LECTURE NOTES ON FINITE ELEMENT ANALYSIS

Prof. Niels Højen Østergaard, University of Applied Sciences Hochschule Rhein-Waal

April 6, 2015

1 Introduction

These lecture notes were written as teaching material for courses in finite element analysis at the University of Applied Sciences Rhein-Waal (Hochschule Rhein-Waal). The notes may both be applied as stand-alone teaching material for courses introducing finite element techniques to students who do not have mechanical engineering as main subject, and as supplementary to text books for students in mechanical engineering.
In the lecture notes, the basic mechanical concepts of bars and plane stress elements are summarized, before the differential equations governing the considered problems are converted to finite element formulations.
Students who are familiar with the general theory of elasticity are encouraged anyway to read the summaries contained in these notes, since it is crusial to have the details of the basic theories in mind when studying the details of finite element methods.
The present section contains a brief introduction to the method and the underlying principles.
The code examples contained in these notes were written using the GNU Octave 3.0.0-beta version. They should work in MATLAB as well, but minor modifications may be required.
Bugs and typos can be reported to niels.ostergaard@hochschule-rhein-waal.de.

1.1 What is finite element analysis (FEA)

The finite element method (FEM) is a method for approximating solutions to differential equations. In mechanical, civil and structural engineering, finite element analysis (FEA) is applied for design of virtually all systems and components, which are too complex to analyze using analytical methods and hand calculatons. Typical results obtained by FEA are stresses and strains, forces and deformations.
Industrial application of FEM for design purposes originates from analysis of aeroplane wings during World War II. Further developments of the method were conducted in the aerospace industry in the late 40s and early 50s, leading to the formulation techniques, which today are implemented in a high number of commercially available computer programms. Examples of such are Ansys, Abaqus, Nastran (developed by NASA), Cosmos (for SolidWorks), and Mechanica (for Pro/Engineer). The method is widely applied for

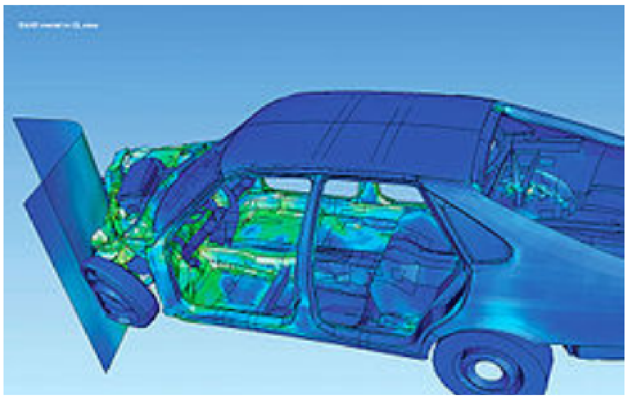

Figure 1: Example of results plot from a finite element analysis (picture licensed under creative commons)

analysis in the automotive industry (see figure 1). Other examples of applications are analysis of fiber composite structures like wind turbine wings, determination of eigen-frequencies of bridges, and calculation of stresses in pressure tanks.
The motivation for learning the theoretical basis of FEM is easy to comprehend. All results presented in smooth looking contour plots as shown in figure 1 intuitively seem correct. However, small changes in model input or mistakes made by the analysist when defining a FEM-model, can in many cases alter the obtained results completely (or said in a different manner: garbage in, garbage out). Therefore, it is crusial to have profound knowledge of the method, in order to assess the correctness of obtained results and define analytical verification cases in order to ensure that the defined model performs as intended. An old saying in mechanical engineering states that "FEM makes a good engineer better and a poor engineer a lot more dangerous". This is our motivation for opening the FEM-black box to have a carefull look inside it.

1.2 The finite element paradigm

This section will introduce the FEM framework and define the most basic terms required to formulate problems with FEM. The procedure is visualized in figure 2.
Initially, we chose a **domain** which is subject of analysis. This can be a mechanical part, a structural component or like in figure 2 a simple plate constituted by a rectangular and a triangular part. The mechanical behavior of the domain in terms of stresses and strains, forces and deformations, are governed by differential equations.The solution to those will be approximated by FEA. The structure is subjected to **external loads** and boundary conditions (**BCDs**), which prescribe which parts of the structure are free to

deform and which are not. In most cases, it is either not feasible or possible to estimate a solution for the entire domain (when this is done, it is called Ritz-methods!). Instead we split the domain into elements and assign a number to each. Each element shape is defined by a number of points in the corners. These are called nodes and are also assigned a number each.
In FEM, the behaviour of the analysed domain is described only in terms of the deformation of the nodes. We may imagine, that a number of springs is attached to each node, and the deformation of the structure is governed solely by the compression or extension of these springs and the behaviour of the domain between the nodes is governed by **interpolation** on basis of the nodal deformations. For each spring, a corresponding nodal force can be calculated based on the spring stiffness. The number of springs attached to each node constitutes a measure for in how many directions the node can deform and are called degrees of freedom (**DOFs**). All stiffness parameters for the domain are assembled in a matrix $[K]$ called the stiffness matrix. This matrix relates all nodal forces $\{F\}$ to all nodal deformations $\{F\}$ by

$$\{F\} = [K]\{d\} \tag{1}$$

If the analyzed system is linear, $[K]$ remains constant no matter which loads and boundary conditions we apply. The most important assumptions, which must be fulfilled in order to assume linearity, are linear elastic material behaviour and that deformations are small compared to the dimensions of the analyzed domain. If the way of writing a system of equations applied in equation 1 seems unfamiliar or even unpleasent, the reader is encouraged to take a look a the unltra-brief introduction to matrix algebra in section 3.
If the last part of this introduction contained words the reader does not recall or understand, DON'T PANIC - we will return to this later. It is presently sufficient if you recall, that a domain is split into elements consisting of a number of nodes. The nodal deformations due to external loads and BCDs are called DOFs and are governed by spring stiffnesses. The mechanical behavior is governed solely by the nodal deformations and determined in the rest of the domain by interpolation.
As mentioned, FEM provides approximated solutions. Opposite *strong* methods like analytical solutions or numerical solutions obtained by direct integration (for example of finite difference equations), the FEM provides an approximated solution, and we therefore refer to the method as *weak*.
Another important property of a FEA-model is how fine the mesh is, or said in a different manner, how many elements the domain has been divided into. Some models, for example constituted by truss and beam elements, can generate high precision results with very few elements (corresponding to very coarse meshes). Other models, for example constituted by plane triangles or solid brick elements, usually require fine meshes. The mesh must be so fine, that the interpolation functions of the seperate elements are sufficiently accurate to approximate the nodal soluton to the problem. As a rule of thumb for such models, coarse meshes can be used when only deformations are of interest for the analyst, while fine meshes are required for accurate calculation of the stresses. However,

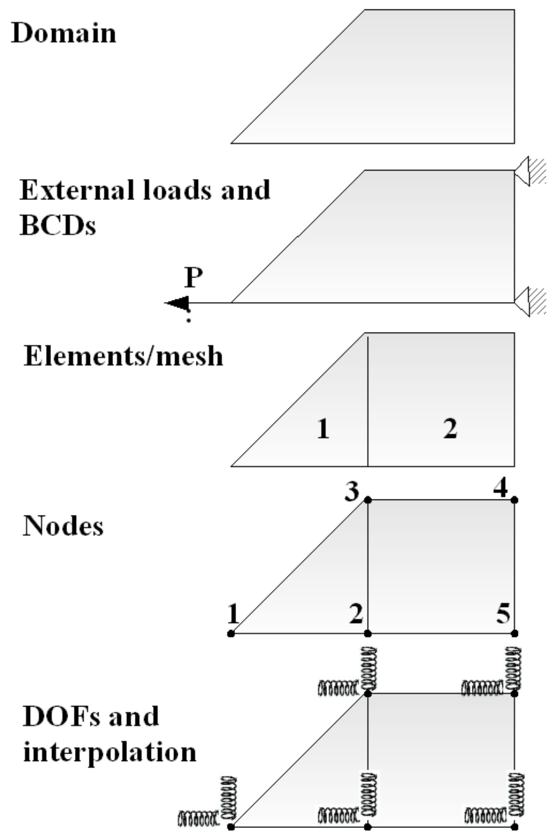

Figure 2: Explaination of the basis of the common procedure by FEA

it applies for both deformations and stresses that results can be considered inaccurate, if re-analysis with a finer mesh cause significant changes in results. We will introduce the term **convergence**, which for FEA-models means that the number of elements or DOFs applied is of a magnitude where re-analysis with a finer mesh only cause small changes in the obtained results. In the following sections, the mechanics of spring systems will be considered along with interpolation by matrix manipulation, since an understanding of both of these concepts will be required.
We will in the following chapters learn that there are different elements and that these due to their different properties are well-posed for different problems.

1.3 Summary of linear equations on matrix form

In this section, a brief summary of how to solve systems of linear equations using matrix algebra will be given. This form is very convenient when solving systems of linear equations using a computer, but also allows us to conduct analytical manipulations of large systems of equations in an efficient manner. In order to provide a brief and simple explenation, the case of a two dimensional linear system will be considered

$$\begin{aligned} y &= a_1 x + b_1 \\ y &= a_2 x + b_2 \end{aligned} \tag{2}$$

The solution (x, y) to this system is constituted by the variables which simultanously fulfill both of the linear equations. The equations are in plane geometry visualized as two straight lines with slopes a_1 and a_2 and intersections with the y-axis b_1 and b_2 where (x, y) are the coordinates of the points where the lines intersect. The two linear equations can be rewritten on the form

$$\begin{aligned} b_1 &= -a_1 x + y \\ b_2 &= -a_2 x + y \end{aligned} \tag{3}$$

On matrix form, equation 3 can be written as follows

$$\left\{ \begin{array}{c} b_1 \\ b_2 \end{array} \right\} = \left[\begin{array}{cc} -a_1 & 1 \\ -a_2 & 1 \end{array} \right] \left\{ \begin{array}{c} x \\ y \end{array} \right\} = \left\{ \begin{array}{c} -a_1 x + y \\ -a_2 x + y \end{array} \right\} \tag{4}$$

It is noted that a matrix is multiplied with a column vector by the technique demonstrated in the equation. Hence, the product is a vector. Utilizing a more compact notation, the system of equations can be written on the form

$$\{b\} = [A]\{x\} \tag{5}$$

In order to obtain the solution to the system of equations, x and y must be isolated. In order to do so, the matrix $[A]$ must be eliminated from the right side of the equation. This can be done by multiplication of the so-called inverse of the matrix $[A]$, which is denoted $[A]^{-1}$. Always remembering that matrix multiplication is a non-commulative mathmatical operation (hence, multiplication from left and right produce different results), the following expression is obtained

$$[A]^{-1}\{b\} = [A]^{-1}[A]\{x\} = \{x\} \tag{6}$$

in which we have utilized that $[A]^{-1}[A] = [I]$. $[I]$ is the identity matrix which for the 2D case is given by

$$[I] = \left[\begin{array}{cc} 1 & 0 \\ 0 & 1 \end{array} \right] \tag{7}$$

It is observed that multiplication of a matrix with its own inverse cancels out the matrix. A matrix with no inverse is called singular.
In the following, the procedure outlined above is demonstrated as a calculated example. The system shown below will have $(x, y) = (2.5, 2.5)$ as solution, which in figure 3 is visualized as the intersection between the two lines (drawn in red and blue). The intersection is marked with black lines.

```
clc; close all; clear all;
%Define coefficients of two linear equations
a1=1; b1=0; a2=-1; b2=5;
%Setup vector containing the b-coefficients
b(1)=b1; b(2)=b2;
%Setup the coefficient matrix
A(1,1)=-a1; A(1,2)=1; A(2,1)=-a2; A(2,2)=1;
%Solve system
X=inv(A)*b';
x=X(1); y=X(2);
%Plot solution
for i=1:101;
    x_plot(i)=20/(101)*i-10;
    y1_plot(i)=a1*x_plot(i)+b1;
    y2_plot(i)=a2*x_plot(i)+b2;
end
plot(x_plot,y1_plot,'b',x_plot,y2_plot,'r');
hold on; plot([x x],[-10 10],'k');
hold on; plot([-10 10],[y y],'k');
axis([-10 10 -10 10])
```

The method, which in this section was illustrated for the 2D case can without loss of generality be applied to systems of equations of larger dimensions (though the solution cannot be visualized in a manner as simple as in the 2D case).

1.4 Interpolation of functions

Interpolation is the art of determining a function $y = f(x)$, in our case a polynomial, satisfying a number of requirements. For example it may be required that he polynomial must pass through a point (x_i, y_i) or that the tangent to the polynomial for a specified x-coordinate must have a specified slope, (x_j, y'_j) (equivalent to specifying $f'(x)$). In order to solve the interpolation problem, the number of coefficients (unknowns) in the polynomial must equal the number of specified requirements to produce n equations with n unknowns.
If a cubic (third order) polynomial is considered, this is given by

$$p(x) = a + bx + cx^2 + dx^3 \tag{8}$$

If the derivative of p is specified in a point x, the following must be fulfilled

$$p'(x) = b + 2cx + 3dx^2 \tag{9}$$

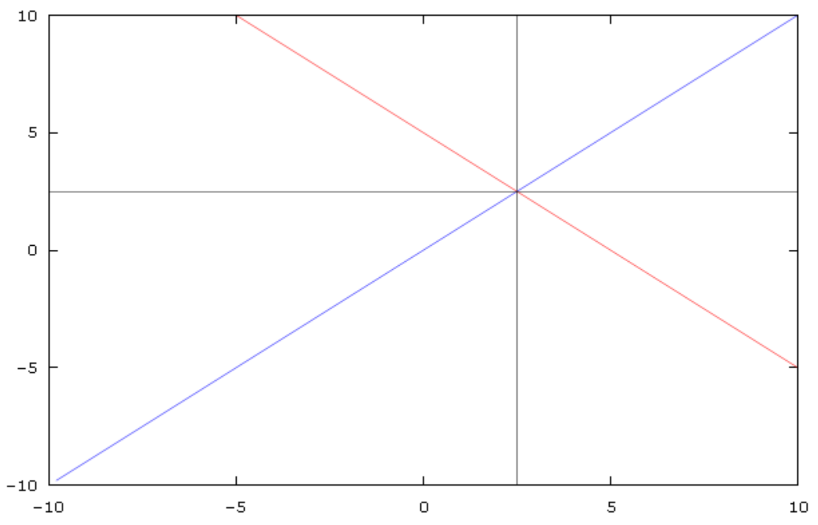

Figure 3: The solution to a 2D linear system of equations

The coefficients of a cubic polynomial passing through four specified points can be obtained by solving the following system

$$\left\{\begin{array}{c} f(x_1) \\ f(x_2) \\ f(x_3) \\ f(x_4) \end{array}\right\} = \left[\begin{array}{cccc} 1 & x_1 & x_1^2 & x_1^3 \\ 1 & x_2 & x_2^2 & x_2^3 \\ 1 & x_3 & x_3^2 & x_3^3 \\ 1 & x_4 & x_4^2 & x_4^3 \end{array}\right] \left\{\begin{array}{c} a \\ b \\ d \\ d \end{array}\right\} \tag{10}$$

It is noted that the system is linear since we solve for coefficients though the interpolation polynomial is non-linear. The non-linearity vanishes, when the coordinates of the specified points are substituted into the coefficient matrix. If a cubic polynomial is required to pass through two prescribed points and furthermore in two points to have a prescribed first order derivative, the linear system to solve becomes

$$\left\{\begin{array}{c} f(x_1) \\ f'(x_1) \\ f(x_2) \\ f'(x_2) \end{array}\right\} = \left[\begin{array}{cccc} 1 & x_1 & x_1^2 & x_1^3 \\ 0 & 1 & 2x_2 & 3x_2^2 \\ 1 & x_3 & x_3^2 & x_3^3 \\ 0 & 1 & 2x_4 & 3x_4^2 \end{array}\right] \left\{\begin{array}{c} a \\ b \\ d \\ d \end{array}\right\} \tag{11}$$

In form of Matlab code, equation 11 takes a form as shown in the following. The polynomial determined will here pass through $(2, 5)$ and $(4, 7)$ and in both of these two points have a first order derivative with a value of -1. The determined polynomial is shown in figure 4.
The matrix based principle for interpolation described here can easily be applied for interpolation with a different polynomial order. However, the reader should note that

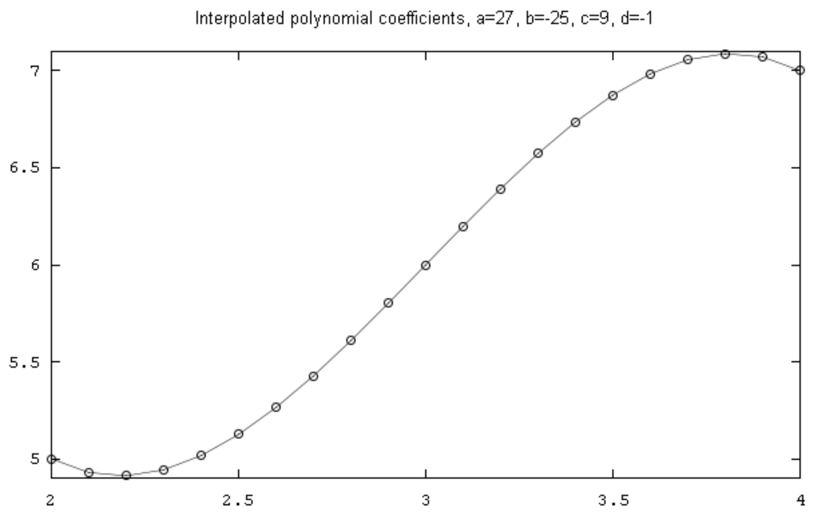

Figure 4: Example of interpolation with cubic (3rd order) polynomial

polynomials of higher order determined by this method tend to oscillate between points leading to an undesired fit. It is therefore common to limit the polynomial order to 3 occasionally extended to 4.
The matrix formulation demonstrated in this section is widely applied in finite element techniques. When applied for FEM, interpolation functions are often referred to by the name **shape functions**.

```
clc; close all; clear all;
%Input x, y=f(x)
x1=2.0; y1=5.0;          x2=4.0; y2=7.0;
%Input y=f'(x) for x1 and x2
dy1=-1.0;                dy2=dy1;
%Formulate problem
f(1,1)=y1; f(2,1)=dy1; f(3,1)=y2; f(4,1)=dy2;
A(1,1)=1;A(1,2)=x1; A(1,3)=x1^2; A(1,4)=x1^3;
A(2,1)=0;A(2,2)=1;  A(2,3)=2*x1; A(2,4)=3*x1^2;
A(3,1)=1;A(3,2)=x2; A(3,3)=x2^2; A(3,4)=x2^3;
A(4,1)=0;A(4,2)=1;  A(4,3)=2*x2; A(4,4)=3*x2^2;
%Calculate the polynomial coefficients
a=inv(A)*f;
%Plot the solution
```

```
for i=1:21;
    x(i)=(x2-x1)/20*(i-1)+x1;
    y(i)=[1 x(i) x(i)^2 x(i)^3]*a;
end
figure;plot(x,y,'ko-')
```

1.5 A single linear elastic spring

In the present section, the stiffness matrix of a single linear elastic spring is derived. By linear elastic, it is meant that the relation between force and deformation is linear, see figure 5.
Initially, the right end of the spring will be fixed by setting $u_2 = 0$. This yields the following equation

$$\sum F_{x,1} = ku_1 \tag{12}$$

From equilibrium ($\Sigma F_x = 0$) it follows that

$$\sum F_{x,2} = -ku_1 = -F_{x,1} \tag{13}$$

Now instead fixing the left end by setting $u_1 = 0$, the following equation must hold

$$\sum F_{x,2} = -F_{x,2} = -ku_1 \tag{14}$$

Superposition of these two equations provides the expressions

$$\begin{aligned} F_{x,1} &= ku_1 + ku_2 && (15) \\ F_{x,2} &= -ku_2 + ku_1 && (16) \end{aligned}$$

These two equations with two unknowns can easily be converted to matrix form

$$\left\{ \begin{array}{c} F_{x,1} \\ F_{x,2} \end{array} \right\} = \left[\begin{array}{cc} k & -k \\ -k & k \end{array} \right] \left\{ \begin{array}{c} u_1 \\ u_2 \end{array} \right\} \tag{17}$$

On compact form, this is written

$$\{F\} = [K]\{d\} \tag{18}$$

1.6 Two linear elastic springs in line

We shall now consider the problem of assembling a stiffness matrix for more than one element. Two springs connected in series will be considered, see figure 6. Initially, only deformation of the first node will be considered. Hence, $u_2 = u_3 = 0$. The first nodal force can be determined by

$$F_{x,1} = k_a u_1 = -F_{x,2} \tag{19}$$

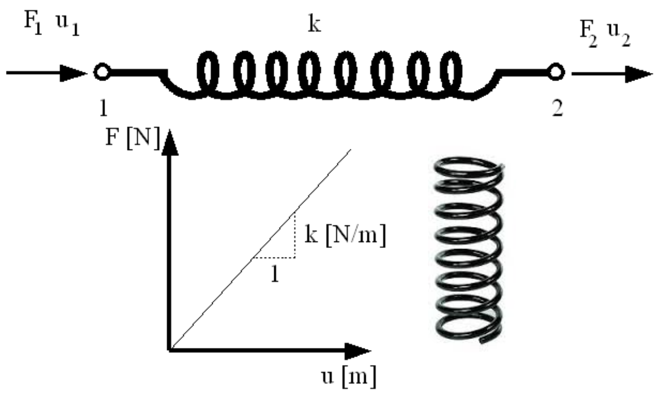

Figure 5: A single elastic spring subjected to end forces F_1 and F_2

Now, setting $u_1 = u_3 = 0$, it follows that

$$\begin{aligned} F_{x,2} &= (k_a + k_b)u_2 \\ F_{x,1} &= -k_a u_2 \\ F_{x,3} &= -k_b u_2 \end{aligned} \tag{20}$$

Finally, by setting $u_1 = u_2 = 0$, the following expression is obtained

$$\begin{aligned} F_{x,3} &= k_b u_3 \\ &= -F_{x,2} \\ F_{x,1} &= 0 \end{aligned} \tag{21}$$

By superposition, the following three equations with three unknowns are obtained

$$\begin{aligned} F_{x,1} &= -k_a u_1 - k_a u_2 \\ F_{x,2} &= -k_a u_1 + (k_a + k_b)u_2 - k_b u_3 \\ F_{x,3} &= -k_b u_2 + k_b u_3 \end{aligned} \tag{22}$$

On matrix form, this corresponds to

$$\left\{ \begin{array}{c} F_{x,1} \\ F_{x,2} \\ F_{x,3} \end{array} \right\} = \left[\begin{array}{ccc} k_a & -k_a & 0 \\ -k_a & k_a + k_b & -k_b \\ 0 & -k_b & k_b \end{array} \right] \left\{ \begin{array}{c} u_1 \\ u_2 \\ u_3 \end{array} \right\} \tag{23}$$

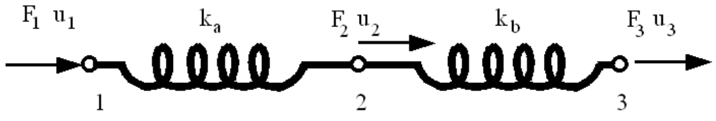

Figure 6: Two elastic springs connected in series

$$\begin{bmatrix} k_a & -k_a & 0 \\ -k_a & k_a + k_b & -k_b \\ 0 & -k_b & k_b \end{bmatrix} = \begin{bmatrix} k_a & -k_a & 0 \\ -k_a & k_a & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & k_b & -k_b \\ 0 & -k_b & k_b \end{bmatrix} \tag{24}$$

Important : It can observed that the total stiffness matrix of the considered system (two springs mounted in line) can be obtained by addition of the stiffness matrices of the seperate elements (in this case springs) when these are expanded with zeros at entries that correspond to DOFs that are not contained in the element. This illustrates the principle of how to assemble a stiffness matrix for multiple elements. It is noted that the DOF corresponding to node 2 has a contribution in stiffness from both strings, since this node is contained in both elements.
In general terms, the stiffness matrix of a single element is called the element stiffness matrix and is denoted $[k]$. The global stiffness matrix for the entire system is denoted $[K]$.
If all DOFs in equation 24 are left active, the stiffness matrix becomes singular. Hence, it has no inverse and the system of equations cannot be solved. The physical interpretation of this is that the spring is pulled without having a point fixed. This corresponds to a system with no boundary conditions. Obviously, this will lead to that the springs starts to move by sliding without deforming. This behabviour is called **rigid body motion**, and cannot be analysed using FEM, since this method is applied for computation of deformations.
In order to solve the system, a node must be constraint. This can be carried out in two different manners: 1) the DOF is along with corresponding F and d entries removed from the system leaving a 2×2 system of equations, 2) A large stiffness (not large enough to cause singularity) is added to the DOF, which shall be fixed, in the stiffness matrix .

1.7 Equivalent stiffness of spring connections

Finally, two very usefull formulaes will be repeated (or introduced) though slightly off-topic. We shall consider the problem of replacing a number of springs with a single spring stiffness acting in the same manner as the larger system. This will be referred to as an equivalent spring stiffness. Initially, considering figure 7-A, two springs in parallel are shown. The total force acting on the system must due to force equilibrium equal the

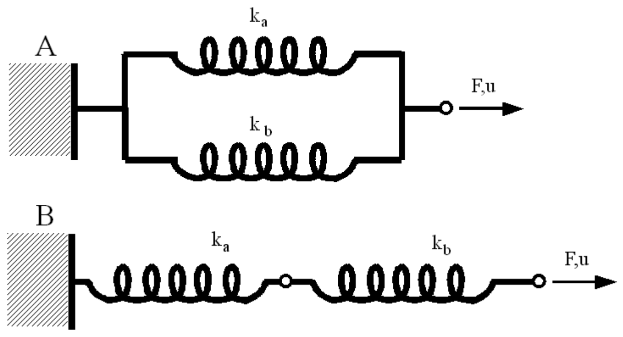

Figure 7: **A**: Two elastic springs acting parallel to each other, **B**: two elastic springs acting in line

sum of forces generated by the two springs. On equation form, it follows that

$$\begin{aligned} F &= F_a + F_b \\ &= k_a u + k_b u \\ &= (k_a + k_b)u \end{aligned} \tag{25}$$

The equivalent spring stiffness can be observed to be given by

$$k_{parallel} = k_a + k_b \tag{26}$$

Now considering figure 7-B showing two springs mounted in series, the deformation of the entire system can be observed to be given by the sum of deformations of both springs. Hence, the following must hold

$$\begin{aligned} u &= u_a + u_b \\ &= \frac{F}{k_a} + \frac{F}{k_b} \\ &= F\left(\frac{1}{k_a} + \frac{1}{k_b}\right) \end{aligned} \tag{27}$$

The equivalent stiffness is now given by

$$\frac{1}{k_{series}} = \left(\frac{1}{k_a} + \frac{1}{k_b}\right) \tag{28}$$

The two equivalent stiffness parameters can on mathematical form be observed to function opposite the way electrical resistances connected respectively in series and in parallel work.

Figure 8: Examples of truss structures, the terminal building of Düsseldorf Airport

2 FEA of truss systems

In this chapter, it will be considered how FEM can be applied for analysis of truss systems. Trusses are bars, which solely are subjected to loads in the longitudinal direction and are also referred to as "two force members". Hence, trusses are not subject of bending or torsion. Systems of bars act as trusses if all joints are constituted by pinned connections and loads solely are applied in the joints. An example of a system dominated by trusses is shown in figure 8.

2.1 Stiffness of a single truss

In figure 9, a single truss subjected to axial load P is shown. If P does not cause plasticity (permanent deformations) anywhere in the cross-section A, the following equation governs the mechanical behaviour

$$\begin{aligned} \frac{F}{A} &= E\frac{\Delta L}{L} \\ \rightarrow F &= \frac{EA}{L}d \end{aligned} \tag{29}$$

We will define the uni-directional stress as $\sigma_x = E\epsilon$ and the corresponding strain (unit deformation) as $\epsilon = \frac{\Delta L}{L}$ Setting $\Delta L = d$ in equation 29, the truss can be observed

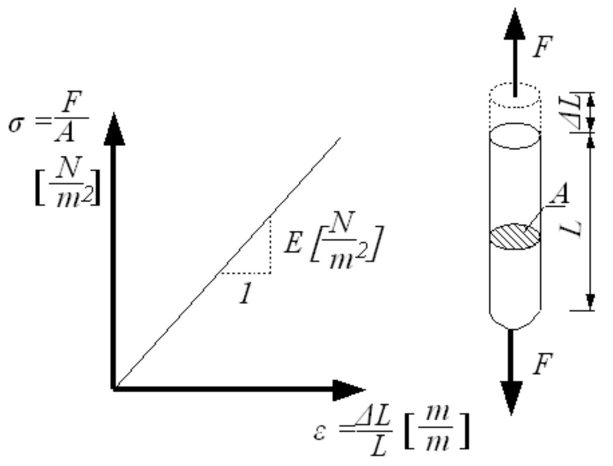

Figure 9: Mathematical relation between force and deformation for a single linear elastic bar/truss subjected solely to axial load

to act like a spring with spring stiffness $k = \frac{EA}{L}$. Having realized this, the stiffness matrix of a truss is in accordance with equation 17 given by

$$[k] = \begin{bmatrix} \frac{EA}{L} & -\frac{EA}{L} \\ -\frac{EA}{L} & \frac{EA}{L} \end{bmatrix} \tag{30}$$

Rearranging this, gives us the following more convenient expression

$$\begin{Bmatrix} F_{x,1} \\ F_{x,2} \end{Bmatrix} = \frac{EA}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} \tag{31}$$

2.2 Accounting for spatial orientation by transformation

This far only springs and trusses oriented along the axes of a plane coordinate system have been considered. However, in systems as shown in figure 8, trusses have various orientations. In order to account for this in the mathmatical formulation of the problem, a truss of arbitrary spatial orientation is considered, see figure 10. The truss will be considered as one single element with (x, y) as global coordinate system and (x', y') as local coordinate system with x' oriented along the longitudinal direction of the truss.

In the local cordinate system, the following must hold in accordance with equation 30

$$\begin{Bmatrix} F'_{x,1} \\ F'_{x,2} \end{Bmatrix} = \frac{EA}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{Bmatrix} u'_1 \\ u'_2 \end{Bmatrix} \tag{32}$$

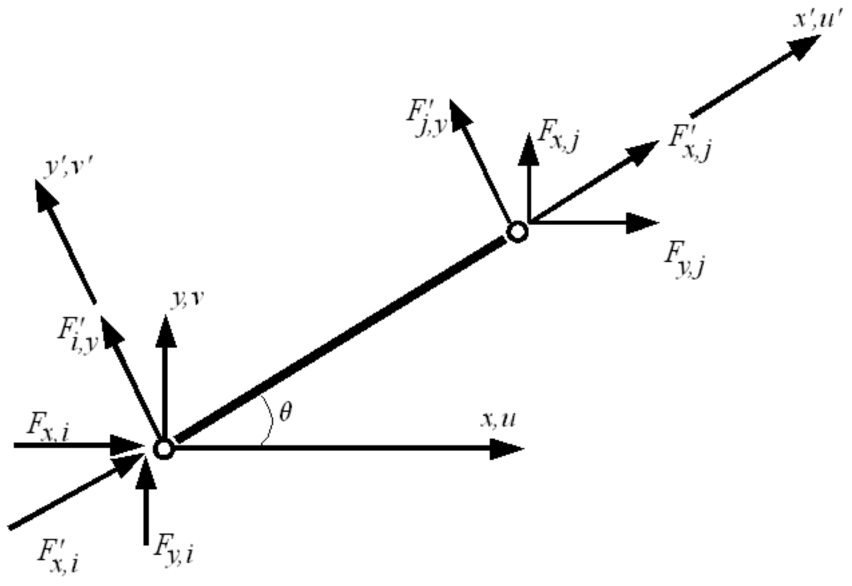

Figure 10: Definition of local (x', y') and global (x, y) coordinate systems for a truss oriented arbitrarily in the xy-plane

Projecting the forces in the (x', y') coordinate system onto x and y gives us the following expressions

$$\begin{aligned}
F'_{x,i} &= F_{x,i}\cos\theta + F_{y,i}\sin\theta \\
F'_{y,i} &= -F_{x,i}\sin\theta + F_{y,i}\cos\theta \\
F'_{x,j} &= F_{x,j}\cos\theta + F_{y,j}\sin\theta \\
F'_{y,j} &= -F_{x,j}\sin\theta + F_{y,j}\cos\theta
\end{aligned}$$

These equations relate forces in local and global coordinates. On matrix form this can be formulated as

$$\begin{Bmatrix} F'_{x,1} \\ F'_{x,2} \\ F'_{x,3} \\ F'_{x,4} \end{Bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & \cos\theta & \sin\theta \\ 0 & 0 & -\sin\theta & \cos\theta \end{bmatrix} \begin{Bmatrix} F_{x,1} \\ F_{x,2} \\ F_{x,3} \\ F_{x,4} \end{Bmatrix} \tag{33}$$

This must also hold for deformations in local and global coordinates. On compact form, this can be written

$$\{F'\} = [T]\{F\}, \{d'\} = [T]\{d\} \tag{34}$$

The governing equations on matrix form are in local coordinates in accordance with equation 30 given by

$$\{F'\} = [k']\{d'\} \tag{35}$$

Substituting the transformation given in equation 34 into this gives

$$\{T\}\{F\} = [k']\{T\}\{d\} \tag{36}$$

The governing equation in global coordinates is now

$$\begin{aligned}\{F\} &= \{T\}^{-1}[k']\{T\}\{d\} \\ &= \{T\}^{T}[k']\{T\}\{d\}\end{aligned} \tag{37}$$

since $[T]^{-1} = [T]^T$ due to the orthogonality property of the transformation matrix. The global stiffness matrix is therefore given by

$$[k] = [T]^T[k']\{T\} \tag{38}$$

Evaluating this equation analytically gives the following results

$$[k] = \frac{EA}{L}\begin{bmatrix} \cos^2\theta & \cos\theta\sin\theta & -\cos^2\theta & -\cos\theta\sin\theta \\ \cos\theta\sin\theta & \sin^2\theta & -\cos\theta\sin\theta & -\sin^2\theta \\ -\cos^2\theta & -\cos\theta\sin\theta & \cos^2\theta & \cos\theta\sin\theta \\ -\cos\theta\sin\theta & -\sin^2\theta & \cos\theta\sin\theta & \sin^2\theta \end{bmatrix} \tag{39}$$

The stresses can be calculated by

$$\sigma = \frac{E}{L}[-\cos\theta - sin\theta \cos\theta sin\theta]\begin{Bmatrix} u_i \\ v_i \\ u_j \\ v_j \end{Bmatrix} \tag{40}$$

If the longitudinal forces are required, these can easily be determined from $F = \sigma A$. The directional sines and cosines can be calculated on basis of geometrical considerations by

$$\cos\theta = \frac{x_j - x_i}{L}, \sin\theta = \frac{y_j - y_i}{L} \tag{41}$$

The theory derived in this section is in combination with the basic knowledge of FEM we gained in chapter 1 sufficient to write an FEM-code for analysis of plane truss systems. The equations can by the same principles as shown above be generalized to three-dimensional truss systems.

2.3 MyfirstFEAcode.m for analysis of truss systems

```
%MyFirstFEACode.m
%example FEA-code for truss-systems, 23022015,FH Rhein-Waal
%For analysis of other geometries,
%modify BCDs (line 50) and loads (line 55)

clc; close all; clear all;

%Input parameters %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
E=210.0*10^9;                    %Module of elasticity [N/mm^2]
Area=0.1*0.05;                   %h x t rectangular cross-section
P=100.0*10^3;                    %Load [N]
Lb=2.0;   db=Lb*tan(60*pi/180); %Frame geometry [m]
BCStiffness=10.0^50.0;           %Stiffness for BCDs and force vector
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Input nodes [x,y] for plane geometry
nodes(1,:)=[0.0 0.0];nodes(2,:)=[0.0 db];nodes(3,:)=[Lb 0];

%%%Input elements [first element node, second element node]
elem(1,:)=[1 3]; elem(2,:)=[2 3];

%%%Save number of nodes and number of elements
n_nodes=length(nodes);  n_elem=length(elem);

%%%Initialize array for stiffness matrix
K(2*n_nodes,2*n_nodes)=0;

%%%Assemble stiffness matrix
for i=1:n_elem;
%%%Set coordinates for node i
coor1=nodes(elem(i,1),:); coor2=nodes(elem(i,2),:);
%%%Calculate element length
L=sqrt((coor2(1)-coor1(1))^2+(coor2(2)-coor1(2))^2);
%%%Calculate directional cosines and sines
c=(coor2(1)-coor1(1))/L;
s=(coor2(2)-coor1(2))/L;
%%%Calculate element stiffness matrix
k=Area*E/L*[c^2    c*s     -c^2    -c*s
            c*s    s^2     -c*s    -s^2
            -c^2  -c*s     c^2     c*s
            -c*s  -s^2     c*s     s^2];
%%%Set nodenumbers for both nodes in element
```

```
n1=elem(i,1); n2=elem(i,2);
%%%Add element stiffness matrix to global stiffness matrix
K(n1*2-1:n1*2,n1*2-1:n1*2)=K(n1*2-1:n1*2,n1*2-1:n1*2)+k(1:2,1:2);
K(n2*2-1:n2*2,n2*2-1:n2*2)=K(n2*2-1:n2*2,n2*2-1:n2*2)+k(3:4,3:4);
K(n1*2-1:n1*2,n2*2-1:n2*2)=K(n1*2-1:n1*2,n2*2-1:n2*2)+k(1:2,3:4);
K(n2*2-1:n2*2,n1*2-1:n1*2)=K(n2*2-1:n2*2,n1*2-1:n1*2)+k(3:4,1:2);
end

%%%Define boundary conditions
K(1,1)=BCStiffness;
K(2,2)=BCStiffness;
K(3,3)=BCStiffness;
K(4,4)=BCStiffness;
%%%Define load vector
F(2*n_nodes)=0;          %Initialize array
F(2*n_nodes)=-P;         %Define load
%%%Solve for deformations
d=inv(K)*F'
%%%Calculate reaction forces
Rx1=-d(1,1)*BCStiffness
Ry1=-d(2,1)*BCStiffness
Rx2=-d(3,1)*BCStiffness
Ry2=-d(4,1)*BCStiffness

%%%Caluclate sectional forces and stresses for all elements
for i=1:n_elem;
%Following five lines equivalent to those used for calculation of [K]
coor1=nodes(elem(i,1),:); coor2=nodes(elem(i,2),:);
L=sqrt((coor2(1)-coor1(1))^2+(coor2(2)-coor1(2))^2);
vec=[coor2(1)-coor1(1) coor2(2)-coor1(2)];
c=(coor2(1)-coor1(1))/L;
s=(coor2(2)-coor1(2))/L;

%Set element deformations
d_e=[d(2*elem(i,1)-1) d(2*elem(i,1)) d(2*elem(i,2)-1) d(2*elem(i,2))]'

%Calculate stresses
Sigma(i)=E/L*[-c -s c s]*;

%Calculate sectional forces
Tens(i)=Sigma(i)*Area;
end
```

2.4 Calculated example I: simple truss system

In this example, the simple truss system shown in figure 11 will be analyzed in order to determine a) the reactions at node 1 and 3, b) the stresses in each truss.

In case the reader somehow has forgotten this, force and moment equilibrium must apply both at global and component level.

$$\sum F_x = 0 \qquad \sum F_y = 0 \qquad \sum M_z = 0 \tag{42}$$

These equations are a special case of $\Sigma F = ma$ by which the acceleration a is 0 for a statical case. The applied equation is (obviously) Newton's second law. It will in the following be shown how this works.
Considering figure 11, the vertical dimension is given by $d_b = L_b tan(\theta)$; We will now aim to determine the reaction forces. Global static equilibrium gives us the following two equations

$$\sum F_x = R_{x,1} + R_{x,2} = 0 \rightarrow R_{x,1} = -R_{x,2} \tag{43}$$

$$\sum F_y = -P + R_{y,1} + R_{y,2} = 0 \tag{44}$$

Equivalently, moment equilibrium provides the equation

$$\sum M_2 = PL_b - R_{x,1}d_b \rightarrow R_{x,1} = P\frac{L_b}{d_b} = 0 \tag{45}$$

Hence, we have $R_{x,2} = -P\frac{L}{d}$. However, this is insufficient to determine all reactions. Considering component-wise equilibrium and considering the overbar, the following must hold

$$\sum F_x = Fcos(\theta) + R_{x,2} = 0 \rightarrow F = \frac{-R_{x,2}}{cos(\theta)} = P\frac{L_b}{d_b}\frac{1}{cos(\theta)} \tag{46}$$

$$\begin{aligned} \sum F_y = -Fsin(\theta) + R_{y,2} = 0 \rightarrow R_{y,2} &= Fsin(\theta) = P\frac{L_b}{d_b}\frac{sin(\theta)}{cos(\theta)} \\ &= P\frac{L_b}{d_b}\frac{d_b}{cos(L_b)} = P \end{aligned} \tag{47}$$

Additionally, the following holds: $R_{y,1} = P - R_{y,2} = P - P = 0$

Having determined all reaction forces analytically, the FEA code $MyfirstFEAcode.m$ can be applied for analysis of the truss system. For the steel material parameters given in the example and bars of dimensions $h \times b = 0.1m \times 0.05m$, $L = 2.0m$ and $\theta = 60deg$, the following reactions are obtained with the code:

$R_{x,1} = 58kN$	$R_{x,2} = -58kN$
$R_{y,1} = 0kN$	$R_{y,2} = 100kN$

The same values are obtained if the input given above is substituted into the equations derived for the equilibrium. The stresses are shown in figure 12,B. These correspond to the values derived analytically using $\sigma = \frac{F}{A}$

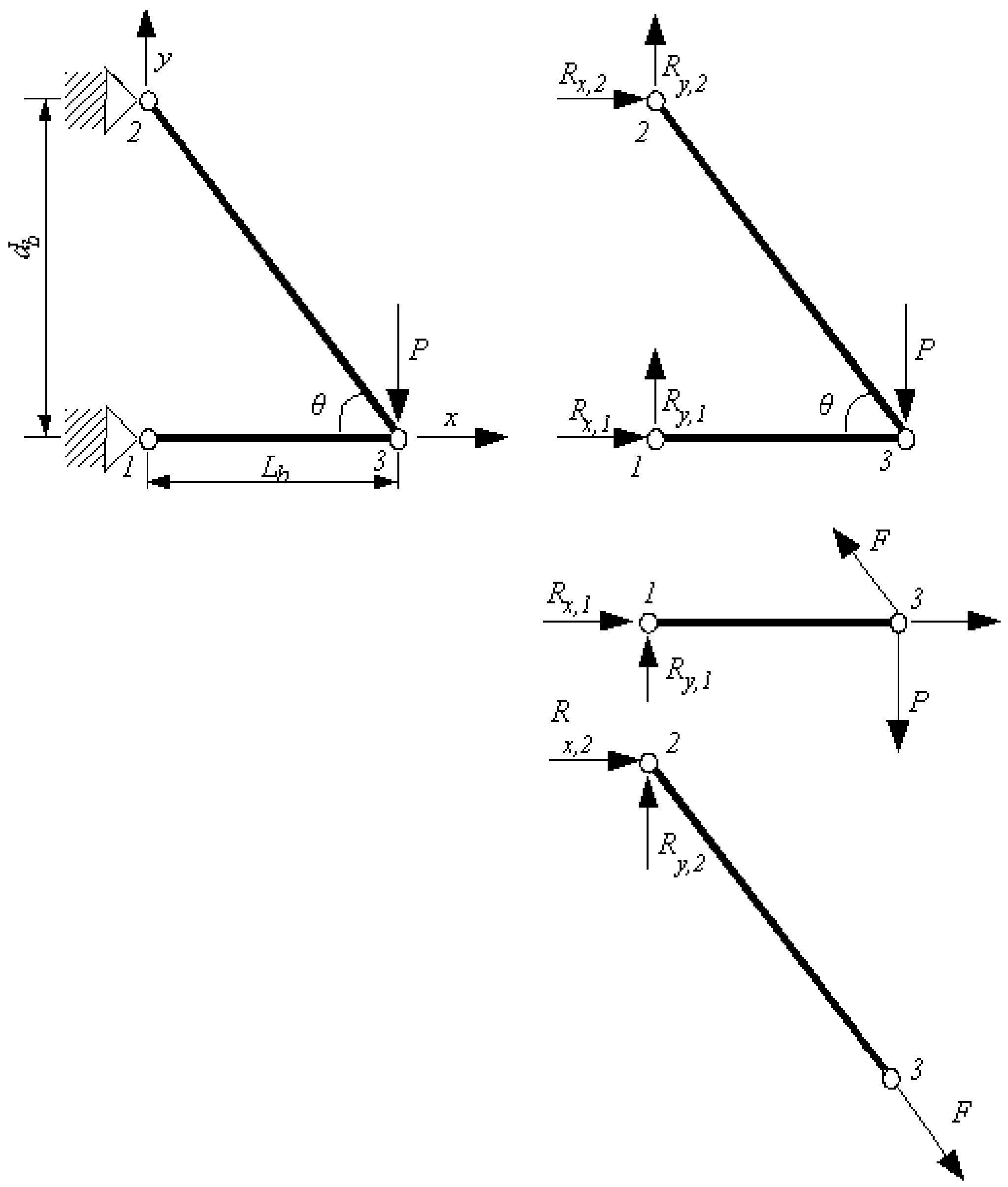

Figure 11: Truss structure which is subject of analysis

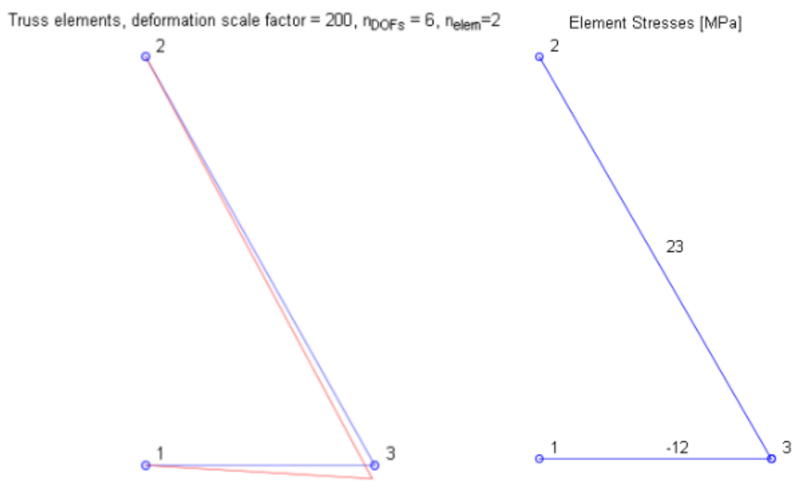
Truss elements, deformation scale factor = 200, n_{DOFs} = 6, n_{elem}=2
2
1
3
Element Stresses [MPa]
2
23
1
-12
3

Figure 12: **left** : undeformed and scaled deformed nodal solution to the FEA-problem, **right** : calculated element stresses, note that tensile stresses are positive while compressive stresses are negative

2.5 Calculated example II: plane frame structure

Larger frames can now be analyzed, see for example figure 13. The node coordinates and element definitions applied for generating this structure is contained in Appendix A.

2.6 Adaption to general FEA framework

In the previous section, the stiffness matrix for a truss was derived on basis of the basic matrix formulations for spring systems. Having demonstrated how effectively this works, we will now turn to a framework for derivation of FEA formulations of more general (or abstract) nature. Initially, energy principles in finite element will briefly be explained, before these are converted to a form appropriate for FEM. Finally, this will be applied to derive the stiffness matrix of the single truss element in equation 30 again applying the general theoretical framework.
Remember not to panic - though looking mean, the expressions are actually mostly harmless for the cases we consider.
The total elastic potential energy of a structure is given by $\Pi = U + V$, in which U is the strain energy and V constitutes the potential of externally applied forces to perform work. Hence, Π is constituted by a term related to internal- and a term related to external forces. The total elastic potential can be formulated in terms of stresses, strains, loads and deformations on the following form (this will in the present context

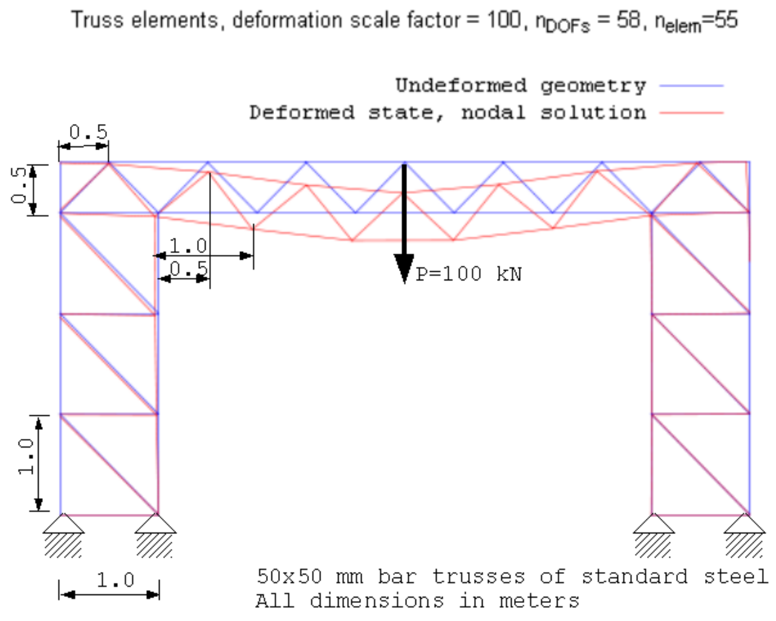
Truss elements, deformation scale factor = 100, n_{DOFs} = 58, n_{elem}=55
Undeformed geometry
Deformed state, nodal solution
0.5
0.5
1.0
0.5
P=100 kN
1.0
1.0
50x50 mm bar trusses of standard steel
All dimensions in meters

Figure 13: FEA model and scaled deformation from nodal solution to large truss frame problem

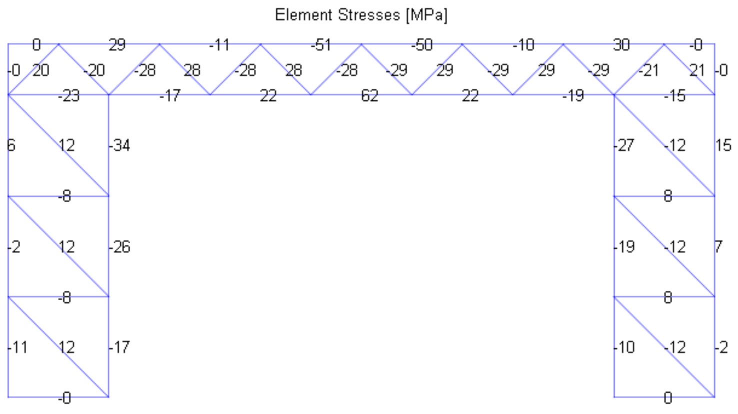

Figure 14: Stresses calculated on basis of nodal solution

simply be postulated)

$$\Pi = U + V = \frac{1}{2}\int_V \{\epsilon\}^T\{\sigma\}dV - \{\delta^e\}^T\{F^e\} \tag{48}$$

We will now apply interpolation. Having the coefficients of the shape functions contained in the column vector $\{a\}$, the nodal displacements will be interpolated as follows

$$\{d^e\} = [A]\{a\} \tag{49}$$

in which $[A]$ is called the coefficient matrix.
In an equivalent manner, the strains will be interpolated as

$$\{\epsilon\} = [C]\{a\} \tag{50}$$

where $[C]$ will be denoted the strain matrix.
Finally, on basis of the constitutive (or material) law linking geometry to equilibrium, the stresses and strains are linked by

$$[\sigma] = [D]\{\epsilon\} \tag{51}$$

(this is no interpolation, but simply the constitutive relations on matrix form). Combining equations 49 and 50, the following expression is obtained for strains

$$\{\epsilon\} = [C]\{a\} = [C][A]^{-1}\{\delta^e\} \tag{52}$$

It is now convenient to define $[B] = [C][A]^{-1}$, which we will denote the strain-displacement matrix and enables us to simplify equation 52 to

$$\{\epsilon\} = [B]\{\delta^e\} \tag{53}$$

Equation 48 can now be rewritten as

$$U + V = \frac{1}{2}\int_V \{\delta^e\}([A]^{-1})^T[C]^T[D][C][A]^{-1}\{\delta^e\}dV - \{\delta^e\}^T\{F^e\} \tag{54}$$

The total elastic potential can on this form be considered a function of a function. We refer to this as a *functional* on integral form. Here comes something slightly tricky: for the independent variables (in our case δ^e) corresponding to equilibrium, the total elastic potential has a stationary value. This means that the first variation is 0, which may sound a bit like gibberish. However, this is for functionals what an extreema fulfilling $f'(x) = 0$ constitutes for a plane curve. The stationarity of the total elastic potential is among the most basic (some say the most basic) principles in mechanical physics and lots and lots of other theorems and principles can be derived from this. However, in our case this means, that if $\frac{\partial(U+V)}{\partial\delta^e}$ equals 0, δ^e defines the equilibrium state. By matrix manipulations we obtain

$$\frac{\partial(U+V)}{\partial\delta^e} = 0 \rightarrow \{F^e\} = \left[\int_V [B]^T[D][B]dV\right]\{\delta^e\} = [k]\{d\}$$

so let us return to the thruss...

The enhanced theoretical framework presented above is not going to ease the derivation in this case. However, a truss element is by nature of such simplicity, that the principle excellently can be explained for trusses in order for the reader to gain insight in how to apply the framework. For more complex elements, this methodology eases the derivations a lot.

The nodal displacements are given by

$$\{\delta^e\} = \left\{ \begin{array}{c} u_1 \\ u_2 \end{array} \right\} \tag{55}$$

(for reference, see figure 10). We will now need to interpolate the displacement field between u_1 and u_2. In order to do so, we apply a linear shape function, such that the deformations $u(x)$ can be determined by

$$\begin{aligned} u(x) &\approx f(x) = a + bx \\ &= \left[\begin{array}{cc} 1 & x \end{array} \right] \left\{ \begin{array}{c} a \\ b \end{array} \right\} \end{aligned} \tag{56}$$

On compact form, this can be written $[f(x)]\{a\}$.

Knowing the values of $f(x)$ on the boundaries $x = 0$ and $x = L$, the following must hold

$$u_1 = f(0) = a, \qquad u_2 = f(L) = a + bL \tag{57}$$

On matrix form, this can be written

$$\left\{ \begin{array}{c} u_1 \\ u_2 \end{array} \right\} = \left[\begin{array}{cc} 1 & 0 \\ 1 & L \end{array} \right] \left\{ \begin{array}{c} a \\ b \end{array} \right\} \tag{58}$$

On compact form, this can be written $\{\delta^e\}$=[A]{a}. Hence, the $[A]$-matrix has now been obtained.

It is noted that the inverse of $[A]$ is given by

$$[A]^{-1} = \left[\begin{array}{cc} 1 & 0 \\ -\frac{1}{L} & \frac{1}{L} \end{array} \right] \tag{59}$$

since this will be needed in the following. We will now consider the element strains in order to derive the $[C]$ matrix. The strain is per definition given by

$$\epsilon = \frac{du}{dx} \approx \frac{df}{dx} = b \tag{60}$$

On matrix form, this gives us

$$\epsilon = \left[\begin{array}{cc} 0 & 1 \end{array} \right] \left\{ \begin{array}{c} a \\ b \end{array} \right\} \tag{61}$$

$$= [C]\{a\} \tag{62}$$

Now having determined the [C] matrix, only the relation between stresses and strains governed by [D] is required. This is easily found by considering Hook's law for uni-directional stress

$$\sigma = E\epsilon \tag{63}$$

It follows that $[D] = E$. The strain-displacement matrix can now be calculated as

$$\begin{aligned}[B] &= [C][A]^{-1} \\ &= \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ -\frac{1}{L} & \frac{1}{L} \end{bmatrix} \\ &= \begin{bmatrix} -\frac{1}{L} & \frac{1}{L} \end{bmatrix} \end{aligned} \tag{64}$$

And that's it! Everything required to determine the element stiffness matrix has been obtained from the analysis. Plugging [B] and [D] into equation 55, we obtain

$$\begin{aligned}[k^e] &= \int_V [B]^T[D][B]dV \\ &= \int_V \left\{ \begin{matrix} -\frac{1}{L} \\ \frac{1}{L} \end{matrix} \right\} E \begin{bmatrix} -\frac{1}{L} & \frac{1}{L} \end{bmatrix} dV \\ &= EAL \begin{bmatrix} \frac{1}{L^2} & -\frac{1}{L^2} \\ -\frac{1}{L^2} & \frac{1}{L^2} \end{bmatrix} \\ &= \frac{EA}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \end{aligned} \tag{65}$$

in which we have taken advantage of that the integral in V simplifies to $\int_V dV = AL$ for bars with constant cross sections, in which A denotes cross-sectional area and L length. The obtained stiffness matrix can be observed to correspond to the result that was derived in equation 30 .

It is common to define the interpolation- or shape function matrix (in this case a row vector) in the following manner

$$\{N\} = [f(x)][A]^{-1} \tag{66}$$

This notation is only included in order for the readers to familiarize themselves with it, since it is commonly used in text books and related litterature. In the following, we will continue to use the notation that was applied in this section. For problems of more complex nature it implies multiple advantages, for example that the solutions throughout the domain easily can be approximated directly on basis of the nodal solutions and the shape functions organized in a matrix.
Summarizing, in order to derive the stiffness matrix, the following matrices are needed:

- The coefficient matrix [A] relating nodal displacements to shape function coefficients

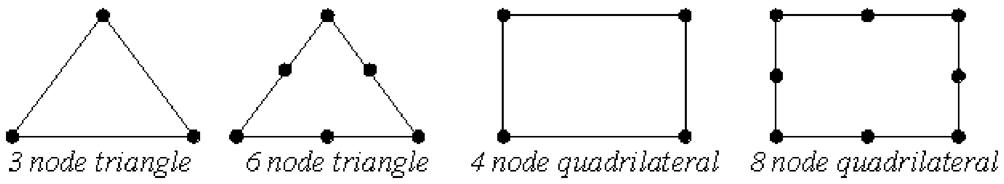

Figure 15: Overview of basic elements for plane stress analysis

- The strain matrix $[C]$ linking element strains to shape function coefficients
- The constitutive matrix $[D]$ linking element stresses to element strains

Recalling that the strain-displacement matrix is given by $[D] = [C][A]^{-1}$, the element stiffness matrix can be obtained by equation 55.

3 Plane elements

In this chapter, FEA analysis of plane stress states will be considered. By plane stress, we mean a state of stress where all components in one direction - usually denoted z - are 0. Under these conditions, the stress state can be described mathematically by two normal components σ_x and σ_y along with a shear component τ_{xy}. For practical purposes it is usually sufficient that the stress components in one direction are small compared to the remaining components. Many problems related to plates subjected to in-plane loads can be analyzed using plane stress conditions. However, it is important to note that the word *plane* refers to the state of stress and not to the geometry. The most commonly encountered case of plane stress in a curved surface is the stress state in the wall of a long pressure champer or a thin-walled pipeline. Here, the stress can be described by normal components in the hoop and longitudinal directions, while the radial stress component in the thickness direction is very small compared to the remaining components.
An overview of the most common 2D elements is shown in figure 15

3.1 Summary of the theory of elasticity for plane stress problems

In figure 16 an infinitisimally small segment of a structure in plane stress is shown. It is noted that we have used a Taylor approximation of first order to develop the stresses over the element leading to partial derivatives. Furthermore, pure normal strain and pure shear strain is showed. It is noticed that while normal strains elongate or compress the segment, shear stresses do not actually change the length of the segment sides, but only distorts the shape.

Hook's generalized law for 2-dimensional stress is given by

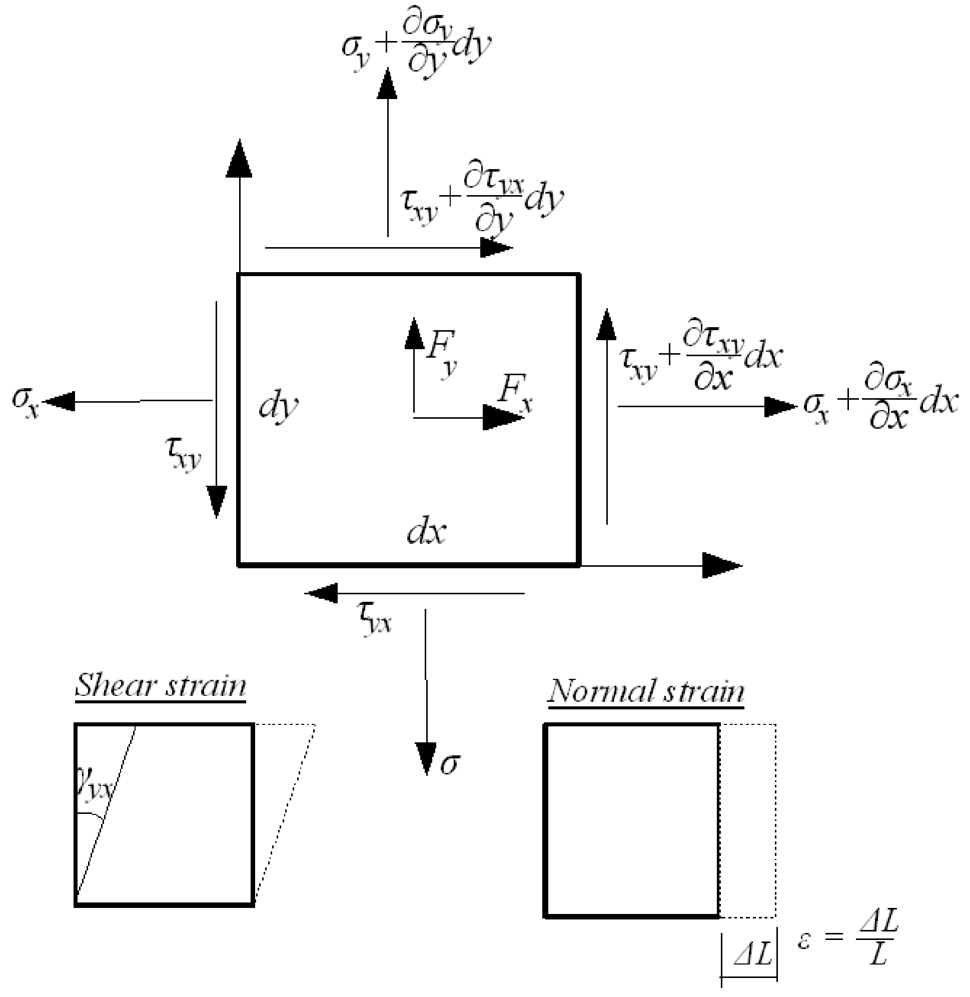
$\sigma_y + \frac{\partial \sigma_y}{\partial y} dy$
$\tau_{xy} + \frac{\partial \tau_{yx}}{\partial y} dy$
F_y
F_x
$\tau_{xy} + \frac{\partial \tau_{xy}}{\partial x} dx$
σ_x
dy
$\sigma_x + \frac{\partial \sigma_x}{\partial x} dx$
τ_{xy}
dx
τ_{yx}
Shear strain
Normal strain
γ_{yx}
σ
ΔL
$\varepsilon = \frac{\Delta L}{L}$

Figure 16: Segment of material of Inifinitisimal proportiones subjected to plane stress

$$\epsilon_x = \frac{\sigma_x}{E} - \frac{\nu\sigma_y}{E} \tag{67}$$

$$\epsilon_y = \frac{\sigma_y}{E} - \frac{\nu\sigma_x}{E} \tag{68}$$

$$\gamma_{xy} = \frac{\tau}{G} = \frac{2(1+\nu)}{E}\tau_{xy} \tag{69}$$

These equations are our constitutive relations relating geometry in form of strains (unit deformations) to equilibrium (stresses). Due to the moment equilibrium it follows that $\tau_{xy} = \tau_{yx}$.

$$\left(\sigma_x + \frac{\partial\sigma_x}{\partial x}dx\right)dy - \sigma_x dy + \left(\tau_{xy} + \frac{\partial\tau_{xy}}{\partial y}dy\right)dx - \tau_{xy}dx + F_x dxdy = 0$$

Neglecting higher order terms, this can be reduced to

$$\left(\frac{\partial\sigma_x}{\partial x} + \frac{\partial\tau_{xy}}{\partial y} + F_x\right)dxdy = 0 \tag{70}$$

Summing up the forces in the y-direction provides a similar equation governing the force equilibrium. The equilibrium equations for a state of plane stress are then given by

$$\frac{\partial\sigma_x}{\partial x} + \frac{\partial\tau_{xy}}{\partial y} + F_x = 0 \tag{71}$$

$$\frac{\partial\sigma_y}{\partial y} + \frac{\partial\tau_{xy}}{\partial x} + F_y = 0 \tag{72}$$

The forces F_x and F_y are volume forces (also referred to as body forces), which act on the entire volume considered. Examples of volume forces are gravity and inertia forces. The derivation for a three dimensional general stress state is obtained in an equivalent manner by force and moment considerations on a unit cube (this will not be repeated here). These are given by

$$\frac{\partial\sigma_x}{\partial x} + \frac{\partial\tau_{xy}}{\partial y} + \frac{\partial\tau_{xz}}{\partial z} + F_x = 0 \tag{73}$$

$$\frac{\partial\sigma_y}{\partial y} + \frac{\partial\tau_{xy}}{\partial x} + \frac{\partial\tau_{yz}}{\partial z} + F_y = 0 \tag{74}$$

$$\frac{\partial\sigma_z}{\partial z} + \frac{\partial\tau_{xz}}{\partial x} + \frac{\partial\tau_{yz}}{\partial y} + F_z = 0 \tag{75}$$

Additionally, a set of compability equations, for plane conditions 3, is required to ensure that the stresses and strain fields are physically possible. These will not be presented in this context. Instead, it willl be considered how a triangular element for plane stress analysis can be constructed.

3.2 The constant-strain-triangular element (CST)

In figure 17, a triangular element with 3 nodes and 6 DOFs is shown. In the following, the stiffness matrix will be derived applying the theoretical framework based on stationarity of the total elastic potential and interpolation by development of the $[A], [C]$ and $[D]$ matrices.
The element is called a constant strain triangle (CST), since we will see that stresses and strains are constant inside each element due to the choice of shape functions. This constitutes a serious limitation in element performance, since the stress and strain fields are not allow to vary. However, as this is the simplest element for analysis of plane stress, it is highly valuable when aiming to understand how plane elements function. The nodal degrees of freedom and corresponding nodal forces are given by

$$\{d^e\} = \begin{Bmatrix} u_i \\ v_i \\ u_j \\ v_j \\ u_k \\ v_k \end{Bmatrix} \qquad \{F^e\} = \begin{Bmatrix} F_{x,i} \\ F_{y,i} \\ F_{x,j} \\ F_{y,j} \\ F_{x,k} \\ F_{y,k} \end{Bmatrix} \tag{76}$$

The following linear shape functions are chosen

$$u(x,y) = a + bx + cy \tag{77}$$
$$v(x,y) = d + ex + fy \tag{78}$$

On matrix form, the following expression is now obtained

$$\begin{Bmatrix} u_i \\ v_i \\ u_j \\ v_j \\ u_k \\ v_k \end{Bmatrix} = \begin{bmatrix} 1 & x_i & y_i & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x_i & y_i \\ 1 & x_j & y_j & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x_j & y_j \\ 1 & x_k & y_k & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x_k & y_k \end{bmatrix} \begin{Bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{Bmatrix} \tag{79}$$

On compact form, this can be written

$$\{d^e\} = [A]\{a\} \rightarrow \{a\} = [A]^{-1}\{d^e\} \tag{80}$$

We note, that the nodal displacements can be written as

$$\begin{Bmatrix} u_i \\ v_i \end{Bmatrix} = [f(x,y)][A]^{-1}\{d^e\} \tag{81}$$

The general definition of the three strain components are

$$\epsilon_x = \frac{\partial u}{\partial x} \qquad \epsilon_y = \frac{\partial v}{\partial y} \qquad \gamma_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \tag{82}$$

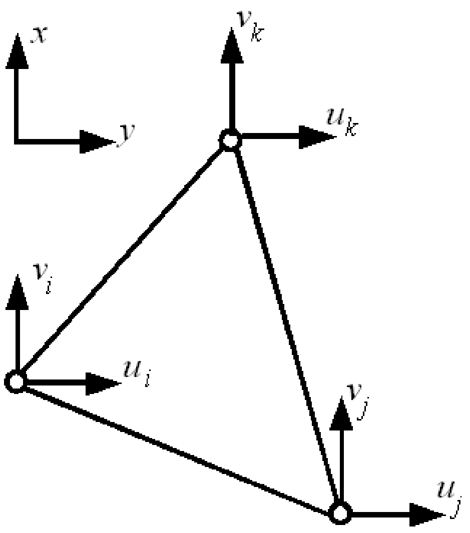

Figure 17: Constant strain triangle (CST)

By differentiation of the shape functions, the following expressions are obtained

$$\epsilon_x = b \qquad \epsilon_x = f \qquad \gamma_{xy} = c + e \tag{83}$$

On matrix form, this corresponds to

$$\{\epsilon\} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 \end{bmatrix} \begin{Bmatrix} a \\ b \\ c \\ d \\ e \\ f \end{Bmatrix} = [C]\{a\} \tag{84}$$

The strain can now be written on the form

$$\{\epsilon\} = [C]\{a\} = [C][A]^{-1}\{d^e\} = [B]\{d^e\} \tag{85}$$

with $[B] = [C][A]^{-1}$.

The constitutive relations given at the beginning of this section can be put on matrix form

$$\left\{ \begin{array}{c} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \end{array} \right\} = \frac{1}{E} \left\{ \begin{array}{ccc} 1 & -\nu & 0 \\ -\nu & 1 & 0 \\ 0 & 0 & 2(1+\nu) \end{array} \right\} \left\{ \begin{array}{c} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{array} \right\} \tag{86}$$

Inverting this matrix equation yields

$$\left\{ \begin{array}{c} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{array} \right\} = \frac{E}{1-\nu^2} \left\{ \begin{array}{ccc} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & \frac{(1+\nu)}{2} \end{array} \right\} \left\{ \begin{array}{c} \epsilon_x \\ \epsilon_y \\ \gamma_{xy} \end{array} \right\} \tag{87}$$

We recognize the constitutive matrix and can now on compact form write

$$\{\sigma\} = [D]\{\epsilon\} = [D][B]\{d^e\} \tag{88}$$

The stiffness matrix can now be determined by the general equation that was derived from the principle of stationarity of the total elastic potential

$$[K] = \int_V [B]^T[D][B]dV = [B]^T[D][B]At \tag{89}$$

The stiffness matrix is not presented in analytical form in this context due to the size of the array.

3.3 Calculated example VII: cantilever beam modeled with CST elements

In this section, a cantilever constituted by a plate will be analyzed, see figure 18. The cantilever will be made of steel with properties as applied in section 2.4. The thickness will be set to $t = 0.0025$, the height to $h = 0.05m$ and the length to $L = 10h$. A numerical implementation of the problem is presented in Appendix B.

An example of the mesh in undeformed and deformed state is shown in figure 19 along with the analytical solution obtained using $u_{max} = \frac{Px^3}{6EI} - \frac{PLx^2}{2}$. The triangular mesh can be observed to produce results that do not correspond with the analytical solution. Figure 20 shows a finer mesh providing better results. It can be observed that a lot of elements are required to obtain an accurate solution with respect to the analytical expression. The FEA approximation divided by the analytical solution are plotted as function of the number of DOFs included in the model. The triangular elements can be observed to constitute a very expensive way of obtaining a result. The triangular mesh is to stiff and does for low numbers of DOFs not bend sufficiently. Furthermore, considering the blue curve obtained with the analytically derived tip deflection of the cantilever, convergence was expected to occur towards a value of 1. However, this can be observed to be overestimated in the analysis. The question is now why the two solutions do not correspond and which of the models delivers the correct result? Considering the

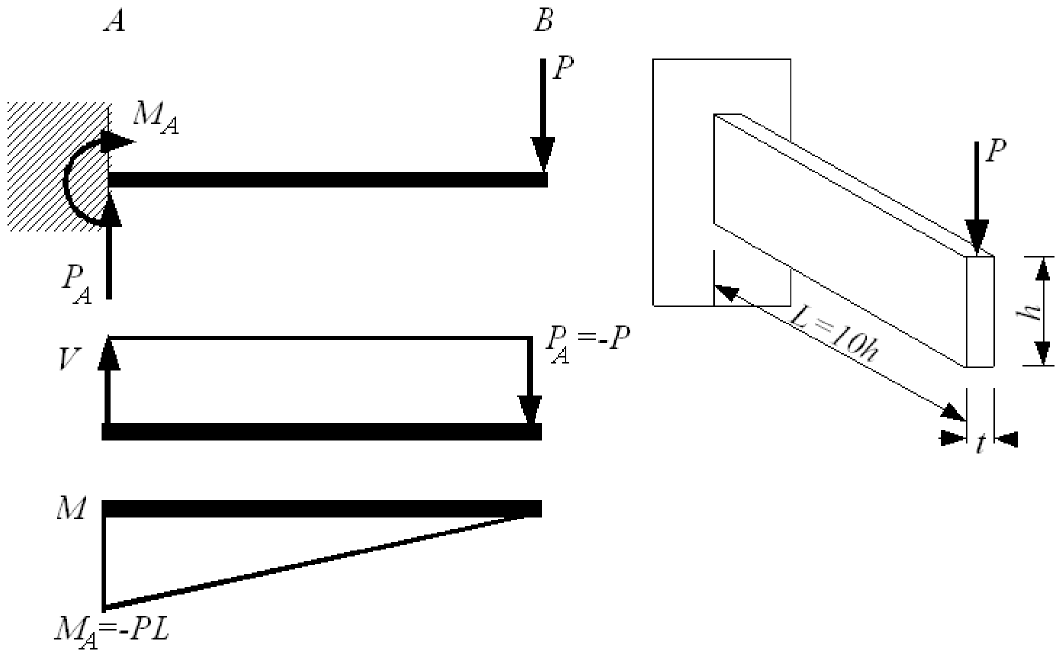

Figure 18: Cantilever plate subjected to end loads

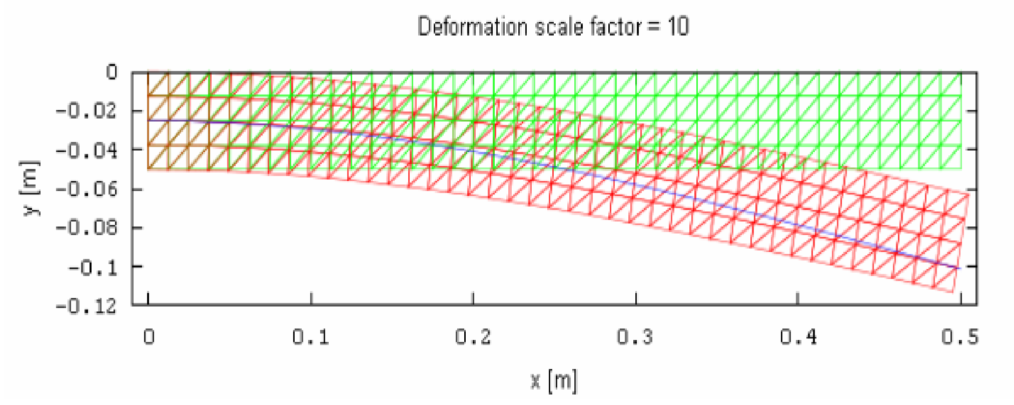

Figure 19: Cantilever plate mesh (green) and scaled nodal deformation solution (red) along with analytical solution (blue)

chosen geometry, we note that $L/h = 10$. This violates the assumption $L/h > 12 - 15$, which serve as basis for the Bernuilli-Euler beam theory. Hence, for the analyzed beam it can no longer be assumed that plane cross-sections remain plane. It follows, that the model of CLT-elements, though expensive from a computational mechanics perspective, provides a better result than the Bernuilli-Euler expression for cantilever beam.

Beams for which $L/h < 12$ can be calculated on basis of the *Timoshenko*-beam theory. This theoretical framework accounts for shear contributions to beam deformations and allow plane cross sections to be distorted due to stress stress. This is illustrated on figure 22 The general theory is complex, but it provides a solution for the specific problem considered, namely a cantilever beam. This is given by

$$v_y = \frac{P}{6EI}\left[3\nu y^2(L-x) + (4+5\nu)\frac{D^2 x}{4} + (3L-x)x^2\right] \tag{90}$$

3.4 Closing remarks regarding choice of elements

For analysis of plane stress, 3-node triangles may provide accurate results with low computational costs in regions where strain gradients are small. However, in general terms, the element performance is poor and for analysis of critical regions, higher order elements should be applied, for example 6-node triangles. The same applies to quadrilateral elements. The 3-node triangle and 4-node quadrilateral elements perform poorly due to the constant stress and strain fields and their lacking ability to represent bending. This is illustrated in figure 23.

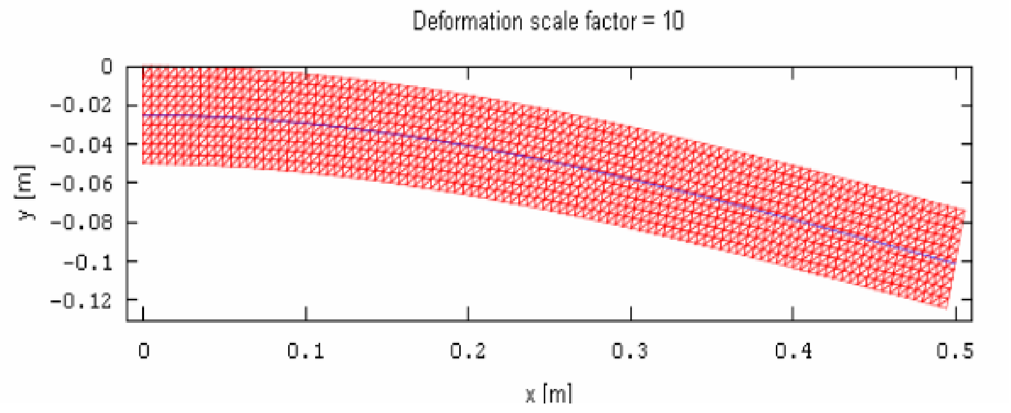

Figure 20: Scaled nodal deformation solution (red) along with analytical solution (blue) obtained with refined mesh

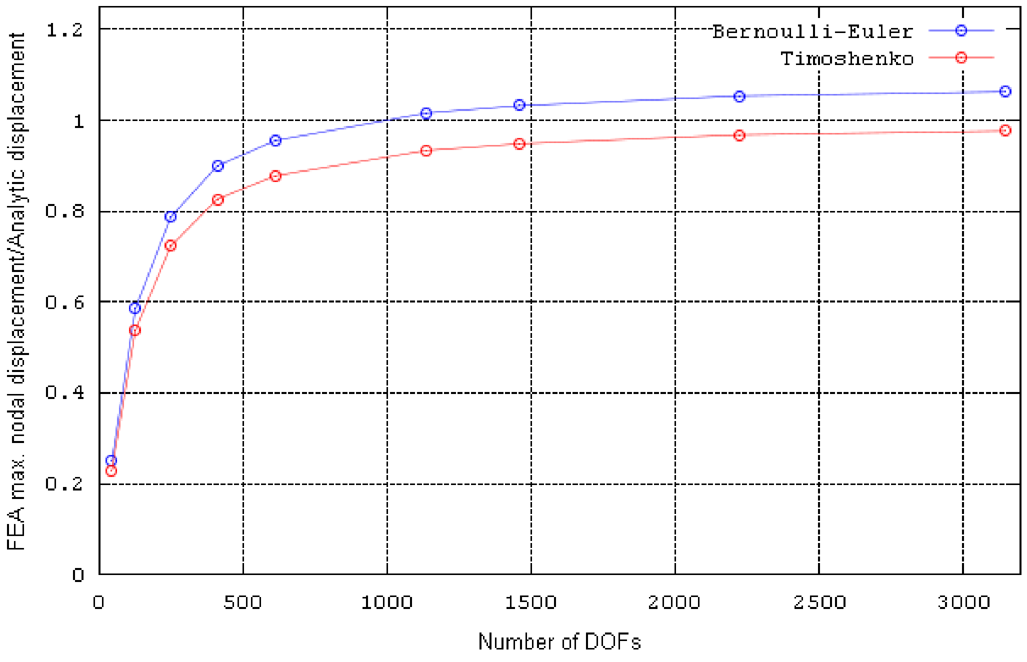

Figure 21: The mesh sensitivity of the maximum (tip) deflection of the cantilever plate

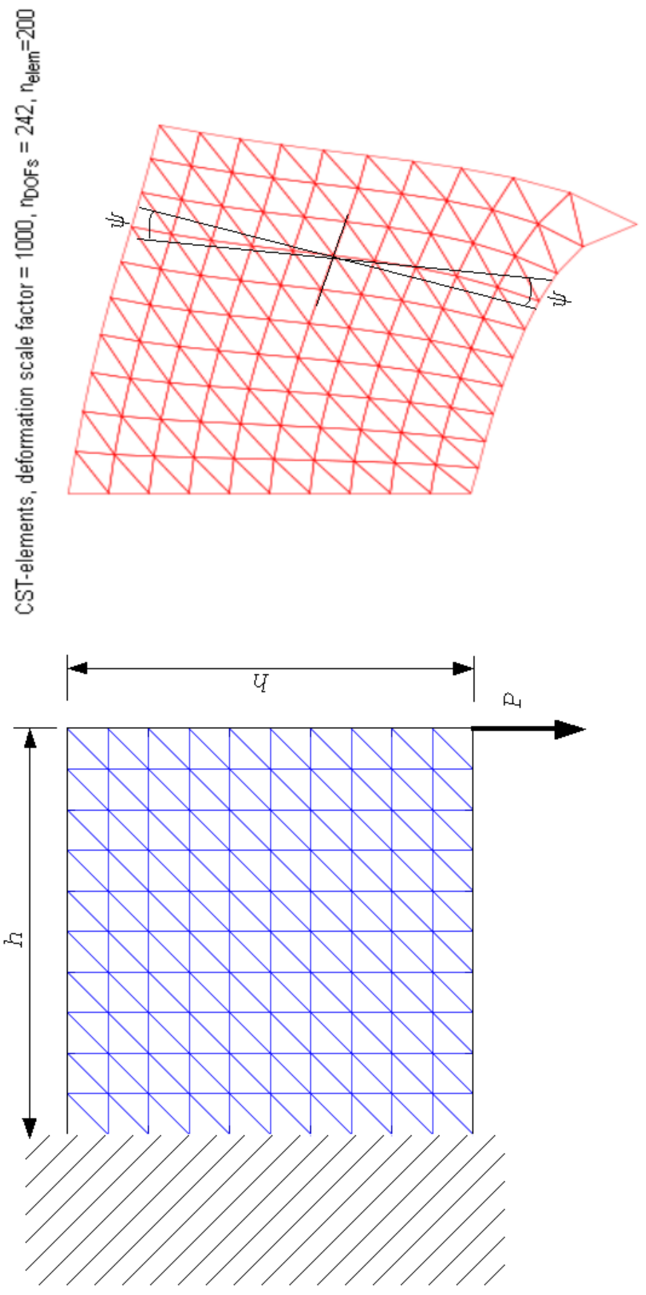
CST-elements, deformation scale factor = 1000, n_DOFs = 242, n_elem=200
ψ
ψ
h
h
P

Figure 22: Illustration of the deflection effect due to non-uniform distribution of shear strains, plane cross-section do not remain plane

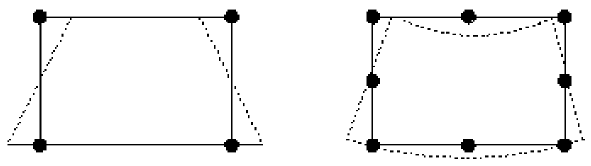

Figure 23: Modes of deformation for a 4 and 8 node quadrilateral elements

4 3D elements

The differential equations governing a general 3D state of stress were contained in section 3.1 and can be derived on basis of the stress component of an infinitisimal cube of material, see figure 24. The derivation of stiffness matrices for 3D elements follow exactly the same principles as applied in these notes for development of $[k]$ for trusses and constant-strain-triangles. However, the expressions are very long and not well-posed for presentation in course material.
In the 3D case, tetrahedrons and brick elements are commonly applied. Tetrahedrons have the advantage, that even complex geomtries can be meshed efficiently, but a very high number of elements are often required in order to obtain convergence. Brick elements provide accurate solutions for low number of elements. However, no generic mesh routine capable of automatically handling all geometries is available for such elements. Therefore, it is more difficult to obtain a 3D brick mesh even using commercial software than it is to generate a mesh of tetrahedrons.
The reader is still not adviced to use basic elements with low numbers of nodes for analysis of sections with high strains gradients. Exactly as in the 2D-case, elements with a high number of nodes capable of representing bending perform a lot more efficiently than simple elements in high-strain regions of an analyzed part.

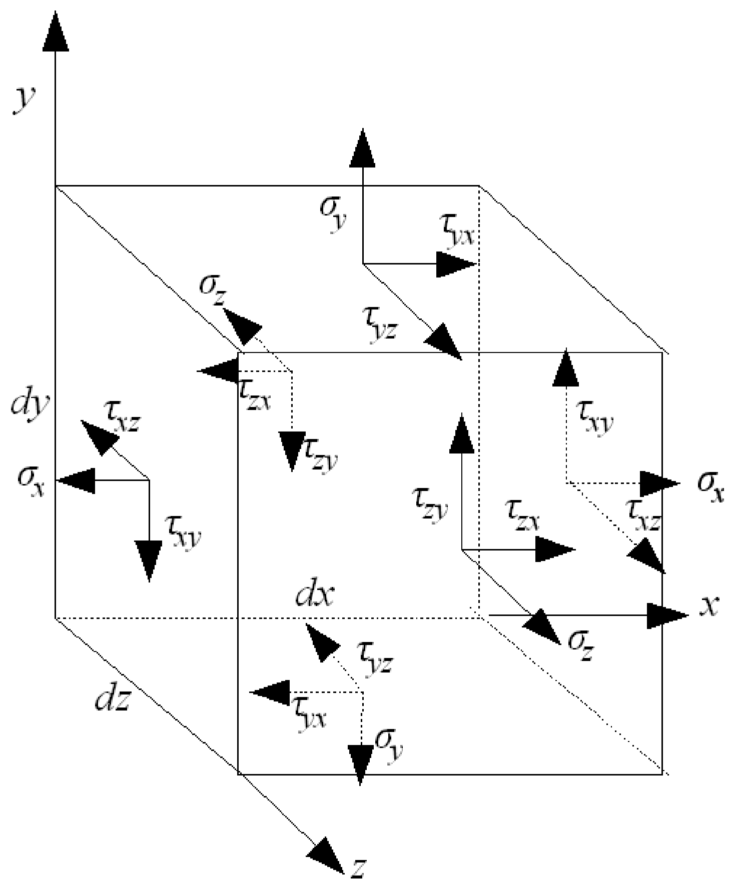

Figure 24: Segment of material of Inifinitisimal proportiones subjected to plane stress

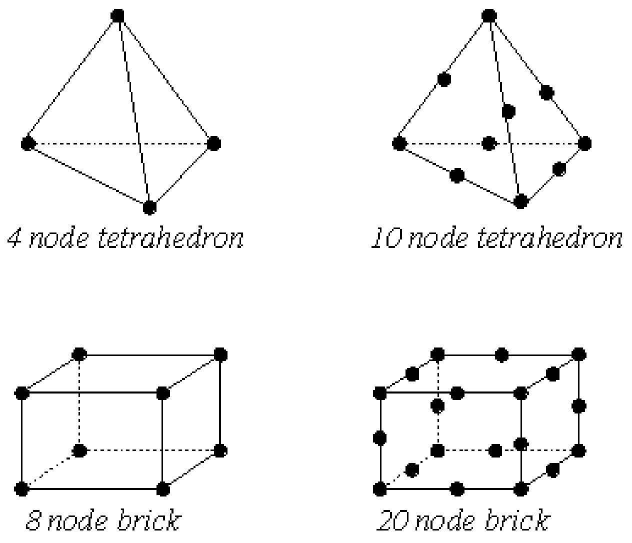

Figure 25: Overview of basic elements for analysis of 3D (general) stress cases

5 Appendix A: node coordinates and element definitions for large frame

```
nodes(1,:)=[0 0];         nodes(2,:)=[0.5 0.5];
nodes(3,:)=[1 0];         nodes(4,:)=[1.5 0.5];
nodes(5,:)=[2 0];         nodes(6,:)=[2.5 0.5];
nodes(7,:)=[3 0];         nodes(8,:)=[3.5 0.5];
nodes(9,:)=[4 0];         nodes(10,:)=[4.5 0.5];
nodes(11,:)=[5 0];        nodes(12,:)=[5.5 0.5];
nodes(13,:)=[6 0];        nodes(14,:)=[6.5 0.5];
nodes(15,:)=[7 0];        nodes(16,:)=[0 0.5];
nodes(17,:)=[7 0.5];
nodes(18,:)=[0 -1];     nodes(19,:)=[1 -1];
nodes(20,:)=[0 -2];     nodes(21,:)=[1 -2];
nodes(22,:)=[0 -3];     nodes(23,:)=[1 -3];
nodes(24,:)=[6 -1];     nodes(25,:)=[7 -1];
nodes(26,:)=[6 -2];     nodes(27,:)=[7 -2];
nodes(28,:)=[6 -3];     nodes(29,:)=[7 -3];

elem(32,:)=[1 18];           elem(33,:)=[3 19];
elem(34,:)=[18 20];          elem(35,:)=[19 21];
elem(36,:)=[20 22];          elem(37,:)=[21 23];
elem(38,:)=[18 19];          elem(39,:)=[20 21];
elem(40,:)=[22 23];          elem(41,:)=[1 19];
elem(42,:)=[18 21];          elem(43,:)=[20 23];
elem(44,:)=[13 24];          elem(45,:)=[15 25];
elem(46,:)=[24 26];          elem(47,:)=[25 27];
elem(48,:)=[26 28];          elem(49,:)=[27 29];
elem(50,:)=[24 25];          elem(51,:)=[26 27];
elem(52,:)=[28 29];          elem(53,:)=[13 25];
elem(54,:)=[24 27];          elem(55,:)=[26 29];
elem(1,:)=[1 2];             elem(2,:)=[1 3];         elem(3,:)=[2 3];
elem(4,:)=[2 4];             elem(5,:)=[3 4];         elem(6,:)=[3 5];
elem(7,:)=[4 5];             elem(8,:)=[4 6];         elem(9,:)=[5 6];
elem(10,:)=[5 7];            elem(11,:)=[6 7];        elem(12,:)=[6 8];
elem(13,:)=[7 8];            elem(14,:)=[7 9];        elem(15,:)=[8 9];
elem(16,:)=[8 10];           elem(17,:)=[9 10];       elem(18,:)=[9 11];
elem(19,:)=[10 11];          elem(20,:)=[10 12];      elem(21,:)=[11 12];
elem(22,:)=[11 13];          elem(23,:)=[12 13];      elem(24,:)=[12 14];
elem(25,:)=[13 14];          elem(26,:)=[14 15];      elem(27,:)=[13 15];
elem(28,:)=[1 16];           elem(29,:)=[2 16];
elem(30,:)=[15 17];          elem(31,:)=[14 17];
```

6 Appendix B: CST element based solution to laterally loaded cantilever beam

```
%FEA example code for analysis of cantilever beam meshed with CST-elements
clc; close all; clear all;

%Input parameters %%%%%%%%%%%%%%%%%%%%%%%%%
E=210*10^9;             %Module of elasticity [N/mm^2]
nu=0.3;                 %Poisson's ratio
P=-1000;                %Load [N]

%Beam dimensions
h=0.05;                   %height
L=10*h;                   %length
t=0.0025;                 %thickness
Area=0.5*(L_elem)^2       %cross-sectional area
L_elem=h/10;              %element length

div_L=L/L_elem; div_h=h/L_elem;
n_elem=div_L*div_h;

BCStiffness=10^50;      %Pseudo Stiffness for boundary conditions

%Generated node coordinate matrix
for i=1:div_L+1;
    for j=1:div_h+1;
      nodenum=(i-1)*(div_h+1)+j;
      nodes(nodenum,:)=[(i-1)*L_elem -(j-1)*L_elem];
    end
end
n_nodes=(div_L+1)*(div_h+1);

%Generate element matrix
for i=1:div_L;
  for j=1:div_h;
     elemnum=(i-1)*div_h+j;
     nodenum1=(i-1)*(div_h+1)+j;
     nodenum2=(i-1)*(div_h+1)+j+1;
     elem(2*elemnum-1,:)=[nodenum1 nodenum2 nodenum1+(div_h+1)];
     elem(2*elemnum,:)=[nodenum2 nodenum1+(div_h+1) nodenum2+(div_h+1)];
  end
end
```

```
n_elem=2*div_L*div_h;

%Calculate C-matrix
C(3,6)=0;
C(1,2)=1; C(2,6)=1; C(3,3)=1; C(3,5)=1;

%Calculate D-matrix
D(3,3)=0;
D(1,1)=1;        D(1,2)=nu;
D(2,1)=nu;       D(2,2)=1;
D(3,3)=0.5*(1-nu);
D=E/(1-nu^2)*D;

%Initialize global stiffness matrix
K=spalloc(2*n_nodes,2*n_nodes);

%Assemble global stiffness matrix
for m=1:n_elem;
    i=elem(m,1);        j=elem(m,2);           k=elem(m,3);
    xi=nodes(elem(m,1),1); yi=nodes(elem(m,1),2);
    xj=nodes(elem(m,2),1); yj=nodes(elem(m,2),2);
    xk=nodes(elem(m,3),1); yk=nodes(elem(m,3),2);

    A(1:6,1:6)=0;
    A(1,1)=1;   A(1,2)=xi;      A(1,3)=yi;
    A(2,4)=1;   A(2,5)=xi;      A(2,6)=yi;
    A(3,1)=1;   A(3,2)=xj;      A(3,3)=yj;
    A(4,4)=1;   A(4,5)=xj;      A(4,6)=yj;
    A(5,1)=1;   A(5,2)=xk;      A(5,3)=yk;
    A(6,4)=1;   A(6,5)=xk;      A(6,6)=yk;

    B=C*inv(A);

    k_elem=transpose(B)*D*B*Area*t;

    K(2*i-1:2*i,2*i-1:2*i)=K(2*i-1:2*i,2*i-1:2*i)+k_elem(1:2,1:2);
    K(2*i-1:2*i,2*j-1:2*j)=K(2*i-1:2*i,2*j-1:2*j)+k_elem(1:2,3:4);
    K(2*i-1:2*i,2*k-1:2*k)=K(2*i-1:2*i,2*k-1:2*k)+k_elem(1:2,5:6);

    K(2*j-1:2*j,2*i-1:2*i)=K(2*j-1:2*j,2*i-1:2*i)+k_elem(3:4,1:2);
    K(2*j-1:2*j,2*j-1:2*j)=K(2*j-1:2*j,2*j-1:2*j)+k_elem(3:4,3:4);
    K(2*j-1:2*j,2*k-1:2*k)=K(2*j-1:2*j,2*k-1:2*k)+k_elem(3:4,5:6);

    K(2*k-1:2*k,2*i-1:2*i)=K(2*k-1:2*k,2*i-1:2*i)+k_elem(5:6,1:2);
    K(2*k-1:2*k,2*j-1:2*j)=K(2*k-1:2*k,2*j-1:2*j)+k_elem(5:6,3:4);
    K(2*k-1:2*k,2*k-1:2*k)=K(2*k-1:2*k,2*k-1:2*k)+k_elem(5:6,5:6);
end

%Generate nodal load vector
F(2*n_nodes)=0;
F(2*n_nodes)=P;

%Set boundary conditions
for m=1:2*(div_h+1);
K(m,m)=BCStiffness;
end

%Solve problem
d=inv(K)*F';

%Postprocessing, elementwise stress calculations
for m=1:n_elem;
    i=elem(m,1);        j=elem(m,2);           k=elem(m,3);
    xi=nodes(elem(m,1),1); yi=nodes(elem(m,1),2);
    xj=nodes(elem(m,2),1); yj=nodes(elem(m,2),2);
    xk=nodes(elem(m,3),1); yk=nodes(elem(m,3),2);

    A(1:6,1:6)=0;
    A(1,1)=1;   A(1,2)=xi;      A(1,3)=yi;
    A(2,4)=1;   A(2,5)=xi;      A(2,6)=yi;
    A(3,1)=1;   A(3,2)=xj;      A(3,3)=yj;
    A(4,4)=1;   A(4,5)=xj;      A(4,6)=yj;
    A(5,1)=1;   A(5,2)=xk;      A(5,3)=yk;
    A(6,4)=1;   A(6,5)=xk;      A(6,6)=yk;

    H=D*C*inv(A);

    dE=[d(2*elem(m,1)-1) d(2*elem(m,1)) d(2*elem(m,2)-1)  //
   d(2*elem(m,2)) d(2*elem(m,3)-1) d(2*elem(m,3))]';

    Sigma(m,:)=H*dE;
  end
```