frac{\partial^2 \mathrm{f}}{\partial \mathrm{u} \partial \mathrm{u}'} \delta \mathrm{u} \delta \mathrm{u}' + \frac{\partial^2 \mathrm{f}}{\partial \mathrm{u}' \partial \mathrm{u}'} \delta \mathrm{u}'^2 \right) \mathrm{dx}. \end{aligned} \tag{110}$$

Let us now compute the difference of the functionals $\Pi(\tilde{\mathrm{u}})$ and $\Pi(\mathrm{u})$ using (106) with $\Pi_\epsilon(\epsilon) = \Pi(\tilde{\mathrm{u}})$

$$\Pi(\tilde{\mathrm{u}}) - \Pi(\mathrm{u}) = \delta\Pi(\mathrm{u}) + \frac{1}{2!} \delta^2\Pi(\mathrm{u}) + \ldots. \tag{111}$$

The necessary condition for an extreme value of the variational problem yields

$$\delta\Pi(\mathrm{u}) = 0 \quad \rightarrow \int_{\mathrm{a}}^{\mathrm{b}} \left(\delta \mathrm{u} \frac{\partial \mathrm{f}}{\partial \mathrm{u}} + \delta \mathrm{u}' \frac{\partial \mathrm{f}}{\partial \mathrm{u}'} \right) \mathrm{dx} = 0. \tag{112}$$

□ For the justification of the next steps we consider the fundamental lemma of the calculus of variations: If a continuous function $\phi : [\mathrm{a}, \mathrm{b}] \rightarrow \mathbb{R}$ satisfies

$$\int_{\mathrm{a}}^{\mathrm{b}} \phi(\mathrm{x}) \, \delta \mathrm{u}(\mathrm{x}) \, \mathrm{dx} = 0 \tag{113}$$

with a twice-differentiable function $\delta \mathrm{u}(\mathrm{x})$ that vanishes at the boundaries, i.e., $\delta \mathrm{u}(\mathrm{a}) = \delta \mathrm{u}(\mathrm{b}) = 0$, then $\phi(\mathrm{x}) \equiv 0$. □

An illustration of the quantities occuring in (113) is pointed out in Figure (11). The proof of this lemma is based on a contrary assumption that $\phi(\mathrm{x}) \neq 0$ while (113) holds, which leads to a contradiction. In this context we refer to standard text books on calculus of variations, see e.g., Lebedev *et al.* (2003), van Brunt (2004).

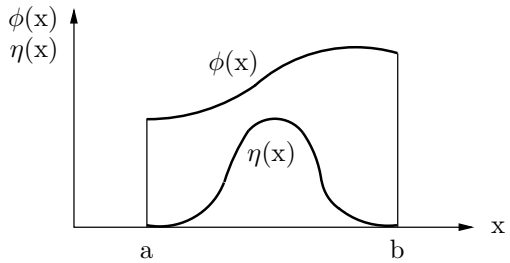


Figure 11: Visualization of the functions $\phi(\mathrm{x})$ and $\eta(\mathrm{x})$, occuring in the fundamental lemma of the calculus of variations

In order to derive the Euler-Lagrange differential equation we have to reformulate (112) in such a way, that

$$\delta\Pi(\mathrm{u}) = 0 \quad \rightarrow \int_{\mathrm{a}}^{\mathrm{b}} [\,\bullet\,]\,\delta\mathrm{u}\,\mathrm{dx} = 0 \quad \rightarrow [\,\bullet\,] = 0. \tag{114}$$

is valid. Thus we firstly perform a partial integration of the second term in $(112)_2$

$$\int_{\mathrm{a}}^{\mathrm{b}} \frac{\partial \mathrm{f}}{\partial \mathrm{u}'}\delta\mathrm{u}'\mathrm{dx} = \int_{\mathrm{a}}^{\mathrm{b}} \frac{\partial \mathrm{f}}{\partial \mathrm{u}'}\mathrm{d}\delta\mathrm{u} = \frac{\partial \mathrm{f}}{\partial \mathrm{u}'}\delta\mathrm{u}\Big|_{\mathrm{a}}^{\mathrm{b}} - \int_{\mathrm{a}}^{\mathrm{b}} \delta\mathrm{u}\frac{\mathrm{d}}{\mathrm{dx}}\left(\frac{\partial \mathrm{f}}{\partial \mathrm{u}'}\right)\mathrm{dx}. \tag{115}$$

Taking the homogenous boundary conditions (103) of $\delta\mathrm{u}$ into account, i.e.,

$$\frac{\partial \mathrm{f}}{\partial \mathrm{u}'}\delta\mathrm{u}\Big|_{\mathrm{a}}^{\mathrm{b}} = 0\,, \quad \text{because } \delta\mathrm{u}(\mathrm{a}) = 0,\ \delta\mathrm{u}(\mathrm{b}) = 0, \tag{116}$$

(112) is simplified to

$$\delta\Pi(\mathrm{u}) = \int_{\mathrm{a}}^{\mathrm{b}} \left[\frac{\partial \mathrm{f}}{\partial \mathrm{u}} - \frac{\mathrm{d}}{\mathrm{dx}}\left(\frac{\partial \mathrm{f}}{\partial \mathrm{u}'}\right)\right]\delta\mathrm{u}(\mathrm{x})\,\mathrm{dx} = 0\,. \tag{117}$$

Applying the fundamental lemma of the calculus of variations leads to the Euler-Lagrange differential equation

$$\boxed{\frac{\partial \mathrm{f}}{\partial \mathrm{u}} - \frac{\mathrm{d}}{\mathrm{dx}}\left(\frac{\partial \mathrm{f}}{\partial \mathrm{u}'}\right) = 0} \tag{118}$$

of the variational problem (95) with (93), where the explixit form of (118) is

$$\frac{\partial \mathrm{f}}{\partial \mathrm{u}} - \frac{\mathrm{d}}{\mathrm{dx}}\left(\frac{\partial \mathrm{f}}{\partial \mathrm{u}'}\right) = \frac{\partial \mathrm{f}}{\partial \mathrm{u}} - \frac{\partial^2 \mathrm{f}}{\partial \mathrm{x}\partial \mathrm{u}'} - \frac{\partial^2 \mathrm{f}}{\partial \mathrm{u}\partial \mathrm{u}'}\mathrm{u}' - \frac{\partial^2 \mathrm{f}}{\partial \mathrm{u}'\partial \mathrm{u}'}\mathrm{u}''.$$

As a first application we consider a simple one dimensional truss of length l. The Young's modulus E is assumed to be constant, the cross-sectional area A(x) and the axial loading p(x) are functions of the location.

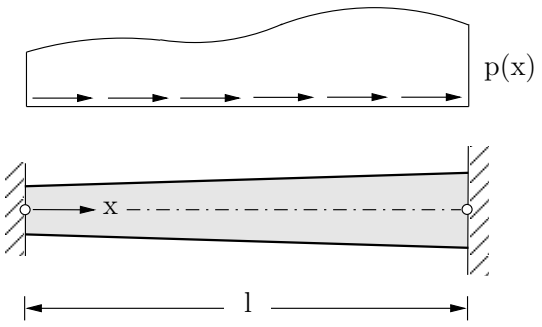


Figure 12: One-dimensional truss with variable stiffness EA(x) and axial loading p(x)

Furthermore, the displacements of the considered system are constrained at both ends of the bar, i.e.

$$\mathrm{u}(0) = 0 \qquad \text{and} \qquad \mathrm{u}(\mathrm{l}) = 0\,. \tag{119}$$

Let us start from the principle of the minimum of total potential energy

$$\Pi(\mathrm{u}) = \Pi^{\mathrm{i}}(\mathrm{u}) + \Pi^{\mathrm{e}}(\mathrm{u}) \rightarrow \text{minimum!}, \tag{120}$$

where Π denotes the total potential energy, Π^{i} the deformation or strain energy (internal energy) and Π^{e} the potential of the external loads. This principle states that a loaded elastic system presents an equilibrium state under some suitable boundary conditions if the total potential energy assumes a stationary value. In the following we focus on linear elastic systems; in this case the potential reaches a minimum. The strain energy is given by

$$\Pi^{\mathrm{i}}(\mathrm{u}) = \frac{1}{2}\int_0^{\mathrm{l}} \mathrm{EA}(\mathrm{x})\mathrm{u}'^2(\mathrm{x})\mathrm{dx} \tag{121}$$

and the potential of the external loads has the form

$$\Pi^{\mathrm{e}}(\mathrm{u}) = -\int_0^{\mathrm{l}} \mathrm{p}(\mathrm{x})\mathrm{u}(\mathrm{x})\mathrm{dx}. \tag{122}$$

Identifying the integrand of (93) with the latter equations yields

$$\mathrm{f}(\mathrm{x}, \mathrm{u}, \mathrm{u}') = \frac{1}{2}\mathrm{EA}(\mathrm{x})\mathrm{u}'^2(\mathrm{x}) - \mathrm{p}(\mathrm{x})\mathrm{u}(\mathrm{x}). \tag{123}$$

Thus, the derivatives of f with respect to u and u′ are

$$\frac{\partial \mathrm{f}}{\partial \mathrm{u}} = -\,\mathrm{p}(\mathrm{x}) \tag{124}$$

and

$$\frac{\mathrm{d}}{\mathrm{dx}}\left(\frac{\partial \mathrm{f}}{\partial \mathrm{u}'}\right) = \frac{\mathrm{d}}{\mathrm{dx}}\left(\mathrm{EA}(\mathrm{x})\mathrm{u}'(\mathrm{x})\right) = \mathrm{E}(\mathrm{A}(\mathrm{x})\mathrm{u}'(\mathrm{x}))'\,, \tag{125}$$

respectively. Evaluating (118) yields the Euler-Lagrange differential equation

$$-\,\mathrm{p}(\mathrm{x}) - \mathrm{E}(\mathrm{A}(\mathrm{x})\mathrm{u}'(\mathrm{x}))' = 0 \tag{126}$$

for the axial displacements of an axially loaded bar.

4.2.1 Boundary Conditions The boundary $\partial\mathcal{B}$ of the (onedimensional) body of interest $\mathcal{B} = [\mathrm{a}, \mathrm{b}]$ is seperated into $\partial\mathcal{B}_\mathrm{u}$ and $\partial\mathcal{B}_\mathrm{t}$, with

$$\partial\mathcal{B} = \partial\mathcal{B}_\mathrm{u} \bigcup \partial\mathcal{B}_\mathrm{t} \quad \text{and} \quad \partial\mathcal{B}_\mathrm{u} \bigcap \partial\mathcal{B}_\mathrm{t} = \emptyset. \tag{127}$$

For the variational problems we have to specify several types of boundary conditions. We distinguish between

1^{st} kind bc's:	These are the Dirichlet boundary conditions or the so-called essential boundary conditions. Here we describe values of the unknown functions u at each point of the boundary $\partial\mathcal{B}_\mathrm{u}$.
2^{nd} kind bc's:	Boundary conditions for the normal derivatives of u applied to the points of the boundary $\partial\mathcal{B}_\mathrm{t}$ are denoted as the Neumann boundary conditions or natural boundary conditions.
3^{rd} kind bc's:	These are mixed boundary conditions, Robin/Cauchy-type boundary condition; they present linear combinations of the 1^{st} and 2^{nd} kind bc's.

The boundary conditions of the second and third kind are the natural boundary conditions. Variational problems subjected to the Dirichlet conditions have been discussed in the last section. Let us now concentrate on a variational problem subjected to Dirichlet and Neumann boundary conditions of the form

$$\Pi(\mathrm{u}) = \int_\mathrm{a}^\mathrm{b} \mathrm{f}(\mathrm{x}, \mathrm{u}, \mathrm{u}')\mathrm{dx} + \mathrm{c} \cdot \mathrm{u}(\mathrm{b}) \rightarrow \text{extreme!} \tag{128}$$

with the constant c and

$$\mathrm{u}(\mathrm{a}) = \mathrm{u}_\mathrm{a} \quad \rightarrow \quad \delta\mathrm{u}(\mathrm{a}) = 0 \quad \text{and} \quad \delta\mathrm{u}(\mathrm{b}) \neq 0. \tag{129}$$

The necessary condition for an extreme value is $\delta\Pi = 0$. After integration by parts we get

$$\delta\Pi(\mathrm{u}) = \int_\mathrm{a}^\mathrm{b} \left[\frac{\partial \mathrm{f}}{\partial \mathrm{u}} - \frac{\mathrm{d}}{\mathrm{dx}}\left(\frac{\partial \mathrm{f}}{\partial \mathrm{u}'}\right)\right] \delta\mathrm{u}\,\mathrm{dx} + \left[\mathrm{c} + \frac{\partial \mathrm{f}(\mathrm{b})}{\partial \mathrm{u}'}\right] \delta\mathrm{u}(\mathrm{b}). \tag{130}$$

Applying the fundamental lemma of the calculus of variations yields for arbitrary $\delta\mathrm{u}(\mathrm{b}) \neq 0$ the set of equations

$$\frac{\partial \mathrm{f}}{\partial \mathrm{u}} - \frac{\mathrm{d}}{\mathrm{dx}}\left(\frac{\partial \mathrm{f}}{\partial \mathrm{u}'}\right) = 0 \quad \text{and} \quad \mathrm{c} + \frac{\partial \mathrm{f}(\mathrm{b})}{\partial \mathrm{u}'} = 0. \tag{131}$$

Equation $(131)_1$ is the Euler-Lagrange differential equation and $(131)_2$ represents the natural boundary condition of the second kind.

As an example for a variational problem with Neumann boundary conditions we consider a one-dimensional bar of length l with constant Young's modulus E, varying cross-sectional

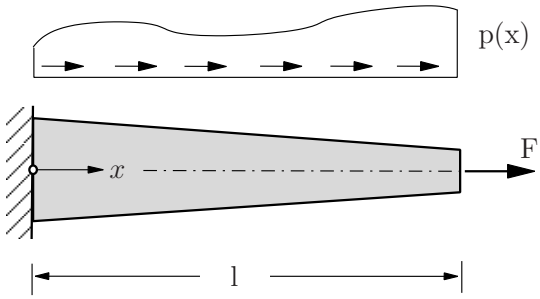


Figure 13: Bar with Dirichlet (left) and Neumann (right) boundary condition

area A(x) and axial loading p(x). The displacement of the system is fixed at the left end and an external force F is applied at the right end of the bar, see Figure 13.

Using again the principle of the minimum of total potential energy

$$\Pi(\mathrm{u}) = \Pi^{\mathrm{i}}(\mathrm{u}) + \Pi^{\mathrm{e}}(\mathrm{u}) \rightarrow \text{minimum!} \tag{132}$$

with the internal part

$$\Pi^{\mathrm{i}}(\mathrm{u}) = \int_{\mathrm{x}=0}^{\mathrm{l}} \frac{1}{2} \mathrm{EA}(\mathrm{x}) \mathrm{u}'^{2}(\mathrm{x}) \, \mathrm{dx} \tag{133}$$

and the external part

$$\Pi^{\mathrm{e}}(\mathrm{u}) = \int_{\mathrm{x}=0}^{\mathrm{l}} -\mathrm{p}(\mathrm{x})\mathrm{u}(\mathrm{x})\mathrm{dx} - \mathrm{F} \cdot \mathrm{u}(\mathrm{l}) \tag{134}$$

with the geometrical boundary condition $\mathrm{u}(0) = 0$. Evaluating (131) with $\mathrm{c} = -\mathrm{F}$ yields

$$\begin{aligned} -\mathrm{p}(\mathrm{x}) - \mathrm{E}[\mathrm{A}(\mathrm{x})\mathrm{u}'(\mathrm{x})]' &= 0 \\ -\mathrm{F} + \mathrm{EA}(\mathrm{l})\mathrm{u}'(\mathrm{l}) &= 0. \end{aligned} \tag{135}$$

Equation $(135)_1$ represents the Euler-Lagrange differential equation of the variational problem. The Neumann boundary condition is reflected by $(135)_2$; it states that the internal force at the right end of the bar $\mathrm{N}(\mathrm{l}) = \mathrm{EA}(\mathrm{l})\mathrm{u}'(\mathrm{l})$ has to be identical to the applied external force F.

4.2.2 Constraint Conditions A variety of applications in the calculus of variations demands the consideration of constraints which reduce the space of possible extremals. This constraints are differential equations, algebraic conditions and inequalities. Let us consider again a first order variational problem with Dirichlet boundary conditions, given by

$$\Pi(\mathrm{u}) = \int_{\mathrm{a}}^{\mathrm{b}} \mathrm{f}(\mathrm{x}, \mathrm{u}(\mathrm{x}), \mathrm{u}'(\mathrm{x}))\mathrm{dx} \rightarrow \text{extreme!}, \tag{136}$$

which is completed by a side condition, here e.g. a differential equation of the form

$$\mathrm{g}(\mathrm{x}, \mathrm{u}(\mathrm{x}), \mathrm{u}'(\mathrm{x})) = 0. \tag{137}$$

In this situation we cannot use the classical concept based on the comparable curve $\tilde{\mathrm{u}}(\mathrm{x})$, see (96), because it does not fulfill the side condition. A possibility is the introduction of a modified comparable curve

$$\tilde{\tilde{\mathrm{u}}}(\mathrm{x}) = \mathrm{u}(\mathrm{x}) + \epsilon_1 \eta_1(\mathrm{x}) + \epsilon_2 \eta_2(\mathrm{x}).$$

The parameters ϵ_1 and ϵ_2 are dependent due to the side condition.

An alternative is based on the Lagrange-Multiplier method. The problem (136) in addition with (137) is equivalent to

$$\mathcal{L}(\mathrm{u}, \lambda) = \int_{\mathrm{a}}^{\mathrm{b}} \mathrm{h}(\mathrm{x}, \mathrm{u}(\mathrm{x}), \mathrm{u}'(\mathrm{x}), \lambda(\mathrm{x}))\mathrm{dx} \rightarrow \text{extreme!}, \tag{138}$$

where we have introduced the abbreviation

$$\mathrm{h}(\mathrm{x}, \mathrm{u}(\mathrm{x}), \mathrm{u}'(\mathrm{x}), \lambda) = \mathrm{f}(\mathrm{x}, \mathrm{u}(\mathrm{x}), \mathrm{u}'(\mathrm{x})) + \lambda(\mathrm{x})\, \mathrm{g}(\mathrm{x}, \mathrm{u}(\mathrm{x}), \mathrm{u}'(\mathrm{x})). \tag{139}$$

The set of independent variables is (u,λ) with the unknown extremal variable u and the Lagrange multiplier λ. In this approach we obtain the Euler-Lagrange differential equations

$$\begin{aligned} \frac{\partial \mathrm{h}}{\partial \mathrm{u}} - \frac{\mathrm{d}}{\mathrm{dx}} \left(\frac{\partial \mathrm{h}}{\partial \mathrm{u}'} \right) &= 0 \\ \frac{\partial \mathrm{h}}{\partial \lambda} \equiv \mathrm{g} &= 0. \end{aligned} \tag{140}$$

As an application of a problem with a differential equation as side condition we consider a one-dimensional bar with homogenous Dirichlet boundary conditions, as depicted in Figure 12. As before we use the principle of the minimum of total potential energy

$$\Pi(\mathrm{u}, \varepsilon) = \int_0^{\mathrm{l}} \left(\frac{1}{2} \mathrm{EA}(\mathrm{x}) \varepsilon^2(\mathrm{x}) - \mathrm{p}(\mathrm{x}) \mathrm{u}(\mathrm{x}) \right) \mathrm{dx} \rightarrow \text{minimum!}, \tag{141}$$

where we have introduced an independent field for the strains. Additionally, we take a side condition into account, which relates the strains ε to the derivative of the displacement field u', i.e. the side condition is

$$\mathrm{g}(\mathrm{x}, \mathrm{u}', \varepsilon) = \mathrm{u}' - \varepsilon = 0. \tag{142}$$

Equivalent to the variational problem with side condition

$$\Pi(\mathrm{u},\varepsilon) \rightarrow \text{minimal!} \qquad \text{with} \qquad \mathrm{g}(\mathrm{x},\mathrm{u}',\varepsilon) = 0 \tag{143}$$

is the demand

$$\mathcal{L}(\mathrm{u},\varepsilon,\lambda) = \int_0^{\mathrm{l}} \mathrm{l}(\mathrm{x},\mathrm{u},\varepsilon,\mathrm{u}',\lambda)\mathrm{dx} \rightarrow \text{extreme!}. \tag{144}$$

Here the integrand of the new functional has the explicit form

$$\mathrm{l}(\mathrm{x},\mathrm{u},\varepsilon,\mathrm{u}',\lambda) = \frac{1}{2}\mathrm{EA}\varepsilon^2 - \mathrm{p}\ \mathrm{u} + \lambda(\mathrm{u}' - \varepsilon) \tag{145}$$

with the Lagrange-multiplier λ. The associated Euler equations of this variational problem are

$$\begin{aligned} \frac{\partial \mathrm{l}}{\partial \mathrm{u}} - \frac{\mathrm{d}}{\mathrm{dx}}\left(\frac{\partial \mathrm{l}}{\partial \mathrm{u}'}\right) &= 0 \quad \rightarrow \quad -\mathrm{p} - \lambda' &= 0, \\ \frac{\partial \mathrm{l}}{\partial \varepsilon} &= 0 \quad \rightarrow \quad \mathrm{EA}\varepsilon - \lambda &= 0, \\ \frac{\partial \mathrm{l}}{\partial \lambda} &= 0 \quad \rightarrow \quad \mathrm{u}' - \varepsilon &= 0. \end{aligned} \tag{146}$$

The last equation characterizes the relation between the displacements and the strains. An interpretation of the Lagrange-multiplier λ is obtained from the second Euler equation, it is the internal force in the axial direction N. Furthermore, this equation can be seen as the "constitutive law" of a linear elastic one-dimensional bar. Using the identification $\lambda = \mathrm{N}$ we observe that 146_1 represents the equilibrium condition of the axially loaded bar.

4.3 Direct Methods

The direct methods are used for solving the variational problem rather than solving the associated Euler-Lagrange differential equations. Consequently, we start directly from the variational problem, e.g. from (93). In this framework it is possible to specify requirements on the functional, which are necessary for the existence of solutions, and moreover, it is possible to design approximative numerical methods.

4.3.1 Ritz's Method The basic idea of Ritz's method, also known as Rayleigh-Ritz method, is to reduce the variational problem of minimizing the functional

$$\Pi(u) = \int_a^b f(x, u(x), u'(x))\, dx,$$

on the space of all feasible functions to a problem which deals with the minimizing of the same functional on a finite dimensional space. For this we choose a function

$$\bar{u}(x) = N_0(x) + \sum_{i=1}^{n} c_i N_i(x) \tag{147}$$

for an approximate solution of the problem with the yet unknown parameters c_i for $i = 1, ..., n$. Now we have to seek for the parameters which minimize the functional. For simplicity we firstly focus on pure Dirichlet boundary conditions

$$u(a) = u_a \quad \text{and} \quad u(b) = u_b.$$

In order to satisfy the boundary conditions a priori we choose functions satisfying

$$N_0(a) = u_a \quad \text{and} \quad N_0(b) = u_b, \tag{148}$$

as well as

$$N_i(a) = N_i(b) = 0 \quad \text{for} \quad i = 1, ...n. \tag{149}$$

The latter functions, satisfying the homogenous boundary conditions, the so-called basis functions, are assumed to be linearly independent, i.e.

$$\sum_{i=1}^{n} c_i N_i(x) = 0 \quad \text{iff} \quad c_i = 0 \quad \text{for} \quad i = 1, ..., n. \tag{150}$$

Inserting the finite sum into the variational problem leads to the approximation

$$\Pi(u) \approx \Pi(\bar{u}) = \int_a^b f(x, \bar{u}, \bar{u}')dx \rightarrow \text{ extreme!} \tag{151}$$

The first variation of the approximated problem reads

$$\delta\Pi(\bar{u}) = \int_a^b \left(\frac{\partial f}{\partial \bar{u}} \delta\bar{u} + \frac{\partial f}{\partial \bar{u}'} \delta\bar{u}' \right) dx = 0 \tag{152}$$

with the abbreviation

$$\delta\bar{u}(x) = \sum_{i=1}^{n} N_i(x) \delta c_i. \tag{153}$$

Writing out (152) yields with $\Pi(c_1, ..., c_n) = \Pi(\bar{u})$

$$\delta\Pi(c_1, ..., c_n) = \sum_{i=1}^{n} \int_a^b \left(\frac{\partial f}{\partial \bar{u}} N_i + \frac{\partial f}{\partial \bar{u}'} N_i' \right) dx \, \delta c_i = 0. \tag{154}$$

Alternatively we can write

$$\frac{\partial \Pi(c_1, ..., c_n)}{\partial c_i} = \int_a^b \left(\frac{\partial f}{\partial \bar{u}} N_i + \frac{\partial f}{\partial \bar{u}'} N_i' \right) dx = 0 \quad \text{for} \quad i = 1, ..., n, \tag{155}$$

because the values δc_i for $i = 1, ..., n$ in (154) are arbitrary. The latter expression represents a system of n equations with n parameters c_i for $i = 1, ..., n$. It should be remarked that the system is linear if $f(\bullet)$ is a quadratic form in the unknown variables.

For a more detailed look into Ritz's method let us consider an axially loaded bar with constant axial rigidity EA. At the left end we fix and at the right end we prescribe a displacement, i.e.

$$u(0) = 0 \quad \text{and} \quad u(l) = u_b, \tag{156}$$

see Figure 14.

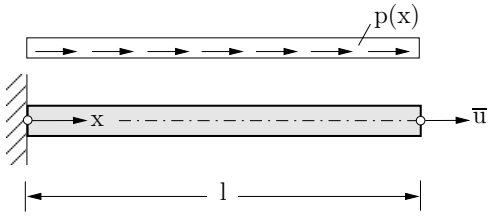


Figure 14: System and boundary conditions

The total potential is

$$\Pi(u) = \int_0^l \left(\frac{1}{2} EA(u')^2 - pu \right) dx \rightarrow \text{ minimal!} \tag{157}$$

In order to satisfy the essential boundary conditions we choose the linear function

$$N_0(x) = \frac{u_b}{l} x. \tag{158}$$

Now we will consider a solution in the form of the polynomial

$$\bar{u}(x) = N_0(x) + c_1 N_1(x) \tag{159}$$

with the base function

$$N_1(x) = x(x - l), \tag{160}$$

which vanishes at the boundaries. So we get with $\Pi(c_1) = \Pi(\bar{u})$ the approximation (161)

$$\Pi(c_1) = \int_0^l \left(\frac{1}{2} EA(\bar{u}'(x))^2 - p\bar{u}(x) \right) dx. \tag{161}$$

Evaluating (155) yields

$$\begin{aligned}\frac{\partial \Pi(c_1)}{\partial c_1} &= \int_0^l (EA\bar{u}'N_1' - pN_1)\,dx \\ &= EA\int_0^l \left((N_0' + c_1N_1')N_1' - \frac{1}{EA}pN_1\right)dx = 0.\end{aligned} \tag{162}$$

After some simple manipulations we get

$$c_1\int_0^l N_1'^2 dx = \frac{p_0}{EA}\int_0^l N_1 dx - \int_0^l N_0'N_1' dx, \tag{163}$$

which leads to

$$c_1\left(\frac{1}{3}l^3\right) = \frac{p_0}{EA}\left(-\frac{l^3}{6}\right) - 0 \quad \rightarrow \quad c_1 = -\frac{p_0}{2EA}. \tag{164}$$

Inserting the solution for c_1 in (159) gives the sought-after solution

$$\bar{u}(x) = \frac{u_b}{l}x + \frac{p}{2EA}x(l-x). \tag{165}$$

In this special case we obtained the exact solution of the variational problem.

4.3.2 Weighted Residual Methods Ritz's method is only applicable if a variational problem exists, which is equivalent to the considered boundary value problem. For problems, which cannot be recast into a variational formulation, we need other methods for the computation of approximate solutions. Let us consider a differential equation of order $2k$

$$f(x, u(x), u'(x), ..., u^{(2k)}(x)) + p(x) = 0\,, \tag{166}$$

in the interval $[a, b]$. The associated 2 k boundary conditions are alluded to the values of the function u (x) and/or their derivatives at the points a and b. For the approximate solutions we choose an ansatz-function of the form

$$\bar{u}(x) = N_0(x) + \sum_{i=1}^{n} c_i N_i(x), \tag{167}$$

which satisfies all boundary conditions. Furthermore, the functions N_0 and $N_1, ..., N_n$ are integrable as well as their derivations of order $2k$. Inserting the ansatz-function (167) in (166) leads to the so-called residual

$$r(x) := f(x, \bar{u}(x), \bar{u}'(x), ...\bar{u}^{(2k)}(x)) + p(x) \neq 0. \tag{168}$$

Now we have to seek the free parameters $c_1, ..., c_n$ in such a way that the residual is small in some sense. For this we distribute the residual in an overall manner throughout the interval. The basic idea is to multiply (168) with the weighting function $\eta(x)$ and integrate over the considered domain, i.e.

$$\int_a^b \eta(x) r(\bar{u}) dx = \int_a^b \eta(x)(f(x, \bar{u}, ..., \bar{u}^{(2k)}) + p(x))dx = 0\,. \tag{169}$$

Based on different definitions of the weighting functions we obtain different types of approximation methods, e.g. the point-collocation method, the subdomain-collocation

method, the method of moment, and the Galerkin methods. The weighting function is chosen by

$$\eta(\mathrm{x}) = \sum_{\mathrm{j}=1}^{\mathrm{n}} \bar{\mathrm{c}}_{\mathrm{j}} \bar{\mathrm{N}}_{\mathrm{j}}(\mathrm{x}) \tag{170}$$

with the arbitrary parameters $\bar{\mathrm{c}}\mathrm{j}$. Inserting the weighting function in the basic equation for the weighted-residual method (169) leads to a set of n equations

$$\begin{aligned} \int_{\mathrm{a}}^{\mathrm{b}} \bar{\mathrm{N}}_1(\mathrm{x})\ \mathrm{r}(\mathrm{x})\mathrm{dx} &= 0 \\ \int_{\mathrm{a}}^{\mathrm{b}} \bar{\mathrm{N}}_2(\mathrm{x})\ \mathrm{r}(\mathrm{x})\mathrm{dx} &= 0 \\ &\ldots \\ \int_{\mathrm{a}}^{\mathrm{b}} \bar{\mathrm{N}}_{\mathrm{n}}(\mathrm{x})\ \mathrm{r}(\mathrm{x})\mathrm{dx} &= 0, \end{aligned} \tag{171}$$

which have to be solved with respect to the free parameters $c_1, ..., c_n$.

Galerkin Method

In the Galerkin method the weighting functions are the same as the approximation basis functions, also known as the Bubnov-Galerkin method.

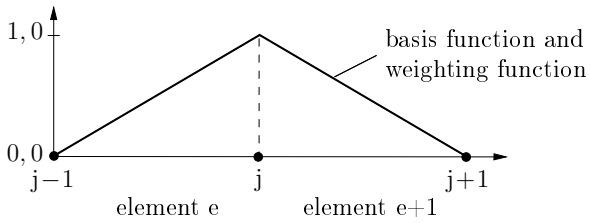


Figure 15: Weighting function for the standard Galerkin method

Thus, we obtain with

$$\bar{N}_i(x) = N_i(x) \quad \text{for} \quad i = 1, 2, ..., n \tag{172}$$

the set of equations

$$\begin{aligned} \int_a^b N_1(x)\, r(x) dx &= 0 \\ \int_a^b N_2(x)\, r(x) dx &= 0 \\ &\dots \\ \int_a^b N_n(x)\, r(x) dx &= 0 \end{aligned} \tag{173}$$

An advantage of this method is that an equivalent variational formulation of the considered boundary value problem is not required. On the other hand, the ansatz-function have to fulfill the essential as well as the natural boundary conditions. Furthermore, for a differential equation of order 2 k, the functions have to be (2 k - 1) times continuously differentiable. This drawback, especially in view of the Finite-Element-Method, can be avoided by applying partial integration of the residual function, in general. After this we can use approximation functions, which have only to be (k - 1) times continuously differentiable and only have to satisfy the essential boundary conditions.

The latter notes are explained in more detail considering the uniform beam fixed at the left end and subjected to a vertical point load as well as a bending moment at the other end, see Figure 16.

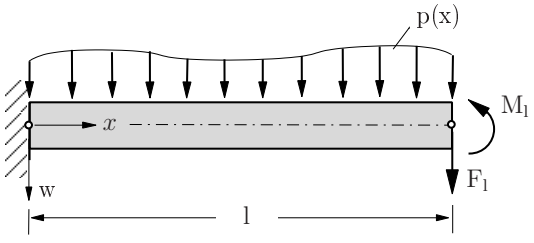


Figure 16: System and boundary conditions

The problem is described in terms of the following boundary value problem:

$$\begin{aligned} EIw''''(x) - p(x) &= 0 \\ w(0) &= 0 \\ w'(0) &= 0 \\ EIw''(l) &= -M(l) \quad ; \quad M(l) = M_l \\ EIw'''(l) &= -Q(l) \quad ; \quad Q(l) = F_l \end{aligned} \tag{174}$$

The essential boundary conditions are $(174)_{2,3}$ and the natural ones are given by $(174)_{4,5}$. Let us now introduce an ansatz-function $\tilde{w}$, then the residual of $(174)_1$ is defined by

$$r(x) := EI\tilde{w}''''(x) - p(x). \tag{175}$$

Furthermore, the residuals of the equations $(174)_{4,5}$ are

$$r_M := EI\tilde{w}''(l) + M_l \quad \text{and} \quad r_F := EI\tilde{w}'''(l) + F_l. \tag{176}$$

Using the concept of the weighted residual method leads to

$$\int_0^l r(x)\eta(x)dx + r_M\eta_M + r_F\eta_F = 0 \tag{177}$$

with the weighting function $\eta(x)$ and the weights η_M as well as η_F. Furthermore, we assume that $\eta(x)$ fulfills the homogeneous boundary conditions $\eta(0) = 0$ and $\eta'(0) = 0$. For the explicit form of this equation we get

$$\int_0^l [EI\tilde{w}''''(x) - p(x)]\eta(x)dx + [EI\tilde{w}''(l) + M_l]\eta_M + [EI\tilde{w}'''(l) + F_l]\eta_F = 0 \tag{178}$$

The partial integration of the term $\int_0^l EI\tilde{w}''''(x)\eta(x)dx$ yields

$$\left.\begin{aligned} \int_0^l EI\tilde{w}''''(x)\eta(x)dx &= EI\tilde{w}'''(x)\eta(x)\Big|_0^l - \int_0^l EI\tilde{w}'''(x)\eta'(x)dx \\ &= EI\tilde{w}'''(x)\eta(x)\Big|_0^l \\ &\quad -EI\tilde{w}''(x)\eta'(x)\Big|_0^l + \int_0^l EI\tilde{w}''(x)\eta''(x)dx \end{aligned}\right\}. \tag{179}$$

After evaluating the boundary terms we get

$$\int_0^l EI\tilde{w}''''(x)\eta(x)dx = EI\tilde{w}'''(l)\eta(l) - EI\tilde{w}''(l)\eta'(l) + \int_0^l EI\tilde{w}''(x)\eta''(x)dx. \tag{180}$$

Inserting this intermediate result in (178)

$$\begin{aligned} &\int_0^l [EI\tilde{w}''(x)\eta''(x) - p(x)\eta(x)]dx \\ &+EI\tilde{w}'''(l)[\eta(l) + \eta_F] + \eta_F F_l + EI\tilde{w}''(l)[\eta_M - \eta'(l)] + \eta_M M_l = 0. \end{aligned} \tag{181}$$

For the arbitrary weights η_{M} and η_{F} we choose

$$\eta_{\mathrm{M}} = \eta'(\mathrm{l}) \quad \text{and} \quad \eta_{\mathrm{F}} = -\eta(\mathrm{l}), \tag{182}$$

in order to reduce (181) to the final expression

$$\int_0^{\mathrm{l}} [\mathrm{EI}\tilde{\mathrm{w}}''(\mathrm{x})\eta''(\mathrm{x}) - \mathrm{p}(\mathrm{x})\eta(\mathrm{x})]\mathrm{dx} - \eta(\mathrm{l})\mathrm{F} + \eta'(\mathrm{l})\mathrm{M}_{\mathrm{l}} = 0\,. \tag{183}$$

It can be seen that the function $\tilde{\mathrm{w}}(\mathrm{x})$ only has to satisfy the essential boundary conditions and $\tilde{\mathrm{w}}(\mathrm{x})$ and $\eta(\mathrm{x})$ only have to be $(\mathrm{k}-1)$ times continuously differentiable.

If we choose different functions for the weighting and the approximation basis functions, i.e.

$$\bar{\mathrm{N}}_{\mathrm{i}}(\mathrm{x}) \neq \mathrm{N}_{\mathrm{i}}(\mathrm{x}) \quad \text{for} \quad \mathrm{i} = 1, 2, ..., \mathrm{n} \tag{184}$$

then the method is dedicated to Petrov-Galerkin.

Exemplification: In order to explain the standard Galerkin method we consider the model problem shown in Fig. 17.

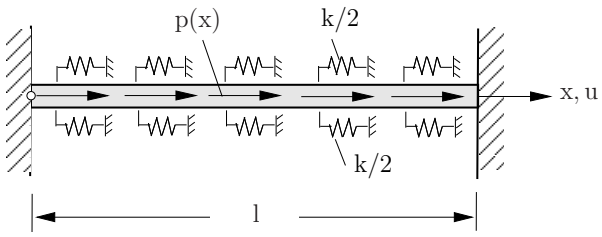


Figure 17: Axially loaded bar with Dirichlet boundary conditions and elastic support

Starting from (169) we obtain for our situation

$$\int_{\mathrm{a}}^{\mathrm{b}} \eta(\mathrm{x})\mathrm{r}(\mathrm{x})\mathrm{dx} = \int_{\mathrm{a}}^{\mathrm{b}} \eta(\mathrm{x})\Big(\mathrm{EA}\bar{\mathrm{u}}''(\mathrm{x}) - \mathrm{k}\ \bar{\mathrm{u}}(\mathrm{x}) + \mathrm{p}(\mathrm{x})\Big)\mathrm{dx} = 0\,. \tag{185}$$

We can directly use the latter equation for the computation of the approximative solution, but in view of the standard procedure in the finite-element procedure let us first partially integrate $\displaystyle\int_0^{\mathrm{l}} \eta(\mathrm{x})\mathrm{EA}\bar{\mathrm{u}}''(\mathrm{x})\mathrm{dx}$:

$$\int_0^{\mathrm{l}} \eta(\mathrm{x})\mathrm{EA}\bar{\mathrm{u}}''(\mathrm{x})\mathrm{dx} = \eta(\mathrm{x})\mathrm{EA}\bar{\mathrm{u}}'(\mathrm{x})\Big|_0^{\mathrm{l}} - \int_0^{\mathrm{l}} \eta'(\mathrm{x})\mathrm{EA}\bar{\mathrm{u}}'(\mathrm{x})\mathrm{dx} \tag{186}$$

Exploiting the homogeneous conditions $\eta(0) = 0$ and $\eta(\mathrm{l}) = 0$ yields

$$\int_0^{\mathrm{l}} \eta(\mathrm{x})\mathrm{EA}\bar{\mathrm{u}}''(\mathrm{x})\mathrm{dx} = -\int_0^{\mathrm{l}} \eta'(\mathrm{x})\mathrm{EA}\bar{\mathrm{u}}'(\mathrm{x})\mathrm{dx}. \tag{187}$$

Thus, the remaining equation reads

$$\int_0^l \eta'(x) EA \bar{u}'(x) dx + \int_0^l \eta(x)\ k\ \bar{u}(x) dx = \int_0^l \eta(x) p(x) dx. \tag{188}$$

As discussed above we choose the same functions for the weighting and the basis functions, i.e.

$$\eta(x) = \sum_{j=1}^{n} \bar{c}_j N_j(x) \quad \text{and} \quad \bar{u}(x) = N_0(x) + \sum_{i=1}^{n} c_i N_i(x) \tag{189}$$

with $N_0 = 0$; the derivatives of this functions are

$$\eta'(x) = \sum_{j=1}^{n} \bar{c}_j N_j'(x) \quad \text{and} \quad \bar{u}'(x) = \sum_{i=1}^{n} c_i N_i'(x). \tag{190}$$

Finally we obtain the set of equations for the arbitrary parameters $c_1, c_2, ..., c_n$

$$\begin{aligned}
&\int_0^l N_1'(x) EA \bar{u}'(x) dx + \int_0^l N_1(x)\ k\ \bar{u}(x) dx = \int_0^l N_1(x) p(x) dx. \\
&\int_0^l N_2'(x) EA \bar{u}'(x) dx + \int_0^l N_2(x)\ k\ \bar{u}(x) dx = \int_0^l N_2(x) p(x) dx. \\
&\qquad\qquad ... \\
&\int_0^l N_n'(x) EA \bar{u}'(x) dx + \int_0^l N_n(x)\ k\ \bar{u}(x) dx = \int_0^l N_n(x) p(x) dx.
\end{aligned} \tag{191}$$

Inserting the relations $(189)_2$ and $(190)_2$ leads in matrix notation to the expression (dropping the x-dependencies)

$$\begin{bmatrix}
\int_0^l (N_1' N_1' + N_1 \tilde{k} N_1) dx & \cdots & \int_0^l (N_1' N_n' + N_1 \tilde{k} N_n) dx \\
\int_0^l (N_2' N_1' + N_2 \tilde{k} N_1) dx & \cdots & \int_0^l (N_2' N_n' + N_2 \tilde{k} N_n) dx \\
\vdots & \ddots & \vdots \\
\int_0^l (N_n' N_1' + N_n \tilde{k} N_1) dx & \cdots & \int_0^l (N_n' N_n' + N_n \tilde{k} N_n) dx
\end{bmatrix}
\begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}
=
\begin{bmatrix}
\int_0^l N_1 \tilde{p} dx \\
\int_0^l N_2 \tilde{p} dx \\
\vdots \\
\int_0^l N_n \tilde{p} dx
\end{bmatrix}$$

with the abbreviations

$$\tilde{k} = \frac{k}{EA} \quad \text{and} \quad \tilde{p} = \frac{p}{EA}. \tag{192}$$

5 Introduction to Matlab

5.1 Introduction

What is MATLAB ?

- MATLAB (abbr. for MATrix LABoratory) is a software package for numerical computation and visualization. The main part consists of a huge number of build-in functions (e.g. linear Algebra, solving differential equations).
- MATLAB is primarily a numerical computation package, while MATHEMATICA and MAPLE are primarily symbolic algebra packages.
- Major MATLAB "clones": SCILAB, OCTAVE

Basic features:

- Automatic dimensioning (i.e., no dimension statements are required for vectors or arrays,
- Case sensitivity (e.g., `a` and `A` are different variables)
- Build-in function are based on and optimized for vector and matrix operations.
- File types:
  - M-files: standard ASCII text files, with a `.m` extension,
  - Mat-files: binary data files, with a `.mat` extension,
  - Mex-files: MATLAB-callable Fortran of C programs, with a `.mex` extension,
- The fundamental data type is the array which encompasses several distinct data objects: integers, doubles, matrices, character strings, structures, and cells.
- Data objects do not need to be declared as specific data types, e.g.,there is no need to declare variables as real or complex.

5.2 Basics

5.2.1 Graphical User Interface of MATLAB

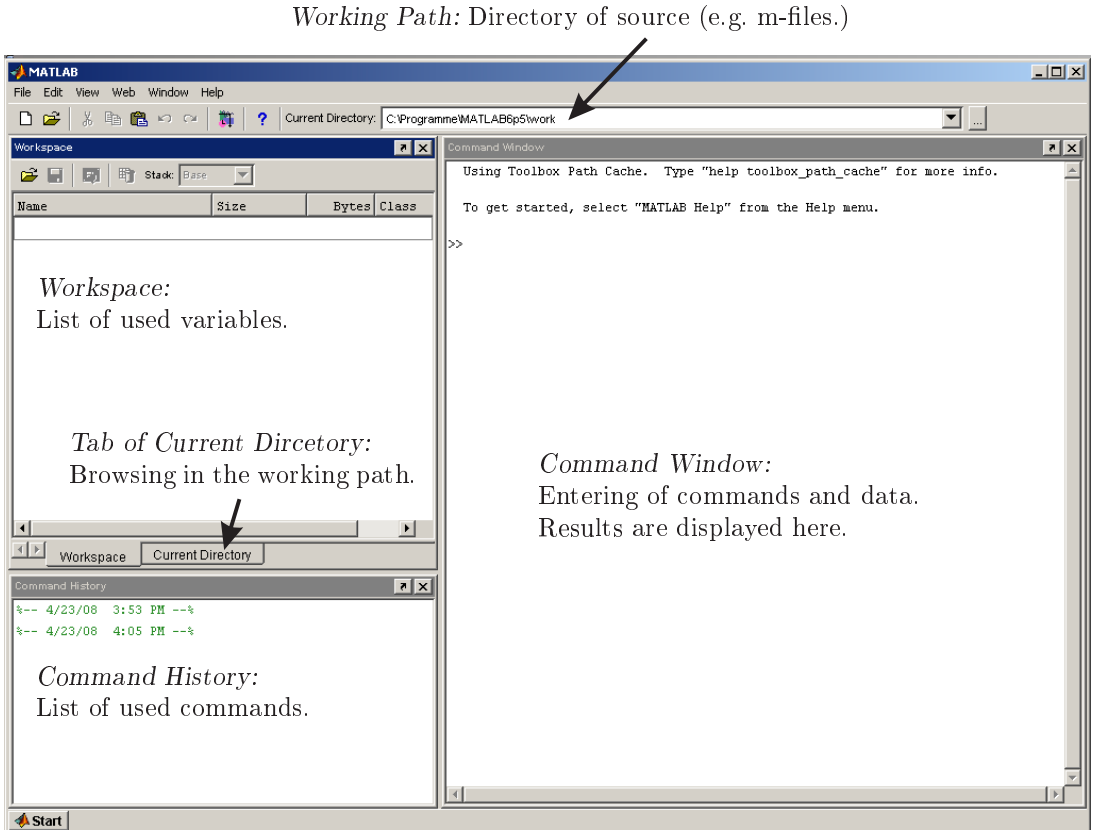


Figure 18: Components of MATLAB 's GUI

5.2.2 Additional Environments

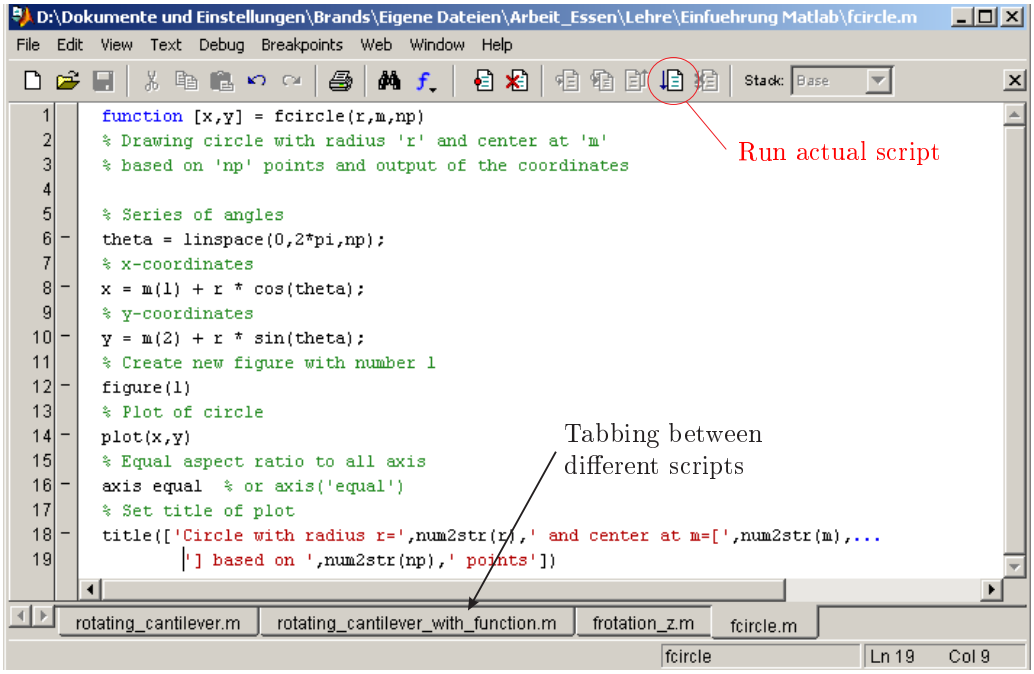


Figure 19: M-Editor

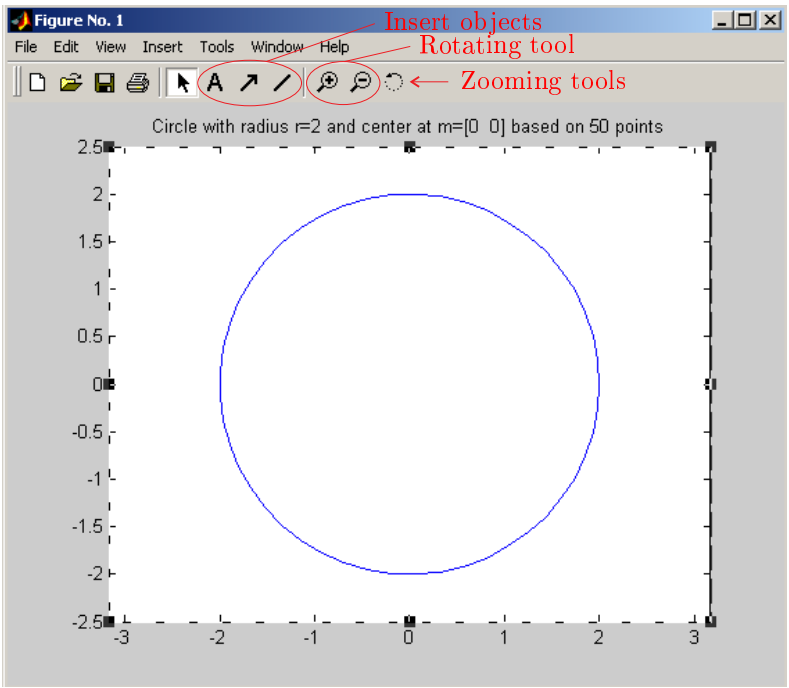


Figure 20: Figure window

5.2.3 General Commands

Help

`help <command>`	help on `<command>`
`lookfor <string>`	lists help containing `<string>`

Directory Operations

`pwd`	show current directory
`cd`	change directory
`dir` / `ls`	list contents of current directory

Example:

```
>> help cos
 COS    Cosine.
    COS(X) is the cosine of the elements of X.
 Overloaded methods
    help sym/cos.m
```

5.2.4 Variables

Operators

`=`	assignment of value
`+ - * / ^`	mathematical operators
, (at end of line)	command with ouput
; (at end of line)	no output
...	linebreak inside of a command

Fix Values

`i,j`	imaginary value $\sqrt{-1}$
`pi`	$\pi = 3.14\ldots$
`inf`	infinity inf
`NaN`	Not a Number, e. g. $\frac{0}{0}$
`eps`	smallest displayable positive number

Management of Variables

`who`	lists variables currently in workspace by name
`whos`	lists variables with their size
`clear`	clear workspace
`clear <var>`	clear variable <var>
`clc` / `clf` / `cla`	clear command / figure / axes

More information: type `help +` at MATLAB -commandline.

Examples:

```
>> ( 47 + 1e+02 * 1.5 + 4^2 ) / 4
ans =
   53.2500
```

`ans` is the variable of the result of the last execution.

```
>> a = ( 47 + 1e+02 * 1.5 + 4^2 ) / 4
a =
   53.2500
```

The result is stored in variable `a`. execution.

5.2.5 Mathematical Functions

`sqrt(x)`	square root
`xp(x)`	exponential funktion
`log(x)` / `log10`	logarithms
`sin(x)`	sinus
`cos(x)`	cosin
`tan(x)`	tangens
`atan(x)`	arcus-tanges (angle $-90° \ldots + 90°$)
`atan2(y,x)`	arcus-tanges (angle $-180° \ldots + 180°$)
`abs(x)`	absolute value of `x`
`sign(x)`	signum (sign of `x`)

More information: type `help elfun` or `help datafun` at MATLAB -commandline.

Example:

```
>> y1 = 2^2+log(pi)*sin(0.75*pi/2)+sqrt(exp(2*pi/3))
y1 =
    7.9072
```

$$y_1 = 2^2 + \ln \pi \sin(0.75\pi/2) + \sqrt{e^{2/3\pi}}$$

5.2.6 Vectors and Matrices

Formulation of vectors and matrices

`[ x1 x2 ...; y1,y2,...]`	vector or matrix (',' or space between columns, ';' or linebreak between rows)
`start:` <`stepsize:`> `end`	colon-operator (stepsize is opt., otherwise = 1)
`linspace(start,end,num_steps)`	linear row-vector
`logspace(start,end,num_steps)`	logarithmical row-vector
`eye(rows,columns)`	Identity [rows x cols]
`ones(rows,columns)`	matrix with all Elements equal 1 [rows x cols]
`zeros(rows,columns)`	zero-matrix [rows x cols]
`rand(rows,columns)`	random matrix [rows x cols]
`a(index)`	element of vector at position `index`
`A(r,c)`	element of matrix at row `r` and column `c`

Examples:

```
>> x=[1 2 3]
x =
     1     2     3
>> y=[4;5;6]
y =
     4
     5
     6
>> A=[1 2 3;4,5,6;7 8,9]
A =
     1     2     3
     4     5     6
     7     8     9
>> A(2,3)
ans =
     6
```

```
>> A(1,2)=8
A =
     1     8     3
     4     5     6
     7     8     9
>> B=A(2:3,1:3)
B =
     4     5     6
     7     8     9
>> v=0:2:8
v =
     0     2     4     6     8
>> v=[8:-2:0]
v =
     8     6     4     2     0
```

5.2.7 Operations on Vectors and Matrices

Operations

`.* .\ ./ .^`	element-wise calculations
`\ /`	left and right division
`transpose(A)` or `A.'`	transpose of `A`
`ctranspose(A)` or `A'`	transpose of `A` (conjugated complex)
`inv(A)`	inverse of `A`
`det(A)`	determinant of `A`

Dimensions

`[M,N] = SIZE(A)`	dimension of matrix and vector
`M = SIZE(A,DIM)`	length of matrix of dimension `DIM`

Mathematical Functions for Vectors and Matrices

`sum(a)`	sum of vector elements
`prod(a)`	product of vector elements
`min(a)`	smallest vector element
`max(a)`	largest vector element
`sort(a)`	element in increasing order
`find(a)`	non-zero elements

Examples:

```
>> A=[1 2 3; 4 5 6; 7 8 9];
>> B=[1 2 3; 2 4 5; 3 7 8];
>> b=[2 4 6 8 10]';
>> v=0:2:8;

>> v*b
ans =
   160
>> v'*b'
ans =
     0     0     0     0     0
     4     8    12    16    20
     8    16    24    32    40
    12    24    36    48    60
    16    32    48    64    80
```

```
>> c=v+b';
>> c=v-b';
>> C=A+B;
>> C=A-B;
>> C=A*B;
>> c=A*v(1:3)'
c =
    16
    34
    52
>> c=v.*b'
c =
     0     8    24    48    80
```

Example:

```
>> A=[5 -3 2; -3 8 4; 2 4 -9];
>> b=[10;20;9];
>> x=A\b
x =
    3.4442
    3.1982
    1.1868
```

Solving the linear system of equations: $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$

$$\begin{aligned} 5x &= 3y - 2z + 10 \\ 8y + 4z &= 3x + 20 \\ 2x + 4y - 9z &= 9 \end{aligned}$$

5.2.8 Scripts and Functions

- Script:
  When you invoke a script, MATLAB simply executes the commands found in the file. Scripts can operate on existing data in the workspace, or they can create new data on which to operate. Although scripts do not return output arguments, any variables that they create remain in the workspace, to be used in subsequent computations.

- Function:
  Functions are .m-files that can accept input arguments and return output arguments. The name of the .m-file and of the function should be the same. Functions operate on variables within their own workspace, separate from the workspace you access at the MATLAB command prompt.

Both types of codes are stored as .m-file.

Structure of functions:

```
function [a_out,b_out] = name_of_function(a_in,b_in)
%         output list                     input list
%
% Description of function can be placed here
% by using the comment-operator '%'. This
% informations can be displyed by typing
% 'help name_of_function' at MATLAB-command-line.
.
.
a_out = a_in - a_out;
b_out = a_in + a_out;
.
.
```

Example: Computation of the length of a vector

```
function vlength = fvectorlength(vector)
%
% Computation of the length 'vlength' of
% a column vector 'vector'

vlength = sqrt(vector'*vector);
```

Calling the function:

```
>> fvectorlength([-1;3;5])
ans =
    5.9161
```

5.2.9 Relational and Logical Operators

`==` , `~=`	equal, not equal		
`<` , `<=`	less than, less than or equal		
`>` , `>=`	greater than, greater than or equal		
`~`	logical not	`&`	element-wise logical AND
`|`	element-wise logical OR	`xor`	logical EXCLUSIVE OR

More informations: type `help +` at MATLAB -commandline.

5.2.10 IF-/CASE Statements and Loops

`IF` expression (e.g. `a==1`)
  statements
`ELSEIF` expression
  statements
`ELSE`
  statements
`END`
	IF statement condition
	
	The ELSE and ELSEIF parts (optional)

`SWITCH` switch-expr (e.g. `id`)
  `CASE` case-expr, (e.g. `1`)
  statement, ..., statement
  `CASE` {case-expr1, case-expr2, case-expr3,...}
  statement, ..., statement
  `OTHERWISE`,
  statement, ..., statement
`END`
	SWITCH statement case

`FOR` variable = expr (e.g. `ii=1:10`) statement, ..., statement `END`	Repeat statements a specific number of times.
`WHILE` expression (e.g. `ii<imax`) statements `END`	Repeat statements an indefinite number of times.

5.2.11 Plotting Figures

Control the figure window

`figure` , `figure(no)`	create, activate figure with number `no`
`subplot(num_rows,num_cols,index)`	create a subplot e.g., `subplot(2,3,2)`: subplot with 2 rows and 3 column, activate 2^{nd} subplot
`gcf`	handle of active figure
`clf`	clear active figure
`delete(id)`	delete object with handle `id`
`close(index)`	delete figure window `index`
`close all`	delete all figure windows
`hold` <`on` | `off`>	keeping existent objects through new plots on/off

2-D Plot Commands

`plot( <x,> y <,plotstyle> ,...)`	plot, linear (<> = opt.)
`fplot(func,range)`	plot of explicit function
`line(x,y)`	Create line through coordinate-vectors `x,y`
`text(x,y,string)`	Place text `string` at position `x,y`
For more informations about 2-D Plots, call `help plot` at command-line.	

3-D Plot Commands

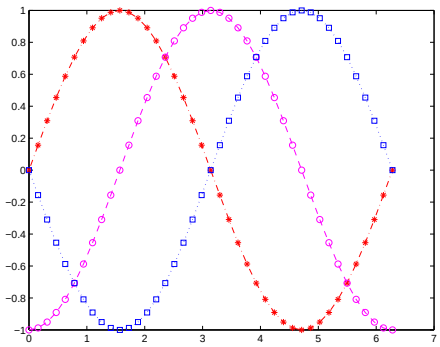
`plot3(x,y,z <,plotstyle> ,...)`	threedimensional plot
`sphere`	plot a sphere
`[x,y,z] = sphere`	store coordinates of a unit sphere to `x,y,z`
`line(x,y,z)`	Create line through coordinate-vectors `x,y,z`
`text(x,y,z,string)`	Place text `string` at position `x,y,z`

Plotstyles

colors:			lines and markes:		
`k` black	`r` red	`g` green	`-` solid	`o` circle	`.` points
`b` blue	`m` magenta	`w` white	`--` dashed	`*` star	`x` x-mark
`c` cyan	`y` yellow		`:` dotted	`+` plus	`-.` dashdot
More infos: `help plot`					

Examples:

```
>> figure(1)
>> clf
>> t=0:pi/20:2*pi;
>> plot(t,sin(t),'-.r*')
>> hold on
>> plot(t,(sin(t-pi/2)),'linestyle','--',...
   'marker','o','color','m')
>> plot(t,sin(t-pi),':bs')
>> hold off
```



Labelling of Axes and Figures

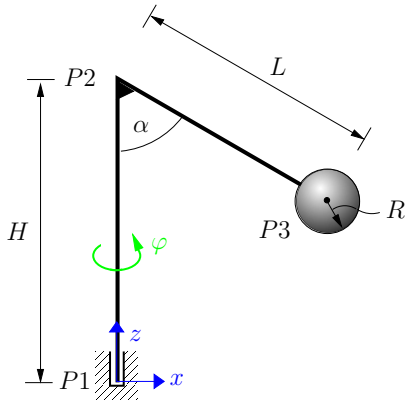
`axis([xmin,xmax,ymin,ymax <,zmin,zmax> ])`	scaling (min=`-inf` or max=`inf` → auto scal.)
`axis <on | off | auto | equal | square>`	several axis commands more: `help axis`
`grid <on | off>`	grid
`gca`	handle to current axis
`cla`	delete current axis
`xlabel(string)`	label of x-axis
`ylabel(string)`	label of y-axis
`zlabel(string)`	label of z-axis
`title(string)`	title of current axis
`legend(string_1,string_2,... <,pos>`	place legend `pos = 0,1,2,3,4,-1`

5.3 Exercises

5.3.1 Animation of a rotating cantilever

The cantilever with the length L is fixed at the top (point P2) of a vertical rod (height H). Both include an angle α. The vertical rod is bounded by a revolute joint at its base (point P1) with the z-axis as rotation axis. At the free end of the cantilever (point P3) a ball is attached and its radius is R.

Program a MATLAB-script (`.m`-file), which animates the rotation of the beside shown system about the vertical axis (z-axis) with angle φ.



5.3.2 Animation of a rotating cantilever using a subroutine

The MATLAB-script in Section 5.3.1 should be modified. Now a self-programed function should compute the rotated coordinates for the animation. The input-parameters should be the original coordinates and the rotation angle φ. The output of the function should be the rotated coordinates.

5.3.3 Source code

to Section 5.3.1

```
clear all; clc; close all;
%--- Geometrical Parameter ---
% Height of vertical rod
H = 10;
% Length of cantilever arm
L = 5;
% Radius of sphere
R = 1;
% Angle between vertical rod and cantilever arm
alpha = 90*pi/180;
% Series of angle of rotation
phi = linspace(0,2*pi,100);
%--- Basic Coordinates ---
% Uniform sphere (r=1)
[xk yk zk] = sphere;
% Scaling of sphere by R
xk = R.*xk;
yk = R.*yk;
zk = R.*zk;
% Points of veritcal rod's ends
P1 = [0 ; 0 ; 0];
P2 = P1 + [0 ; 0 ; H];
% Point of sphere's center in inital position
P30 = [ L * sin(alpha) ; 0 ; -L * cos(alpha)];
%--- Plot Procedure ---
% Create new figure with number 1
figure(1)
% Limits of axis
axis([-5  5 -5 5 0 15])
% Equal aspect ratio to all axis
axis equal
% Show grid
grid on
% Plot of vertical rod
line([P1(1) P2(1)],[P1(2) P2(2)],[P1(3) P2(3)] ,...
     'linewidth', 2 , 'color' , 'k');
%--- Begin of Animation ---
for istep = 1 : size(phi,2)
  % Delete previous plot objects if necessary
  if istep > 1
      % Sphere
      delete(id_kugel)
      % cantilever arm
      delete(id_line)
  end
  % Rotary matrix
  R = [ cos(phi(istep)) -sin(phi(istep)) 0;
        sin(phi(istep))  cos(phi(istep)) 0;
```

```
           0                   0         1];
  % Actual point of sphere's center
  P3 = P1 + P2 + R*P30;
  % Plot of sphere
  id_kugel = surface(P3(1)+xk,P3(2)+yk,P3(3)+zk);
  % Hold current graph
  hold on
  % Plot of cantilever arm
  id_line = line([P2(1) P3(1)],[P2(2) P3(2)],[P2(3) P3(3)] ,...
                 'linewidth', 2 , 'color' , 'k');
  % Update of screen
  drawnow
end
```

to Section 5.3.2

Replaced parts in the MATLAB-script:

```
...
  % Delete previous plot objects if necessary
  if istep > 1
      % Sphere
      delete(id_kugel)
      % cantilever arm
      delete(id_line)
  end
  %--- Using of new function frotation_z ---
  % Actual point of sphere's center
  P3 = P1 + P2 + frotation_z(phi(istep),P30);
  %-------------------------------------------
  % Plot of sphere
  id_kugel = surface(P3(1)+xk,P3(2)+yk,P3(3)+zk);

...
```

New function `frotation`:

```
function [xyz_new] = rotation_z(phi,xyz_old)
% Rotation of a point with the coordinates 'xyz'
% around the z-axis with angle 'phi'
%
% input:  phi     [1x1] ... angle
%         xyz_old [3x1] ... vector of coordinates
% output: xyz_new [3x1] ... vector of rotated coordinates

% Rotary matrix
R = [ cos(phi) -sin(phi) 0;
      sin(phi)  cos(phi) 0;
          0        0     1];
% Compute rotated coordinates
xyz_new = R * xyz_old;
```

6 Linear Finite Element Method

The Finite Element Method (FEM) is a general purpose procedure for engineering analysis of solids and structures as well as for computations of heat transfer and fluid mechanics, etc. The ancestor of the Finite Element Method is matrix structural analysis. The first formulations in discrete aeroelasticity date from the 1930s. A formal unification of force and displacement methods was then presented by Argyris and Kelsey (1954,1955). The direct stiffness method proposed by Turner (1959) can be seen as the first appearance of the hitherto unnamed Finite Element Method. A paper by Turner, Clough, Martin and Topp (1956), which applied this method to vibration analysis, is often considered the first reference in history to FEM.

The name “finite element", however, was established in a paper by Clough (1960) which attracted very little attention. There are many other early contributions concerning FEM, of which we cite here only Zienkiewicz and Cheung (1967). Contributions to the history of the method can be found e.g. in Felippa (2000) and Clough (2004).

We refer to Zienkiewicz and Taylor (2000), Bathe (1996), Stein *et al.* (2004a,b), Belytschko, Liu and Moran (2000), Oden (1972), Wriggers (2001), Cook, Malkus and Plesha (1989), Dhatt, Touzot and Cantin (1985) and Hughes (1987) as standard texts in this field.

6.1 Illustration of the Finite Element Concept

In solid-, structural- and fluid mechanical analysis in modern engineering technology, it is impossible to calculate exact (analytical) solutions due to the complexity of the (initial) boundary value problems. On account of their complexity, it is necessary to compute approximate solutions which converge in some manner to given (unknown) analytical solutions. For the explanation of the basic steps used for a finite element analysis we consider a truss structure as depicted in Figure 21 and restrict ourselves to geometrical and physical linear systems.

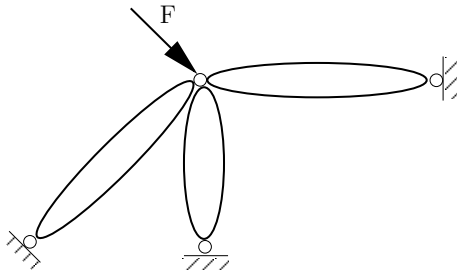


Figure 21: Physical domain $\mathcal{B}$ of a plane truss structure.

The typical steps for a finite element analysis are as follows:

i) idealization of the real system,

ii) division of the idealized structure into finite elements,

iii) approximation of the basic field variables (e.g. displacement-, pressure-, temperature-field) with appropriate ansatz functions,

iv) computation of the element matrizes,

v) assembly of the individual elements taking into account the Dirichlet boundary conditions;

vi) calculation of the solution to the final algebraic system of equations.

6.1.1 Idealization of the Physical System The actual physical system must be idealized and described by a mathematical model. This model can be a set of partial differential equations or a variational functional.

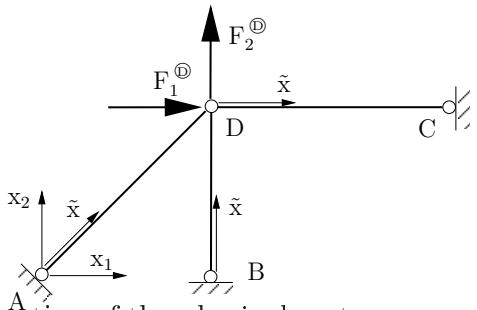


Figure 22: Idealization of the physical system as a plane truss structure, global coordinate system x_1, x_2 and parametrization of the individual trusses with $\tilde{x}$.

As an example, we assume in this simple system the existence of a total potential energy functional in the form

$$\Pi(\boldsymbol{u}) = \int_{\mathcal{B}} \frac{1}{2} \mathrm{EA} \left[\tilde{u}'(\tilde{x})\right]^2 \, \mathrm{d}\tilde{x} - F_1^{\textcircled{D}} \, u_1^{\textcircled{D}} - F_2^{\textcircled{D}} \, u_2^{\textcircled{D}} \, ,$$

with some additional Dirichlet boundary conditions at the points A, B and C (Figure 22). Here, EA characterizes the axial stiffness of the bar, $\tilde{u}(\tilde{x})$ the local displacement field and $\tilde{u}'(\tilde{x})$ the derivative of the local displacements with respect to the local coordinate $\tilde{x}$ of each individual bar. The last two terms in the above-mentioned total potential represent the work done by the load vector

$$\boldsymbol{F}^{\textcircled{D}} = (F_1^{\textcircled{D}}, F_2^{\textcircled{D}})^{\mathrm{T}} \tag{193}$$

acting at point (node) D. In the equilibrium state Π must be minimal:

$$\delta\Pi(u, \delta u) = \int_{\mathcal{B}} \mathrm{EA}\tilde{u}'(\tilde{x})\delta\tilde{u}'(\tilde{x}) \, \mathrm{d}\tilde{x} - F_1^{\textcircled{D}} \, \delta u_1^{\textcircled{D}} - F_2^{\textcircled{D}} \, \delta u_2^{\textcircled{D}} = 0, \tag{194}$$

where $\delta\tilde{u}$ is the virtual diplacement field of the individual trusses, and $\delta u_1^{\textcircled{D}}, \delta u_2^{\textcircled{D}}$ represent the horizontal and vertical virtual displacements of node D.

Division of the System into Finite Elements: The body under consideration $\mathcal{B}$ is approximated by $\mathcal{B}^{\mathrm{h}}$. Let $\mathtt{num_{ele}}$ denote the number of finite elements $\mathcal{B}^{\mathrm{e}}$, then we write

$$\mathcal{B} \approx \mathcal{B}^{\mathrm{h}} = \bigcup_{e=1}^{\mathtt{num_{ele}}} \mathcal{B}^{\mathrm{e}}.$$

This leads to the following representation of (194)

$$\delta\Pi(u,\delta u) = \sum_{e=1}^{num_{ele}} \underbrace{\int_{\mathcal{B}^e} EA\tilde{u}'(\tilde{x})\delta\tilde{u}'(\tilde{x})\, d\tilde{x}}_{=:\,\delta\Pi^{e,int}} - F_1^{\textcircled{D}}\,\delta u_1^{\textcircled{D}} - F_2^{\textcircled{D}}\,\delta u_2^{\textcircled{D}} = 0, \tag{195}$$

where $\delta\Pi^{e,int}$ is the contribution of a typical element to the first variation of the total potential energy. The division of the whole structure into individual elements is depicted in Figure 23. Here we have chosen three two-noded finite elements for simplicity.

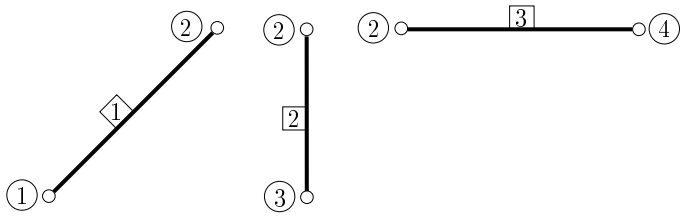


Figure 23: Division of the structure into three finite elements $\mathcal{B}^e$ for e = 1,2,3.

Approximation of the Basic Variables: The individual finite elements play a major role in this approximation method. Over the region $\mathcal{B}^e$ we interpolate the variable of interest, so we introduce the local coordinate system $\tilde{x}_1, \tilde{x}_2$ for each individual finite element $\mathcal{B}^e$ with length l^e, as seen in Figure 24. The approximation of the local displacements for the chosen two-noded element with the linear shape functions

$$\begin{aligned} N^{①}(\tilde{x}) &= \frac{1}{2} - \frac{\tilde{x}}{l^e} \quad \rightarrow \quad N^{①}(-\frac{l^e}{2}) = 1, \qquad N^{①}(+\frac{l^e}{2}) = 0, \\ N^{②}(\tilde{x}) &= \frac{1}{2} + \frac{\tilde{x}}{l^e} \quad \rightarrow \quad N^{②}(-\frac{l^e}{2}) = 0, \qquad N^{②}(+\frac{l^e}{2}) = 1, \end{aligned}$$

is given in terms of the local nodal displacements $\tilde{d}^{①}$ and $\tilde{d}^{②}$ as

$$\tilde{u}_h(\tilde{x}) = N^{①}\tilde{d}^{①} + N^{②}\tilde{d}^{②} = \underline{\mathbb{N}}\,\underline{\tilde{\boldsymbol{d}}}^e\,. \tag{196}$$

$\underline{\mathbb{N}}$ denotes the matrix of shape functions and $\underline{\tilde{\boldsymbol{d}}}^e$ the matrix of the discrete local displacements for a typical finite element; these matrices are given by

$$\underline{\mathbb{N}} = [\,N^{①},\, N^{②}\,] \quad \text{and} \quad \underline{\tilde{\boldsymbol{d}}}^e = \begin{bmatrix} \tilde{d}^{①} \\ \tilde{d}^{②} \end{bmatrix}.$$

Furthermore, we need the derivatives of the displacement interpolations with respect to $\tilde{x}$, see (195). The formal derivative of (196) yields

$$\frac{d\tilde{u}_h(\tilde{x})}{d\tilde{x}} = \tilde{u}_h'(\tilde{x}) = N^{①}_{,\tilde{x}}\,\tilde{d}^{①} + N^{②}_{,\tilde{x}}\,\tilde{d}^{②}.$$

In the use of the Finite Element Method it is common to arrange the derivatives of the shape functions into so-called $\underline{\mathbb{B}}$ -matrices to obtain expressions of the form

$$\tilde{u}_h'(\tilde{x}) = \underline{\mathbb{B}}\,\underline{\tilde{\boldsymbol{d}}}^e \quad \text{with} \quad \underline{\mathbb{B}} = [\,N^{①}_{,\tilde{x}},\, N^{②}_{,\tilde{x}}\,]. \tag{197}$$

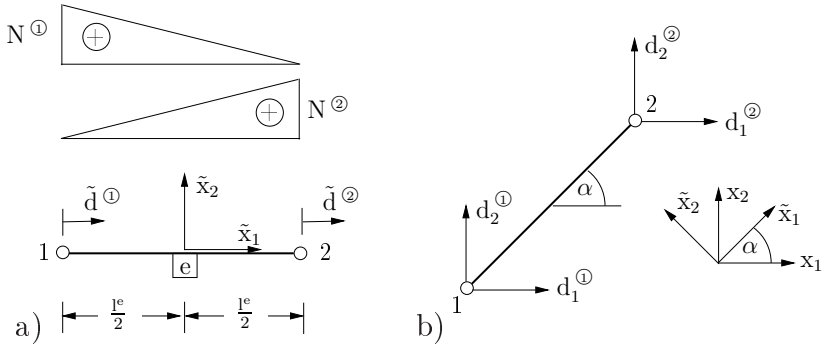


Figure 24: Finite element $\mathcal{B}^{\mathrm{e}}$ in the a) local and b) global coordinate system.

We now transform the local discrete nodal displacements $\underline{\tilde{\boldsymbol{d}}}^{\mathrm{e}} = (\tilde{\mathrm{d}}^{①}, \tilde{\mathrm{d}}^{②})^{\mathrm{T}}$ into the global nodal displacements $\underline{\boldsymbol{d}}^{\mathrm{e}} = (\mathrm{d}_1^{①}, \mathrm{d}_2^{①}, \mathrm{d}_1^{②}, \mathrm{d}_2^{②})^{\mathrm{T}}$, see Figure 24 for details. This transformation is accomplished by using the transformation matrix $\underline{\boldsymbol{T}}$ as shown below

$$\underline{\tilde{\boldsymbol{d}}}^{\mathrm{e}} = \underline{\boldsymbol{T}}\,\underline{\boldsymbol{d}}^{\mathrm{e}} \quad \rightarrow \quad \begin{bmatrix} \tilde{\mathrm{d}}^{①} \\ \tilde{\mathrm{d}}^{②} \end{bmatrix} = \left[\begin{array}{cc|cc} \cos\alpha & \sin\alpha & 0 & 0 \\ 0 & 0 & \cos\alpha & \sin\alpha \end{array}\right] \begin{bmatrix} \mathrm{d}_1^{①} \\ \mathrm{d}_2^{①} \\ \hline \mathrm{d}_1^{②} \\ \mathrm{d}_2^{②} \end{bmatrix} . \tag{198}$$

Inserting the transformation rule (198) into (197) yields

$$\tilde{\mathrm{u}}_{\mathrm{h}}'(\tilde{\mathrm{x}}) = \underline{\mathbb{B}}\,\underline{\boldsymbol{T}}\,\underline{\boldsymbol{d}}^{\mathrm{e}} .$$

Using the same interpolation for the virtual displacement fields $\delta\tilde{\mathrm{u}}(\tilde{\mathrm{x}})$ as for the displacement field, i.e.,

$$\delta\tilde{\mathrm{u}}_{\mathrm{h}}(\tilde{\mathrm{x}}) = \mathrm{N}^{①}\delta\tilde{\mathrm{d}}^{①} + \mathrm{N}^{②}\delta\tilde{\mathrm{d}}^{②} = \underline{\mathbb{N}}\,\delta\underline{\tilde{\boldsymbol{d}}}^{\mathrm{e}}, \tag{199}$$

the equation for the approximation of $\delta\tilde{\mathrm{u}}'$ is converted into the form

$$\delta\mathrm{u}_{\mathrm{h}}'(\tilde{\mathrm{x}}) = \underline{\mathbb{B}}\,\underline{\boldsymbol{T}}\,\delta\underline{\boldsymbol{d}}^{\mathrm{e}},$$

where $\delta\underline{\tilde{\boldsymbol{d}}}^{\mathrm{e}}$ and $\delta\underline{\boldsymbol{d}}^{\mathrm{e}}$ characterize the element vectors of discrete virtual displacements in the local and global coordinate systems, respectively. Inserting these relationships into $\delta\Pi^{\mathrm{e,int}}$ along with the element-wise constant matrix $\underline{\boldsymbol{T}}$ yields

$$\delta\Pi_{\mathrm{h}}^{\mathrm{e,int}} = \delta\underline{\boldsymbol{d}}^{\mathrm{eT}} \underbrace{\underline{\boldsymbol{T}}^{\mathrm{T}}\,\underline{\tilde{\boldsymbol{k}}}^{\mathrm{e}}\,\underline{\boldsymbol{T}}}_{=:\,\underline{\boldsymbol{k}}^{\mathrm{e}}} \boldsymbol{d}^{\mathrm{e}} \quad \text{with} \quad \underline{\tilde{\boldsymbol{k}}}^{\mathrm{e}} = \int_{\mathcal{B}^{\mathrm{e}}} \underline{\mathbb{B}}^{\mathrm{T}}\mathrm{EA}\underline{\mathbb{B}}\,\mathrm{d}\tilde{\mathrm{x}}, \tag{200}$$

where $\underline{\tilde{\boldsymbol{k}}}^{\mathrm{e}}$ and $\underline{\boldsymbol{k}}^{\mathrm{e}}$ are the element stiffness matrices in the local and global coordinate systems, respectively.

6.1.2 The Assembly Procedure Another major component of the Finite Element Method is the assembly procedure. For a descriptive motivation of this step we consider the algebraic expression

$$\delta\Pi_{\mathrm{h}} = \sum_{\mathrm{e}=1}^{\mathtt{num_{ele}}} \delta\underline{\boldsymbol{d}}^{\mathrm{eT}}\underline{\boldsymbol{k}}^{\mathrm{e}}\underline{\boldsymbol{d}}^{\mathrm{e}} - \mathrm{F}_1^{②}\delta\mathrm{d}_1^{②} - \mathrm{F}_2^{②}\delta\mathrm{d}_2^{②} = 0, \tag{201}$$

using the assignments

$$\{\mathrm{A},\mathrm{B},\mathrm{C},\mathrm{D}\} \quad \rightarrow \quad \{1,3,4,2\}$$

for the global nodes, compare Figures 22 and 23. Here we must take into account the global node numbers of the discretized system in order to assemble the global system of equations:

$$\delta\Pi_{\mathrm{h}} = \delta\underline{\boldsymbol{D}}^{\mathrm{T}}(\underline{\boldsymbol{K}}\,\underline{\boldsymbol{D}} - \underline{\boldsymbol{P}}) = 0 \tag{202}$$

where the global displacement vector is

$$\underline{\boldsymbol{D}} = (\mathrm{D}_1,\mathrm{D}_2,\mathrm{D}_3,\mathrm{D}_4,\mathrm{D}_5,\mathrm{D}_6,\mathrm{D}_7,\mathrm{D}_8)^{\mathrm{T}},$$

analogously, we define $\delta\underline{\boldsymbol{D}}$; the global load vector is then

$$\underline{\boldsymbol{P}} = (0,0,\mathrm{F}_1^{\textcircled{2}},\mathrm{F}_2^{\textcircled{1}},0,0,0,0)^{\mathrm{T}},$$

and the global stiffness matrix $\underline{\boldsymbol{K}}$ has to be assembled.

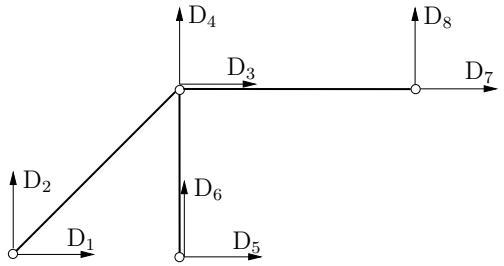


Figure 25: Global degrees of freedom $\mathrm{D}_1,\mathrm{D}_2,\mathrm{D}_3,\mathrm{D}_4,\mathrm{D}_5,\mathrm{D}_6,\mathrm{D}_7,\mathrm{D}_8$.

.

The arrangement of the actual nodal and virtual nodal displacements in the global vectors $\underline{\boldsymbol{D}}$ and $\delta\underline{\boldsymbol{D}}$ controls the assembly process. For further clarification we introduce the connectivity matrix

local node \ element	1	2	3
1	1	3	2
2	2	2	4

The connectivity matrix assigns the local nodes to the global ones (addressed in the gray shaded boxes). For example, the local nodes 1 and 2 of element 2 are the global nodes 3 and 2, respectively. Neglecting the displacement boundary conditions at this point, we obtain a global stiffness matrix $\underline{\boldsymbol{K}} \in \mathbb{R}^{8\times 8}$ because we have 4 global nodes, each of which is endowed with 2 degrees of freedom. The assembly process is denoted by

$$\underline{\boldsymbol{K}} = \mathop{\mathsf{A}}_{e=1}^{\mathrm{nele}} \underline{\boldsymbol{k}}^{\mathrm{e}}, \tag{203}$$

where $\mathop{\mathsf{A}}_{e=1}^{\mathrm{nele}}$ represents the assembling operator.

For the simple case depicted in Figures 22 and 23, we obtain the element stiffness matrices $\boldsymbol{k}^{\mathrm{e}}$ with respect to the global coordiante system as well as the element vectors for the virtual $\delta\underline{\boldsymbol{d}}^{\mathrm{e}}$ and actual $\underline{\boldsymbol{d}}^{\mathrm{e}}$ displacements for $\mathrm{e} = 1, 2, 3$ having the form

$$\delta\underline{\boldsymbol{d}}^{\boxed{1}} = \begin{bmatrix} \delta\mathrm{D}_1 \\ \delta\mathrm{D}_2 \\ \delta\mathrm{D}_3 \\ \delta\mathrm{D}_4 \end{bmatrix}, \quad \underline{\boldsymbol{k}}^{\boxed{1}} = \begin{bmatrix} \mathrm{k}^{\boxed{1}}_{11} & \mathrm{k}^{\boxed{1}}_{12} & \mathrm{k}^{\boxed{1}}_{13} & \mathrm{k}^{\boxed{1}}_{14} \\ \mathrm{k}^{\boxed{1}}_{21} & \mathrm{k}^{\boxed{1}}_{22} & \mathrm{k}^{\boxed{1}}_{23} & \mathrm{k}^{\boxed{1}}_{24} \\ \mathrm{k}^{\boxed{1}}_{31} & \mathrm{k}^{\boxed{1}}_{32} & \mathrm{k}^{\boxed{1}}_{33} & \mathrm{k}^{\boxed{1}}_{34} \\ \mathrm{k}^{\boxed{1}}_{41} & \mathrm{k}^{\boxed{1}}_{42} & \mathrm{k}^{\boxed{1}}_{43} & \mathrm{k}^{\boxed{1}}_{44} \end{bmatrix}, \quad \underline{\boldsymbol{d}}^{\boxed{1}} = \begin{bmatrix} \mathrm{D}_1 \\ \mathrm{D}_2 \\ \mathrm{D}_3 \\ \mathrm{D}_4 \end{bmatrix},$$

and

$$\delta\underline{\boldsymbol{d}}^{\boxed{2}} = \begin{bmatrix} \delta\mathrm{D}_5 \\ \delta\mathrm{D}_6 \\ \delta\mathrm{D}_3 \\ \delta\mathrm{D}_4 \end{bmatrix}, \quad \underline{\boldsymbol{k}}^{\boxed{2}} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \mathrm{k}^{\boxed{2}}_{22} & 0 & \mathrm{k}^{\boxed{2}}_{24} \\ 0 & 0 & 0 & 0 \\ 0 & \mathrm{k}^{\boxed{2}}_{42} & 0 & \mathrm{k}^{\boxed{2}}_{44} \end{bmatrix}, \quad \underline{\boldsymbol{d}}^{\boxed{2}} = \begin{bmatrix} \mathrm{D}_5 \\ \mathrm{D}_6 \\ \mathrm{D}_3 \\ \mathrm{D}_4 \end{bmatrix},$$

and

$$\delta\underline{\boldsymbol{d}}^{\boxed{3}} = \begin{bmatrix} \delta\mathrm{D}_3 \\ \delta\mathrm{D}_4 \\ \delta\mathrm{D}_7 \\ \delta\mathrm{D}_8 \end{bmatrix}, \quad \underline{\boldsymbol{k}}^{\boxed{3}} = \begin{bmatrix} \mathrm{k}^{\boxed{3}}_{11} & 0 & \mathrm{k}^{\boxed{3}}_{13} & 0 \\ 0 & 0 & 0 & 0 \\ \mathrm{k}^{\boxed{3}}_{31} & 0 & \mathrm{k}^{\boxed{3}}_{33} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \underline{\boldsymbol{d}}^{\boxed{3}} = \begin{bmatrix} \mathrm{D}_3 \\ \mathrm{D}_4 \\ \mathrm{D}_7 \\ \mathrm{D}_8 \end{bmatrix}.$$

Performing the summation in (201) with

$$\sum_{\mathrm{e}=1}^{\mathtt{num_{ele}}} \delta\underline{\boldsymbol{d}}^{\mathrm{eT}} \underline{\boldsymbol{k}}^{\mathrm{e}} \underline{\boldsymbol{d}}^{\mathrm{e}} = \delta\underline{\boldsymbol{D}}^{\mathrm{T}} \, \underline{\boldsymbol{K}} \, \underline{\boldsymbol{D}} \tag{204}$$

yields the explicit form

$$\underline{\boldsymbol{K}} = \begin{bmatrix} \mathrm{k}^{\boxed{1}}_{11} & \mathrm{k}^{\boxed{1}}_{12} & \mathrm{k}^{\boxed{1}}_{13} & \mathrm{k}^{\boxed{1}}_{14} & & & & \\ \mathrm{k}^{\boxed{1}}_{21} & \mathrm{k}^{\boxed{1}}_{22} & \mathrm{k}^{\boxed{1}}_{23} & \mathrm{k}^{\boxed{1}}_{24} & & & & \\ \mathrm{k}^{\boxed{1}}_{31} & \mathrm{k}^{\boxed{1}}_{32} & \mathrm{k}^{\boxed{1}}_{33} + \mathrm{k}^{\boxed{3}}_{11} & \mathrm{k}^{\boxed{1}}_{34} & 0 & 0 & \mathrm{k}^{\boxed{3}}_{13} & 0 \\ \mathrm{k}^{\boxed{1}}_{41} & \mathrm{k}^{\boxed{1}}_{42} & \mathrm{k}^{\boxed{1}}_{43} & \mathrm{k}^{\boxed{1}}_{44} + \mathrm{k}^{\boxed{2}}_{44} & 0 & \mathrm{k}^{\boxed{2}}_{42} & 0 & 0 \\ & & 0 & 0 & 0 & 0 & & \\ & & 0 & \mathrm{k}^{\boxed{2}}_{24} & 0 & \mathrm{k}^{\boxed{2}}_{22} & & \\ & & \mathrm{k}^{\boxed{3}}_{31} & 0 & & & \mathrm{k}^{\boxed{3}}_{33} & 0 \\ & & 0 & 0 & & & 0 & 0 \end{bmatrix}.$$

In the next step we account for the displacement (Dirichlet) boundary conditions. If we assume zero boundary displacements at nodes 1,3 and 4, we must modify the assembling

equations. To achieve this we delete the rows and columns numbered 1, 2, 5, 6, 7, 8 (associated with D_1, D_2, D_5, D_6, D_7 and D_8, respectively) of $\underline{\underline{\boldsymbol{K}}}$ as well as the assigned rows in $\underline{\boldsymbol{P}}$. The result is the reduced set of equations

$$\overline{\underline{\boldsymbol{D}}}^{\mathrm{T}}(\overline{\underline{\boldsymbol{K}}}\,\overline{\underline{\boldsymbol{D}}} - \overline{\underline{\boldsymbol{P}}}) = 0 \quad \forall\, \overline{\underline{\boldsymbol{D}}}^{\mathrm{T}} \quad \rightarrow \quad \overline{\underline{\boldsymbol{K}}}\,\overline{\underline{\boldsymbol{D}}} = \overline{\underline{\boldsymbol{P}}}, \tag{205}$$

with the explicit forms of the reduced matrices $\overline{\underline{\boldsymbol{K}}}, \overline{\underline{\boldsymbol{D}}}, \overline{\underline{\boldsymbol{P}}}$:

$$\begin{bmatrix} k^{\boxed{1}}_{33} + k^{\boxed{3}}_{11} & k^{\boxed{1}}_{34} \\ k^{\boxed{1}}_{43} & k^{\boxed{1}}_{44} + k^{\boxed{2}}_{44} \end{bmatrix} \begin{bmatrix} D_3 \\ D_4 \end{bmatrix} = \begin{bmatrix} F^{\textcircled{2}}_1 \\ F^{\textcircled{2}}_2 \end{bmatrix} \tag{206}$$

Solving the Systems of Equations: We now solve the reduced system of equations with respect to (D_3, D_4). In this simple situation we can compute the inverse of the coefficient matrix directly as shown below

$$\begin{bmatrix} D_3 \\ D_4 \end{bmatrix} = \frac{1}{\det\,[\overline{\underline{\boldsymbol{K}}}]} \begin{bmatrix} k^{\boxed{1}}_{44} + k^{\boxed{2}}_{44} & -k^{\boxed{1}}_{34} \\ -k^{\boxed{1}}_{43} & k^{\boxed{1}}_{33} + k^{\boxed{3}}_{11} \end{bmatrix} \begin{bmatrix} F^{\textcircled{2}}_1 \\ F^{\textcircled{2}}_2 \end{bmatrix} \tag{207}$$

where

$$\det\,[\overline{\underline{\boldsymbol{K}}}] = (k^{\boxed{1}}_{33} + k^{\boxed{3}}_{11})(k^{\boxed{1}}_{44} + k^{\boxed{2}}_{44}) - k^{\boxed{1}}_{43}\, k^{\boxed{1}}_{34}. \tag{208}$$

In general, well-established numerical techniques are used to solve the system of equations. We distinguish between direct methods, such as Gaussian direct elimination and Cholesky reduction, and iterative methods, such as conjugate gradient methods and relaxation schemes.

6.1.3 Remarks Concerning the Implementation For an efficient implementation we have to design more sophisticated procedures for the assembly process as indicated above. We therefore introduce an integer array, which indicates the locations of the entries of the element stiffness matrices

$$\underline{\boldsymbol{k}}^{\mathrm{e}} \in \mathbb{R}^{k^{\mathrm{e}}_{\mathtt{dim}} \times k^{\mathrm{e}}_{\mathtt{dim}}}$$

in the global stiffness matrix

$$\underline{\underline{\boldsymbol{K}}} \in \mathbb{R}^{\mathtt{num_{eq}} \times \mathtt{num_{eq}}}.$$

The dimension of the element stiffness matrix is governed by

$$k^{\mathrm{e}}_{\mathtt{dim}} = \mathtt{nodes_{ele}} \;*\; \mathtt{dof_{nodes}}, \tag{209}$$

where $\mathtt{nodes_{ele}}$ denotes the nodes per element and $\mathtt{dof_{nodes}}$ represents the global degrees of freedom per node. If we let $\mathtt{num_{dbc}}$ be the number of Dirichlet boundary conditions, then the dimension of the global stiffness matrix is given as $\mathtt{num_{eq}} \times \mathtt{num_{eq}}$, where

$$\mathtt{num_{eq}} = \mathtt{num_{nodes}} \;*\; \mathtt{dof_{nodes}} \;-\; \mathtt{num_{dbc}} \tag{210}$$

with the number of nodes of the whole system designated as $\mathtt{num_{nodes}}$.

Explanation of the Assembly Process: In order to explain the assembly process we consider the plane truss structure shown in Figure 26. Each individual truss element has two nodes

$$\mathtt{nodes_{ele}} = 2$$

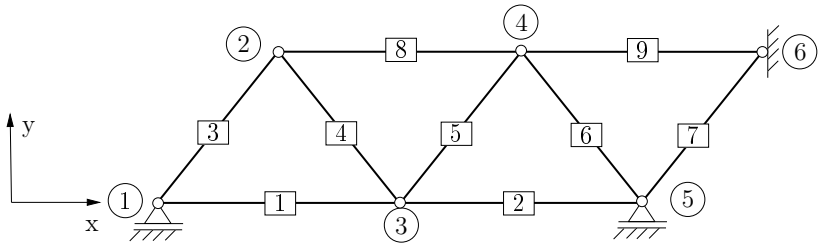


Figure 26: A plane truss structure with nine finite elements.

with two degrees of freedom (the displacements in x- and y-direction)

$$\texttt{dof}_{\texttt{nodes}} = 2.$$

The connectivity matrix of the nine truss elements' discretization is given as

element / local node	1	2	3	4	5	6	7	8	9
1	1	3	1	2	3	4	5	2	4
2	3	5	2	3	4	5	6	4	6

For the computer implementation we use the form

$$[\texttt{connectivity}] = \begin{bmatrix} 1\,3\,1\,2\,3\,4\,5\,2\,4 \\ 3\,5\,2\,3\,4\,5\,6\,4\,6 \end{bmatrix}. \tag{211}$$

The boundary value problem is discretized into nine truss elements, six nodes and four (zero) displacement boundary conditions, i.e.,

$$\texttt{num}_{\texttt{ele}} = 9, \quad \texttt{num}_{\texttt{nodes}} = 6, \quad \texttt{num}_{\texttt{dbc}} = 4.$$

Evaluating (209) and (210) yields

$$\underline{\boldsymbol{k}}^{\mathrm{e}} \in \mathbb{R}^{4\times 4} \quad \text{and} \quad \underline{\boldsymbol{K}} \in \mathbb{R}^{8\times 8}.$$

The assembly process is accomplished with an integer array

$$\texttt{assemble}_{\texttt{id}} \in \mathbb{R}^{\texttt{dof}_{\texttt{nodes}} \times \texttt{num}_{\texttt{nodes}}}$$

which assigns the location of the individual entries from the element stiffness matrix $[\boldsymbol{k}^{\mathrm{e}}]_{\mathrm{IJ}}$ into the global stiffness matrix $[\boldsymbol{K}]_{\mathrm{KL}}$. For the boundary value problem shown in Figure 26 we obtain

dof / global node i	1	2	3	4	5	6
${}^{\mathrm{i}}\mathrm{d}_1$	1	2	4	6	8	0
${}^{\mathrm{i}}\mathrm{d}_2$	0	3	5	7	0	0

.

As done we introduce an array $\texttt{assemble}_{\texttt{id}}$ for the computer implementation

$$[\texttt{assemble}_{\texttt{id}}] = \begin{bmatrix} 1\,2\,4\,6\,8\,0 \\ 0\,3\,5\,7\,0\,0 \end{bmatrix} . \tag{212}$$

In order to explain the assembling procedure

$$\underline{\boldsymbol{K}} = \mathop{\mathsf{A}}_{e=1}^{\text{nele}} \underline{\boldsymbol{k}}^{\text{e}} \tag{213}$$

we describe the procedure used to assemble elements in the connectivity matrix (211). First, we pre-allocate the global stiffness matrix $\underline{\boldsymbol{K}}$ with zeros, i.e.,

$$\text{K}_{\text{IJ}} = 0.0 \quad \text{for} \quad \text{I, J} = 1, ...8 \, .$$

Next, we pass through all elements to build the (local) element stiffness matrices and to update the global stiffness matrix. The iteration over the first element in the assembly process is denoted by

$$\underline{\boldsymbol{K}} = \mathop{\mathsf{A}}_{\text{e}} \underline{\boldsymbol{k}}^{\text{e}} \quad \text{for} \quad \text{e} = 1 \tag{214}$$

with the element stiffness matrix

$$\underline{\boldsymbol{k}}^{\boxed{1}} = \begin{array}{c|c} \begin{array}{cccc} 1 & 0 & 4 & 5 \end{array} & \text{global dof} \\ \hline \begin{bmatrix} \text{k}_{11}^{\boxed{1}} & \text{k}_{12}^{\boxed{1}} & \text{k}_{13}^{\boxed{1}} & \text{k}_{14}^{\boxed{1}} \\ \text{k}_{21}^{\boxed{1}} & \text{k}_{22}^{\boxed{1}} & \text{k}_{23}^{\boxed{1}} & \text{k}_{24}^{\boxed{1}} \\ \text{k}_{31}^{\boxed{1}} & \text{k}_{32}^{\boxed{1}} & \text{k}_{33}^{\boxed{1}} & \text{k}_{34}^{\boxed{1}} \\ \text{k}_{41}^{\boxed{1}} & \text{k}_{42}^{\boxed{1}} & \text{k}_{43}^{\boxed{1}} & \text{k}_{44}^{\boxed{1}} \end{bmatrix} & \begin{array}{c} 1 \\ 0 \\ 4 \\ 5 \end{array} \end{array} \, .$$

The first two rows and columns of $\underline{\boldsymbol{k}}^{\boxed{1}}$ are associated with the vertical and horizontal degrees of freedom of the global node 1; The third and fourth row and column are associated with the global node 3. For a further explanation see the first column in the connectivity array (211).

In order to update the global stiffness matrix we now use the information given in the $\texttt{assemble}_{\texttt{id}}$-array (212) and this yields the following update

$$\begin{aligned} &\text{K}_{11} \Longleftarrow K_{11} + k_{11}^{\boxed{1}} \, , \quad &&\text{K}_{14} \Longleftarrow K_{14} + k_{13}^{\boxed{1}} \, , \quad &&\text{K}_{15} \Longleftarrow K_{15} + k_{14}^{\boxed{1}} \, , \\ &\text{K}_{41} \Longleftarrow K_{41} + k_{31}^{\boxed{1}} \, , \quad &&\text{K}_{44} \Longleftarrow K_{44} + k_{33}^{\boxed{1}} \, , \quad &&\text{K}_{45} \Longleftarrow K_{45} + k_{34}^{\boxed{1}} \, , \\ &\text{K}_{51} \Longleftarrow K_{51} + k_{41}^{\boxed{1}} \, , \quad &&\text{K}_{54} \Longleftarrow K_{54} + k_{43}^{\boxed{1}} \, , \quad &&\text{K}_{55} \Longleftarrow K_{55} + k_{44}^{\boxed{1}} \, . \end{aligned}$$

For the next increment in the assembly loop, i.e.,

$$\underline{\boldsymbol{K}} = \mathop{\mathsf{A}}_{\text{e}} \underline{\boldsymbol{k}}^{\text{e}} \quad \text{for} \quad \text{e} = 2, \tag{215}$$

we update the global stiffness matrix by adding the entries of the element stiffness matrix

$$\underline{\boldsymbol{k}}^{\boxed{2}} = \begin{array}{c|cccc|c} & 4 & 5 & 8 & 0 & \text{global dof} \\ \hline & \mathrm{k}^{\boxed{2}}_{11} & \mathrm{k}^{\boxed{2}}_{12} & \mathrm{k}^{\boxed{2}}_{13} & \mathrm{k}^{\boxed{2}}_{14} & 4 \\ & \mathrm{k}^{\boxed{1}}_{21} & \mathrm{k}^{\boxed{2}}_{22} & \mathrm{k}^{\boxed{2}}_{23} & \mathrm{k}^{\boxed{2}}_{24} & 5 \\ & \mathrm{k}^{\boxed{2}}_{31} & \mathrm{k}^{\boxed{2}}_{32} & \mathrm{k}^{\boxed{2}}_{33} & \mathrm{k}^{\boxed{1}}_{34} & 8 \\ & \mathrm{k}^{\boxed{2}}_{41} & \mathrm{k}^{\boxed{2}}_{42} & \mathrm{k}^{\boxed{2}}_{43} & \mathrm{k}^{\boxed{1}}_{44} & 0 \end{array} .$$

to the respective destination storage space in $\underline{\boldsymbol{K}}$. Element 2 has global nodes 3 and 5: see the second column in the connectivity matrix (211). Furthermore, global node number 3 is associated with the fourth and fifth global degree of freedom whereas global node number 5 makes only one contribution to the whole system of equations: namely, the eighth one. This information is saved in the third and fifth column of the $\texttt{assemble}_{\texttt{id}}$-array (212). In detail we perform the following updates for the second element:

$$\begin{aligned} &\mathrm{K}_{44} \Longleftarrow K_{44} + k^{\boxed{2}}_{11} , \quad \mathrm{K}_{45} \Longleftarrow K_{45} + k^{\boxed{2}}_{12} , \quad \mathrm{K}_{48} \Longleftarrow K_{48} + k^{\boxed{2}}_{13} , \\ &\mathrm{K}_{54} \Longleftarrow K_{54} + k^{\boxed{2}}_{21} , \quad \mathrm{K}_{55} \Longleftarrow K_{55} + k^{\boxed{2}}_{22} , \quad \mathrm{K}_{58} \Longleftarrow K_{58} + k^{\boxed{2}}_{23} , \\ &\mathrm{K}_{84} \Longleftarrow K_{84} + k^{\boxed{2}}_{31} , \quad \mathrm{K}_{85} \Longleftarrow K_{85} + k^{\boxed{2}}_{32} , \quad \mathrm{K}_{88} \Longleftarrow K_{88} + k^{\boxed{2}}_{33} . \end{aligned}$$

The steps are repeated until the iteration over the elements is finished.

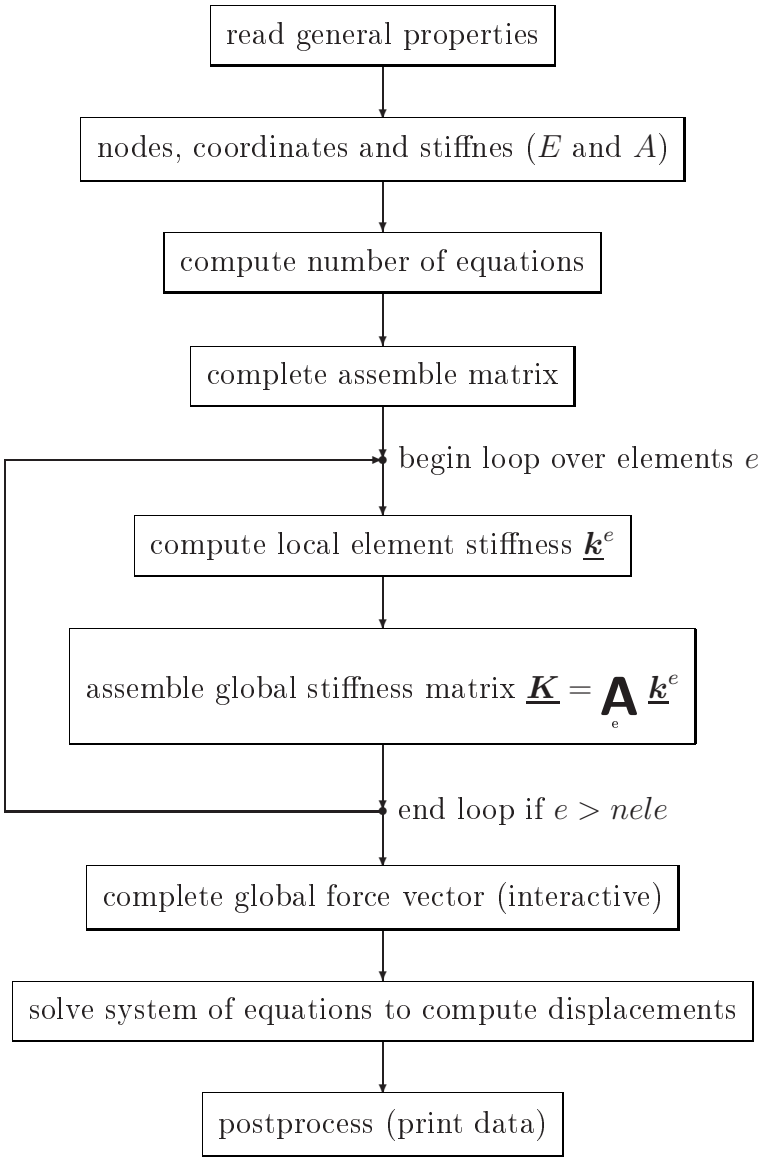


Figure 27: Workflow of `PlaneTrussElementStiffness.m`

A Simple Plane Truss FEM-Code: To continue further into the main procedures, we consider a program-listing of a rudimentary FEM program for plane truss structures written in MATLAB:

```
% MAIN PROGRAM
%-------------------------------------------------------------------
% FEM for simple plane struss structures
%-------------------------------------------------------------------
clear all
close all
%-read general properties-------------------------------------------
num_ele   = input('give the total number of elements. num_ele = ');
num_nodes = input('give the total number of nodes.  num_nodes = ');
disp(' ');
%-define nodes, coordinates and stiffnes (E and A) for elements------
for i = 1:num_ele
    fprintf('element no.%g\n',i);
    connectivity(1,i) = input('  global node for 1st local node = ');
    connectivity(2,i) = input('  global node for 2nd local node = ');
    elementdata(i,:)  = input('  element data           [ E A ] = ');
    disp(' ');
end
for i = 1:num_nodes
    fprintf('node no.%g\n',i);
    X(:,i) = input('  global node coordinates [ x y ] = ')';
    disp(' ');
end
%-compute number of equations--------------------------------------
disp('give the total number of displacement boundary conditions')
num_bc = input('                                        num_bc = ');
num_eq = num_nodes*2 - num_bc;
%-assemble matrix---------------------------------------------------
assembleid = -ones(2,num_nodes);
disp('    input of global dofs with zero boundary displacements')
disp('                                    for i from 1 to num_bc')
for i = 1:num_bc
    node = ...
    input('        global node of constrained degree of freedom = ');
    dof = ...
    input('          degree of freedom to constrained (1=x,2=y) = ');
    assembleid(dof,node) = 0;
end
idof = 0;
for i = 1:num_nodes
    for j = 1:2
        if assembleid(j,i) ~= 0
            idof = idof + 1;
            assembleid(j,i) = idof;
        end
    end
end
```

```
%-compute global stiffnes matrix-------------------------------------
K = zeros(num_eq,num_eq);
for i = 1:num_ele
    k = PlaneTrussElementStiffness(elementdata(i,1), ...
    elementdata(i,2),X(:,connectivity(1,i)),X(:,connectivity(2,i)));
    I = assembleid(:,connectivity(1,i));
    J = assembleid(:,connectivity(2,i));
    IJ = find(( [ I J ] ) ~= 0);
    K = PlaneTrussAssemble(K,k,I,J,IJ);
end
disp(' ');
%-global force vector-------------------------------------------------
R = zeros(num_eq,1);
disp('give the total number of nodes with non-zero load vectors')
num_loads = ...
          input('                                    num_loads = ');
disp('             input of load vectors for i from 1 to num_loads')
for i = 1:num_loads
    lbc = input('  no. of global node with non-zero load vector = ');
    if assembleid(2*lbc-1) ~= 0 & assembleid(2*lbc) ~= 0
        R(assembleid(2*lbc-1)) = ...
        input('                   force in global x-direction = ');
        R(assembleid(2*lbc))   = ...
        input('                   force in global y-direction = ');
    elseif assembleid(2*lbc-1) ~= 0
        R(assembleid(2*lbc-1)) = ...
        input('                   force in global x-direction = ');
    elseif assembleid(2*lbc) ~= 0
        R(assembleid(2*lbc))   = ...
        input('                   force in global y-direction = ');
    elseif assembleid(2*lbc-1) == 0 & assembleid(2*lbc) == 0
        disp('all degrees of freedom in this node are constrained!');
        pause(1);
    end
end
%-displacement computed by Gaussian elimination----------------------
u = K\R;
%-print the data------------------------------------------------------
disp(' ');disp('element connectivity table');disp(' ');
disp('  ============================')
disp('  |  e  |  Node i  |  Node j  |')
disp('  ============================')
for i = 1:num_ele
    fprintf('  |  %g  |     %g    |     %g    |', ...
    i,connectivity(1,i),connectivity(2,i))
    disp(' ');
end
disp('  ============================')
disp(' ');disp('global stiffness matrix'); K
disp(' ');disp('global force vector    '); R
disp(' ');disp('displacement vector    '); u
```

`PlaneTrussElementStiffness.m` is the subroutine used to calculate the element stiffness matrix in the global coordinate system. The input values are the modulus of elasticity `E`, the area of cross-section `A` and the coordinates

```
node_i(1) = x(1)
node_i(2) = y(1)
node_j(1) = x(2)
node_j(2) = y(2)
```

for the individual nodes i and j of the element, see Figure 28.

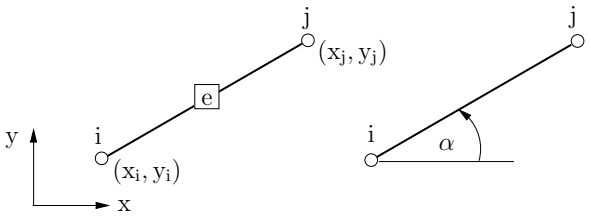


Figure 28: Interpretation of the variables occuring in the subroutine `PlaneTrussElementStiffness.m`

The output variable is the element stiffness matrix

$$\underline{\boldsymbol{k}}^{\mathrm{e}} = \underline{\boldsymbol{T}}^{\mathrm{T}}\, \underline{\tilde{\boldsymbol{k}}}^{\mathrm{e}}\, \underline{\boldsymbol{T}} \quad \text{with} \quad \underline{\tilde{\boldsymbol{k}}}^{\mathrm{e}} = \int_{\mathcal{B}^{\mathrm{e}}} \underline{\mathbb{B}}^{\mathrm{T}} \mathrm{EA} \underline{\mathbb{B}}\, \mathrm{d}\tilde{\mathrm{x}}$$

in the global x,y-coordinate system.

```
function y = PlaneTrussElementStiffness(E,A,node_i,node_j)
%-compute element stiffnes matrix-----------------------------------
L = sqrt((node_j(1) - node_i(1))^2 + (node_j(2) - node_i(2))^2);
if node_j(1) == node_i(1)
    if node_i(2) > node_j(2)
        alpha = -pi/2;
    elseif node_i(2) < node_j(2)
        alpha = pi/2;
    end
else
    alpha = atan((node_j(2) - node_i(2))/(node_j(1) - node_i(1)));
end
C = cos(alpha);
S = sin(alpha);
y = E*A/L*[ C*C  C*S -C*C -C*S ;
            C*S  S*S -C*S -S*S ;
           -C*C -C*S  C*C  C*S ;
           -C*S -S*S  C*S  S*S ];
```

The subroutine `PlaneTrussAssemble.m` is the control routine for the assembly process

$$\underline{\boldsymbol{K}} = \mathop{\mathsf{A}}_{e=1}^{\mathrm{nele}} \underline{\boldsymbol{k}}^{\mathrm{e}}.$$

If the element has only one active degree of freedom we utilize `Assemble1.m`, if the element has two active degrees of freedom we utilize `Assemble2.m`, and so on.

```
function y = PlaneTrussAssemble(K,k,I,J,IJ)
%-------------------------------------------------------------------
%Input:  element stiffnes matrix
%Output: updated global stiffnes matrix
%-------------------------------------------------------------------
if size(find([ I' J' ] ~= 0),2)     == 1
    K = Assemble1(K,k,I,J,IJ);
elseif size(find([ I' J' ] ~= 0),2) == 2
    K = Assemble2(K,k,I,J,IJ);
elseif size(find([ I' J' ] ~= 0),2) == 3
    K = Assemble3(K,k,I,J,IJ);
elseif size(find([ I' J' ] ~= 0),2) == 4
    K = Assemble4(K,k,I,J,IJ);
end
y = K;


function y = Assemble1(K,k,I,J,IJ)
%-------------------------------------------------------------------
% Assembly routine for element with one active dof
i = find([ I' J' ] ~= 0);
switch i
    case 1,  a = I(1);
    case 2,  a = I(2);
    case 3,  a = J(1);
    case 4,  a = J(2);
end
K(a,a) = K(a,a) + k(IJ,IJ);
y = K;


function y = Assemble2(K,k,I,J,IJ)
%-------------------------------------------------------------------
% Assembly routine for element with two active dofs
if size(find(I ~= 0),1) == 2
    a = I(1);
    b = I(2);
elseif size(find(J ~= 0),1) == 2
    a = J(1);
    b = J(2);
elseif (size(find(I ~= 0),1) == 1) & (size(find(J ~= 0),1) == 1)
    a = I(find(I ~= 0));
    b = J(find(J ~= 0));
end
i = IJ(1);
j = IJ(2);
K(a,a) = K(a,a) + k(i,i);
K(a,b) = K(a,b) + k(i,j);
K(b,a) = K(b,a) + k(j,i);
K(b,b) = K(b,b) + k(j,j);
y = K;
```

```
function y = Assemble3(K,k,I,J,IJ)
%-------------------------------------------------------------------
% Assembly routine for element with three active dofs
if size(find(I ~= 0),1) == 1
    a = I(find(I ~= 0));
    b = J(1);
    c = J(2);
elseif size(find(J ~= 0),1) == 1
    a = I(1);
    b = I(2);
    c = J(find(J ~= 0));
end
K(a,a) = K(a,a) + k(IJ(1),IJ(1));
K(a,b) = K(a,b) + k(IJ(1),IJ(2));
K(a,c) = K(a,c) + k(IJ(1),IJ(3));
K(b,a) = K(b,a) + k(IJ(2),IJ(1));
K(b,b) = K(b,b) + k(IJ(2),IJ(2));
K(b,c) = K(b,c) + k(IJ(2),IJ(3));
K(c,a) = K(c,a) + k(IJ(3),IJ(1));
K(c,b) = K(c,b) + k(IJ(3),IJ(2));
K(c,c) = K(c,c) + k(IJ(3),IJ(3));
y = K;

function y = Assemble4(K,k,I,J,IJ)
%-------------------------------------------------------------------
% Assembly routine for element with four active dofs
a = I(1);
b = I(2);
c = J(1);
d = J(2);
K(a,a) = K(a,a) + k(1,1);
K(a,b) = K(a,b) + k(1,2);
K(a,c) = K(a,c) + k(1,3);
K(a,d) = K(a,d) + k(1,4);
K(b,a) = K(b,a) + k(2,1);
K(b,b) = K(b,b) + k(2,2);
K(b,c) = K(b,c) + k(2,3);
K(b,d) = K(b,d) + k(2,4);
K(c,a) = K(c,a) + k(3,1);
K(c,b) = K(c,b) + k(3,2);
K(c,c) = K(c,c) + k(3,3);
K(c,d) = K(c,d) + k(3,4);
K(d,a) = K(d,a) + k(4,1);
K(d,b) = K(d,b) + k(4,2);
K(d,c) = K(d,c) + k(4,3);
K(d,d) = K(d,d) + k(4,4);
y = K;
```

In order to clearly explain the handling of the small program we will walk through each step while explaining the relevant details. Below we consider a descriptive example of the plane truss structure as depicted in Figure 29a, which is discretized with two elements and three global nodes: see Figure 29b.

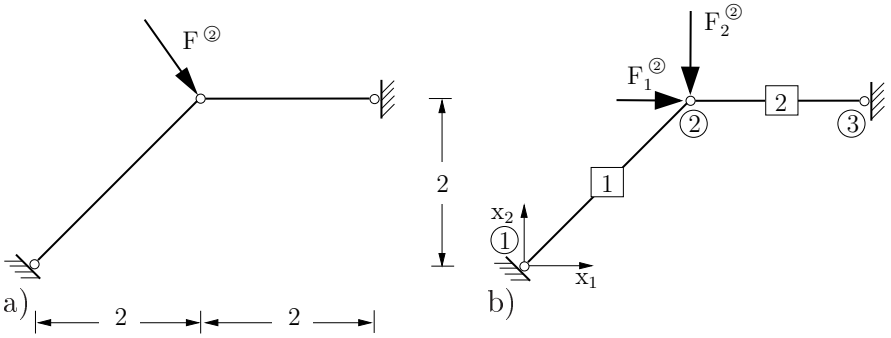


Figure 29: Plane truss structure: a) Dimensions (in meters) and boundary conditions, b) discretization of the structure.

Consequently, after starting the program, we have to apply the following input:

```
>> PlaneTruss
give the total number of elements. num_ele = 2
give the total number of nodes.  num_nodes = 3
```

In the next step we provide information about the truss' connectivity and stiffness. Element 1 consists of the global nodes 1 and 2; element 2 has the nodes 2 and 3. Thus the connectivity matrix is given as

local node \ element	1	2
1	1	2
2	2	3

.

For numerical analysis Young's modulus is set to $\mathrm{E} = 10000\,\mathrm{kN/m^2}$ and the cross sections for elements 1 and 2 are set as

$$0.02\,\mathrm{m}^2 \quad \text{and} \quad 0.015\,\mathrm{m}^2,$$

respectively. Accordingly, the next entries are entered as

```
element no.1
  global node for 1st local node = 1
  global node for 2nd local node = 2
  element data           [ E A ] = [10000 0.02]

element no.2
  global node for 1st local node = 2
  global node for 2nd local node = 3
  element data           [ E A ] = [10000 0.015]
```

Next we put in the (x_1, x_2) coordinates of all nodes: here we use the notation $x = x_1$ and $y = x_2$.

```
node no.1
  global node coordinates [ x y ] = [0 0]

node no.2
  global node coordinates [ x y ] = [2 2]

node no.3
  global node coordinates [ x y ] = [4 2]
```

The total number of displacement boundary conditions is four: the horizontal and vertical displacements of the nodes 1 and 3 are set to zero. We apply

$$d_1^{①} = d_2^{①} = d_1^{③} = d_2^{③} = 0,$$

or equivalently

$$D_1 = D_2 = D_5 = D_6 = 0.$$

The input values are:

```
give the total number of displacement boundary conditions
                                        num_bc = 4
    input of global dofs with zero boundary displacements
                                    for i from 1 to num_bc
        global node of constrained degree of freedom = 1
          degree of freedom to constrained (1=x,2=y) = 1
        global node of constrained degree of freedom = 1
          degree of freedom to constrained (1=x,2=y) = 2
        global node of constrained degree of freedom = 3
          degree of freedom to constrained (1=x,2=y) = 1
        global node of constrained degree of freedom = 3
          degree of freedom to constrained (1=x,2=y) = 2
```

As shown in the previous figures a local vector $\boldsymbol{F}^{②}$ is acting on node 2 where

$$\boldsymbol{F}^{②} = [F_1^{②}, F_2^{②}]^{\mathrm{T}} = [0.01\,\mathrm{kN}, -0.05\,\mathrm{kN}]^{\mathrm{T}}.$$

```
give the total number of nodes with non-zero load vectors
                                      num_loads = 1
             input of load vectors for i from 1 to num_loads
  no. of global node with non-zero load vector = 2
                  force in global x-direction = 0.01
                  force in global y-direction = -0.05
```

The first output is the element connectivity table. Below we have changed the arrangement of the rows and columns

```
============================
|  e  |  Node i  |  Node j  |
============================
|  1  |     1    |     2    |
|  2  |     2    |     3    |
============================
```

Thereafter the global stiffness matrix $\underline{\boldsymbol{K}}$ and the global force vector $\underline{\boldsymbol{P}}$ are printed

$$\underline{\boldsymbol{K}} = \begin{bmatrix} 110.3553 & 35.3553 \\ 35.3553 & 35.3553 \end{bmatrix}, \quad \underline{\boldsymbol{P}} = \begin{bmatrix} 0.0100 \\ -0.0500 \end{bmatrix}$$

The solution vector $\underline{\boldsymbol{u}}$ of the system of equations is obtained by using a Gaussian elimination:

$$\begin{bmatrix} \mathrm{D}_3 \\ \mathrm{D}_4 \end{bmatrix} = \begin{bmatrix} 0.0008 \\ -0.0022 \end{bmatrix}.$$

To verify this solution we consider the normal forces and the external loads acting on the global node 2, and evaluate the equilibrium conditions as shown in Figure 30.

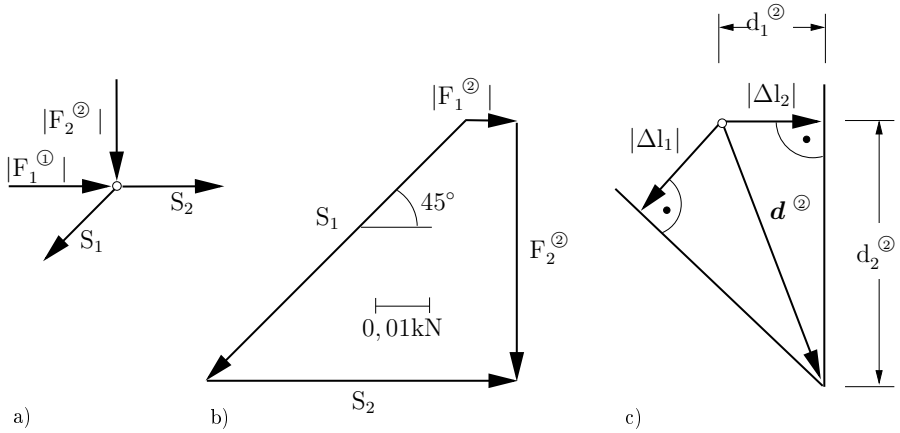


Figure 30: a) Normal forces $\mathrm{S}_1, \mathrm{S}_2$ and load vector $\boldsymbol{F}$ acting on node 2; b) polygon of forces for the free-body diagram; c) displacement of node 2.

As illustrated we obtain from Figure 30_b

$$\mathrm{S}_1 = -0.07071\,\mathrm{kN} \qquad \mathrm{S}_2 = -0.06\,\mathrm{kN}.$$

From these quantities we obtain the elongations of the rods $\Delta \mathrm{l}_\mathrm{i} = \mathrm{S}_\mathrm{i}\mathrm{l}_\mathrm{i}/(\mathrm{EA}_\mathrm{i})$:

$$\Delta \mathrm{l}_1 = -0.001\,\mathrm{m} \qquad \Delta \mathrm{l}_2 = -0.0008\,\mathrm{m}.$$

Evaluation of these relationships in Figure 30c, where $\Delta \mathrm{l}_1$ and $\Delta \mathrm{l}_2$ are the projections of the resulting displacement $\underline{\mathbf{d}}^{②}$ of node 2 onto the center lines of the individual rods, results in the displacement values

$$\mathrm{d}_1^{②} = 0.0008 \quad \text{and} \quad \mathrm{d}_2^{②} = -0.0022,$$

which agree with the numerical results.

♠ *Exercises* for 6.1.3

Compute the global stiffness matrix, the global load vector and the displacements of the plane truss structure shown in Figure (31). The dimensions are given in meters, all rods have a cross-section area of $0.02\,\mathrm{m}^2$ and the Young's modulus is $10000\,\mathrm{kN/m}^2$. For the calculation use the discretization as shown in Figure (32).

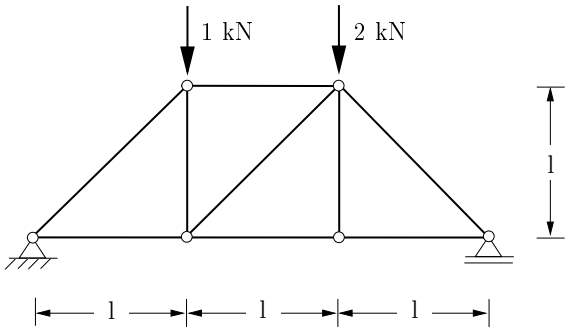


Figure 31: A plane truss structure with two vertical loads.

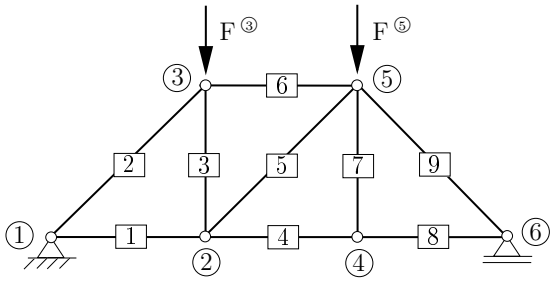


Figure 32: Discretization of the plane truss structure with nine finite elements and six global nodes.

♠

6.2 Generalizations

In Section 6.1 we motivated the finite element concept by considering a simple plane truss structure. Let us now consider a two-/three-dimensional domain $\mathcal{B}$, parameterized by the coordinates $\boldsymbol{x}$. As outlined before, the basic concept of the Finite Element Method is the subdivision of the spatial domain $\mathcal{B}$ into $\texttt{num}_{\texttt{ele}}$ non-overlapping finite elements $\mathcal{B}^{\mathrm{e}}$, which generally have a simple geometry. Subsequently, we choose some appropriate shape functions for the as yet unknown state variables. Thus, we get a geometrical approximation $\mathcal{B}^h$ of the original body, i.e.,

$$\mathcal{B} \approx \mathcal{B}^{\mathrm{h}} = \bigcup_{\mathrm{e}=1}^{\texttt{num}_{\texttt{ele}}} \mathcal{B}^{\mathrm{e}} \,. \tag{216}$$

A typical discretization of a two-dimensional body with triangular finite elements $\mathcal{B}^e$ is depicted in Figure 33a.

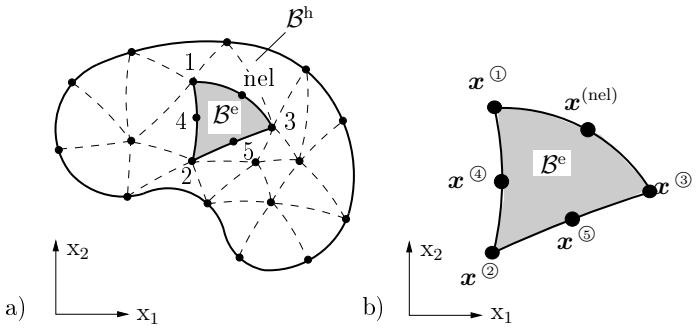


Figure 33: a) Discretization of the body $\mathcal{B}$ with triangular finite elements $\mathcal{B}^e$. b) A typical triangular finite element with six nodes per element ($\texttt{nel} = 6$).

A single element has $\texttt{nel}$ nodal points, see Figure 33b, which are used to define of the element geometry and which also carry the nodal degrees of freedom in the so-called isoparametric representation. The degrees of freedom are defined as the nodal values of the basic state-field variables. For a linear problem we obtain the (linear) relation

$$\underline{\mathbf{K}}\,\underline{\mathbf{D}} = \underline{\mathbf{P}} \tag{217}$$

between the global state vector $\underline{\mathbf{D}}$, and the global conjugate vector $\underline{\mathbf{P}}$. The global stiffness matrix $\underline{\mathbf{K}}$ is calculated by the assembling process

$$\underline{\mathbf{K}} = \mathop{\mathsf{A}}_{e\,=\,1}^{\mathrm{nele}} \underline{\mathbf{k}}^{\mathrm{e}} \,, \tag{218}$$

whereas the global state vector is identified by a reassignment step of the element state vectors $\underline{\mathbf{d}}^{\mathrm{e}}$. Depending on the physical problem the state and conjugate vectors may have the following representations:

physical problem:	state vector:	conjugate vector:
heat conduction	temperature	heat flux
incompressible potential flow	velocity-potential	volume flow
solid mechanics	displacements	mechanical force
electrostatics	electric potential	charge density

In the sequel, we discuss some well-known elements used for the approximation of basic scalar-valued and vector-valued field variables $\vartheta(\boldsymbol{x})$ and $\boldsymbol{u}(\boldsymbol{x})$, respectively. For the first case we denote the approximation of field variable of one typical element by $\vartheta^{\mathrm{h}}(\boldsymbol{x})$ and set

$$\vartheta^{\mathrm{h}} = \sum_{\mathrm{I}=1}^{\mathtt{nel}} \mathrm{N}^{\mathrm{I}}(\bullet)\, \mathrm{d}^{\mathrm{I}} = \underline{\mathbb{N}}(\bullet)\, \underline{\mathbf{d}}^{\mathrm{e}} \,, \tag{219}$$

where N_{I} characterizes the shape function associated with the node I, d^{I} the discrete nodal value at node I, $\underline{\boldsymbol{N}}$ the shape function matrix

$$\underline{\mathbb{N}} = [\mathrm{N}^{①}, \mathrm{N}^{②}, ..., \mathrm{N}^{(\mathrm{nel})}] \tag{220}$$

and $\underline{\boldsymbol{d}}^{\mathrm{e}}$ the element-state vector

$$\underline{\mathbf{d}}^{\mathrm{e}} = [\mathrm{d}^{①}, \mathrm{d}^{②}, ..., \mathrm{d}^{(\mathtt{nel})}]^{\mathrm{T}}. \tag{221}$$

An interpolation of a two-dimensional vectorial field variable $\boldsymbol{u}(\boldsymbol{x})$ yields

$$\begin{bmatrix} \mathrm{u}_1^{\mathrm{h}} \\ \mathrm{u}_2^{\mathrm{h}} \end{bmatrix} = \sum_{\mathrm{I}=1}^{\mathtt{nel}} \mathrm{N}^{\mathrm{I}}(\bullet) \begin{bmatrix} \mathrm{d}_1^{\mathrm{I}} \\ \mathrm{d}_2^{\mathrm{I}} \end{bmatrix}, \tag{222}$$

or, in a more compact notation

$$\boldsymbol{u}^{\mathrm{h}} = \sum_{\mathrm{I}=1}^{\mathtt{nel}} \mathrm{N}^{\mathrm{I}}(\bullet)\, \underline{\mathbf{d}}^{\mathrm{I}} = \underline{\mathbb{N}}(\bullet)\, \underline{\mathbf{d}}^{\mathrm{e}} \,. \tag{223}$$

The matrix containing the shape functions is organized as follows

$$\underline{\mathbb{N}} = \begin{bmatrix} \mathrm{N}^{①} & 0 & \mathrm{N}^{②} & 0 & ... & \mathrm{N}^{(\mathrm{nel})} & 0 \\ 0 & \mathrm{N}^{①} & 0 & \mathrm{N}^{②} & ... & 0 & \mathrm{N}^{(\mathrm{nel})} \end{bmatrix}, \tag{224}$$

and the element-state variable vector appears as

$$\underline{\mathbf{d}}^{\mathrm{e}} = [\mathrm{d}_1^{①}, \mathrm{d}_2^{①}, \mathrm{d}_1^{②}, \mathrm{d}_2^{②}, ..., \mathrm{d}_1^{(\mathtt{nel})}, \mathrm{d}_2^{(\mathtt{nel})}]^{\mathrm{T}}. \tag{225}$$

For the construction of straight-sided triangle and tetrahedra elements it might be useful to work with dimensionless natural coordinates λ. In this section we can use analytical expressions for the integration of polynomials in λ over the finite element domain $\mathcal{B}^{\mathrm{e}}$. Note that the shape functions are general, but the analytical integration formulas are restricted to the straight-sided elements. If we are interested in the construction of geometrically more flexible elements, then an isoparametric representation is a useful tool; viz. we use the identical approximation of the element geometry and the unknown field variables. In this framework it is easy to construct shape functions for higher-order elements with curved boundaries satisfying consistency requirements (interelement continuity and completeness). Furthermore, the integration of field quantities over the element domain is performed via numerical quadrature. In order to achieve convergence of refined meshes, i.e. wherein the finite element solution should tend toward the analytical solution to the underlying mathematical model, several requirements have to be fulfilled. These are completeness, compatibility and stability of the elements, see Hinton and Owen (1981).

Completeness: Element shape functions must represent the highest complete polynomial in cartesian coordinates. In two dimensions a complete polynomial of order n is

$$\vartheta(\mathrm{x},\mathrm{y}) = \sum_{\mathrm{r}=1}^{\mathrm{p}} \alpha_{\mathrm{r}}\, \mathrm{x}^{\mathrm{i}}\, \mathrm{y}^{\mathrm{j}} \quad \text{with} \quad \mathrm{i}+\mathrm{j} \leq \mathrm{n}\,, \tag{226}$$

and the number of terms is

$$\mathrm{p} = (\mathrm{n}+1)(\mathrm{n}+2)/2\,. \tag{227}$$

An illustration of complete polynomials is given by Pascal's triangle:

$$\begin{array}{ccccccc} & & & 1 & & & \\ & & \mathrm{x} & & \mathrm{y} & & \\ & \mathrm{x}^2 & & \mathrm{xy} & & \mathrm{y}^2 & \\ \mathrm{x}^3 & & \mathrm{x}^2\mathrm{y} & & \mathrm{xy}^2 & & \mathrm{y}^3 \end{array} \tag{228}$$

A complete polynomial of order n in three dimensions can be found by

$$\vartheta(\mathrm{x},\mathrm{y},\mathrm{z}) = \sum_{\mathrm{r}=1}^{\mathrm{p}} \alpha_{\mathrm{r}}\, \mathrm{x}^{\mathrm{i}}\, \mathrm{y}^{\mathrm{j}}\, \mathrm{z}^{\mathrm{k}} \quad \text{with} \quad \mathrm{i}+\mathrm{j}+\mathrm{k} \leq \mathrm{n}\,, \tag{229}$$

with the number of terms

$$\mathrm{p} = (\mathrm{n}+1)(\mathrm{n}+2)(\mathrm{n}+3)/6\,. \tag{230}$$

Let m be the highest order of a spatial derivative occurring in the potential Π to be minimized. The compatibility requirement states that the shape functions must be m times continuously differentiable within $\mathcal{B}^{\mathrm{e}}$ and that they must be (m-1) times continuous across element boundaries. Both requirements, completeness and compatibility, are two main aspects of the consistency between the discrete and the mathematical models.

Furthermore, stability is associated with solution uniqueness properties of the discrete and mathematical model, e.g. nonphysical zero-energy modes have to be precluded.

6.3 Triangular and Tetrahedral Elements

6.3.1 Triangular Elements

Linear Triangular Element: Triangular elements with linear ansatz functions belong to the first developments in the field of the Finite Element Method. The node numbers $\mathrm{I} = 1,2,3$ are arranged counterclockwise. Here we consider an element with scalar-valued state variables. The approximation of the state variable then reads

$$\vartheta^{\mathrm{h}}(\mathrm{x},\mathrm{y}) = \alpha_1 + \alpha_2\, \mathrm{x} + \alpha_3\, \mathrm{y} \quad \text{in} \quad \mathcal{B}^{\mathrm{e}}\,. \tag{231}$$

We denote the coordinates and state variables at the individual nodes with

$$(\mathrm{x}^{\mathrm{I}}, \mathrm{y}^{\mathrm{I}}),\ \mathrm{I} = 1,2,3 \qquad \text{and} \qquad \mathrm{d}^{\mathrm{I}},\ \mathrm{I} = 1,2,3\,, \tag{232}$$

respectively, and then we obtain

$$\begin{array}{rcl} \mathrm{d}^{①} &=& \alpha_1 + \alpha_2\, \mathrm{x}^{①} + \alpha_3\, \mathrm{y}^{①}, \\ \mathrm{d}^{②} &=& \alpha_1 + \alpha_2\, \mathrm{x}^{②} + \alpha_3\, \mathrm{y}^{②}, \\ \mathrm{d}^{③} &=& \alpha_1 + \alpha_2\, \mathrm{x}^{③} + \alpha_3\, \mathrm{y}^{③}\,. \end{array} \tag{233}$$

From (233) we calculate α_1, α_2 and α_3 and insert them in (231) to obtain the approximation

$$\begin{aligned}\vartheta^{\mathrm{h}}(\mathrm{x},\mathrm{y}) = \frac{1}{2\mathrm{A}^{\mathrm{e}}}\Big[&(\mathrm{a}_1+\mathrm{b}_1\,\mathrm{x}+\mathrm{c}_1\,\mathrm{y})\mathrm{d}^{①}+\\ &+(\mathrm{a}_2+\mathrm{b}_2\,\mathrm{x}+\mathrm{c}_2\,\mathrm{y})\mathrm{d}^{②}+\\ &+(\mathrm{a}_3+\mathrm{b}_3\,\mathrm{x}+\mathrm{c}_3\,\mathrm{y})\mathrm{d}^{③}\Big]\end{aligned} \tag{234}$$

with the abbreviations

$$\begin{aligned}&\mathrm{a}_1 = \mathrm{x}^{②}\mathrm{y}^{③} - \mathrm{x}^{③}\mathrm{y}^{②}, \quad \mathrm{b}_1 = \mathrm{y}^{②} - \mathrm{y}^{③}, \quad \mathrm{c}_1 = \mathrm{x}^{③} - \mathrm{x}^{②}\\ &\mathrm{a}_2 = \mathrm{x}^{③}\mathrm{y}^{①} - \mathrm{x}^{①}\mathrm{y}^{③}, \quad \mathrm{b}_2 = \mathrm{y}^{③} - \mathrm{y}^{①}, \quad \mathrm{c}_2 = \mathrm{x}^{①} - \mathrm{x}^{③}\\ &\mathrm{a}_3 = \mathrm{x}^{①}\mathrm{y}^{②} - \mathrm{x}^{②}\mathrm{y}^{①}, \quad \mathrm{b}_3 = \mathrm{y}^{①} - \mathrm{y}^{②}, \quad \mathrm{c}_3 = \mathrm{x}^{②} - \mathrm{x}^{①}.\end{aligned} \tag{235}$$

The element area A^{e} can be computed from

$$2\,\mathrm{A}^{\mathrm{e}} = \det\begin{bmatrix}1 & 1 & 1\\ \mathrm{x}^{①} & \mathrm{x}^{②} & \mathrm{x}^{③}\\ \mathrm{y}^{①} & \mathrm{y}^{②} & \mathrm{y}^{③}\end{bmatrix} = \begin{cases}(\mathrm{x}^{②}\mathrm{y}^{③} - \mathrm{x}^{③}\mathrm{y}^{②}) +\\ +(\mathrm{x}^{③}\mathrm{y}^{①} - \mathrm{x}^{①}\mathrm{y}^{③}) +\\ +(\mathrm{x}^{①}\mathrm{y}^{②} - \mathrm{x}^{②}\mathrm{y}^{①}).\end{cases} \tag{236}$$

Now we can reformulate the approximation of the state variable in $\mathcal{B}^e$:

$$\vartheta^{\mathrm{h}}(\mathrm{x},\mathrm{y}) = \sum_{\mathrm{I}=1}^{3} \mathrm{N}^{\mathrm{I}}(\mathrm{x},\mathrm{y})\,\mathrm{d}^{\mathrm{I}} \quad \text{with} \quad \mathrm{N}^{\mathrm{I}} = \frac{1}{2\mathrm{A}^{\mathrm{e}}}(\mathrm{a}_{\mathrm{I}} + \mathrm{b}_{\mathrm{I}}\,\mathrm{x} + \mathrm{c}_{\mathrm{I}}\,\mathrm{y})\ . \tag{237}$$

The ansatz function N^{I} is equal to 1 at the associated node $(\mathrm{x}_{\mathrm{I}}, \mathrm{y}_{\mathrm{I}})$ and zero at the remaining nodes. Furthermore, element matrices $\underline{\mathbf{k}}^{\mathrm{e}}$are typically built up by the integration of products of derivatives of the ansatz-functions with respect to cartesian coordinates. Starting from (237) the cartesian derivatives of N^{I} are

$$\mathrm{N}^{\mathrm{I}}_{,\mathrm{x}} = \frac{1}{2\mathrm{A}^{\mathrm{e}}}\mathrm{b}_{\mathrm{I}} \quad \text{and} \quad \mathrm{N}^{\mathrm{I}}_{,\mathrm{y}} = \frac{1}{2\mathrm{A}^{\mathrm{e}}}\mathrm{c}_{\mathrm{I}}. \tag{238}$$

For the integration over triangular areas the area coordinates are convenient. In this

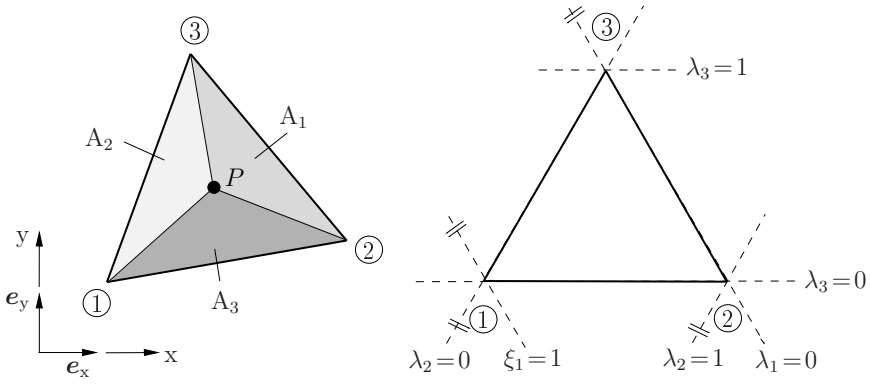


Figure 34: Triangular element with linear ansatz functions in area coordinates.

context the position of a point P in $\mathcal{B}^{\mathrm{e}}$ can be identified by λ_1, λ_2 and λ_3, which are defined as

$$\lambda_{\mathrm{I}} = \frac{\mathrm{A}_{\mathrm{I}}}{\mathrm{A}^{\mathrm{e}}} \quad \text{for} \quad \mathrm{I} = 1,2,3 \tag{239}$$

with the sub-areas shown in Fig. 34. From $\mathrm{A}_1 + \mathrm{A}_2 + \mathrm{A}_3 = \mathrm{A}^{\mathrm{e}}$ we conclude

$$\lambda_1 + \lambda_2 + \lambda_3 = 1; \tag{240}$$

the middle-point of the triangle is located at $\lambda_1 = \lambda_2 = \lambda_3 = 1/3$. A linear approximation of the state-variable in area coordinates makes direct use of the nodal values $\mathrm{d}^{\mathrm{I}}|_{\mathrm{I}=1,2,3}$:

$$\vartheta^{\mathrm{h}}(\lambda_1, \lambda_2, \lambda_3) = \lambda_1 \, \mathrm{d}^{①} + \lambda_2 \, \mathrm{d}^{②} + \lambda_3 \, \mathrm{d}^{③} \, . \tag{241}$$

Thus, we conclude from (241) and (237) $\lambda_{\mathrm{I}} = \mathrm{N}_{\mathrm{I}}$. Cartesian and area-coordinates are related through

$$\begin{bmatrix} 1 \\ \mathrm{x} \\ \mathrm{y} \end{bmatrix} = \underline{\mathbf{A}} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{bmatrix} \quad \text{with} \quad \underline{\mathbf{A}} = \begin{bmatrix} 1 & 1 & 1 \\ \mathrm{x}^{①} & \mathrm{x}^{②} & \mathrm{x}^{③} \\ \mathrm{y}^{①} & \mathrm{y}^{②} & \mathrm{y}^{③} \end{bmatrix} \tag{242}$$

or through the inverse relationship

$$\begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \end{bmatrix} = \underbrace{\frac{1}{2\mathrm{A}^{\mathrm{e}}} \left[\begin{array}{c|c|c} \mathrm{x}^{②}\mathrm{y}^{③} - \mathrm{x}^{③}\mathrm{y}^{②} & \mathrm{y}^{②} - \mathrm{y}^{③} & \mathrm{x}^{③} - \mathrm{x}^{②} \\ \mathrm{x}^{③}\mathrm{y}^{①} - \mathrm{x}^{①}\mathrm{y}^{③} & \mathrm{y}^{③} - \mathrm{y}^{①} & \mathrm{x}^{①} - \mathrm{x}^{③} \\ \mathrm{x}^{①}\mathrm{y}^{②} - \mathrm{x}^{②}\mathrm{y}^{①} & \mathrm{y}^{①} - \mathrm{y}^{②} & \mathrm{x}^{②} - \mathrm{x}^{①} \end{array}\right]}_{\underline{\mathbf{A}}^{-1}} \begin{bmatrix} 1 \\ \mathrm{x} \\ \mathrm{y} \end{bmatrix} \tag{243}$$

For the calculation of the partial derivatives of $\vartheta^{\mathrm{h}}(\lambda_1, \lambda_2, \lambda_3)$ with respect to cartesian coordinates we apply the chain rule

$$\begin{aligned} \frac{\partial \vartheta^{\mathrm{h}}}{\partial \mathrm{x}} &= \frac{\partial \vartheta^{\mathrm{h}}}{\partial \lambda_1}\frac{\partial \lambda_1}{\partial \mathrm{x}} + \frac{\partial \vartheta^{\mathrm{h}}}{\partial \lambda_2}\frac{\partial \lambda_2}{\partial \mathrm{x}} + \frac{\partial \vartheta^{\mathrm{h}}}{\partial \lambda_3}\frac{\partial \lambda_3}{\partial \mathrm{x}} \, , \\ \frac{\partial \vartheta^{\mathrm{h}}}{\partial \mathrm{y}} &= \frac{\partial \vartheta^{\mathrm{h}}}{\partial \lambda_1}\frac{\partial \lambda_1}{\partial \mathrm{y}} + \frac{\partial \vartheta^{\mathrm{h}}}{\partial \lambda_2}\frac{\partial \lambda_2}{\partial \mathrm{y}} + \frac{\partial \vartheta^{\mathrm{h}}}{\partial \lambda_3}\frac{\partial \lambda_3}{\partial \mathrm{y}} \, . \end{aligned} \tag{244}$$

Evaluation of the partial derivatives of the area-coordinates with respect to x and y leads, by use of (243), to

$$\begin{bmatrix} \dfrac{\partial \lambda_1}{\partial \mathrm{x}} & \dfrac{\partial \lambda_1}{\partial \mathrm{y}} \\ \dfrac{\partial \lambda_2}{\partial \mathrm{x}} & \dfrac{\partial \lambda_2}{\partial \mathrm{y}} \\ \dfrac{\partial \lambda_3}{\partial \mathrm{x}} & \dfrac{\partial \lambda_3}{\partial \mathrm{y}} \end{bmatrix} = \underline{\mathbf{A}}^{-1} \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \, , \tag{245}$$

with the individual terms

$$\begin{aligned} \frac{\partial \lambda_1}{\partial \mathrm{x}} &= \frac{1}{2\mathrm{A}^{\mathrm{e}}}(\mathrm{y}^{②} - \mathrm{y}^{③}) = \frac{\mathrm{b}_1}{2\mathrm{A}^{\mathrm{e}}} \, , \quad \frac{\partial \lambda_1}{\partial \mathrm{y}} = \frac{1}{2\mathrm{A}^{\mathrm{e}}}(\mathrm{x}^{③} - \mathrm{x}^{②}) = \frac{\mathrm{c}_1}{2\mathrm{A}^{\mathrm{e}}} \, , \\ \frac{\partial \lambda_2}{\partial \mathrm{x}} &= \frac{1}{2\mathrm{A}^{\mathrm{e}}}(\mathrm{y}^{③} - \mathrm{y}^{①}) = \frac{\mathrm{b}_2}{2\mathrm{A}^{\mathrm{e}}} \, , \quad \frac{\partial \lambda_2}{\partial \mathrm{y}} = \frac{1}{2\mathrm{A}^{\mathrm{e}}}(\mathrm{x}^{①} - \mathrm{x}^{③}) = \frac{\mathrm{c}_2}{2\mathrm{A}^{\mathrm{e}}} \, , \\ \frac{\partial \lambda_3}{\partial \mathrm{x}} &= \frac{1}{2\mathrm{A}^{\mathrm{e}}}(\mathrm{y}^{①} - \mathrm{y}^{②}) = \frac{\mathrm{b}_3}{2\mathrm{A}^{\mathrm{e}}} \, , \quad \frac{\partial \lambda_3}{\partial \mathrm{y}} = \frac{1}{2\mathrm{A}^{\mathrm{e}}}(\mathrm{x}^{②} - \mathrm{x}^{①}) = \frac{\mathrm{c}_3}{2\mathrm{A}^{\mathrm{e}}} \, . \end{aligned} \tag{246}$$

Compare these results with (238) and (235). Now the gradient of ϑ^{h} appears through evaluation of (244) by use of (246) and (241) in the form

$$\begin{bmatrix} \dfrac{\partial \vartheta^{\mathrm{h}}}{\partial \mathrm{x}} \\ \dfrac{\partial \vartheta^{\mathrm{h}}}{\partial \mathrm{y}} \end{bmatrix} = \frac{1}{2\mathrm{A}^{\mathrm{e}}} \begin{bmatrix} \mathrm{b}_1\, \mathrm{d}^{①} + \mathrm{b}_2\, \mathrm{d}^{②} + \mathrm{b}_3\, \mathrm{d}^{③} \\ \mathrm{c}_1\, \mathrm{d}^{①} + \mathrm{c}_2\, \mathrm{d}^{②} + \mathrm{c}_3\, \mathrm{d}^{③} \end{bmatrix} = \underline{\mathbb{B}}^{\mathrm{e}}\, \underline{\mathbf{d}}^{\mathrm{e}} , \tag{247}$$

with the $\underline{\mathbb{B}}^{\mathrm{e}}$ matrix and the element vector of unknowns $\underline{\mathbf{d}}^{\mathrm{e}}$:

$$\underline{\mathbb{B}}^{\mathrm{e}} = \frac{1}{2\mathrm{A}^{\mathrm{e}}} \begin{bmatrix} \mathrm{b}_1 & \mathrm{b}_2 & \mathrm{b}_3 \\ \mathrm{c}_1 & \mathrm{c}_2 & \mathrm{c}_3 \end{bmatrix} , \quad \underline{\mathbf{d}}^{\mathrm{e}} = \begin{bmatrix} \mathrm{d}^{①} \\ \mathrm{d}^{②} \\ \mathrm{d}^{③} \end{bmatrix} . \tag{248}$$

2D linear elasticity: For plane elasticity problems the linear interpolation for the displacements is

$$\boldsymbol{u}^{\mathrm{h}} = \underline{\mathbb{N}}\, \underline{\mathbf{d}}^{\mathrm{e}} \quad \text{with} \quad \underline{\mathbb{N}} = \left[\begin{array}{cc|cc|cc} \mathrm{N}^{①} & 0 & \mathrm{N}^{②} & 0 & \mathrm{N}^{③} & 0 \\ 0 & \mathrm{N}^{①} & 0 & \mathrm{N}^{②} & 0 & \mathrm{N}^{③} \end{array}\right] , \tag{249}$$

and the element displacement vector $\underline{\mathbf{d}}^{\mathrm{e}} = [\mathrm{d}_{\mathrm{x}}^{①},\ \mathrm{d}_{\mathrm{y}}^{①},\ \mathrm{d}_{\mathrm{x}}^{②},\ \mathrm{d}_{\mathrm{y}}^{②},\ \mathrm{d}_{\mathrm{x}}^{③},\ \mathrm{d}_{\mathrm{y}}^{③}]^{\mathrm{T}}$. The strains within a typical element in matrix notation are

$$\underline{\boldsymbol{\varepsilon}} = \underline{\mathbb{B}}^{\mathrm{e}} \underline{\mathbf{d}}^{\mathrm{e}} , \quad \text{with} \quad \underline{\boldsymbol{\varepsilon}} = [\varepsilon_{11}, \varepsilon_{22}, 2\varepsilon_{12}]^{\mathrm{T}} \tag{250}$$

and the element $\underline{\mathbb{B}}^{\mathrm{e}}$ matrix

$$\begin{aligned} \underline{\mathbb{B}}^{e} &= \\ &= \frac{1}{2\mathrm{A}^{\mathrm{e}}} \left[\begin{array}{c|c|c|c|c|c} \mathrm{y}^{②} - \mathrm{y}^{③} & 0 & \mathrm{y}^{③} - \mathrm{y}^{①} & 0 & \mathrm{y}^{①} - \mathrm{y}^{②} & 0 \\ 0 & \mathrm{x}^{③} - \mathrm{x}^{②} & 0 & \mathrm{x}^{①} - \mathrm{x}^{③} & 0 & \mathrm{x}^{②} - \mathrm{x}^{①} \\ \mathrm{x}^{③} - \mathrm{x}^{②} & \mathrm{y}^{②} - \mathrm{y}^{③} & \mathrm{x}^{①} - \mathrm{x}^{③} & \mathrm{y}^{③} - \mathrm{y}^{①} & \mathrm{x}^{②} - \mathrm{x}^{①} & \mathrm{y}^{①} - \mathrm{y}^{②} \end{array}\right] . \end{aligned} \tag{251}$$

The element stiffness matrix is given by

$$\underline{\mathbf{k}}^{\mathrm{e}} = \mathrm{h} \int_{\mathcal{B}^{\mathrm{e}}} \underline{\mathbb{B}}^{\mathrm{eT}} \underline{\mathbb{C}}^{(\mathrm{V})} \underline{\mathbb{B}}^{\mathrm{e}}\, \mathrm{da} , \tag{252}$$

where h is the constant element thickness and

$$\underline{\mathbb{C}}^{(\mathrm{V})} = \begin{bmatrix} \mathbb{C}_{11} & \mathbb{C}_{12} & \mathbb{C}_{14} \\ \mathbb{C}_{21} & \mathbb{C}_{22} & \mathbb{C}_{24} \\ \mathbb{C}_{41} & \mathbb{C}_{42} & \mathbb{C}_{44} \end{bmatrix} \tag{253}$$

represents the reduced coefficient scheme of the elasticities in Voigt notation. All terms in (252) are constant, thus we obtain

$$\underline{\mathbf{k}}^{\mathrm{e}} = \mathrm{h}\ \mathrm{A}^{\mathrm{e}}\ \underline{\mathbb{B}}^{\mathrm{eT}}\ \underline{\mathbb{C}}^{(\mathrm{V})}\ \underline{\mathbb{B}}^{\mathrm{e}} . \tag{254}$$

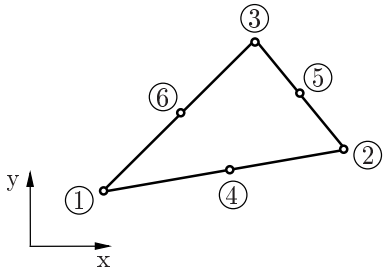


Figure 35: Straight-sided quadratic triangle.

Quadratic Triangular Element: Let us now consider the straight-sided quadratic triangle with three mid-side nodes as shown in Figure 35. Here we focus on the approximation of a vector-valued state vector, the displacement field $\boldsymbol{u}$. Thus, we have two nodal displacement components at each node:

$$\underline{\mathbf{d}}^{\mathrm{I}} = \begin{bmatrix} \mathrm{d}_{\mathrm{x}}^{\mathrm{I}} \\ \mathrm{d}_{\mathrm{y}}^{\mathrm{I}} \end{bmatrix} , \ \mathrm{I} = 1, ..., 6 . \tag{255}$$

The shape functions for a complete polynomial, see Pascal's triangle (228), of quadratic order is formulated in terms of

$$1,\ \mathrm{x},\ \mathrm{y},\ \mathrm{x}^2,\ \mathrm{xy},\ \mathrm{y}^2 . \tag{256}$$

For the approximation of a 2D vector-valued state variable we set

$$\begin{aligned} \mathrm{u}_{\mathrm{x}}^{\mathrm{h}}(\mathrm{x},\mathrm{y}) &= \alpha_1 + \alpha_2\,\mathrm{x} + \alpha_3\,\mathrm{y} + \alpha_4\,\mathrm{x}^2 + \alpha_5\,\mathrm{y}^2 + \alpha_6\,\mathrm{xy} , \\ \mathrm{u}_{\mathrm{y}}^{\mathrm{h}}(\mathrm{x},\mathrm{y}) &= \beta_1 + \beta_2\,\mathrm{x} + \beta_3\,\mathrm{y} + \beta_4\,\mathrm{x}^2 + \beta_5\,\mathrm{y}^2 + \beta_6\,\mathrm{xy} . \end{aligned} \tag{257}$$

This is equivalent to the formulation of the displacement vector in terms of area coordinates $\lambda_1, \lambda_2, \lambda_3$, i.e.,

$$\begin{aligned} \mathrm{u}_{\mathrm{x}}^{\mathrm{h}}(\lambda_1,\lambda_2,\lambda_3) &= \sum_{\mathrm{I}=1}^{6} \mathrm{N}^{\mathrm{I}}(\lambda_1,\lambda_2,\lambda_3)\,\mathrm{d}_{\mathrm{x}}^{\mathrm{I}} , \\ \mathrm{u}_{\mathrm{y}}^{\mathrm{h}}(\lambda_1,\lambda_2,\lambda_3) &= \sum_{\mathrm{I}=1}^{6} \mathrm{N}^{\mathrm{I}}(\lambda_1,\lambda_2,\lambda_3)\,\mathrm{d}_{\mathrm{y}}^{\mathrm{I}} , \end{aligned} \tag{258}$$

where the shape functions are

$$\begin{aligned} &\mathrm{N}^{①} = \lambda_1(2\lambda_1 - 1) , \quad && \mathrm{N}^{②} = \lambda_2(2\lambda_2 - 1) , \quad && \mathrm{N}^{③} = \lambda_3(2\lambda_3 - 1) , \\ &\mathrm{N}^{④} = 4\lambda_1\lambda_2 , && \mathrm{N}^{⑤} = 4\lambda_2\lambda_3 , && \mathrm{N}^{⑥} = 4\lambda_3\lambda_1 . \end{aligned} \tag{259}$$

A visualization of a corner node and mid-side node shape function is given in Figures 36. Introducing the element displacement vector

$$\boldsymbol{d}^{\mathrm{e}} = [\mathrm{d}_{\mathrm{x}}^{①}, \mathrm{d}_{\mathrm{y}}^{①}, \mathrm{d}_{\mathrm{x}}^{②}, \mathrm{d}_{\mathrm{y}}^{②}, ..., \mathrm{d}_{\mathrm{y}}^{⑥}]^{\mathrm{T}} \tag{260}$$

and the matrix of the ansatz functions

$$\underline{\mathbb{N}} = \left[\begin{array}{cc|cc|cc|cc|cc|cc} \mathrm{N}^{①} & 0 & \mathrm{N}^{②} & 0 & \mathrm{N}^{③} & 0 & \mathrm{N}^{④} & 0 & \mathrm{N}^{⑤} & 0 & \mathrm{N}^{⑥} & 0 \\ 0 & \mathrm{N}^{①} & 0 & \mathrm{N}^{②} & 0 & \mathrm{N}^{③} & 0 & \mathrm{N}^{④} & 0 & \mathrm{N}^{⑤} & 0 & \mathrm{N}^{⑥} \end{array}\right] , \tag{261}$$

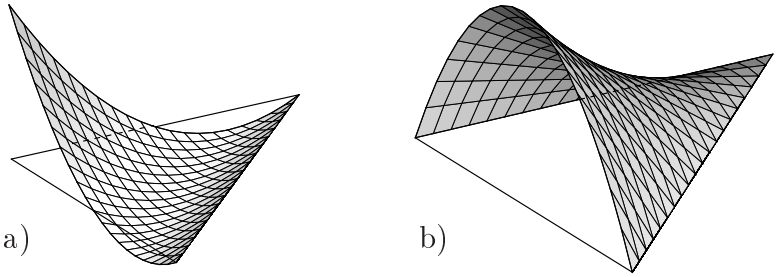


Figure 36: Shape functions of a) corner node and b) mid-side node.

yields

$$\boldsymbol{u}^{\mathrm{h}} = \sum_{\mathrm{I}=1}^{6} \mathrm{N}^{\mathrm{I}}(\lambda_1, \lambda_2, \lambda_3) \begin{bmatrix} \mathrm{d}_{\mathrm{x}}^{\mathrm{I}} \\ \mathrm{d}_{\mathrm{y}}^{\mathrm{I}} \end{bmatrix} = \underline{\mathbb{N}}\, \underline{\mathbf{d}}^{\mathrm{e}} \,. \tag{262}$$

Partial derivative calculations: The strain field of this element contains a complete linear interpolation of the strain components $\varepsilon_{11}, \varepsilon_{22}$ and ε_{12} in $\mathcal{B}^{\mathrm{e}}$. Based on the notion used in (250) we obtain

$$\begin{aligned} \varepsilon_{11}^{\mathrm{h}} &= \frac{\partial \mathrm{u}_{\mathrm{x}}^{\mathrm{h}}}{\partial \mathrm{x}} = \sum_{\mathrm{I}=1}^{6} \frac{\partial \mathrm{N}^{\mathrm{I}}}{\partial \mathrm{x}} \mathrm{d}_{\mathrm{x}}^{\mathrm{I}} \,, \\ \varepsilon_{22}^{\mathrm{h}} &= \frac{\partial \mathrm{u}_{\mathrm{y}}^{\mathrm{h}}}{\partial \mathrm{y}} = \sum_{\mathrm{I}=1}^{6} \frac{\partial \mathrm{N}^{\mathrm{I}}}{\partial \mathrm{y}} \mathrm{d}_{\mathrm{y}}^{\mathrm{I}} \,, \\ 2\varepsilon_{12}^{\mathrm{h}} &= \frac{\partial \mathrm{u}_{\mathrm{x}}^{\mathrm{h}}}{\partial \mathrm{y}} + \frac{\partial \mathrm{u}_{\mathrm{y}}^{\mathrm{h}}}{\partial \mathrm{x}} = \sum_{\mathrm{I}=1}^{6} \left(\frac{\partial \mathrm{N}^{\mathrm{I}}}{\partial \mathrm{y}} \mathrm{d}_{\mathrm{x}}^{\mathrm{I}} + \frac{\partial \mathrm{N}^{\mathrm{I}}}{\partial \mathrm{x}} \mathrm{d}_{\mathrm{y}}^{\mathrm{I}} \right) \,, \end{aligned} \tag{263}$$

with the associated matrix representation

$$\begin{bmatrix} \varepsilon_{11}^{\mathrm{h}} \\ \varepsilon_{22}^{\mathrm{h}} \\ 2\varepsilon_{12}^{\mathrm{h}} \end{bmatrix} = \sum_{\mathrm{I}=1}^{6} \begin{bmatrix} \mathrm{N}_{,\mathrm{x}}^{\mathrm{I}} & 0 \\ 0 & \mathrm{N}_{,\mathrm{y}}^{\mathrm{I}} \\ \mathrm{N}_{,\mathrm{y}}^{\mathrm{I}} & \mathrm{N}_{,\mathrm{x}}^{\mathrm{I}} \end{bmatrix} \begin{bmatrix} \mathrm{d}_{\mathrm{x}}^{\mathrm{I}} \\ \mathrm{d}_{\mathrm{y}}^{\mathrm{I}} \end{bmatrix} = \sum_{\mathrm{I}=1}^{6} \underline{\mathbb{B}}^{\mathrm{I}} \underline{\mathbf{d}}^{\mathrm{I}} = \underline{\mathbb{B}}^{\mathrm{e}} \underline{\mathbf{d}}^{\mathrm{e}} \,. \tag{264}$$

For the computation of the cartesian derivatives we apply the chain rule to $(258)_1$

$$\begin{aligned} \frac{\partial \mathrm{u}_{\mathrm{x}}^{\mathrm{h}}}{\partial \mathrm{x}} &= \sum_{\mathrm{i}=1}^{3} \frac{\partial \mathrm{u}_{\mathrm{x}}^{\mathrm{h}}}{\partial \lambda_{\mathrm{i}}} \frac{\partial \lambda_{\mathrm{i}}}{\partial \mathrm{x}} = \sum_{\mathrm{i}=1}^{3} \frac{\partial}{\partial \lambda_{\mathrm{i}}} \left[\sum_{\mathrm{I}=1}^{6} \mathrm{N}^{\mathrm{I}}\, \mathrm{d}_{\mathrm{x}}^{\mathrm{I}} \right] \frac{\partial \lambda_{\mathrm{i}}}{\partial \mathrm{x}} \,, \\ \frac{\partial \mathrm{u}_{\mathrm{x}}^{\mathrm{h}}}{\partial \mathrm{y}} &= \sum_{\mathrm{i}=1}^{3} \frac{\partial \mathrm{u}_{\mathrm{x}}^{\mathrm{h}}}{\partial \lambda_{\mathrm{i}}} \frac{\partial \lambda_{\mathrm{i}}}{\partial \mathrm{y}} = \sum_{\mathrm{i}=1}^{3} \frac{\partial}{\partial \lambda_{\mathrm{i}}} \left[\sum_{\mathrm{I}=1}^{6} \mathrm{N}^{\mathrm{I}}\, \mathrm{d}_{\mathrm{x}}^{\mathrm{I}} \right] \frac{\partial \lambda_{\mathrm{i}}}{\partial \mathrm{y}} \,. \end{aligned} \tag{265}$$

For the straight-sided element with mid-side nodes the derivatives of λ_{i} with respect to x and y are given in (246). Finally, we get through (259)

$$\begin{aligned} \frac{\partial \mathrm{u}_{\mathrm{x}}^{\mathrm{h}}}{\partial \mathrm{x}} = \frac{1}{2\mathrm{A}^{\mathrm{e}}} \Big\{ & \left[(4\,\lambda_1 - 1) \mathrm{d}_{\mathrm{x}}^{①} + 4\,\lambda_2\, \mathrm{d}_{\mathrm{x}}^{④} + 4\,\lambda_3\, \mathrm{d}_{\mathrm{x}}^{⑥} \right] (\mathrm{y}^{②} - \mathrm{y}^{③}) \\ & + \left[(4\,\lambda_2 - 1) \mathrm{d}_{\mathrm{x}}^{②} + 4\,\lambda_1\, \mathrm{d}_{\mathrm{x}}^{④} + 4\,\lambda_3\, \mathrm{d}_{\mathrm{x}}^{⑤} \right] (\mathrm{y}^{③} - \mathrm{y}^{①}) \\ & + \left[(4\,\lambda_3 - 1) \mathrm{d}_{\mathrm{x}}^{③} + 4\,\lambda_2\, \mathrm{d}_{\mathrm{x}}^{⑤} + 4\,\lambda_1\, \mathrm{d}_{\mathrm{x}}^{⑥} \right] (\mathrm{y}^{①} - \mathrm{y}^{②}) \Big\} \,. \end{aligned} \tag{266}$$

Sorting with respect to the ordered unknowns we write

$$\frac{\partial u_x^h}{\partial x} = \sum_{i=1}^{6} N_{,x}^I d_x^I \tag{267}$$

with

$$\begin{aligned}
N_{,x}^{①} &= (4\lambda_1 - 1)(y^{②} - y^{③}),\\
N_{,x}^{②} &= (4\lambda_2 - 1)(y^{③} - y^{①}),\\
N_{,x}^{③} &= (4\lambda_3 - 1)(y^{①} - y^{②}),\\
N_{,x}^{④} &= 4\lambda_2(y^{②} - y^{③}) + 4\lambda_1(y^{③} - y^{①}),\\
N_{,x}^{⑤} &= 4\lambda_3(y^{③} - y^{①}) + 4\lambda_2(y^{①} - y^{②}),\\
N_{,x}^{⑥} &= 4\lambda_3(y^{②} - y^{③}) + 4\lambda_1(y^{①} - y^{②}).
\end{aligned} \tag{268}$$

The remaining terms are derived analogously.

Integration: A main advantage of using area coordinates instead of cartesian ones is the existence of analytical formulae for the integration over the domain $\mathcal{B}^e$ or the surface $\partial\mathcal{B}^e$ of it, see Eisenberg and Malvern (1973). Integration over $\mathcal{B}^e$ yields

$$\int_{\mathcal{B}^e} \lambda_1^h \lambda_2^l \lambda_3^m \, dA = 2A^e \frac{h!\, l!\, m!}{(h + l + m + 2)!}\,, \tag{269}$$

and over a pathlength $\mathcal{S} \subset \partial\mathcal{B}^e$

$$\int_{\mathcal{S}\subset\partial\mathcal{B}^e} \lambda_i^h \lambda_j^l \, dS = S \frac{h!\, l!}{(h + l + 1)!} \quad \text{for} \quad i \neq j\,, \tag{270}$$

where h, l, m are integers.

6.3.2 Tetrahedral Elements

Linear Tetrahedral Element: The spatial counterpart of the two-dimensional linear triangle is the four-noded (linear) tetrahedron. The corner nodes $\mathrm{P}_1, \mathrm{P}_2, \mathrm{P}_3, \mathrm{P}_4$ are ordered in such a way that the three directional vectors

$$\overrightarrow{\mathrm{P}_1\mathrm{P}_2}, \quad \overrightarrow{\mathrm{P}_1\mathrm{P}_3}, \quad \overrightarrow{\mathrm{P}_1\mathrm{P}_4} \tag{271}$$

form a right-handed system. Analogously to the area coordinates of the triangular elements

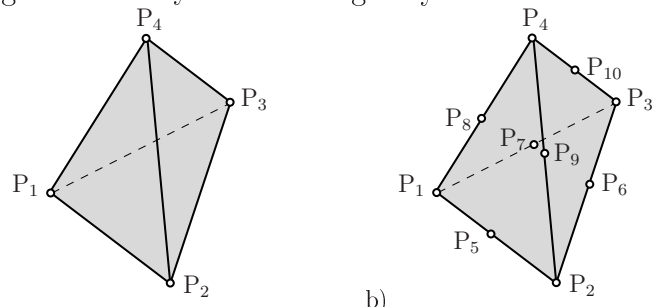


Figure 37: a) Four-noded (linear) tetrahedral element and b) Ten-noded (quadratic) tetrahedral element.

we define the volume coordinates for the tetrahedron via

$$\lambda_{\mathrm{I}} = \frac{\mathrm{V}_{\mathrm{I}}}{\mathrm{V}^{\mathrm{e}}} \quad \text{for} \quad \mathrm{I} = 1, 2, 3, 4\,, \tag{272}$$

with the subvolumes of the tetrahedron built up by the internal node P and the associated corner nodes

$$\begin{aligned} \mathrm{V}_1 &= \mathrm{vol}(\mathrm{P}, \mathrm{P}_2, \mathrm{P}_3, \mathrm{P}_4) \\ \mathrm{V}_2 &= \mathrm{vol}(\mathrm{P}, \mathrm{P}_1, \mathrm{P}_3, \mathrm{P}_4) \\ \mathrm{V}_3 &= \mathrm{vol}(\mathrm{P}, \mathrm{P}_1, \mathrm{P}_2, \mathrm{P}_4) \\ \mathrm{V}_4 &= \mathrm{vol}(\mathrm{P}, \mathrm{P}_1, \mathrm{P}_2, \mathrm{P}_3) \end{aligned} \tag{273}$$

for all points P in $\mathcal{B}^{\mathrm{e}}$, $\mathrm{V}^{\mathrm{e}} = \mathrm{vol}(\mathrm{P}_1\mathrm{P}_2\mathrm{P}_3\mathrm{P}_4)$ is the volume of the tetrahedron. In these coordinates we obtain the representation

$$\begin{aligned} \mathrm{x} &= \lambda_1\,\mathrm{x}^{①} + \lambda_2\,\mathrm{x}^{②} + \lambda_3\,\mathrm{x}^{③} + \lambda_4\,\mathrm{x}^{④}\,, \\ \mathrm{y} &= \lambda_1\,\mathrm{y}^{①} + \lambda_2\,\mathrm{y}^{②} + \lambda_3\,\mathrm{y}^{③} + \lambda_4\,\mathrm{y}^{④}\,, \\ \mathrm{z} &= \lambda_1\,\mathrm{z}^{①} + \lambda_2\,\mathrm{z}^{②} + \lambda_3\,\mathrm{z}^{③} + \lambda_4\,\mathrm{z}^{④}\,, \end{aligned} \tag{274}$$

where $(\mathrm{x}, \mathrm{y}, \mathrm{z})$ are the coordinates of an arbitrary point P in $\mathcal{B}^{\mathrm{e}}$. Since $\sum_{\mathrm{I}} \mathrm{V}_{\mathrm{I}} = \mathrm{V}^{\mathrm{e}}$ we obtain

$$\lambda_1 + \lambda_2 + \lambda_3 + \lambda_4 = 1\,. \tag{275}$$

The shape functions in terms of the volume coordinates are

$$\mathrm{N}^{①} = \lambda_1; \quad \mathrm{N}^{②} = \lambda_2; \quad \mathrm{N}^{③} = \lambda_3; \quad \mathrm{N}^{④} = \lambda_4\,. \tag{276}$$

Equation (275) and (274) appear in matrix form as

$$\begin{bmatrix} 1 \\ \mathrm{x} \\ \mathrm{y} \\ \mathrm{z} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 1 & 1 & 1 \\ \mathrm{x}^{①} & \mathrm{x}^{②} & \mathrm{x}^{③} & \mathrm{x}^{④} \\ \mathrm{y}^{①} & \mathrm{y}^{②} & \mathrm{y}^{③} & \mathrm{y}^{④} \\ \mathrm{z}^{①} & \mathrm{z}^{②} & \mathrm{z}^{③} & \mathrm{z}^{④} \end{bmatrix}}_{\underline{\mathbf{A}}} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \end{bmatrix} \quad \rightarrow \quad \begin{bmatrix} 1 \\ \boldsymbol{x} \end{bmatrix} = \underline{\mathbf{A}}\,\boldsymbol{\lambda}\,. \tag{277}$$

with the inverse relation

$$\boldsymbol{\lambda} = \underline{\mathbf{A}}^{-1} \begin{bmatrix} 1 \\ \boldsymbol{x} \end{bmatrix} = \frac{1}{6\mathrm{V}^{\mathrm{e}}} \operatorname{adj}\underline{\mathbf{A}} \begin{bmatrix} 1 \\ \boldsymbol{x} \end{bmatrix} \tag{278}$$

and the volume of the tetrahodron

$$\mathrm{V}^{\mathrm{e}} = \frac{1}{6} \det \underline{\mathbf{A}} \,. \tag{279}$$

Quadratic Tetrahedral Element: The construction of a straight-sided quadratic tetrahedron with mid-side nodes is a direct extension of the above discussed elements. Here we only represent the shape functions

$$\begin{aligned} \mathrm{N}^{\mathrm{I}} &= \lambda_{\mathrm{I}}(2\lambda_{\mathrm{I}} - 1) \quad \text{for} \quad \mathrm{I} = 1,2,3,4 \,, \\ \mathrm{N}^{⑤} &= 4\lambda_1\lambda_2 \,, \quad \mathrm{N}^{⑥} = 4\lambda_2\lambda_3 \,, \quad \mathrm{N}^{⑦} = 4\lambda_3\lambda_1 \,, \\ \mathrm{N}^{⑧} &= 4\lambda_1\lambda_4 \,, \quad \mathrm{N}^{⑨} = 4\lambda_2\lambda_4 \,, \quad \mathrm{N}^{⑩} = 4\lambda_3\lambda_4 \,. \end{aligned} \tag{280}$$

Integration: The integration of polynomials in volume coordinates can be performed by

$$\int_{\Omega^{\mathrm{e}}} \lambda_1^{\mathrm{h}} \, \lambda_2^{\mathrm{l}} \, \lambda_3^{\mathrm{m}} \, \lambda_4^{\mathrm{n}} \, \mathrm{dV} = 6\mathrm{V}^{\mathrm{e}} \, \frac{\mathrm{h!\, l!\, m!\, n!}}{(\mathrm{h+l+m+n+3})!} \,. \tag{281}$$

6.4 Isoparametric Concept

In the sequel we focus on the *isoparametric element concept*. The idea is to use the same interpolation functions for the unknown basic variables (e.g. the temperature or the displacements) and the geometry of the considered body of interest. Let us consider a domain $\mathcal{B}$ in the physical space $\boldsymbol{x}$. In **one dimension** the geometry of an individual finite element $\mathcal{B}^{\mathrm{e}}$ is approximated by

$$\mathrm{x} = \sum_{\mathrm{I}=1}^{\mathrm{nel}} \mathrm{N}^{\mathrm{I}}(\xi)\, \mathrm{x}^{\mathrm{I}}, \tag{282}$$

where the shape functions are formulated in the dimensionless natural or intrinsic coordinates $-1 \leq \xi \leq 1$ and x^{I} for I = 1,...,nel characterize the nodal coordinates. An example of a quadratic (three-noded) bar element is depicted in Figure 38. The approximation of

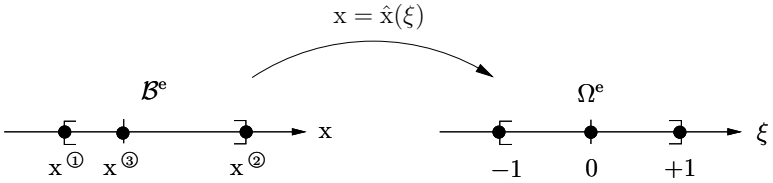


Figure 38: Isoparametric representation of a quadratic bar element.

a scalar-valued function ϑ is as follows

$$\vartheta^{\mathrm{h}} = \sum_{\mathrm{I}=1}^{\mathrm{nel}} \mathrm{N}^{\mathrm{I}}(\xi)\, \mathrm{d}^{\mathrm{I}}\,, \tag{283}$$

where we use the same shape funcions as in (282). For the computation of the element matrices we need the derivative of ϑ^{h} with respect to x. But, as pointed out in (283), the field variable ϑ^{h} is parameterized in ξ, so we have to apply the chain rule

$$\frac{\mathrm{d}\vartheta^{\mathrm{h}}}{\mathrm{dx}} = \frac{\mathrm{d}\vartheta^{\mathrm{h}}}{\mathrm{d}\xi}\frac{\mathrm{d}\xi}{\mathrm{dx}}\,, \tag{284}$$

where $\dfrac{\mathrm{d}\vartheta^{\mathrm{h}}}{\mathrm{d}\xi}$ can be computed from (283). As mentioned above, we use the same interpolation functions for the approximation of the geometry as for the approximation of the basic variable ϑ. Thus, we can compute the derivative of x with respect to ξ, i.e.

$$\frac{\mathrm{dx}^{\mathrm{h}}}{\mathrm{d}\xi} = \sum_{\mathrm{I}=1}^{\mathrm{nel}} \frac{\mathrm{dN}^{\mathrm{I}}}{\mathrm{d}\xi}\, \mathrm{x}^{\mathrm{I}} = \mathrm{J}_{11} \quad \rightarrow \quad \mathrm{dx}^{\mathrm{h}} = \mathrm{J}_{11}\, \mathrm{d}\xi\,, \tag{285}$$

and obtain the inverse relation

$$\frac{\mathrm{d}\xi^{\mathrm{h}}}{\mathrm{dx}} = \mathrm{J}_{11}^{-1}\,. \tag{286}$$

Inserting (286) in (284) leads to the required derivative

$$\frac{\mathrm{d}\vartheta^{\mathrm{h}}}{\mathrm{dx}^{\mathrm{h}}} = \mathrm{J}_{11}^{-1}\frac{\mathrm{d}\vartheta^{\mathrm{h}}}{\mathrm{d}\xi} = \sum_{\mathrm{I}=1}^{\mathrm{nel}} \mathrm{J}_{11}^{-1}\frac{\mathrm{dN}^{\mathrm{I}}}{\mathrm{d}\xi}\, \mathrm{d}^{\mathrm{I}} = \sum_{\mathrm{I}=1}^{\mathrm{nel}} \mathbb{B}^{\mathrm{I}}\, \mathrm{d}^{\mathrm{I}}. \tag{287}$$

For the illustrated bar element in Figure 38 we choose the shape functions

$$\mathrm{N}^{①} = \frac{1}{2}(\xi^2 - \xi)\,, \quad \mathrm{N}^{②} = \frac{1}{2}(\xi^2 + \xi)\,, \quad \mathrm{N}^{③} = 1 - \xi^2\,. \tag{288}$$

The cartesian nodal coordinates x^{I} are associated with the intrinsic nodal values

$$\mathrm{x}^{①} \to \xi^{①} = -1\,, \quad \mathrm{x}^{②} \to \xi^{②} = 1\,, \quad \mathrm{x}^{③} \to \xi^{③} = 0\,. \tag{289}$$

Thus, we obtain the nodal values

$$\mathrm{N}^{①}(\xi^{①}) = 1\,, \quad \mathrm{N}^{②}(\xi^{②}) = 1\,, \quad \mathrm{N}^{③}(\xi^{③}) = 1 \tag{290}$$

and $\mathrm{N}^{\mathrm{I}}(\xi^{\mathrm{J}}) = 0$ for $\mathrm{I} \neq \mathrm{J}$. Furthermore, from the completeness requirements we obtain

$$\mathrm{N}^{①} + \mathrm{N}^{②} + \mathrm{N}^{③} = 1\,. \tag{291}$$

These main characteristics are also valid for two- and three-dimensional elements: $\mathrm{N}^{\mathrm{I}}(\bullet)$ has the value 1 at node I and vanishes over all element sides that do not contain node I. The polynomial order, say n, over one side induces that the side must have n+1 nodes to ensure compatibility. Evaluating (285) using (288) yields

$$\mathrm{J}_{11} = \frac{1}{2}(2\xi - 1)\,\mathrm{x}^{①} + \frac{1}{2}(2\xi + 1)\,\mathrm{x}^{②} - 2\xi\,\mathrm{x}^{③}\,. \tag{292}$$

Remark: For the computation of the element matrices we typically have to evaluate integrals of the form

$$\int_{\mathcal{B}^{\mathrm{e}}} \mathbb{B}^{\mathrm{I}}(\xi)\,\mathbb{B}^{\mathrm{J}}(\xi)\,\mathrm{dx}. \tag{293}$$

Inserting the substitution $(285)_2$ leads to the integral

$$\int_{\Omega^{\mathrm{e}}} \mathbb{B}^{\mathrm{I}}\,\mathbb{B}^{\mathrm{J}}\,\mathrm{J}_{11}\,\mathrm{d}\xi \tag{294}$$

defined over $\Omega^{\mathrm{e}} = [-1, 1]$. Ω^{e} is the representation of the domain $\mathcal{B}^{\mathrm{e}}$ in the isoparametric space, parameterized in ξ. Numerical integration schemes for (294) are discussed in Section 6.5.

6.5 Numerical Integration

In the finite element approximations we have to evaluate integrals over the finite element domain $\mathcal{B}^e$ and parts of its boundary $\partial\mathcal{B}^e$. An analytical integration is, for the most finite elements, too cumbersome or even impossible. The fundamental method for the approximation of

$$\mathrm{I} = \int_{\mathrm{a}}^{\mathrm{b}} \mathrm{f(x)}\, \mathrm{dx} \tag{295}$$

is the numerical quadrature. There exist several quadrature rules, some well-known ones being the following:
i) midpoint rule

$$\mathrm{I} = \int_{\mathrm{a}}^{\mathrm{b}} \mathrm{f(x)dx} \approx (\mathrm{b}-\mathrm{a})\, \mathrm{f}\left(\frac{\mathrm{a}+\mathrm{b}}{2}\right) \tag{296}$$

ii) trapezoidal rule

$$\mathrm{I} = \int_{\mathrm{a}}^{\mathrm{b}} \mathrm{f(x)dx} \approx \frac{\mathrm{b}-\mathrm{a}}{2}\, [\mathrm{f(a)} + \mathrm{f(b)}] \tag{297}$$

iii) Simpson rule

$$\mathrm{I} = \int_{\mathrm{a}}^{\mathrm{b}} \mathrm{f(x)dx} \approx \frac{\mathrm{b}-\mathrm{a}}{6}\, [\mathrm{f(a)} + 4\mathrm{f}\left(\frac{\mathrm{a}+\mathrm{b}}{2}\right) + \mathrm{f(b)}] \tag{298}$$

The midpoint and the trapezoidal rules are exact for constant and linear functions, i.e. $\mathrm{f}''(\xi) = 0$ for any $\xi \in [\mathrm{a}, \mathrm{b}]$. The quadrature error of the Simpson formular is proportional to $(\mathrm{b}-\mathrm{a})^5 \mathrm{f}''''(\xi)$, $\xi \in [\mathrm{a}, \mathrm{b}]$, thus a cubic polynomial is exactly integrated.

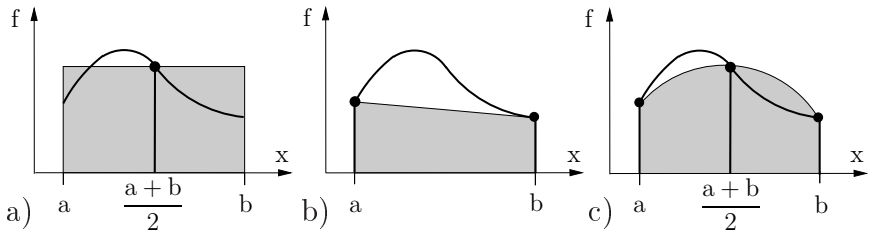


Figure 39: Illustration of the a) midpoint-, b) trapezoidal-, and c) Simpson rule.

These integration rules are of the Newton-Cotes type: they use values for the function evaluation at the interval boundaries and in the considered interval. Gauss quadrature is the most useful method in finite element applications because the integration points (Gauss points) for the evaluation of f(x) are chosen in an optimal way. For a brief explanation of the method we consider the one-dimensional integral (295), which is first transformed into the isoparametric space Ω^e and then approximated by the quadrature rule. Let us consider an integral of the form

$$\mathrm{I} = \int_{\mathrm{a}}^{\mathrm{b}} \mathrm{f(x)}\, \mathrm{dx} = \int_{\Omega^{\mathrm{e}}} \mathrm{f}(\xi)\, \mathrm{J}_{11}\, \mathrm{d}\xi \approx \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{w_i}\, \mathrm{f}(\xi_{\mathrm{i}})\ , \tag{299}$$

in this context see (293), with $\Omega^e = [-1, 1]$. The basic idea of the Gaussian quadrature is to integrate a function of order p = (2 n - 1)

$$\mathrm{f}(\xi) = \mathrm{c}_0 + \mathrm{c}_1\xi + \mathrm{c}_2\xi^2 + \ldots + \mathrm{c}_{2\mathrm{n}-1}\xi^{2\mathrm{n}-1} \tag{300}$$

exactly with n-function evaluations at $\xi_1, ..., \xi_n$ integration points, i.e.

$$I = \int_{-1}^{1} f(\xi)\, d\xi = \sum_{i=1}^{n} f(\xi_i) w_i \,, \tag{301}$$

where w_i are the weight factors. To illustrate the determination of the optimal Gauß points and weight factors we consider a linear $n = 1$ and a cubic $n = 2$ polynomial (300). For $n = 1$ we want to integrate a function with polynomial degree lower or equal to

$$2n - 1 = 2 \cdot 1 - 1 = 1 \,.$$

with exactly one integration point ξ_1 and one weight factor w_1, i.e.

$$\int_{-1}^{1} (c_0 + c_1 \xi)\, d\xi = f(\xi_1) w_1 \,. \tag{302}$$

Let us deal with both terms on the left-hand side separately. With $c_1 = 0$ and $c_2 = 0$ we get

$$\begin{aligned} \int_{-1}^{1} c_0\, d\xi = 2c_0 = c_0 w_1 \quad &\rightarrow \quad w_1 = 2 \,, \\ \int_{-1}^{1} c_1 \xi\, d\xi = 0 = c_1 \xi_1 w_1 \quad &\rightarrow \quad \xi_1 = 0 \,, \end{aligned} \tag{303}$$

respectively. Analogously, for $n = 2$ we obtain the exact same result whenever $f(\xi)$ is a polynomial of degree not greater than 3, i.e.

$$\int_{-1}^{1} f(\xi)\, d\xi = f(\xi_1) w_1 + f(\xi_2) w_2 \,, \tag{304}$$

by an appropriate choice of ξ_1, ξ_2 and w_1, w_2. Evaluating the integral for the cubic polynomial yields

$$\int_{-1}^{1} f(\xi)\, d\xi = c_0 \int_{-1}^{1} d\xi + c_1 \int_{-1}^{1} \xi\, d\xi + c_2 \int_{-1}^{1} \xi^2\, d\xi + c_3 \int_{-1}^{1} \xi^3\, d\xi \,. \tag{305}$$

We now set $c_0 = 1$ and c_1, c_2, c_3 to zero and get

$$\int_{-1}^{1} d\xi = 2 = 1 \cdot w_1 + 1 \cdot w_2 . \tag{306}$$

In the next step the only non-zero coefficient is $c_1 = 1$, it follows that

$$\int_{-1}^{1} \xi\, d\xi = 0 = \xi_1 w_1 + \xi_2 w_2 \,. \tag{307}$$

Furthermore, we obtain for the only non-zero coefficients

$$\begin{aligned} c_2 = 1: \qquad & \int_{-1}^{1} \xi^2\, d\xi = \frac{2}{3} = \xi_1^2 w_1 + \xi_2^2 w_2 \\ c_3 = 1: \qquad & \int_{-1}^{1} \xi^3\, d\xi = 0 = \xi_1^3 w_1 + \xi_2^3 w_2 \,. \end{aligned} \tag{308}$$

The solution to the set of equations (306)-(308) is

$$w_1 = w_2 = 1 \,, \quad \xi_1 = -1/\sqrt{3} \,, \quad \xi_2 = -1/\sqrt{3} \,. \tag{309}$$

For $n = 1, 2, 3, 4$ the integration points (Gauss points) and weight factors are summarized in Table (2)

Table 2: Gauss points and weight factors for $\int_{-1}^{1} \mathrm{f}(\xi)\, \mathrm{d}\xi = \sum_{\mathrm{i}=1}^{\mathrm{n_{GP}}} \mathrm{f}(\xi_\mathrm{i})\mathrm{w_i}$.

$\mathrm{n_{GP}}$	Gauss Points ξ_i	Weightfactors $\mathrm{w_i}$
1	0.0	2.0
2	$1/\sqrt{3}$	1.0
	$-1/\sqrt{3}$	1.0
3	$\sqrt{0.6}$	$5/9$
	0.0	$8/9$
	$-\sqrt{0.6}$	$5/9$
4	$\pm\sqrt{\frac{3+2\sqrt{1.2}}{7}}$	$\frac{1}{2} - \frac{1}{6\sqrt{1.2}}$
	$\pm\sqrt{\frac{3-2\sqrt{1.2}}{7}}$	$\frac{1}{2} + \frac{1}{6\sqrt{1.2}}$

Numerical integration in two dimensions for triangular elements via:

$$\mathrm{I} = \int_0^1 \int_0^{1-\xi} \mathrm{f}(\xi,\eta)\, \det(\underline{\boldsymbol{J}})\, \mathrm{d}\xi \mathrm{d}\eta \approx \sum_{\mathrm{i}=1}^{\mathrm{n_{GP}}} \mathrm{f}(\xi_\mathrm{i},\eta_\mathrm{i})\, \det[\underline{\boldsymbol{J}}(\xi_\mathrm{i},\eta_\mathrm{i})]\, \mathrm{w_i} \tag{310}$$

Table 3: Numerical quadrature of (310) with $\mathrm{n_{GP}}$ Gauss points.

$\#\mathrm{n_{GP}}$	Degree of precision	ξ_i	η_i	$\mathrm{w_i}$
1	1	$\xi_1 = 0.3333633333333$	$\eta_1 = \xi_1$	$\mathrm{w}_1 = 0.5$
3	2	$\xi_1 = 0.1666666666667$ $\xi_2 = 0.6666666666667$ $\xi_3 = \xi_1$	$\eta_1 = \xi_1$ $\eta_2 = \xi_1$ $\eta_3 = \xi_2$	$\mathrm{w}_1 = 0.3333333333333$ $\mathrm{w}_2 = \mathrm{w}_1$ $\mathrm{w}_3 = \mathrm{w}_1$
7	5	$\xi_1 = 0.1012865073235$ $\xi_2 = 0.7974269853531$ $\xi_3 = \xi_1$ $\xi_4 = 0.4701420641051$ $\xi_5 = \xi_4$ $\xi_6 = 0.0597158717898$ $\xi_7 = 0.3333333333333$	$\eta_1 = \xi_1$ $\eta_2 = \xi_1$ $\eta_3 = \xi_2$ $\eta_4 = \xi_6$ $\eta_5 = \xi_4$ $\eta_6 = \xi_4$ $\eta_7 = \xi_7$	$\mathrm{w}_1 = 0.1259391805448$ $\mathrm{w}_2 = \mathrm{w}_1$ $\mathrm{w}_3 = \mathrm{w}_1$ $\mathrm{w}_4 = 0.1323941527885$ $\mathrm{w}_5 = \mathrm{w}_4$ $\mathrm{w}_6 = \mathrm{w}_4$ $\mathrm{w}_7 = 0.225$
13	7	$\xi_1 = 0.0651301029022$ $\xi_2 = 0.8697397941956$ $\xi_3 = \xi_1$ $\xi_4 = 0.3128654960049$ $\xi_5 = 0.6384441885698$ $\xi_6 = 0.0486903154253$ $\xi_7 = \xi_5$ $\xi_8 = \xi_4$ $\xi_9 = \xi_6$ $\xi_{10} = 0.2603459660790$ $\xi_{11} = 0.4793080678419$ $\xi_{12} = \xi_{10}$ $\xi_{13} = 0.3333333333333$	$\eta_1 = \xi_1$ $\eta_2 = \xi_1$ $\eta_3 = \xi_2$ $\eta_4 = \xi_6$ $\eta_5 = \xi_4$ $\eta_6 = \xi_5$ $\eta_7 = \xi_6$ $\eta_8 = \xi_5$ $\eta_9 = \xi_4$ $\eta_{10} = \xi_{10}$ $\eta_{11} = \xi_{10}$ $\eta_{12} = \xi_{11}$ $\eta_{13} = \xi_{13}$	$\mathrm{w}_1 = 0.0533472356088$ $\mathrm{w}_2 = \mathrm{w}_1$ $\mathrm{w}_3 = \mathrm{w}_1$ $\mathrm{w}_4 = 0.0771137608903$ $\mathrm{w}_5 = \mathrm{w}_4$ $\mathrm{w}_6 = \mathrm{w}_4$ $\mathrm{w}_7 = \mathrm{w}_4$ $\mathrm{w}_8 = \mathrm{w}_4$ $\mathrm{w}_9 = \mathrm{w}_4$ $\mathrm{w}_{10} = 0.1756152574332$ $\mathrm{w}_{11} = \mathrm{w}_{10}$ $\mathrm{w}_{12} = \mathrm{w}_{10}$ $\mathrm{w}_{13} = -0.1495700444677$

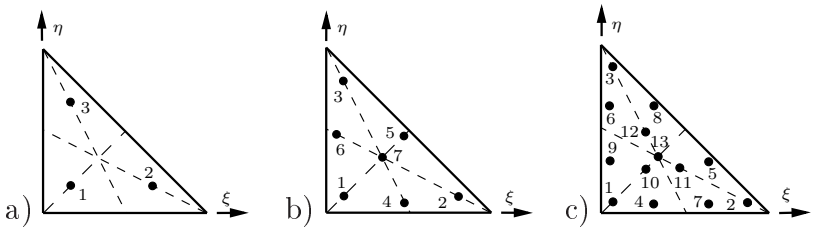


Figure 40: Integration points for a) 3-, b) 7-, and c) 13- Gauss point quadrature.

Numerical integration in two dimensions for quadrilateral elements:

$$\begin{aligned} \mathrm{I} = & \int_{(\mathrm{x})} \int_{(\mathrm{y})} \mathrm{f}(\mathrm{x},\mathrm{y})\, \mathrm{dy}\, \mathrm{dx} = \int_{-1}^{1} \int_{-1}^{1} \mathrm{f}(\xi,\eta)\, \det(\underline{\boldsymbol{J}})\, \mathrm{d}\xi\, \mathrm{d}\eta \\ \approx & \sum_{\mathrm{i}=1}^{\mathrm{n_{GP}}} \mathrm{f}(\xi_{\mathrm{i}},\eta_{\mathrm{i}})\, \det[\underline{\boldsymbol{J}}(\xi_{\mathrm{i}},\eta_{\mathrm{i}})]\, \widetilde{\mathrm{w}}_{\mathrm{i}}\,. \end{aligned} \tag{311}$$

The Gauss points and the values of the weight factors can be computed by the method discussed in the context of the one-dimensional Gauss point quadrature. This is termed "direct method" and yields, e.g. for

$$\left.\begin{aligned} \xi_1 &= \sqrt{\frac{2}{3}}\,, & \eta_1 &= 0 \\ \xi_2 &= -\frac{1}{\sqrt{6}}\,, & \eta_2 &= \frac{1}{\sqrt{2}} \\ \xi_3 &= -\frac{1}{\sqrt{6}}\,, & \eta_3 &= \frac{1}{\sqrt{2}} \end{aligned}\right\} \quad \widetilde{\mathrm{w}}_1 = \widetilde{\mathrm{w}}_2 = \widetilde{\mathrm{w}}_3 = \frac{4}{3}\,. \tag{312}$$

Further integration formulae of the direct method are given in Dhatt *et al.* (1984). More suitable for quadrilateral elements is the bi-directional method, which has less geometrical bias than the direct methods. Bi-directional integration means that

$$\mathrm{I} = \int_{(\mathrm{x})} \int_{(\mathrm{y})} \mathrm{f}(\mathrm{x},\mathrm{y})\, \mathrm{dy}\, \mathrm{dx} \approx \sum_{\mathrm{i}=1}^{\mathrm{n_{GP\xi}}} \sum_{\mathrm{j}=1}^{\mathrm{n_{GP\eta}}} \mathrm{f}(\xi_{\mathrm{i}},\eta_{\mathrm{j}})\, \det[\underline{\mathbf{J}}(\xi_{\mathrm{i}},\eta_{\mathrm{j}})]\, \mathrm{w}_{\mathrm{i}}\, \mathrm{w}_{\mathrm{j}}\,. \tag{313}$$

Here, we selected a one-dimensional integration formula for ξ and η, with $\mathrm{n_{GP\xi}}$ and $\mathrm{n_{GP\xi}}$ integration points in ξ and η directions, respectively, see Table 2.

Numerical integration in three dimensions for brick-type elements:

$$\begin{aligned} \mathrm{I} = & \int_{\mathrm{x}} \int_{\mathrm{y}} \int_{\mathrm{z}} \mathrm{f}(\mathrm{x},\mathrm{y},\mathrm{z})\, \mathrm{dz}\, \mathrm{dy}\, \mathrm{dx} = \int_{-1}^{1} \int_{-1}^{1} \int_{-1}^{1} \mathrm{f}(\xi,\eta,\zeta)\, \det(\underline{\boldsymbol{J}})\, \mathrm{d}\xi\, \mathrm{d}\eta\, \mathrm{d}\zeta \\ \approx & \sum_{\mathrm{i}=1}^{\mathrm{n_{GP}}} \mathrm{f}(\xi_{\mathrm{i}},\eta_{\mathrm{i}},\zeta_{\mathrm{i}}))\, \det[\underline{\boldsymbol{J}}(\xi_{\mathrm{i}},\eta_{\mathrm{i}},\zeta_{\mathrm{i}})]\, \widetilde{\mathrm{w}}_{\mathrm{i}} \end{aligned} \tag{314}$$

Using the same arguments as for the numerical integration of quadrilateral elements we apply a tri-directional method. In this case the integral in (314) is approximated via

$$\mathrm{I} \approx \sum_{\mathrm{i}=1}^{\mathrm{n_{GP\xi}}} \sum_{\mathrm{j}=1}^{\mathrm{n_{GP\eta}}} \sum_{\mathrm{k}=1}^{\mathrm{n_{GP\zeta}}} \mathrm{f}(\xi_{\mathrm{i}},\eta_{\mathrm{j}},\zeta_{\mathrm{k}})\, \det[\underline{\mathbf{J}}(\xi_{\mathrm{i}},\eta_{\mathrm{j}},\zeta_{\mathrm{k}})]\, \mathrm{w}_{\mathrm{i}}\, \mathrm{w}_{\mathrm{j}}\, \mathrm{w}_{\mathrm{k}}\,. \tag{315}$$

The number of integration points in ξ, η and ζ direction are denoted by $\mathrm{n_{GP\xi}}, \mathrm{n_{GP\eta}}$ and $\mathrm{n_{GP\zeta}}$, respectively.

Table 4: Numerical quadrature of (316).

$\#n_{GP}$	Degree of precision	ξ_i	η_i	ζ_i	$\widetilde{w}_i$
1	1	$\xi_1 = 1/4$	$\eta_1 = 1/4$	$\zeta_1 = \xi_1$	$w_1 = 1/6$
4	2	$\xi_1 = a$ $\xi_2 = a$ $\xi_3 = a$ $\xi_4 = b$	$\eta_1 = a$ $\eta_2 = a$ $\eta_3 = b$ $\eta_4 = a$	$\zeta_1 = a$ $\zeta_2 = b$ $\zeta_3 = a$ $\zeta_4 = a$	$w_1 = 1/24$ $w_2 = w_1$ $w_3 = w_1$ $w_4 = w_1$
		$a = \frac{5-\sqrt{5}}{20}, \quad b = \frac{5+3\sqrt{5}}{20}$			
5	3	$\xi_1 = c$ $\xi_2 = d$ $\xi_3 = d$ $\xi_4 = d$ $\xi_4 = e$	$\eta_1 = c$ $\eta_2 = d$ $\eta_3 = d$ $\eta_4 = e$ $\eta_4 = d$	$\zeta_1 = c$ $\zeta_2 = d$ $\zeta_3 = e$ $\zeta_4 = d$ $\zeta_4 = d$	$w_1 = -2/15$ $w_2 = 3/40$ $w_3 = w_2$ $w_4 = w_2$ $w_5 = w_2$
		$c = \frac{1}{4}, \quad d = \frac{1}{6}, \quad e = \frac{1}{2}$			

Numerical integration in three dimensions of tetrahedral elements:

$$\begin{aligned} I &= \int_x \int_y \int_z f(x,y,z)\, dz\, dy\, dx \\ &= \int_0^1 \int_0^{1-\xi} \int_0^{1-\xi-\eta} f(\xi,\eta,\zeta)\, \det(\underline{\boldsymbol{J}})\, d\xi\, d\eta\, d\zeta \\ &\approx \sum_{i=1}^{n_{GP}} f(\xi_i,\eta_i,\zeta_i))\, \det[\underline{\boldsymbol{J}}(\xi_i,\eta_i,\zeta_i)]\, \widetilde{w}_i \end{aligned} \tag{316}$$

6.6 Quadrilateral Element

This section describes the formal derivation of the variational problem arising in linear elasticity and its Finite-Element discretization in the context of the isoparametric concept.

6.6.1 Variational Formulation In this section we deduce the weak form of the boundary value problem. In the framework of linear elastostatics our body of interest $\mathcal{B} \in \mathcal{R}^3$ is parametrized in $\boldsymbol{x} \in \mathcal{B}$. Starting with the balance of linear momentum we consider firstly the strong form of the boundary value problem

$$\operatorname{div}[\boldsymbol{\sigma}] + \bar{\boldsymbol{f}} = \boldsymbol{0} \; \forall \; \boldsymbol{x} \in \mathcal{B}\,, \tag{317}$$

wherein $\boldsymbol{\sigma}$ and $\bar{\boldsymbol{f}}$ denote the Cauchy-stress matrix and the volume force vector, respectively. In the context of the Finite-Element Method we take into account cartesian coordinates only, therefore we skip the base vectors of all tensorial coefficients and use matrix notation. We concentrate on small strains, thus, the strain matrix is defined as the symmetric part of the displacement gradient

$$\boldsymbol{\varepsilon} := \frac{1}{2}[\nabla \boldsymbol{u} + \nabla^T \boldsymbol{u}]\,. \tag{318}$$

For the complete description of the boundary value problem we need boundary conditions

$$\boldsymbol{u} = \bar{\boldsymbol{u}} \text{ on } \partial\mathcal{B}_u \quad \text{and} \quad \boldsymbol{t} = \boldsymbol{\sigma} \cdot \boldsymbol{n} = \bar{\boldsymbol{t}} \text{ on } \partial\mathcal{B}_\sigma. \tag{319}$$

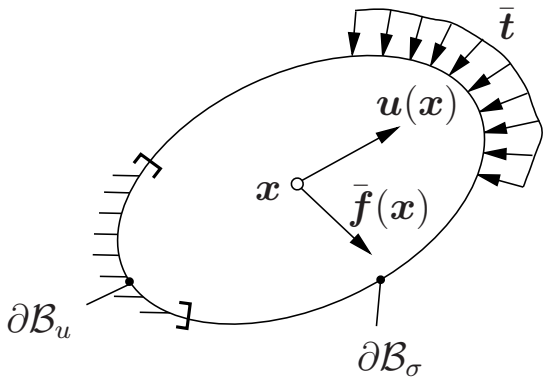


Figure 41: Illustration of the boundary value conditions

In order to obtain a principally solvable partial differential equation we apply the Galerkin method. Thus, we multiply the strong form with a test function $\delta\boldsymbol{u}$ and integrate over the domain $\mathcal{B}$ and obtain

$$G := -\int_{\mathcal{B}} (\operatorname{div}[\boldsymbol{\sigma}] \cdot \delta\boldsymbol{u} + \bar{\boldsymbol{f}} \cdot \delta\boldsymbol{u}) dv = 0. \tag{320}$$

Reformulating $\operatorname{div}\boldsymbol{\sigma} \cdot \delta\boldsymbol{u}$ and applying the divergence theorem leads to

$$\left.\begin{aligned} \sigma_{ij,j}\delta u_i &= (\sigma_{ij}\delta u_i)_{,j} - \sigma_{ij}\delta u_{i,j} \\ &= \operatorname{div}[\boldsymbol{\sigma}] \cdot \delta\boldsymbol{u} + \boldsymbol{\sigma} \cdot \operatorname{grad}[\delta\boldsymbol{u}] \\ \Leftrightarrow \operatorname{div}[\boldsymbol{\sigma}] \cdot \delta\boldsymbol{u} &= \operatorname{div}[\boldsymbol{\sigma}\,\delta\boldsymbol{u}] - \boldsymbol{\sigma} \cdot \operatorname{grad}[\delta\boldsymbol{u}] \end{aligned}\right\}. \tag{321}$$

Inserting this expression into equation (320) leads to

$$G = -\int_{\mathcal{B}} \mathrm{div}[\boldsymbol{\sigma}\,\delta\boldsymbol{u}]dv + \int_{\mathcal{B}} \mathrm{grad}[\delta\boldsymbol{u}]\,\cdot\,\boldsymbol{\sigma}dv - \int_{\mathcal{B}} \delta\boldsymbol{u}\cdot\bar{\boldsymbol{f}}dv = 0\,. \tag{322}$$

Applying the Gauss integral and considering the Cauchy theorem $\boldsymbol{\sigma}\,\boldsymbol{n} = \boldsymbol{t}$ we obtain

$$-\int_{\mathcal{B}} \mathrm{div}[\boldsymbol{\sigma}\,\delta\boldsymbol{u}]dv = -\int_{\partial\mathcal{B}} \delta\boldsymbol{u}\cdot(\boldsymbol{\sigma}\,\boldsymbol{n})da = -\int_{\partial\mathcal{B}} \delta\boldsymbol{u}\cdot\boldsymbol{t}\;da\,. \tag{323}$$

Setting this relation into the previous equation yields

$$G = \int_{\mathcal{B}} \boldsymbol{\sigma}\cdot\mathrm{grad}[\delta\boldsymbol{u}] - \int_{\mathcal{B}} \delta\boldsymbol{u}\cdot\bar{\boldsymbol{f}}dv - \int_{\partial\mathcal{B}_t} \delta\boldsymbol{u}\cdot\bar{\boldsymbol{t}}da = 0\,. \tag{324}$$

Now we compute the virtual strains which are given by

$$\delta\boldsymbol{\varepsilon} = \frac{1}{2}(\mathrm{grad}[\delta\boldsymbol{u}] + \mathrm{grad}^T[\delta\boldsymbol{u}]) = \mathrm{grad}^{sym}[\delta\boldsymbol{u}]\,. \tag{325}$$

Evaluation of the balance of angular momentum and consideration of the local form leads to the symmetry of the stress matrix

$$\boldsymbol{\sigma} = \boldsymbol{\sigma}^T \tag{326}$$

and therefore the equation holds

$$\boldsymbol{\sigma}\cdot\mathrm{grad}[\delta\boldsymbol{u}] = \boldsymbol{\sigma}\cdot\mathrm{grad}^{sym}[\delta\boldsymbol{u}] = \boldsymbol{\sigma}\cdot\delta\boldsymbol{\varepsilon}\,. \tag{327}$$

Finally we obtain the weak form of the boundary value problem

$$G = \underbrace{\int_{\mathcal{B}} \delta\boldsymbol{\varepsilon} : \boldsymbol{\sigma}dv}_{G_{int}} \underbrace{- \int_{\mathcal{B}} \delta\boldsymbol{u}\cdot\bar{\boldsymbol{f}}dv - \int_{\partial\mathcal{B}_t} \delta\boldsymbol{u}\cdot\bar{\boldsymbol{t}}\;da}_{G_{ext}} = 0\,, \tag{328}$$

wherein G_{int} and G_{ext} represent the internal and external part, respectively.

6.6.2 Finite-Element-Discretization In the context of the Finite-Element-Method is based on the approximation of the primary variables

$$\boldsymbol{u}^h(\boldsymbol{\xi}) = \sum_{I=1}^{n_{ele}} N_I(\boldsymbol{\xi}) \boldsymbol{d}_I . \tag{329}$$

Herein, $\boldsymbol{\xi} = [\xi, \eta]^T$ denotes the vector of parametric coordinates. With view to the isoparametric concept the geometry is approximated in the same way

$$\boldsymbol{x}^h(\boldsymbol{\xi}) = \sum_{I=1}^{n_{ele}} N_I(\boldsymbol{\xi}) \boldsymbol{x}_I \, . \tag{330}$$

Please note, that superscript h denotes approximated quantities. Exemplarily we consider the discretization of a typical 2D- four node isoparametric element. The transformation of the parametric domain into the physical domain can be interpreted as a transformation from the reference configuration to the actual configuration, see Fig. 42.

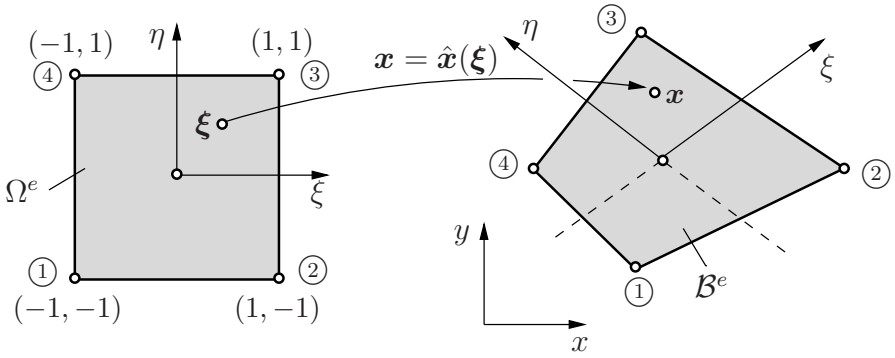


Figure 42: Illustration of the transformatiom from the parametric domain into the physical domain.

The bilinear ansatz functions $\boldsymbol{N}_I$ in (329) and (330) interpolate between the quantities at the nodes of the element and are given for quadrilateral elements by

$$N_I(\xi, \eta) = \frac{1}{4}(1 + \xi\xi_I)(1 + \eta\eta_I) \, . \tag{331}$$

If we insert the parametric coordinates of the nodes we obtain the explicit ansatz functions

$$\left.\begin{aligned} N_1 &= \frac{1}{4}(1-\xi)(1-\eta) \\ N_2 &= \frac{1}{4}(1+\xi)(1-\eta) \\ N_3 &= \frac{1}{4}(1+\xi)(1+\eta) \\ N_4 &= \frac{1}{4}(1-\xi)(1+\eta) \end{aligned}\right\} . \tag{332}$$

In Fig. 42 you see a graphical illustration of the bilinear ansatz functions

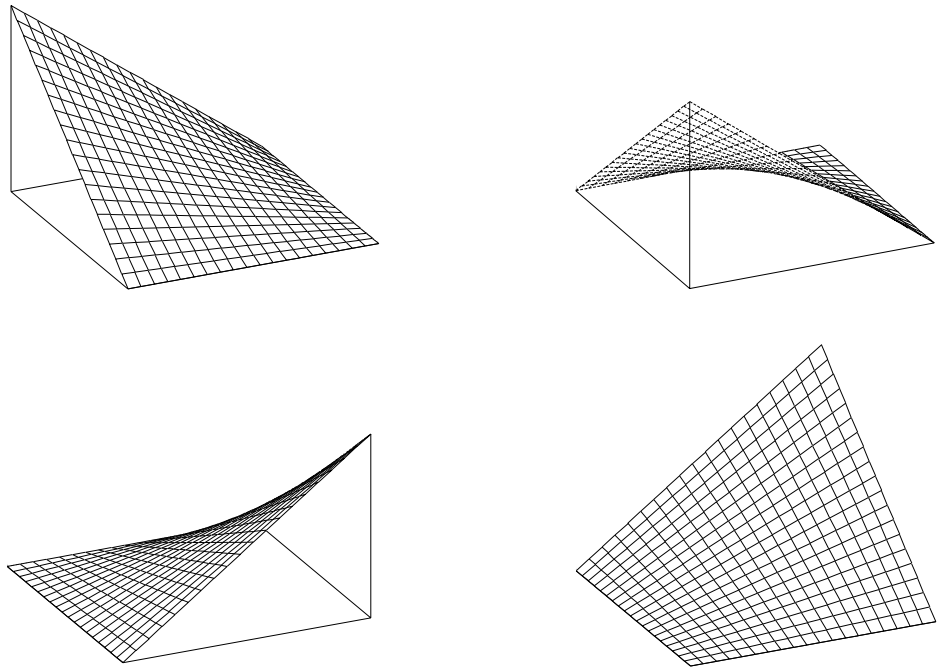


Figure 43: Bilinear ansatz functions for a four node element

Due to the fact that the stress and strain matrices are symmetric, it is not necessary to consider all 9 coefficients. For this purpose we now switch to the reduced matrix notation, where the stress and strain matrix are denoted as vectors. Then we obtain the relation

$$\boldsymbol{\sigma} = \begin{bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \end{bmatrix} = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \dfrac{1-2\nu}{2} \end{bmatrix} \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ 2\varepsilon_{xy} \end{bmatrix} = \mathbb{C}\,\boldsymbol{\varepsilon}, \tag{333}$$

wherein the elasticity matrix is denoted by $\mathbb{C}$. Since the strains are expressed by the symmetric part of the displacement gradient $\boldsymbol{\varepsilon} := \frac{1}{2}(\text{grad}[\boldsymbol{u}] + \text{grad}^T[\boldsymbol{u}])$ we obtain

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \gamma_{xy} \end{bmatrix} = \begin{bmatrix} u_{x,x} \\ u_{y,y} \\ u_{x,y} + u_{y,x} \end{bmatrix}, \tag{334}$$

and note that we are able to express the stresses via derivation of the displacements with respect to $\boldsymbol{x}$. Herein, the displacements $\boldsymbol{u}$ have to be approximated by

$$\boldsymbol{u}^h = \sum_{I=1}^{4} N_I \boldsymbol{d}_I = \boldsymbol{N}^e \boldsymbol{d}^e \tag{335}$$

with $\boldsymbol{N}$ being the ansatzfunction matrix defined by

$$\left[\begin{array}{c} u_x^h \\ u_y^h \end{array}\right] = \left[\begin{array}{cc|cc|cc|cc} N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 & 0 \\ 0 & N_1 & 0 & N_2 & 0 & N_3 & 0 & N_4 \end{array}\right] \left[\begin{array}{c} d_{1x} \\ d_{1y} \\ \hline d_{2x} \\ d_{2y} \\ \hline d_{3x} \\ d_{3y} \\ \hline d_{4x} \\ d_{4y} \end{array}\right] . \tag{336}$$

In the same way we are able to compute the approximation of the virtual displacements

$$\delta \boldsymbol{u}^h = \sum_{I=1}^{4} N_I(\xi) \delta \boldsymbol{d}_I = \boldsymbol{N}^e \delta \boldsymbol{d}^e \,. \tag{337}$$

For the approximation of all solution fields occurring in the weak form of equilibrium we need the approximation of strains, thus, we insert the displacement approximations into (334) and obtain

$$\boldsymbol{\varepsilon}^h = \sum_{I=1}^{4} N_{I,x}(\boldsymbol{\xi}) \boldsymbol{d}_I = \sum_{I=1}^{4} \boldsymbol{B}_I(\boldsymbol{\xi}) \boldsymbol{d}_I \,. \tag{338}$$

Herein, we introduced the $\boldsymbol{B}_I$-matrix at the nodes given by

$$\boldsymbol{B}_I = \left[\begin{array}{cc} N_{I,x} & 0 \\ 0 & N_{I,y} \\ N_{I,y} & N_{I,x} \end{array}\right] . \tag{339}$$

This notation can be abbreviated by the introduction of a $\boldsymbol{B}^e$-matrix per element following

$$\boldsymbol{\varepsilon}^h = \boldsymbol{B}^e \boldsymbol{d}^e = \left[\begin{array}{cc|cc|cc|cc} N_{1,x} & 0 & N_{2,x} & 0 & N_{3,x} & 0 & N_{4,x} & 0 \\ 0 & N_{1,y} & 0 & N_{2,y} & 0 & N_{3,y} & 0 & N_{4,y} \\ N_{1,y} & N_{1,x} & N_{2,y} & N_{2,x} & N_{3,y} & N_{3,x} & N_{4,y} & N_{4,x} \end{array}\right] \left[\begin{array}{c} d_{1x} \\ d_{1y} \\ \hline d_{2x} \\ d_{2y} \\ \hline d_{3x} \\ d_{3y} \\ \hline d_{4x} \\ d_{4y} \end{array}\right] . \tag{340}$$

Analogously we obtain for the virtual strains the approximation

$$\delta \boldsymbol{\varepsilon}^h = \sum_{I=1}^{4} \boldsymbol{B}_I \delta \boldsymbol{d}_I = \boldsymbol{B}^e \delta \boldsymbol{d}^e \,. \tag{341}$$

Due to the fact that the ansatz functions depend on the parametric coordinates $\boldsymbol{\xi}$, we have to apply the chain rule for the computation of the derivatives of N with respect to $\boldsymbol{x}$ in (339) and obtain

$$\frac{\partial N_I}{\partial \boldsymbol{x}} = \frac{\partial N_I}{\partial \boldsymbol{\xi}} \frac{\partial \boldsymbol{\xi}}{\partial \boldsymbol{x}} = \frac{\partial N_I}{\partial \boldsymbol{\xi}} \boldsymbol{J}^{-1} \,. \tag{342}$$

Herein we introduced the inverse of the Jacobi matrix $\boldsymbol{J}$. The Jacobi matrix $\boldsymbol{J} = \dfrac{\partial \boldsymbol{x}}{\partial \boldsymbol{\xi}}$ transforms quantities in the parametric to the physical coordinate system. The partial derivatives occurring in (342) can be computed by

$$\begin{bmatrix} N_{I,\xi} \\ \\ N_{I,\eta} \end{bmatrix} = \begin{bmatrix} \dfrac{1}{4}\xi_I(1+\eta\eta_I) \\ \\ \dfrac{1}{4}\eta_I(1+\xi\xi_I) \end{bmatrix} . \tag{343}$$

Inserting (333) and all approximated fields into the weak form we obtain the approximated weak form for one finite element

$$\left.\begin{aligned} G_e \approx G_e^h &= \int_{\mathcal{B}} \boldsymbol{B}^e \delta\boldsymbol{d}^e \mathbb{C} \boldsymbol{B}^e \boldsymbol{d}^e \, dv - \int_{\mathcal{B}} \boldsymbol{N}^e \delta\boldsymbol{d}^e \cdot \boldsymbol{f} \, dv + \int_{\partial\mathcal{B}} \boldsymbol{N}^e \delta\boldsymbol{d}^e \cdot \boldsymbol{t} \, da \\ &= \delta\boldsymbol{d}^{eT} \left[\underbrace{\int_{\mathcal{B}} \boldsymbol{B}^{eT} \mathbb{C} \boldsymbol{B}^e dv}_{\boldsymbol{k}^e} \boldsymbol{d}^e - \underbrace{\int_{\mathcal{B}} \boldsymbol{N}^{eT} \boldsymbol{f} dv + \int_{\partial\mathcal{B}} \boldsymbol{N}^{eT} \boldsymbol{t} da}_{\boldsymbol{r}^e} \right] \\ &= (\delta\boldsymbol{d}^{eT}) \boldsymbol{k}^e \boldsymbol{d}^e - \boldsymbol{r}^e = 0 \end{aligned}\right\} . \tag{344}$$

Herein $\boldsymbol{k}^e$ denotes the element stiffness matrix and $\boldsymbol{r}^e$ is the abbreviation for the element force vector.

6.7 Beam Elements

In this chapter we consider the finite element formulation for beams under bending conditions. Here, the theory according to Euler-Bernoulli is treated.

6.7.1 Beam Theory according to Euler-Bernoulli for simple bending In this section we consider the Euler-Bernoulli beam element under pure bending conditions, i.e. without axial forces.

Continuous Formulation: The equilibrium conditions can be derived from the infinitesimal beam element with the length dx. For the axial force $N(x)$, shear force $Q(x)$ and the moment $M(x)$, the following known differential equations are given

$$\left.\begin{array}{rcl} N(x) & := & 0 \\ \dfrac{dQ(x)}{dx} & = & -p(x) \\ Q(x) & = & \dfrac{dM(x)}{dx} \end{array}\right\} \tag{345}$$

with the loading function $p(x)$. Inserting equation $(345)_3$ in $(345)_2$ yields the linear, inhomogeneous, common differential equation of second order

$$\frac{d^2M(x)}{dx^2} = -p(x)\ . \tag{346}$$

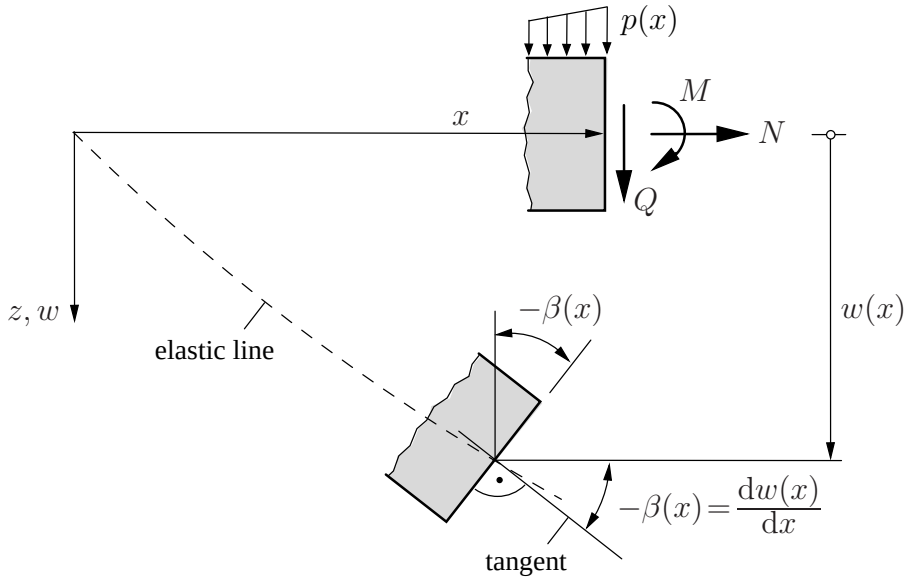


Figure 44: Beam according to Euler-Bernoulli theory

The essential kinematic assumptions of the Euler-Bernoulli beam theory are

1. constant cross-section and
2. "normal remains normal"

The axial displacements in direction of the beam axis over the cross-section result from the assumption of constant cross-sections in

$$u(x,z) = z \cdot \beta(x)\,. \tag{347}$$

For the definition of geometric quantities associated to the beam theory considered here, see Figure 44. The second Bernoulli-hypothesis yields

$$-\beta(x) = w'(x)\,, \tag{348}$$

with the deflections w. Hence, the strains in beam axis can be computed from the derivatives of the elastic line

$$\varepsilon_x = u_{,x} = z \cdot \beta' = -z \cdot w_{,xx} \tag{349}$$

and in simplified notation we write

$$\varepsilon_x = -z w''(x)\,. \tag{350}$$

At first, we only analyze pure bending problems, i.e. no influence of axial force loads in beam axis are considered. The integral over the cross-section yields the bending moment

$$M_y(x) = \int_A \sigma_x(x,z) \cdot z\, dA\,, \tag{351}$$

wherein the normal stresses are calculated by Hooke's law

$$\sigma_x = E \cdot \varepsilon_x\,. \tag{352}$$

With equation (350) and (351) it follows

$$M_y(x) = \int_A -Ez^2 w''(x) dA = -Ew''(x) \int_A z^2 dA = -EIw''(x)\,. \tag{353}$$

We obtain the shear force from the equilibrium equation $(345)_2$. According to Bernoulli, the associated potential for a beam of length l is

$$\Pi = \frac{1}{2}\int_l EI(w'')^2 dx - \int_l p\, w\ . \tag{354}$$

Ansatz equations: We consider a typical element k with the node numbers i and j for the FE-discretization, see Figure 45.

The second derivatives of the deflection w appear in equation (354), i.e. the constitutive equation have to be at least two-times continuously differentiable. We choose

$$\boxed{\tilde{w}(\tilde{x}) = C_0 + C_1\tilde{x} + C_2\tilde{x}^2 + C_3\tilde{x}^3}\,. \tag{355}$$

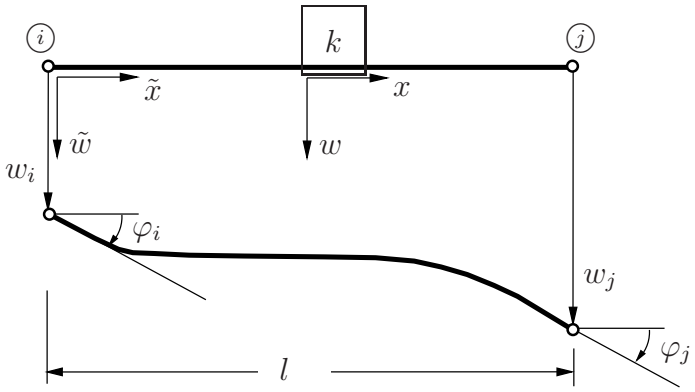


Figure 45: Definition of the beam element.

This assumption is

1. linearly independent (because we consider powers of $\tilde{x}$)
2. two-times continuously differentiable
3. geometrically conform (the completion of the boundary conditions, w und w', is possible). Geometrically conform means, that the deflection and curvature at the boundary to the neighbour elements are continuous. The order of the differential equation in (353) is $2n = 4$, i.e. $n = 2$. The order of the derivatives of the essential boundary- and transition conditions is $n - 1 = 1$ with the Ritz method.
4. sufficient for the description of, at least constant, internal force variables ($M = -EIw''$ requires only quadratic ansatzfunctions for w for this purpose)
5. sufficient for the description of rigid body deflections (translation with $\tilde{w} = c_0$ and rotation with $\tilde{w} = C_0 + C_1\tilde{x}$).

The computation of the derivatives of $\tilde{w}$ yields

$$\left.\begin{array}{rcl} \tilde{w}'(\tilde{x}) & = & C_1 + 2C_2\tilde{x} + 3C_3\tilde{x^2} \\ \tilde{w}''(\tilde{x}) & = & 2C_2 + 6C_3\tilde{x} \\ \tilde{w}'''(\tilde{x}) & = & 6C_3 \\ \tilde{w}''''(\tilde{x}) & = & 0 \end{array}\right\} . \tag{356}$$

Due to the fact that the dependence $Q(x) \sim w'''(x)$ exists, we find that due to equation $(356)_3$ the ansatz delivers the exact solution if we do not apply uniform loads. This coherence is also justifiable as follows: Because the ansatz (355) satisfies the homogeneous differential equation $w'''' = 0$, it describes a solution of equilibrium equation and delivers the exact solution in this case.

We use Hermite polynomials in the dimensionless quantity ξ as ansatz function. The ansatz appears in the form

$$w(\xi) = N_1(\xi)w_1 + \bar{N}_1(\xi)\varphi_1 + N_2(\xi)w_2 + \bar{N}_2(\xi)\varphi_2 = \boldsymbol{N}^e\boldsymbol{d}^e$$

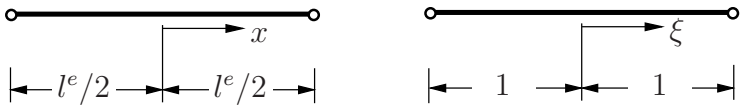


Figure 46: Implementation of natural coordinates.

and has to satisfy the boundary conditions

$$\left.\begin{array}{rclrcl} w(\xi=-1) & = & w_1 & w_{,x}(\xi=-1) & = & \varphi_1 \\ w(\xi=1) & = & w_2 & w_{,x}(\xi=1) & = & \varphi_2 \end{array}\right\} . \tag{357}$$

Herefrom we obtain

$$\left.\begin{array}{rclcrcl} N_1 & = & \frac{1}{4}(2-3\xi+\xi^3) & , & N_2 & = & \frac{1}{4}(2+3\xi-\xi^3) \\ \bar{N}_1 & = & \frac{1}{4}(1-\xi-\xi^2+\xi^3)\frac{l^e}{2} & , & \bar{N}_2 & = & \frac{1}{4}(-1-\xi+\xi^2+\xi^3)\frac{l^e}{2} \end{array}\right\} . \tag{358}$$

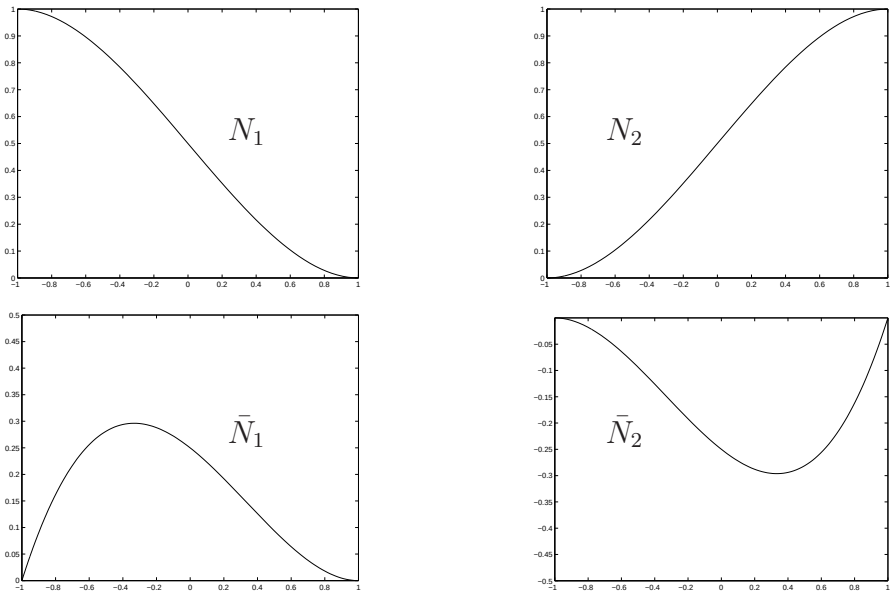


Figure 47: Hermite-polynomials

By applying the chain rule

$$w' = \frac{dw}{dx} = \frac{dw}{d\xi}\frac{d\xi}{dx} = \frac{2}{l}\frac{dw}{d\xi}$$

we receive

$$w' = N_{1,\xi}\frac{2}{l}w_1 + \bar{N}_{1,\xi}\frac{2}{l}\varphi_1 + N_{2,\xi}\frac{2}{l}w_2 + \bar{N}_{2,\xi}\frac{2}{l}\varphi_2 .$$

We obtain the second derivatives analogously with

$$w'' = \frac{d^2w}{dx^2} = \frac{d^2w}{d\xi^2}\frac{d^2\xi}{dx^2} = \frac{4}{l^2}\frac{d^2w}{d\xi^2}$$

to

$$w'' = \underbrace{\frac{4}{l^2}\left[\begin{array}{cccc} N_{1,\xi\xi} & \bar{N}_{1,\xi\xi} & N_{2,\xi\xi} & \bar{N}_{2,\xi\xi} \end{array}\right]}_{\boldsymbol{B}^e} \left[\begin{array}{c} w_1 \\ \varphi_1 \\ w_2 \\ \varphi_2 \end{array}\right] = \boldsymbol{B}^e \boldsymbol{d}^e \tag{359}$$

with the element displacement vector $\boldsymbol{d}^{eT} = [\ w_1 \quad \varphi_1 \quad w_2 \quad \varphi_2\ ]$.

Discrete formulation: Starting from the minimum of the total potenial given in (354) we obtain as a neccessary condition

$$\delta\Pi = \int_l \delta w'' EI w'' dx - \int_l \delta w\ p\ dx = 0$$

and the description for a typical element

$$\delta\Pi^{(e)} = \int_{l^e} \delta w''(x) EI w''(x) dx - \int_{l^e} \delta w(x)\ p(x) dx\ ,$$

respectively. The transformation to the natural coordinates delivers with equation (359) and
$\delta w'' = \boldsymbol{B}^e \delta\, \boldsymbol{d}^e$ and $dx = \dfrac{l}{2} d\xi$

$$\delta\Pi^{(e)} = \delta\boldsymbol{d}^{eT} \int_{-1}^{1} \boldsymbol{B}^{eT} EI \boldsymbol{B}^e \frac{l}{2}\, d\xi\, \boldsymbol{d}^e - \delta\boldsymbol{d}^{eT} \int_{-1}^{1} \boldsymbol{N}^{eT} p \frac{l}{2}\, d\xi\,.$$

Herefrom, we identify the element stiffness matrix $\boldsymbol{k}^e$ and the element loading vector $\boldsymbol{p}^e$

$$\boldsymbol{k}^e := \int_{-1}^{1} \boldsymbol{B}^{eT} EI \boldsymbol{B}^e \frac{l}{2}\, d\xi \qquad \text{and} \qquad \boldsymbol{p}^e = \int_{-1}^{1} \boldsymbol{N}^{eT} p \frac{l}{2}\, d\xi\,.$$

From this it follows, that

$$\boldsymbol{k}^e = \frac{EI}{l^3} \left[\begin{array}{cccc} 12 & 6l & -12 & 6l \\ 6l & 4l^2 & -6l & 4l^2 \\ -12 & -6l & 12 & -6l \\ 6l & 4l^2 & -6l & 4l^2 \end{array}\right]$$

and for $p(\xi) = \text{constant}$, we obtain

$$\boldsymbol{p}^e = \frac{pl}{2} \begin{bmatrix} 1 \\ l/6 \\ 1 \\ -l/6 \end{bmatrix} .$$

The discrete total problem arises from

$$\delta\Pi = \sum_e \{\delta \boldsymbol{d}^{eT} \boldsymbol{k}^{(e)} \boldsymbol{d}^e - \delta \boldsymbol{d}^{eT} \boldsymbol{p}^e\} = 0 \qquad \forall \quad \delta \boldsymbol{d}^e$$

and for the global problem we obtain

$$\boldsymbol{K}\boldsymbol{D} = \boldsymbol{P} \quad \text{with} \quad \boldsymbol{K} = \mathop{\mathsf{A}}_{e=1}^{\text{nele}} \boldsymbol{k}^e$$

whereby $\boldsymbol{K}$ is the global stiffness matrix, $\boldsymbol{P}$ the global loading vector and $\boldsymbol{D}$ the global displacement vector.

Calculation of the internal force variables: We compute the moment M from the constitutive law

$$M(\xi) = -EIw(\xi)_{,xx} = -EI\boldsymbol{B}(\xi)\boldsymbol{d}^e \tag{360}$$

Please note, that the location of evaluation has to be taken into account. For the definition of the internal force variables

$$Q_l = -R_i\,, \quad M_l = M_i\,, \quad Q_r = R_j\,, \quad M_r = -M_j \tag{361}$$

compare Figure 48.

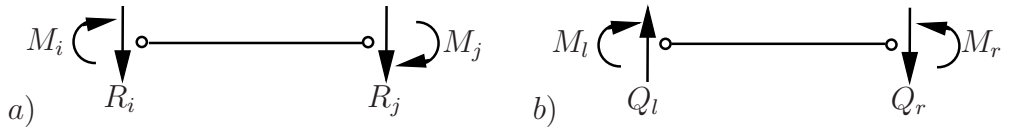


Figure 48: Definition of the internal force variables with respect to a) Finite-element-method and b) Statics.

The internal force variables are given by

$$\left.\begin{aligned} M_i = M(\xi = -1) &= -\frac{EI}{l^2}[-6, -4l, 6, -2l]\boldsymbol{d}^e \\ M_j = M(\xi = 1) \quad &= -\frac{EI}{l^2}[6, 2l, -6, 4l]\boldsymbol{d}^e \end{aligned}\right\}, \tag{362}$$

whereas the shear force is not given by a constitutive law but computed via equilibrium

$$Q = M' = -\frac{EI}{l^2}\frac{2}{l}[6, 3l, -6, 3l]\boldsymbol{d}^e \,. \tag{363}$$

Complete beam element

Here we consider a complete beam element formulation, where also axial loads are taken into account, cf. Fig. 49.

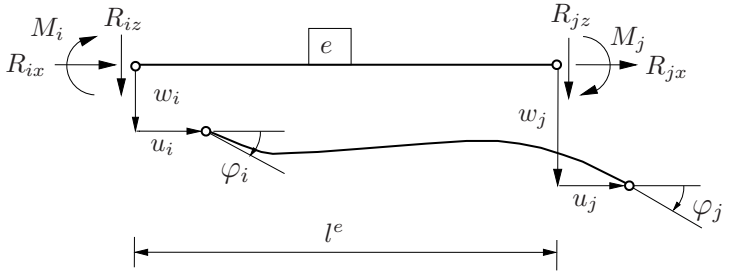


Figure 49: Completed flat beam element.

The flat Bernoulli beam with axial load and constant uniform load in $x-$ and $z-$ direction delivers

$$\begin{bmatrix} R_{ix} \\ R_{iz} \\ M_i \\ R_{jx} \\ R_{jz} \\ M_j \end{bmatrix}_L = \frac{E}{l} \begin{bmatrix} A & 0 & 0 & -A & 0 & 0 \\ 0 & 12\frac{I}{l^2} & 6\frac{I}{l} & 0 & -12\frac{I}{l^2} & 6\frac{I}{l} \\ 0 & 6\frac{I}{l} & 4I & 0 & -6\frac{I}{l} & 4I \\ -A & 0 & 0 & A & 0 & 0 \\ 0 & -12\frac{I}{l^2} & -6\frac{I}{l} & 0 & 12\frac{I}{l^2} & -6\frac{I}{l} \\ 0 & 6\frac{I}{l} & 4I & 0 & -6\frac{I}{l} & 4I \end{bmatrix}_L \begin{bmatrix} u_i \\ w_i \\ \varphi_i \\ u_j \\ w_j \\ \varphi_j \end{bmatrix}_L - \frac{l}{2} \begin{bmatrix} q_x \\ q_z \\ q_z\frac{l}{6} \\ q_x \\ q_z \\ -q_z\frac{l}{6} \end{bmatrix}_L \tag{364}$$

$$\boldsymbol{r}^e_L = \boldsymbol{k}^e_L \boldsymbol{d}^e_L - \boldsymbol{p}^e_L \tag{365}$$

in matrix notation. Since the above presented relations are applied to the description in the local coordinate system, a description with respect to a global coordinate system is neccessary for general boundary value problems.

→ Transformation of the element relations is neccessary, cf. Figure 50

$$\begin{bmatrix} x_L \\ z_L \end{bmatrix} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x_G \\ z_G \end{bmatrix} \tag{366}$$

The transformation for u, w and R_x, R_z respectively follows analogously, thus, we obtain

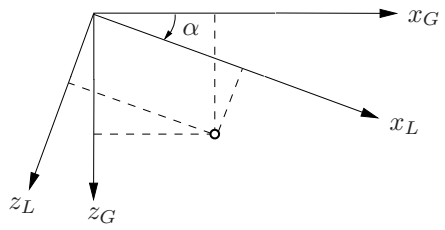


Figure 50: Transformation of coordinates

$$\begin{bmatrix} u_i \\ w_i \\ \varphi_i \\ u_j \\ w_j \\ \varphi_j \end{bmatrix}_L = \begin{bmatrix} c & s & 0 & 0 & 0 & 0 \\ -s & c & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & c & s & 0 \\ 0 & 0 & 0 & -s & c & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_i \\ w_i \\ \varphi_i \\ u_j \\ w_j \\ \varphi_j \end{bmatrix}_G \tag{367}$$

with the abbreviation

$$\begin{aligned} c &:= \cos\alpha \\ s &:= \sin\alpha\,, \end{aligned}$$

which can be reformulated by

$$\boldsymbol{d}_L^e = \boldsymbol{T}^e \boldsymbol{d}_G^e\,. \tag{368}$$

The global stiffness matrix results from the discrete potential $\Pi^i = \sum_e \Pi^e$

$$\begin{aligned} \Pi^i &= \frac{1}{2} \mathop{\mathsf{A}}_{e=1}^{\text{nele}} \boldsymbol{d}_L^{eT} \boldsymbol{k}_L^e \boldsymbol{d}_L^e \\ &= \frac{1}{2} \mathop{\mathsf{A}}_{e=1}^{\text{nele}} \boldsymbol{d}_G^{eT} \boldsymbol{T}^{eT} \boldsymbol{k}_L^e \boldsymbol{T}^e \boldsymbol{d}_G^e \\ &=: \frac{1}{2} \mathop{\mathsf{A}}_{e=1}^{\text{nele}} \boldsymbol{d}_G^{eT} \boldsymbol{k}_G^e \boldsymbol{d}_G^e \end{aligned} \tag{369}$$

to

$$\boldsymbol{K} = \mathop{\mathsf{A}}_{e=1}^{\text{nele}} \boldsymbol{k}_G^e = \mathop{\mathsf{A}}_{e=1}^{\text{nele}} \boldsymbol{T}^{eT} \boldsymbol{k}_L^e \boldsymbol{T}^e \tag{370}$$

global loading vector:

$$\boldsymbol{P} = \mathop{\mathsf{A}}_{e=1}^{\text{nele}} \boldsymbol{p}_G^e = \mathop{\mathsf{A}}_{e=1}^{\text{nele}} \boldsymbol{T}^{eT} \boldsymbol{p}_L^e \tag{371}$$

Example:

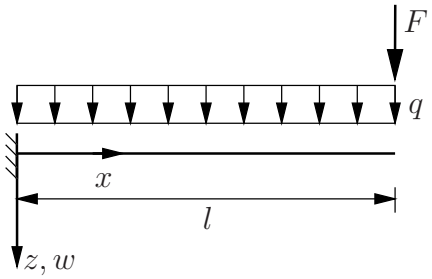


Figure 51: Cantilever with loading case 1. (LC1) and loading case 2. (LC2)

FEM-calculation with one element

$$\boldsymbol{K}\boldsymbol{d} = \boldsymbol{P} \tag{372}$$

$$\frac{EI}{l^3}\begin{bmatrix} 12 & 6l & -12 & 6l \\ & 4l^2 & -6l & 2l^2 \\ & \text{sym.} & 12 & -6l \\ & & & 4l^2 \end{bmatrix}\begin{bmatrix} w_1 \\ \varphi_1 \\ w_2 \\ \varphi_2 \end{bmatrix} = \underbrace{\begin{bmatrix} 1 \\ \frac{l}{6} \\ 1 \\ -\frac{l}{6} \end{bmatrix} q\frac{l}{2}}_{LC1} \text{ and } = \underbrace{\begin{bmatrix} 0 \\ 0 \\ F \\ 0 \end{bmatrix}}_{LC2}. \tag{373}$$

We implement the boundary conditions

$$w(0) = 0 \;\rightarrow\; w_1 = 0 \tag{374}$$

$$w'(0) = \varphi(0) = 0 \;\rightarrow\; \varphi_1 = 0 \tag{375}$$

and eliminate the first and second row/ column and obtain

$$\frac{EI}{l^3}\begin{bmatrix} 12 & -6l \\ -6l & 4l^2 \end{bmatrix}\begin{bmatrix} w_2 \\ \varphi_2 \end{bmatrix} = \underbrace{\begin{bmatrix} q\frac{l}{2} \\ -q\frac{l^2}{12} \end{bmatrix}}_{LC1} \text{ and } = \underbrace{\begin{bmatrix} F \\ 0 \end{bmatrix}}_{LC2} \text{respectively.} \tag{376}$$

The reduction of the system of equations delivers

$$\varphi_2 = \frac{ql^3}{6EI}, \quad w_2 = \frac{ql^4}{8EI} \qquad \text{LC1} \tag{377}$$

$$\varphi_2 = \frac{Fl^3}{2EI}, \quad w_2 = \frac{Fl^3}{3EI} \qquad \text{LC2} \tag{378}$$

Note with respect to loading case 1:
Exact solution $\rightarrow$ cubical w

FE-ansatz for w is cubic
$\rightarrow$ exact elastic curve and exact internal force variables

Note with respect to loading case 2:
Exact solution $\rightarrow$ quadratic w
FE-ansatz for w is cubic
$\rightarrow$ not exact, but the kinematic quantities at the nodes are exact

```
      subroutine
 elmt06(d,ul,xl,ix,tl,s,p,ndf,ndm,nst,isw)
c-----------------------------------------------------------------------
c     elmt06  : Two-dimensional beam element
c
c     List of parameters
c       d(1-2)       material- and element parameters
c       d(1) = yo    - modulus of elasticity
c       d(2) = moin  - geometrical moment of inertia
c       ul(ndf,*)    solution vector
c       xl(ndm,nel)  joint coordinates
c       s(nst,nst)   stiffness matrix
c       p(nst)       residual
c       ndf          number of degrees of freedom
c       ndm          dimensions of the joints
c       nst          number of degrees of freedom of the elements
c       isw          execution parameters
c-----------------------------------------------------------------------
      implicit none
      include   'bdata.h'
      include   'cdata.h'
      include   'debugs.h'
      include   'eldata.h'
      include   'iofile.h'
      integer ix(*)
      integer ndf,ndm,nst,isw
      integer ll,i,j,l
      real*8 d(*),ul(ndf,*),xl(ndm,*),tl(*),s(nst,nst),p(*)
      real*8 bmat(nst),btc(nst)
      real*8 sg(2),wg(2)
      real*8 vl(4),momnod(2)
      real*8 yo,moin,sl,dvol,dy,wxx,mom
      logical errck,pinput
c
      if(isw.eq.1) then
c-----------------------------------------------------------------------
c...  Import of material values
    1 if(ior.lt.0) write(*,3000)
        errck=pinput(d,2)
if(errck) go to 1
        if(ior.lt.0) then
  write(*,2000) d(1),d(2)
end if
write(iow,2000) d(1),d(2)
c
      elseif(isw.eq.2) then
c-----------------------------------------------------------------------
c...  element check with respect to mistakes
        if(d(2) .le. 0.d0) then
          write(*,*)' geometrical moment of inertia .le. 0'
          stop
```

```
        endif
c
      elseif(isw.eq.3 .or. isw.eq.4) then
c-----------------------------------------------------------------------
c...  Calculation of the stiffness matrix and of the residual
c...  Material- and element parameters
        yo    = d(1)
        moin  = d(2)
c...  Length
        sl = xl(1,2) - xl(1,1)
dy = xl(2,2) - xl(2,1)
if (dy .gt. 1.e-9 .or. dy .lt. -1.e-9) then
  write(*,*) 'Only horizontal beam elements'
  write(*,*) 'are allowed in the basic version!'
          stop
end if
c...  Calculation of the local displacements
call pzero(vl,4*1)
        vl(1) = ul(1,1)
        vl(2) = ul(2,1)
        vl(3) = ul(1,2)
        vl(4) = ul(2,2)
c...  Number of Gauss-points and calculation of the nodes
        ll = 2
        sg(1) = - 1.d0/dsqrt(3.d0)
        sg(2) = + 1.d0/dsqrt(3.d0)
        wg(1) = 1.d0
        wg(2) = wg(1)
c.... Gauss loop
        do l = 1,ll
          dvol = 0.5d0*sl*wg(l)
          call pzero(bmat,nst)
          call bmat06(bmat,sg(l),sl)
c...  Calculation of w''(xi)
     wxx = 0.d0
     do j=1,nst
              wxx = wxx + bmat(j)*vl(j)
     end do
c...  Calculation of internal force variables (Moment) (at Gauss-points)
  mom = yo*moin*wxx
          if(isw .eq. 4)then
c.... Output instructions, internal force variables (at joints) in the output file,
c                Usage of convention of internal force variables of the statics
         if (l.eq.1) then
      momnod(1) = -1.d0*(yo*moin/(sl*sl))*(-6.d0*vl(1) -
     &                 4.d0*sl*vl(2) + 6.d0*vl(3) - 2.d0*sl*vl(4))
      momnod(2) = -1.d0*(yo*moin/(sl*sl))*(6.d0*vl(1) +
     &                 2.d0*sl*vl(2) - 6.d0*vl(3) + 4.d0*sl*vl(4))
      write(*,100)
      write(*,101)
      write(iow,100)
```

```
      write(iow,101)
    end if
            write(*,103) ix(l),xl(1,l),xl(2,1),momnod(l)
            write(iow,103) ix(l),xl(1,l),xl(2,1),momnod(l)
c
          elseif(isw .eq. 3)then
c...  Calculation of the residual
            do i=1,nst
      p(i) = p(i) - bmat(i)*mom*dvol
    end do
c...  Compute B^T C
    call pzero(btc,4*1)
            do i  = 1,nst
              btc(i) = bmat(i)*yo*moin*dvol
            enddo
c...  Compute B^T C B = stiffness matrix
            do i  = 1,nst
              do j  = 1,nst
                s(i,j) = s(i,j) + btc(i)*bmat(j)
              enddo
            enddo
          endif    ! End of calculation of local stiffness matrix
        end do   ! Gauss-loop
c
c
      endif   !isw=1,2,3,4,5,6,7,8
c
c-----------------------------------------------------------------------
c...  format instructions
 100  format(15x, 'internal force variables')
 101  format(4x,'joint',3x,'x',5x,'y',8x,'M')
 103  format(5x,i3,2x,f5.2,1x,f5.2,1x,e12.5)
2000  format(5x,'beam element:'//
     &   5x,'modulus of elasticity                    yo = ',e12.5/
     &   5x,'geometrical moment of inertia            moin = ',e12.5/)
3000  format(' Input: Yo,moin'/'   >',$)
      end
c
c
c
      subroutine bmat06(bmat,xi,sl)
c-----------------------------------------------------------------------72
c     Calculation of the B-operator
c       List of parameters
c       bmat(4)  B-operator
c       xi         Gauss-point
c       sl         Length of element
c-----------------------------------------------------------------------
      implicit none
      real*8 bmat(4)
      real*8 sl,xi
```

```
c
c...  Calculation of the B-operator
      bmat(1) = 6.d0*xi/(sl*sl)
      bmat(2) = -1.d0/sl + 3.d0*xi/sl
      bmat(3) = - 6.d0*xi/(sl*sl)
      bmat(4) = 1.d0/sl + 3.d0*xi/sl
      return
      end
c
```

6.8 Plate Elements

Plates carry off the main part of the loading via bending (cf. beams) and have a great technical importance. They are geometrically described by the thickness-measurement h which is much smaller than the length-measurement l_x, l_y, i.e.

$$h \ll l_x, l_y \,. \tag{379}$$

We differentiate between shear-rigid and shear-weak formulations. If we start from the assumption that the cross-section planes remain even, we obtain the shear-weak formulation by Reissner & Mindlin, respectively. In this case, independent approximations for the deflection $w(x)$ and the curvature of the cross-section $\beta_x(x)$, $\beta_y(x)$ are considered. If we take into account the additional assumption of remaining normals ("normal remains normal"), we arrive at the shear-rigid theory of the Kirchhoff-plate. In this case, C^1-consistent approximations for $w(x)$ are required, because the slope of the cross-section is computed by the derivatives $\beta_y = -w_{,y}$ and $\beta_x = -w_{,x}$. Hence, β_x and β_y are independent variables.

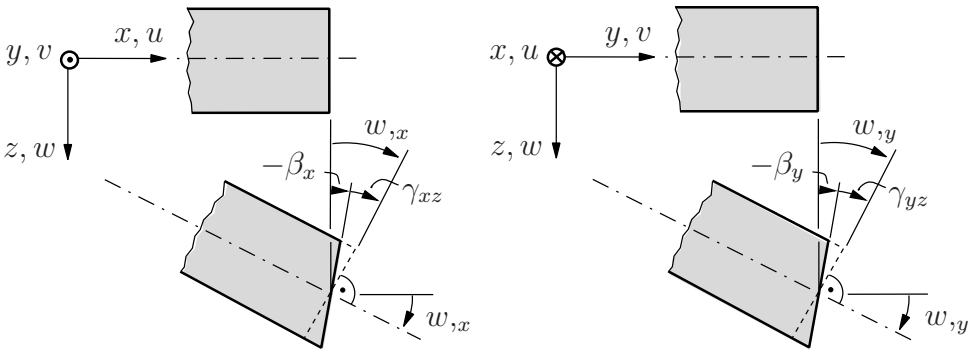


Figure 52: Kinematics.

6.8.1 Kinematics The deformation in the x-z-plane is given by the twist of the cross-section

$$w_{,x} = -\beta_x + \gamma_{xz} \,. \tag{380}$$

Hence, the average shear strain results in

$$\gamma_{xz} = w_{,x} + \beta_x \,. \tag{381}$$

The u-displacement in x-direction is given via the curvature of the cross-section β_x

$$u = \beta_x z \,. \tag{382}$$

Analogously, the deformation in the y-z-plane and therefore the twist of the cross-section results in

$$w_{,y} = -\beta_y + \gamma_{yz} \,. \tag{383}$$

Hence, we obtain the average shear strain

$$\gamma_{yz} = w_{,y} + \beta_y \,. \tag{384}$$

The v-displacement in y-direction reads

$$v = \beta_y z \,. \tag{385}$$

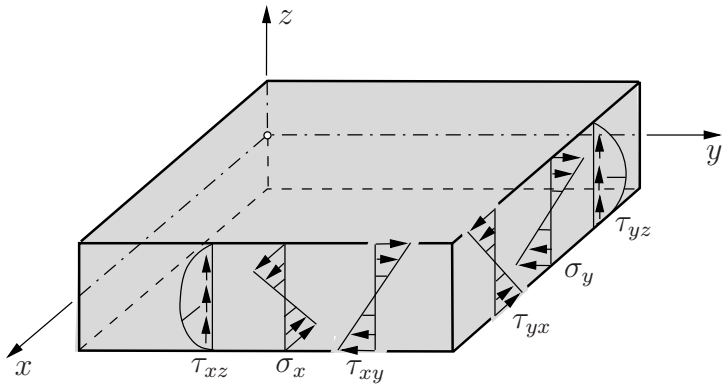


Figure 53: Stress distribution.

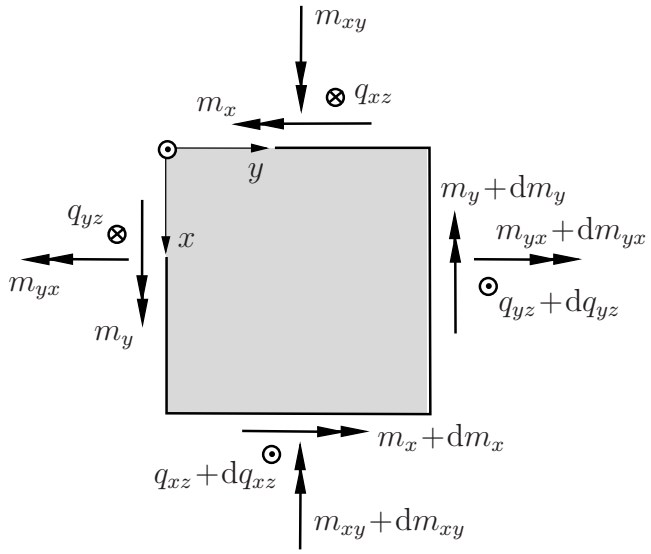


Figure 54: Internal force variables.

6.8.2 Kirchhoff Theory With respect to the shear-rigid KIRCHHOFF-theory the assumption of remaining normals is evaluated and

$$\gamma_{xz} = \gamma_{yz} = 0 \tag{386}$$

is aplied. Hence, for the curvatures of the cross-sections it follows that

$$\beta_x = -w_{,x} \qquad \text{und} \qquad \beta_y = -w_{,y}\,, \tag{387}$$

as well as for the displacements

$$u = -zw_{,x} \qquad \text{and} \qquad v = -zw_{,y}\,. \tag{388}$$

For the strains we obtain

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{bmatrix} = \begin{bmatrix} u_{,x} \\ v_{,y} \\ u_{,y} + v_{,x} \end{bmatrix} = \begin{bmatrix} -zw_{,xx} \\ -zw_{,yy} \\ -z(w_{,xy} + w_{,yx}) \end{bmatrix} = z \begin{bmatrix} -w_{,xx} \\ -w_{,yy} \\ -2w_{,xy} \end{bmatrix} = z\boldsymbol{\kappa}\,, \tag{389}$$

whereby $\boldsymbol{\kappa}$ characterizes the curvatures. With the material law

$$\begin{bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{bmatrix} = \frac{E}{1-\nu^2} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \dfrac{1-\nu}{2} \end{bmatrix} \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{bmatrix} \quad \text{and } \boldsymbol{\sigma} = \mathbb{C}\,\boldsymbol{\varepsilon}, \quad \text{respectively}, \tag{390}$$

we are able to calculate the stress resultants by the integration over the thickness

$$m_x = \int_{(z)} \sigma_x z\, dz\,, \quad m_y = \int_{(z)} \sigma_y z\, dz\,, \quad m_{xy} = \int_{(z)} \tau_{xy} z\, dz\,. \tag{391}$$

With (390) and (389) we obtain

$$\boldsymbol{M} := \begin{bmatrix} m_x \\ m_y \\ m_{xy} \end{bmatrix} = \int_{(z)} \boldsymbol{\sigma} z\, dz = \int_{(z)} \mathbb{C} z^2 \boldsymbol{\kappa}\, dz = \mathbb{C} \int_{-h/2}^{h/2} z^2\, dz\, \boldsymbol{\kappa} \tag{392}$$

and therefore

$$\boldsymbol{M} = \frac{Eh^3}{12(1-\nu^2)} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \dfrac{1-\nu}{2} \end{bmatrix} \boldsymbol{\kappa} =: \mathbb{C}_B \boldsymbol{\kappa}\,. \tag{393}$$

The principle of virtual work reads

$$\delta\Pi = \delta\Pi_i + \delta\Pi_a \tag{394}$$

with the internal and external parts

$$\delta\Pi_i = \int_{(A)} \delta\boldsymbol{\kappa}^T \boldsymbol{M}\, dA \qquad \text{and} \qquad \delta\Pi_a = \int_{(A)} \delta w q_z\, dA\,. \tag{395}$$

In the linear elastic case, the potential of the inner forces reads

$$\Pi_i = \frac{1}{2} \int_{(A)} \boldsymbol{\kappa}^T \mathbb{C}_B \boldsymbol{\kappa}\, dA\,. \tag{396}$$

The ansatz functions with respect to the KIRCHHOFF- theory have to be continuous in the displacement and in the derivatives of the displacements, i.e. we require C^1-continuous functions.

6.8.3 16 Parameter Element by Kirchhoff- Theory According to the KIRCHHOFF theory with respect to a rectangular plate element, the approximations of the Bernoulli-beam with 4 Hermite polynomials

$$w(\xi) = N_1(\xi) w_1 + \bar{N}_1(\xi)\beta_1 + N_2(\xi) w_2 + \bar{N}_2(\xi)\beta_2 \tag{397}$$

are transferred to the case of a plate, thus, we require 16 ansatz functions from a formal 2-dimensional expansion of (397). The ansatz for $w_{①}$ is given by

$$N_{11} = N_{①}(\xi) N_{①}(\eta)\,, \tag{398}$$

and the approximations for $\beta_{y①}$ and $\beta_{x①}$ are

$$N_{1\bar{1}} = N_{①}(\xi)\bar{N}_{①}(\eta)\,; \quad N_{\bar{1}1} = \bar{N}_{①}(\xi)N_{①}(\eta)\,. \tag{399}$$

With the node displacement vector

$$\boldsymbol{d}_I^T = [w_I, \beta_{xI}, \beta_{yI}, w_{xyI}] \quad \text{for } I = 1, ..., 4 \tag{400}$$

we obtain the element displacement vector

$$\boldsymbol{d}^{eT} = [\boldsymbol{d}_①^T, \boldsymbol{d}_②^T, \boldsymbol{d}_③^T, \boldsymbol{d}_④^T]\,. \tag{401}$$

From this we get the approximation for the deflection

$$w^h = \sum_{I=1}^{4} \boldsymbol{N}_I \cdot \boldsymbol{d}_I\,, \tag{402}$$

wherein

$$\left.\begin{array}{ll} \boldsymbol{N}_① & = [N_{11}, N_{\bar{1}1}, N_{1\bar{1}}, N_{\bar{1}\bar{1}}] \\ \boldsymbol{N}_② & = [N_{21}, N_{\bar{2}1}, N_{2\bar{1}}, N_{\bar{2}\bar{1}}] \\ \boldsymbol{N}_③ & = [N_{22}, N_{\bar{2}2}, N_{2\bar{2}}, N_{\bar{2}\bar{2}}] \\ \boldsymbol{N}_④ & = [N_{12}, N_{\bar{1}2}, N_{1\bar{2}}, N_{\bar{1}\bar{2}}] \end{array}\right\}. \tag{403}$$

This ansatz is cubically complete with ansatz functions N_I, $\bar{N}_I$ following the scheme

$$\left.\begin{array}{ccccccc} & & & 1 & & & \\ & & \xi & & \eta & & \\ & \xi^2 & & \xi\eta & & \eta^2 & \\ \xi^3 & & \xi^2\eta & & \xi\eta^2 & & \eta^3 \\ & \xi^3\eta & & \xi^2\eta^2 & & \xi\eta^3 & \\ & & \xi^3\eta^2 & & \xi^2\eta^3 & & \\ & & & \xi^3\eta^3 & & & \end{array}\right\}. \tag{404}$$

Herewith, we are able to represent rigid-body displacements and twists, constant curvatures and constant torsions. Furthermore, homogeneous differential equations (no loading) are exactly reflected if linear stress resultants and constant shear forces occur. The approximation for the strains is then given by

$$\boldsymbol{\kappa}^h = \begin{bmatrix} -w_{,xx} \\ -w_{,yy} \\ -2w_{,xy} \end{bmatrix} = \sum_{I=1}^{4} \boldsymbol{B}_I \boldsymbol{d}_I \tag{405}$$

with the B-matrix

$$\boldsymbol{B}_I = \begin{bmatrix} -N_{I,xx} \\ -N_{I,yy} \\ -2N_{I,xy} \end{bmatrix} \tag{406}$$

and the particular terms

$$N_{I,xx} = N_{I,\xi\xi}\frac{4}{l_x^2}\,; \quad N_{I,yy} = N_{I,\eta\eta}\frac{4}{l_y^2}\,; \quad N_{I,xy} = N_{I,\xi\eta}\frac{4}{l_x l_y}\,. \tag{407}$$

The stiffness matrix is then computed by

$$k_{IJ}^e = \int_{(A)} \boldsymbol{B}_I^{eT} \mathbb{C}_B \boldsymbol{B}_J^e \, dA\,. \tag{408}$$

The integration can be computed by numerical quadratures or analytically.

6.8.4 Reissner-Mindlin Theory With respect to the shear-weak formulation by Reissner, Mindlin the displacements are computed by

$$u = z\beta_x \qquad \text{and} \qquad v = z\beta_y \,. \tag{409}$$

Then the strains are given by

$$\boldsymbol{\varepsilon} = \begin{bmatrix} z\beta_{x,x} \\ z\beta_{y,y} \\ z(\beta_{x,y} + \beta_{y,x}) \end{bmatrix} = z\boldsymbol{\kappa} \,, \tag{410}$$

and a material law has to be provided also for the shear stresses which are not in the plate plain, i.e.

$$\begin{bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{bmatrix} = \frac{E}{1-\nu^2} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \dfrac{1-\nu}{2} \end{bmatrix} \begin{bmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{bmatrix} \;; \quad \begin{bmatrix} \tau_{xz} \\ \tau_{yz} \end{bmatrix} = \kappa_s \, \mu \, \mathbf{1} \begin{bmatrix} \gamma_{xz} \\ \gamma_{yz} \end{bmatrix} \tag{411}$$

which can be expressed in the total notation by

$$\boldsymbol{\sigma} = \mathbb{C} \, \boldsymbol{\varepsilon} \qquad \text{and} \qquad \boldsymbol{\tau} = \boldsymbol{G}\boldsymbol{\gamma} \,, \tag{412}$$

respectively. Herein, κ_s is the shear correction coefficient. In addition to the stress resultants m_x, m_y, m_{xy} in (392), we consider

$$q_{xz} = \int_{(z)} \tau_{xz} \, dz \qquad \text{and} \qquad q_{yz} = \int_{(z)} \tau_{yz} \, dz \tag{413}$$

and obtain the additional stress resultants

$$\boldsymbol{Q} = \begin{bmatrix} q_{xz} \\ q_{yz} \end{bmatrix} = \int_{-h/2}^{h/2} \boldsymbol{G}\boldsymbol{\gamma} \, dz = \kappa_s \, \mu \, h \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \boldsymbol{\gamma} = \mathbb{C}_s \boldsymbol{\gamma} \,. \tag{414}$$

The principle of virtual work states that

$$\delta\Pi = \delta\Pi_i + \delta\Pi_a \,, \tag{415}$$

with the internal

$$\delta\Pi_i = \int_{(A)} \delta\boldsymbol{\kappa}^T \boldsymbol{M} \, dA + \int_{(A)} \delta\boldsymbol{\gamma}^T \boldsymbol{Q} \, dA \tag{416}$$

and external part

$$\delta\Pi_a = \int_{(A)} \delta w q_z \, dA \,. \tag{417}$$

The internal potential reads then

$$\Pi_i = \frac{1}{2} \int_{(A)} \boldsymbol{\kappa}^T \mathbb{C}_B \boldsymbol{\kappa} \, dA + \frac{1}{2} \int_{(A)} \boldsymbol{\gamma}^T \mathbb{C}_S \boldsymbol{\gamma} \, dA \tag{418}$$

6.9 Isoparametric Four-Node-Element of the Shear-Elastic Theory

The degrees of freedom at the nodes are w, β_x, β_y. Thus, we consider the approximation

$$\boldsymbol{u}^h = \begin{bmatrix} w \\ \beta_x \\ \beta_y \end{bmatrix} = \sum_{I=1}^{n_{el}} N_I \boldsymbol{d}_I \,, \tag{419}$$

with the ansatz functions

$$N_I = \frac{1}{4}(1+\xi\xi_I)(1+\eta\eta_I)\,. \tag{420}$$

The approximations for the curvatures are given by

$$\boldsymbol{\kappa}^h = \begin{bmatrix} \beta^h_{x,x} \\ \beta^h_{y,y} \\ \beta^h_{x,y} + \beta^h_{y,x} \end{bmatrix} = \sum_{I=1}^{n_{el}} \boldsymbol{B}^I_b \boldsymbol{d}_I \quad \text{with} \quad \boldsymbol{B}^I_b = \begin{bmatrix} N_{I,x} & 0 \\ 0 & N_{I,y} \\ N_{I,y} & N_{I,x} \end{bmatrix} , \tag{421}$$

whereas the shear-strains are approximated by

$$\gamma^h = \begin{bmatrix} w^h_{,x} + \beta^h_x \\ w^h_{,y} + \beta^h_y \end{bmatrix} = \sum_{I=1}^{n_{el}} \boldsymbol{B}^I_s \boldsymbol{d}_I \quad \text{with} \quad \boldsymbol{B}^I_s = \begin{bmatrix} N_{I,x} & N_I & 0 \\ N_{I,y} & 0 & N_I \end{bmatrix} . \tag{422}$$

The Jacobi matrix is computed by

$$\begin{bmatrix} N_{I,x} \\ N_{I,y} \end{bmatrix} = \boldsymbol{J}^{-1} \begin{bmatrix} N_{I,\xi} \\ N_{I,\eta} \end{bmatrix} , \tag{423}$$

hence, we obtain the element stiffness matrix decomposed into two parts by

$$\delta\Pi^e_i = \delta\boldsymbol{d}^{eT} \left[\underbrace{\int_{(A)} \boldsymbol{B}^{eT}_b \mathbb{C}_b \boldsymbol{B}^e_b \, dA}_{=:\boldsymbol{k}^e_b} + \underbrace{\int_{(A)} \boldsymbol{B}^{eT}_s \mathbb{C}_s \boldsymbol{B}^e_s \, dA}_{=:\boldsymbol{k}^e_s} \right] \boldsymbol{d}^e \,. \tag{424}$$

6.10 Axis-Symmetric Shell Elements

The calculation of rotation-symmetric shells is of large practical interest. If the loading is also rotation-symmetric besides to the geometry of the shell, we obtain one-dimensional elements undergoing bending- and membrane loading. These shells are described by strains and curvatures of the mid plain, which are referred to as "generalized" strains. In the case of axis-symmetric problems, the displacements of the mid area is described in terms of the tangential displacement $u(s)$ and the radial displacement $w(s)$. For an illustration of the degrees of freedom see Figure 55.

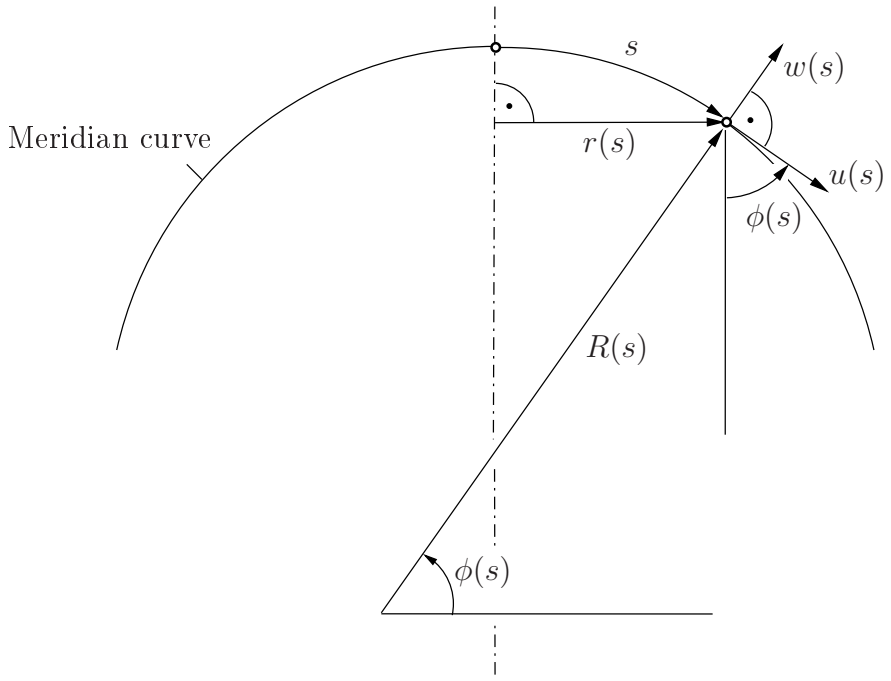


Figure 55: Rotation-symmetric shell: Resulting (meridian curve) and displacements.

$R(s)$ characterizes the curvature radius. The displacements in the considered plane are written in the form

$$\boldsymbol{u} = u\boldsymbol{e}_\varphi + w\boldsymbol{e}_r \tag{425}$$

with the orthogonal basis $\boldsymbol{e}_\varphi$ and $\boldsymbol{e}_r$. The meridian strain is defined as

$$\varepsilon_s := \frac{d}{ds}[\boldsymbol{u}] \cdot \boldsymbol{e}_\varphi \,, \tag{426}$$

and is computed from the projection of the displacement gradient in the direction of $\boldsymbol{e}_\varphi$, i.e.

$$\varepsilon_s = \left[u_{,s}\, \boldsymbol{e}_\varphi + u\frac{d\boldsymbol{e}_\varphi}{ds} + w_{,s}\, \boldsymbol{e}_r + w\frac{d\boldsymbol{e}_r}{ds} \right] \cdot \boldsymbol{e}_\varphi \,. \tag{427}$$

According to Fig. 6.10 we obtain

$$d\boldsymbol{e}_r = \boldsymbol{e}_\varphi d\varphi \qquad \text{und} \qquad d\boldsymbol{e}_\varphi = -\boldsymbol{e}_r d\varphi \,. \tag{428}$$

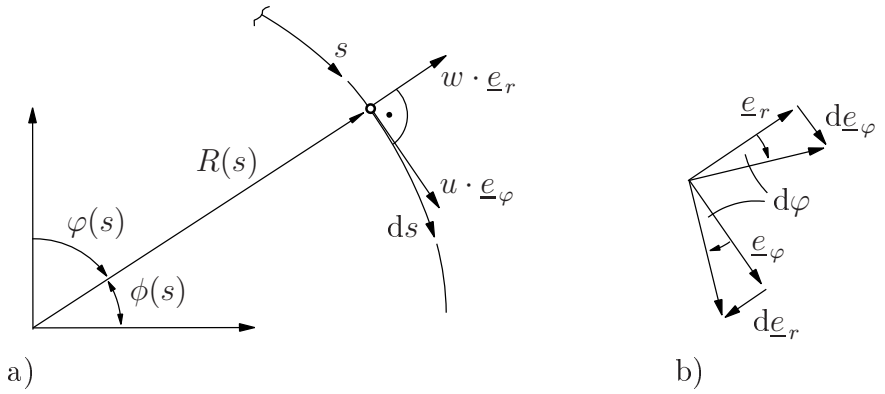


Figure 56: Basis vectors and infinitesimal basis vectors.

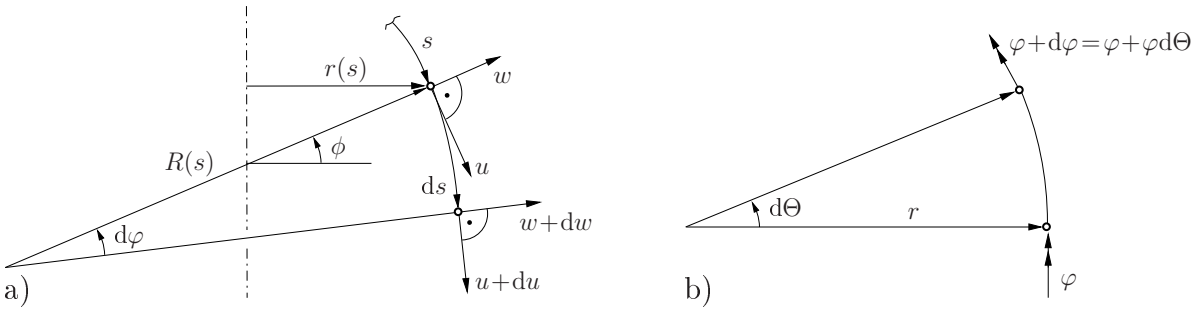


Figure 57: Infinitesimale elements in a) front and b) top view.

If we insert these relations into equation (427) and include $\boldsymbol{e}_\varphi \cdot \boldsymbol{e}_r = 0$ and $\boldsymbol{e}_\varphi \cdot \boldsymbol{e}_\varphi = 1$, we obtain

$$\varepsilon_s = u_{,s} + w\frac{d\varphi}{ds}\,. \tag{429}$$

The coherence $ds = R\,d\varphi$ yields the relation

$$\varepsilon_s = u_{,s} + \frac{w}{R}\,. \tag{430}$$

The enlargement of the radius due to the mid-plain-displacement is computed by

$$dr = w\cos\phi + u\sin\phi\,. \tag{431}$$

Hence, we obtain the membrane-strain in circumferential direction with

$$\varepsilon_\Theta = (w\cos\phi + u\sin\phi)/r\,. \tag{432}$$

With the introduction of a infinitesimal length element $r\,d\Theta$ in circumferential direction, see Figure 57_2, we are now able to compute the bending-strains.

Kirchhoff-Love Formulation: According to the Kichhoff-Love-Hypothesis (normal remains normal), the rotation of the shell mid-plain arises from $\beta(s) := \dfrac{d}{ds}[\boldsymbol{u}] \cdot \boldsymbol{e}_r$ to

$$\beta = w_{,s} - \frac{u}{R}\,. \tag{433}$$

The bending-strains (curvatures) are calculated by

$$\kappa_s = \frac{d}{ds}\beta = \frac{d}{ds}\left[\frac{dw}{ds} - \frac{u}{R}\right] = w_{,ss} - \left(\frac{u}{R}\right)_{,s} . \tag{434}$$

$$\kappa_\Theta = \frac{1}{rd\Theta}\left[\beta d\Theta \sin\phi\right] = \frac{1}{rd\Theta}\left[\left(\frac{dw}{ds} - \frac{u}{R}\right) d\Theta \sin\phi\right] = \frac{\sin\phi}{r}\left(w_{,s} - \frac{u}{R}\right) . \tag{435}$$

With respect to the Kirchhoff-Love theory we consider the assumptions: i) normal remains normal, ii) continuity of planarity of the cross-sections and iii) $\varepsilon_z = 0$ (no strains in thickness direction), and obtain the generalized strain-measurements with

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_s \\ \varepsilon_\Theta \\ \kappa_s \\ \kappa_\Theta \end{bmatrix} = \begin{bmatrix} u_{,s} + w/R \\ (w\cos\phi + u\sin\phi)/r \\ w_{,ss} - (u/R)_{,s} \\ \sin\phi/r\,(w_{,s} - u/R) \end{bmatrix} . \tag{436}$$

Reissner-Mindlin Formulation: With respect to the shear-elastic Reissner-Mindlin-theory, the assumption, normal remains normal, is abandoned. With the introduction of the shear-angle

$$\gamma = w_{,s} - \beta \tag{437}$$

the generalized strains result in

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_s \\ \varepsilon_\Theta \\ \kappa_s \\ \kappa_\Theta \\ \gamma \end{bmatrix} = \begin{bmatrix} u_{,s} + w/R \\ w\cos\phi + u\sin\phi/r \\ \beta_{,s} - (u/R)_{,s} \\ \sin\phi/r\,(\beta - u/R) \\ w_{,s} - \beta \end{bmatrix} . \tag{438}$$

With the elasticity-law we can now calculate the stresses and afterwards the stress resultants by integration over the shell thickness. With the coordinate z in thickness direction we obtain

$$\left.\begin{aligned} \sigma_s &= \frac{E}{1-\nu^2}\left[(\varepsilon_s + \nu\varepsilon_\Theta) + z\,(\kappa_s + \nu\kappa_\Theta)\right] \\ \sigma_\Theta &= \frac{E}{1-\nu^2}\left[(\varepsilon_\Theta + \nu\varepsilon_s) + z\,(\kappa_\Theta + \nu\kappa_s)\right] \\ \tau_m &= \frac{5}{6}\mu\gamma = \frac{5}{6}\frac{E}{2(1+\nu)}\gamma \end{aligned}\right\} , \tag{439}$$

whereby $-\frac{h}{2} \le z \le \frac{h}{2}$ is taken into account and h describes the shell thickness. The factor $\frac{5}{6}$ is referred to as shear-correction factor for rectangular cross-sections and τ_m is a medial shear stress. With the integration over the thickness of the cross-sections we get the membrane parts (n_s, n_Θ), the bending parts (m_s, m_Θ) and the shear parts (q_s) of the stress resultants. In particular, we obtain the membrane parts

$$\left.\begin{aligned} n_s &= \int_{-\frac{h}{2}}^{\frac{h}{2}} \sigma_s dz = \frac{Eh}{1-\nu^2}[\varepsilon_s + \nu\varepsilon_\Theta] \\ n_\Theta &= \int_{-\frac{h}{2}}^{\frac{h}{2}} \sigma_\Theta dz = \frac{Eh}{1-\nu^2}[\varepsilon_\theta + \nu\varepsilon_s] \end{aligned}\right\} \tag{440}$$

the bending parts

$$\left.\begin{aligned} m_s &= \int_{-\frac{h}{2}}^{\frac{h}{2}} \sigma_s z dz = \frac{Eh^3}{12(1-\nu^2)}[\kappa_s + \nu\kappa_\Theta] \\ m_\Theta &= \int_{-\frac{h}{2}}^{\frac{h}{2}} \sigma_\Theta z dz = \frac{Eh^3}{12(1-\nu^2)}[\kappa_\Theta + \nu\kappa_s] \end{aligned}\right\} \tag{441}$$

and the shear part

$$q_s = \int_{-\frac{h}{2}}^{\frac{h}{2}} \tau dz = \frac{5}{6}\frac{Eh}{2(1+\nu)}\gamma\,. \tag{442}$$

These results can be reformulated in matrix notation and we obtain for the Kirchhoff-Love-theory

$$\begin{bmatrix} n_s \\ n_\Theta \\ m_s \\ m_\Theta \end{bmatrix} = \begin{bmatrix} \bar{\mathbb{C}}_M & \\ & \bar{\mathbb{C}}_B \end{bmatrix} \begin{bmatrix} \varepsilon_s \\ \varepsilon_\Theta \\ \kappa_s \\ \kappa_\Theta \end{bmatrix}, \tag{443}$$

wherein the block matrices are given by

$$\bar{\mathbb{C}}_M = \frac{Eh}{1-\nu^2}\begin{bmatrix} 1 & \nu \\ \nu & 1 \end{bmatrix} \qquad \text{und} \qquad \bar{\mathbb{C}}_B = \frac{Eh^3}{12(1-\nu^2)}\begin{bmatrix} 1 & \nu \\ \nu & 1 \end{bmatrix}. \tag{444}$$

For the Reissner-Mindlin-theory we obtain

$$\begin{bmatrix} n_s \\ n_\Theta \\ m_s \\ m_\Theta \\ q_s \end{bmatrix} = \begin{bmatrix} \bar{\mathbb{C}}_M & & \\ & \bar{\mathbb{C}}_B & \\ & & \bar{\mathbb{C}}_s \end{bmatrix} \begin{bmatrix} \varepsilon_s \\ \varepsilon_\Theta \\ \kappa_s \\ \kappa_\Theta \\ \gamma \end{bmatrix} \tag{445}$$

with the additional term

$$\bar{\mathbb{C}}_s = \frac{5}{6}\cdot\frac{Eh}{2(1+\nu)}\,. \tag{446}$$

The elastic potential for linear problems is given by

$$\Pi = \Pi_i + \Pi_a = \frac{1}{2}\int_V \boldsymbol{\varepsilon}^T \tilde{\mathbb{C}}\boldsymbol{\varepsilon} dV - \int_A \boldsymbol{u}^T \boldsymbol{q} dA\,, \tag{447}$$

with the loading force $\boldsymbol{q}$. Inserting the integration over thickness we get

$$\Pi = \frac{1}{2}\int_A \boldsymbol{\varepsilon}^T \bar{\mathbb{C}}\boldsymbol{\varepsilon} dA - \int_A \boldsymbol{u}^T \boldsymbol{q} dA\,, \tag{448}$$

with the relations (443) or (445), respectively. According to the Kirchhoff-Love-theory, the vector of the generalized displacements reads

$$\boldsymbol{u}^T = [u, w, w_{,s}] \tag{449}$$

and with respect to the Reissner-Mindlin-theory

$$\boldsymbol{u}^T = [u, w, \beta]\,. \tag{450}$$

Due to the fact that the rotation-symmetry of the geometry and the loading is assumed, the integration in circumferential-direction can be calculated in advance, because all incoming quantities are no functions of the circumferential angle Θ. With

$$dA = r\,d\Theta\,ds \tag{451}$$

we obtain the potential

$$\Pi = \frac{1}{2}\int_s \boldsymbol{\varepsilon}^T \bar{\mathbb{C}}\boldsymbol{\varepsilon} 2\pi r ds - \int_s \boldsymbol{u}^T \boldsymbol{q} 2\pi r ds\,, \tag{452}$$

which is now formulated in integrals over the meridian direction only. The equilibrium state is achieved if satisfying the condition $\delta\Pi = 0$, i.e.

$$\delta\Pi = \int_s \delta\boldsymbol{\varepsilon}^T \bar{\mathbb{C}}\boldsymbol{\varepsilon} 2\pi r ds - \int_s \delta\boldsymbol{u}^T \boldsymbol{q} 2\pi r ds = 0\,. \tag{453}$$

6.10.1 Two-Node Axis-Symmetric Shell Element We consider an element with 6 degrees of freedom, see e.g. Zienkiewicz & Taylor. The global node displacement vector for node 1 is written in the form

$$\boldsymbol{u}^G_{①} = [u^G_{①}, w^G_{①}, \beta^G_{①}]^T\,. \tag{454}$$

The strains and therefore also the stress-strain relations are defined in the local displacement quantities

$$\boldsymbol{u}^L_{①} = [u^L_{①}, w^L_{①}, \beta^L_{①}]^T\,. \tag{455}$$

The transformation from the global to the local displacement quantities is calculated by

$$\begin{bmatrix} u^L_{①} \\ w^L_{①} \\ \beta^L_{①} \end{bmatrix} = \begin{bmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u^G_{①} \\ w^G_{①} \\ \beta^G_{①} \end{bmatrix} \tag{456}$$

or in the abbreviated form by

$$\boldsymbol{u}^L_1 = \boldsymbol{T} \cdot \boldsymbol{u}^G_1\,. \tag{457}$$

We consider a straight element here, where the transformation matrix is constant. The local element displacement vector is given by

$$\boldsymbol{u}^{(e)L} = \begin{bmatrix} \boldsymbol{u}^L_1 \\ \boldsymbol{u}^L_2 \end{bmatrix} = \begin{bmatrix} \boldsymbol{T} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{T} \end{bmatrix} \begin{bmatrix} \boldsymbol{u}^G_1 \\ \boldsymbol{u}^G_2 \end{bmatrix} = \boldsymbol{T}^{(e)} \boldsymbol{u}^{(e)G}\,. \tag{458}$$

For straight elements we consider

$$R \to \infty \quad \text{from this it follows} \quad \frac{1}{R} \to 0\,. \tag{459}$$

With respect to the Kirchhoff-Love-theory we obtain the simplified description

$$\boldsymbol{\varepsilon} = \begin{bmatrix} \varepsilon_s \\ \varepsilon_\Theta \\ \kappa_s \\ \kappa_\Theta \end{bmatrix} = \begin{bmatrix} u_{,s} \\ (w\cos\phi + u\sin\phi)/r \\ w_{,ss} \\ (\sin\phi w_{,s})/r \end{bmatrix} . \tag{460}$$

The displacement-ansatz has to be at least C^0-continuous in u and at least C^1-continuous in w. Here, we consider linear approximations for u and cubical approximations for w, i.e.

$$\left.\begin{array}{rcl} u_h^L(\xi) & = & (1-\xi)u_{①}^L + \xi u_{②}^L \\ w_h^L(\xi) & = & (1-3\xi^2+2\xi^3)w_{①}^L + l(\xi-2\xi^2+\xi^3)\beta_{①}^L + \\ & & (3\xi^2-2\xi^3)w_{②}^L + l(-\xi^2+\xi^3)\beta_{②}^L \end{array}\right\} \tag{461}$$

with the natural coordinate $\xi = s/l$. We obtain the approximations for the strains as a function of the local element displacements from equation (460) and (461) and application of the chain rule for $N(\xi)_{,s} = N(\xi)_{,\xi}\cdot\xi_{,s} = N(\xi)_{,\xi}\cdot 1/l$ to

$$\begin{bmatrix} \varepsilon_s^h \\ \varepsilon_\theta^h \\ \kappa_s^h \\ \kappa_\theta^h \end{bmatrix} = \begin{bmatrix} -1/l & 0 & 0 \\ (1-\xi)\sin\phi/r & (1-3\xi^2+2\xi^3)\cos\phi/r & l(\xi-2\xi^2+\xi^3)\cos\phi/r \\ 0 & (-6+12\xi)/l^2 & (-4+6\xi)/l \\ 0 & (-6\xi+6\xi^2)\sin\phi/(rl) & (1-4\xi+3\xi^2)\sin\phi/r \end{bmatrix} \begin{bmatrix} u_{①}^L \\ w_{①}^L \\ \beta_{①}^L \end{bmatrix}$$
$$+ \begin{bmatrix} 1/l & 0 & 0 \\ \xi\sin\phi/r & (3\xi^2-2\xi^3)\cos\phi/r & l(-\xi^2+\xi^3)\cos\phi/r \\ 0 & (6-12\xi)/l^2 & (2-6\xi)/l \\ 0 & (6\xi-6\xi^2)\sin\phi/(rl) & (-2\xi+3\xi^2)\sin\phi/r \end{bmatrix} \begin{bmatrix} u_{②}^L \\ w_{②}^L \\ \beta_{②}^L \end{bmatrix}$$

and in shortform

$$\boldsymbol{\varepsilon}_h^{(e)}(\xi) = \boldsymbol{B}_{①}\cdot\boldsymbol{u}_{①}^L + \boldsymbol{B}_{②}\cdot\boldsymbol{u}_{②}^L = \boldsymbol{B}^{(e)}\boldsymbol{u}^{(e)L} . \tag{462}$$

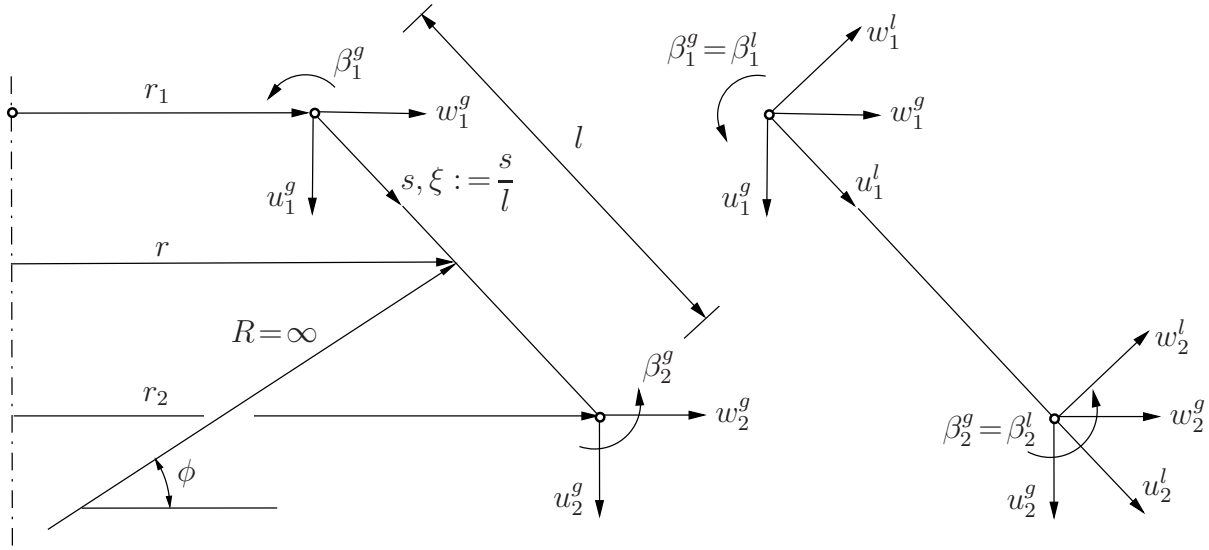


Figure 58: Two-Node axis-symmetric shell element.

If we insert the approximations in $\delta\Pi$, we obtain

$$\delta\Pi_i^{(e)} = \int_{(l)} \delta\boldsymbol{\varepsilon}^T \bar{\mathbb{C}} \boldsymbol{\varepsilon} 2\pi r ds \tag{463}$$

$$= \delta\boldsymbol{u}^{(e)T} \int_0^1 \boldsymbol{B}^{(e)T} \bar{\mathbb{C}} \boldsymbol{B}^{(e)} 2\pi r l d\xi \boldsymbol{u}^{(e)} \tag{464}$$

for the internal part of a typical element. With the transformation relation from equation (457) we get the representation in global quantities

$$\delta\Pi_i^{(e)} = \delta\boldsymbol{u}^{(e)G^T} \boldsymbol{T}^{(e)^T} \int_0^1 \boldsymbol{B}^{(e)T} \bar{\mathbb{C}} \boldsymbol{B}^{(e)} 2\pi r l d\xi \boldsymbol{T}^{(e)} \boldsymbol{u}^{(e)G} \tag{465}$$

$$= \delta\boldsymbol{u}^{(e)G^T} \boldsymbol{K}^{(e)^0} \boldsymbol{u}^{(e)G} . \tag{466}$$

At first the variable r, which is adjustable with ξ, has to be reformulated for the integration of the element stiffness matrix in terms of natural coordinates

$$r = (1 - \xi) r_{①} + \xi r_{②} . \tag{467}$$

Hence, the largest polynomial in $\boldsymbol{K}^{(e)}$ in ξ has the order of 7. In the context of Gauss-point integration we therefore require 4 Gauss-points, because a polynomial of $(2n-1)$th order is exactly integrated with n points.

Example: Cylindric water supply tank

In the following example a cylindric water supply tank is analyzed. The tank is clamped at the bottom and has a free boundary at the top. The analytical distribution of stress resultants over the height will be compared with the stress resultants obtained from the FE-calculation.

Figure 59 illustrates the system

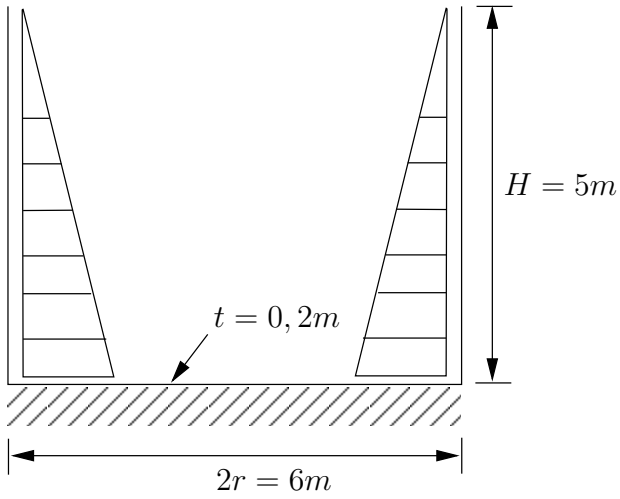


Figure 59: Cross-section of the cylindric water supply tank.

We consider the following input data: $t = 0,2m$, $E = 3 \cdot 10^7 kN/m^2$, $\nu = 0,17m$, $H = 5m$, $r = 3m$, $\gamma = 10kN/m^2$. The analytical distribution of stress resultant m_s follows the equation

$$m_s = \frac{\gamma H}{2\lambda^2} e^{-\lambda(H-z)} \left[-\sin(\lambda(H-z)) - \left(\frac{1}{\lambda H} - 1 \right) \cos(\lambda(H-z)) \right]$$

with the abbreviation

$$\text{with} \quad \lambda = \frac{\sqrt[4]{3(1-\mu^2)}}{\sqrt{ah}} .$$

If we insert the given data, we obtain

$$m_s = 8,753 \cdot e^{-1,69(5-z)} \left[-\sin(1,69(5-z)) + 0,8817 \cdot \cos(1,69(5-z)) \right] .$$

The example is calculated also in the framework of the Finite-Element Method with 5 and with 50 shell elements. Figure 60 illustrates the different stress resultants m_s over the height. We observe a much better aggreement with the analytical solution if 50 elements are used.

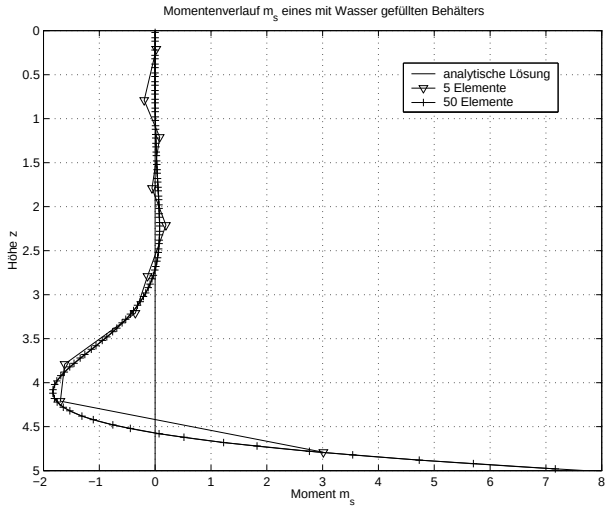


Figure 60: Distribution of stress resultant m_s over the height.

```
      subroutine elmt05(d,ul,xl,ix,tl,s,p,ndf,ndm,nst,isw)
c-----------------------------------------------------------------------
c     elmt05  : Main-subroutine for a rotation-symmetric shell element
c               with rotation-symmetric loading
c
c     Parameter list
c       d(1-3)       Material- and element parameter
c       d(1) = yo    - Modulus of elasticity
c       d(2) = nu    - Poisson's number
c       d(3) = thick - Shell thickness
c       d(4)         Number of Gauss-points
c       ul(ndf,*)    Solution vector
c       xl(ndm,nel)  Coordinates of nodes
c       ix(nel)      Global node number
c       tl(nst)      Temperature values of nodes
c       s(nst,nst)   Stiffness matrix
c       p(nst)       Load vector, residuum
c       ndf          Number of degrees of freedom
c       ndm          Dimension of nodes
c       nst          Number of degress of freedom of elements
c       isw          Execution flag
c-----------------------------------------------------------------------
      implicit none
      include   'bdata.h'
      include   'cdata.h'
      include   'debugs.h'
      include   'eldata.h'
      include   'iofile.h'
      integer ix(*)
      integer ndf,ndm,nst,isw
      integer nod,ll,i,j,kk,l
      real*8 d(*),ul(ndf,*),xl(ndm,*),tl(*),s(nst,nst),p(*)
      real*8 aa(4,4),bmat(4,nst),btc(nst,4)
      real*8 sg(5),wg(5)
      real*8 eps(4),sig(4),vl(6),help(6,1),coor(2)
      real*8 yo,nu,thick,sn,cs,sl,rr,Pi,dvol
      logical errck,pinput
c
      if(isw.eq.1) then
c-----------------------------------------------------------------------
c...  Input of material values
    1 if(ior.lt.0) write(*,3000)
        errck=pinput(d,4)
        if(errck) go to 1
        if(ior.lt.0) then
          write(*,2000) d(1),d(2),d(3),d(4)
        end if
        write(iow,2000) d(1),d(2),d(3),d(4)
c
      elseif(isw.eq.2) then
c-----------------------------------------------------------------------
```

```
c...  Element check for mistakes
        if(d(3) .le. 0.d0) then
          write(*,*)'  shell_thickness .le. 0'
          stop
        endif
c
      elseif(isw.eq.3 .or. isw.eq.4) then
c-----------------------------------------------------------------------
c...  Calculation of the stiffness matrix and the residuum
c...  Material- and element parameter
        yo    = d(1)
        nu    = d(2)
        thick = d(3)
        Pi = 4.d0*datan(1.d0)
c...  Length and angle
        sn = xl(2,2) - xl(2,1)
        cs = xl(1,2) - xl(1,1)
        sl = dsqrt(cs*cs + sn*sn)
        sn = sn/sl
        cs = cs/sl
c...  Calculation of the local displacements
        call pzero(vl,6*1)
        vl(1) =       cs*ul(1,1) + sn*ul(2,1)
        vl(2) = -1.d0*sn*ul(1,1) + cs*ul(2,1)
        vl(3) = ul(3,1)
        vl(4) =       cs*ul(1,2) + sn*ul(2,2)
        vl(5) = -1.d0*sn*ul(1,2) + cs*ul(2,2)
        vl(6) = ul(3,2)
c...  Elasticity matrix and in advance integration
        call pzero(aa,4*4)
        call dmat05(yo,nu,thick,aa)
c...  Labeling of the output list
        if (isw.eq.4) then
           write(iow,100)
           write(iow,101)
        end if
c...  Number of Gauss-points and calculation of the sampling points
        ll = idint(d(4))
        call gauss05(ll,sg,wg)
c.... Gauss loop
        do l = 1,ll
          rr = (1.d0 - sg(l))*xl(2,1) + sg(l)*xl(2,2)
          dvol = rr*2.d0*Pi*sl*wg(l)
          call pzero(bmat,4*nst)
          call bmat05(bmat,sg(l),sl,sn,cs,rr)
c...  Calculation of the strains
          call pzero(eps,4*1)
          do i=1,4
            do j=1,nst
              eps(i) = eps(i) + bmat(i,j)*vl(j)
            end do
```

```
          end do
c...  Calculation of the internal force variables
          call pzero(sig,4*1)
          do i=1,4
            do j=1,4
              sig(i) = sig(i) + aa(i,j)*eps(j)
            end do
          end do
          if(isw .eq. 4)then
c.... Display statements, internal force variables into output file
          open(1, file='beisp1.dat')
          coor(1) = xl(1,1) + sl*cs*sg(l)
          coor(2) = xl(2,1) + sl*sn*sg(l)
          write(iow,103) coor(1),coor(2),sig(1),sig(2),sig(3),sig(4)
          write(1,103) coor(1),coor(2),sig(1),sig(2),sig(3),sig(4)

          elseif(isw .eq. 3)then
c...  Calculation of the residuum
          call pzero(help,6*1)
          do i=1,nst
            do j=1,4
              help(i,1) = help(i,1) + bmat(j,i)*sig(j)*dvol
            end do
          end do
          call trans05(help,nst,1,cs,sn,3,2)
          do i=1,nst
            p(i) = p(i) - help(i,1)
          end do
c...  Calculate B^T C
          call pzero(btc,6*4)
          do i  = 1,nst
            do j  = 1,4
               do kk = 1,4
                  btc(i,j) = btc(i,j) + bmat(kk,i)*aa(kk,j)*dvol
               enddo
            enddo
          enddo
c...  Calculate B^T C B = stiffness matrix
          do i  = 1,nst
            do j  = 1,nst
              do kk = 1,4
                s(i,j) = s(i,j) + btc(i,kk)*bmat(kk,j)
              enddo
            enddo
          enddo
        endif    ! end of calculation of the local stiffness matrix
        end do   ! Gauss-loop
c...  Transformation of the stiffness matrix
        if(isw .eq. 3) then
          call trans05(s,nst,nst,cs,sn,ndf,1)
          call trans05(s,nst,nst,cs,sn,ndf,2)
```

```
        endif
c
c
      endif   !isw=1,2,3,4,5,6,7,8
c
c-----------------------------------------------------------------------
c...  Formate statements
 100  format(15x, 'internal_force_variables')
 101  format(6x,'z',5x,'r',10x,'n_s',8x,'n_theta',8x,'m_s',
     &            8x,'m_theta')
 103  format(5x,f5.2,1x,f5.2,1x,e12.5,1x,e12.5,1x,e12.5,1x,e12.5)
2000  format(5x,'Axis-symmetric shell element:'//
     &   5x,'Modulus of elasticity               yo = ',e12.5/
     &   5x,'Poisson's number                    nu = ',e12.5/
     &   5x,'Shell thickness                      t = ',e12.5/
     &   5x,'Number of Gauss-points              ll = ',I5//)
3000  format(' Input: Yo, nu, t, ll'/'   >',$)
      end
c
c
c
      subroutine gauss05(ll,sg,wg)
c-----------------------------------------------------------------------
c     Sampling points and weighting fo Gauss-point-integration from 0 - 1
c     ll    - Numer of Gauss-points
c     sg(5) - Sampling points
      wg(5) - Weightings
c-----------------------------------------------------------------------
      implicit none
      integer ll
      real*8 sg(5),wg(5)
c
c...  1 Gauss-points
      if(ll .eq. 1) then
        sg(1) = 0.5d0
        wg(1) = 1.d0
c...  2 Gauss-points
      elseif(ll .eq. 2) then
        sg(1) = 0.5d0*(1.d0 - 1.d0/dsqrt(3.d0))
        sg(2) = 0.5d0*(1.d0 + 1.d0/dsqrt(3.d0))
        wg(1) = 0.5d0
        wg(2) = wg(1)
c...  3 Gauss-points
      elseif(ll .eq. 3) then
        sg(1) = 0.5d0*(1.d0 - dsqrt(3.d0/5.d0))
        sg(2) = 0.5d0
        sg(3) = 0.5d0*(1.d0 + dsqrt(3.d0/5.d0))
        wg(1) = 5.d0/18.d0
        wg(2) = 4.d0/9.d0
        wg(3) = wg(1)
c...  4 Gauss-points
```

```
      elseif(ll .eq. 4) then
        sg(1) = 0.5d0*(1.d0 - dsqrt((15.d0 + 2.d0*dsqrt(30.d0))/35.d0))
        sg(2) = 0.5d0*(1.d0 - dsqrt((15.d0 - 2.d0*dsqrt(30.d0))/35.d0))
        sg(3) = 0.5d0*(1.d0 + dsqrt((15.d0 - 2.d0*dsqrt(30.d0))/35.d0))
        sg(4) = 0.5d0*(1.d0 + dsqrt((15.d0 + 2.d0*dsqrt(30.d0))/35.d0))
        wg(1) = .25d0*(1.d0 - dsqrt(30.d0)/18.d0)
        wg(2) = .25d0*(1.d0 + dsqrt(30.d0)/18.d0)
        wg(3) = wg(2)
        wg(4) = wg(1)
c...  5 Gauss-points
      elseif(ll .eq. 5) then
        sg(1) = 0.5d0*(1.d0 - dsqrt(5.d0 + 4.d0*dsqrt(5.d0/14.d0))/3.d0)
        sg(2) = 0.5d0*(1.d0 - dsqrt(5.d0 - 4.d0*dsqrt(5.d0/14.d0))/3.d0)
        sg(3) = 0.5d0
        sg(4) = 0.5d0*(1.d0 + dsqrt(5.d0 - 4.d0*dsqrt(5.d0/14.d0))/3.d0)
        sg(5) = 0.5d0*(1.d0 + dsqrt(5.d0 + 4.d0*dsqrt(5.d0/14.d0))/3.d0)
        wg(1) = (322.d0 - 13.d0*dsqrt(70.d0))/1800.d0
        wg(2) = (322.d0 + 13.d0*dsqrt(70.d0))/1800.d0
        wg(3) =  64.d0/225.d0
        wg(4) = wg(2)
        wg(5) = wg(1)
      else
        write(*,*)'    Number of Gauss-points is not correct'
        stop
      endif
      return
      end
c
      subroutine bmat05(bmat,xi,sl,sn,cs,rr)
c----------------------------------------------------------------------72
c     Calculation of the B-Operator
c       Paramter list
c       bmat(4,6)  B-Operator
c       xi         Gauss-point
c       sl         Length of element
c       sn         Sinus
c       cs         Cosinus
c       rr         Radius
c----------------------------------------------------------------------
      implicit none
      real*8 bmat(4,6)
      real*8 sl,sn,cs,rr,xi
c
c...  Calculation of the B-Operator
      bmat(1,1) = -1.d0/sl
      bmat(2,1) = (1.d0 - xi)*sn/rr
      bmat(2,2) = (1.d0 -  3.d0*xi*xi + 2.d0*xi*xi*xi)*cs/rr
      bmat(2,3) = (xi - 2.d0*xi*xi + xi*xi*xi)*cs*sl/rr
      bmat(3,2) = -(6.d0 - 12.d0*xi)/(sl*sl)
      bmat(3,3) = -(4.d0 - 6.d0*xi)/sl
      bmat(4,2) = -(6.d0*xi - 6.d0*xi*xi)*sn/(rr*sl)
```

```
      bmat(4,3) = -(-1.d0 + 4.d0*xi - 3.d0*xi*xi)*sn/rr
      bmat(1,4) = 1.d0/sl
      bmat(2,4) = xi*sn/rr
      bmat(2,5) = (3.d0*xi*xi - 2.d0*xi*xi*xi)*cs/rr
      bmat(2,6) = (-1.d0*xi*xi + xi*xi*xi)*cs*sl/rr
      bmat(3,5) = -(-6.d0 + 12.d0*xi)/(sl*sl)
      bmat(3,6) = -(2.d0 - 6.d0*xi)/sl
      bmat(4,5) = -(-6.d0*xi + 6.d0*xi*xi)*sn/(rr*sl)
      bmat(4,6) = -(2.d0*xi - 3.d0*xi*xi)*sn/rr
      return
      end
c
      subroutine dmat05(yo,nu,thick,aa)
c-----------------------------------------------------------------------72
c     Calculation of the material matrix,
c     in advance integration over the shell thickness
c     Parameter list
c       yo      Modulus of elasticity
c       nu      Poisson's number
c       thick   Shell thickness
c       aa(4,4) Material matrix
c-----------------------------------------------------------------------
      implicit none
      real*8 yo,nu,thick,fakt
      real*8 aa(4,4)
c
      fakt = yo*thick/(1.d0-nu*nu)
      aa(1,1) = fakt * 1.d0
      aa(1,2) = fakt * nu
      aa(2,1) = aa(1,2)
      aa(2,2) = aa(1,1)
      aa(3,3) = fakt * thick*thick/12.d0
      aa(3,4) = nu * aa(3,3)
      aa(4,3) = aa(3,4)
      aa(4,4) = aa(3,3)
c
      return
      end
c
c
      subroutine trans05(a,n,m,cs,sn,k,iswl)
c-----------------------------------------------------------------------
c     Transformation of a matrix
c        a(n,m)      Matrix
c        n,m         Rows, columns
c        cs          Cosinus
c        sn          Sinus
c        k           Step size of the loop
c        iswl        Control-variable
c-----------------------------------------------------------------------
      implicit none
```

```
c
      integer i,j,n,m,k,iswl
      real*8 t,cs,sn
      real*8 a(n,m)
c
c...  S * T
      if (iswl.eq.1) then
        do i=1,m,k
          do j=1,n
            t        = a(j,i)*cs - a(j,i+1)*sn
            a(j,i+1) = a(j,i)*sn + a(j,i+1)*cs
            a(j,i)   = t
          end do
        end do
c
c...  T^T * S * T
      elseif (iswl.eq.2) then
        do i=1,n,k
          do j=1,m
            t        = a(i,j)*cs - a(i+1,j)*sn
            a(i+1,j) = a(i,j)*sn + a(i+1,j)*cs
            a(i,j)   = t
          end do
        end do
      end if
c
      return
      end
```

References

[1] D. Braess. *Finite Elemente.* Springer, 1992.