Petr Krysl

Finite Element Modeling of Structures with Shells and Beams

Pressure Cooker Press
San Diego

Contents

1

Introduction

1.1 Organization of the book

We use visual means for highlighting certain statements. Important bits of information are boxed:

This is how important information will be set off. For instance, here is an important equation

$$\exp \mathrm{i}\phi = \cos\phi + \mathrm{i}\sin\phi$$

At times there will be a warning:

This is a common mistake!

The electronic book has built-in links to additional information and resources. This is likely to prove useful when searching for text or code. The external links lead to a website storing the PDF tutorials, Python scripts, Abaqus model files, and other resources. Use a capable PDF viewer that will allow you to control how to open these files. Some browsers will attempt to display the Abaqus model files, with useless results. In that case attempt to make them save the file to the hard drive, to open them later with Abaqus.

The material in the regular sections presents the basic information. At the end of each chapter, there may be a special section with detailed background information, additional explanations, tables, and such. This is not required reading at the basic level: if you need it or if you are plain interested, go for it; otherwise you can leave it till it becomes useful.

1.2 Software

The software used in this textbook consists of

- Abaqus/CAE, a finite element program with a graphical user interface;
- the open-source language Python (used for scripting Abaqus).

The PDF version of the textbook includes links to outside sources of information: model files, tutorials, Python source files, ... **Note: until you reach the box that says otherwise, the links in the margin are only examples of what the linking to external resources looks like later in the book: these few links don't work!**

Abaqus
- CAE file

Abaqus examples are typically accompanied by a model database. The link to such a database (extension ".cae") is provided in the margin. The web browser should open such data bases with Abaqus.

Abaqus
- CAE file
- tutorial

Sometimes the Abaqus example comes with a tutorial. The tutorial is a PDF file which shows the process step-by-step. The web browser would typically open the PDF file when you click the link.

Abaqus
- CAE file
- INP file
- tutorial

In a few instances the Abaqus model is derived from an input file (extension ".inp"). Then one of the links will point to an INP file. This is a plain-text file which can be edited with any text editor (`notepad++` is a free offering on Windows and Linux). If the file is opened in a web browser instead of saved to your local disk storage (which depends on your particular computing platform), save the file with the ".inp" extension. As an example, in the Chrome browser, right-click on the text displayed in the browser and select "`Save as...`". In the selection box "`Save as type`" choose "`All Files`", and make sure to delete the "`.txt`" extension, if there is one attached to the name. Then in the folder where the file was saved you have the "`.inp`" file which can be now edited with any text editor. The tutorials will explain how to use the INP file.

Python
- script

Some finite element calculations are carried out with Python. Links to the Python script files (extension `.py`) are available in the margin. Save the file and then load it in development environment of choice (see Section 1.2).

Animation
- video

Some points are illustrated with animations. Links to video files are available in the margin, and are typically opened in a web browser. If that is not the case, locate the file in your Downloads folder and drag it to a browser window. (There are also specialized applications that can display video files.)

Links to outside resources below this box should all work. If you find a problem with the downloads, please let the author know (`pkrysl@ucsd.edu`).

1.2.1 Abaqus software

Abaqus is a suite of commercial finite element codes. It consists of Abaqus Standard, which is a general purpose finite element software, and Abaqus Explicit for dynamic analysis. It is now owned by Dassault Systèmes and is part of the SIMULIA range of products, see

`www.simulia.com/products/unified_fea.html`

In this book we will interact with the finite element analysis in Abaqus Standard through the Abaqus/CAE, a graphical user interface. Abaqus® is a registered trade mark of Dassault Systèmes. For product information, please refer to the website `http://www.3ds.com`. This book has been tested with the Abaqus/CAE Teaching Edition and the Student Edition, as they were current in 2023.

This course is not about Abaqus. The goal is not to teach the reader the operation of the software. The particular FE program used is not important, the focus is on the theory behind the software and on the modeling practices that make a good FE analyst.

Abaqus is tightly coupled to **Python**, a very popular and widely used scripting language. The Abaqus software can be driven by scripts, which makes it a very flexible and powerful tool. In addition, many exercises in this book will be supported by numerical or symbolic computations with Python in the form of script files.

1.2.2 Cloud computing

The easiest way of running most Python examples from the textbook (except those that require Abaqus/CAE) may well be provided by the cloud computing environment `jupyter.org/try`. Select Jupyter notebook, and then paste blocks of code or in fact the entire code of an example into the first cell of the notebook and select for instance "`Run all`" or "`Run Cells`". This will create an output cell, with printouts and graphics.

The programming environment at `repl.it` is also very easy to use. Simply copy the text of the Python program into the `main.py` window and click on Run. Registration is free (as of 2023).

1.2.3 Local Python environment

The examples in this book were written with modules from the so-called SciPy stack (SciPy = Scientific Python). (For those interested in having a local Python environment and who do not mind tinkering, a good option is to visit `www.scipy.org/install.html`.) The recommended option is the Spyder IDE, downloadable for free from `www.spyder-ide.org/download`.

1.2.4 Running Python in Abaqus

The examples in this book are worked with the Abaqus/CAE program. This program can be driven by a Python script. When the user operates the Graphical User Interface (GUI), for instance to partition a face, Abaqus/CAE writes the commands to accomplish this to a few files: the main ones being the replay file, `abaqus.rpy`, the journal file (with extension `.jnl`), and the recovery file (with extension `.rec`). These are all Python scripts that can be executed from the "File" menu.

Abaqus/CAE Student Edition does not write the replay file and the journal file. Also, the recovery file is erased once the user quits the GUI.

These Python scripts can be edited by the user to accomplish slightly modified tasks, for instance the dimensions of the part may be changed, or the material properties can be defined to work with user inputs, etc.

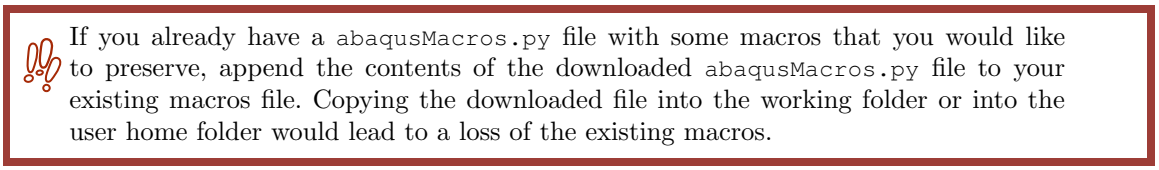
The Abaqus Scripting Interface is an application programming interface (API) to the modeling and model data. The scripting interface is an extension of the Python object-oriented programming language: the interface scripts *are* Python scripts. One can (a) create and modify the components of an Abaqus model (including, but not limited to, parts, materials, loads, and steps); (b) manage analysis jobs; (c) manage output databases; (d) postprocess the results of an analysis.

As an example, a few useful material definitions (steel, aluminum, some composite materials, ...) can be created by running the `useful_materials.py` script. If the script is opened in the web browser, right-click the page and choose "Save as". Then run from the "File" menu of the Abaqus/CAE program using "Run script...".

Python
- script

For an easy scripting of Abaqus/CAE check out the macros: Invoke the "`Macro Manager`" from the "`File`" menu, and start recording the macro. Then execute a few Abaqus commands, such

as create a sketch and extrude it into a part. Then stop the recording of the macro, and voilà: the file called `abaqusMacros.py` appears in the working folder (typically `c:\temp`). The macro is named, and its name should now appear on the list of the "`Macro Manager`". Should it not appear automatically, feel free to click "`Reload`". The contents of the `abaqusMacros.py` script file may now be inspected with an editor of your choice. It consists essentially of definitions of Python functions. Each function is an Abaqus macro.


If you already have a `abaqusMacros.py` file with some macros that you would like to preserve, append the contents of the downloaded `abaqusMacros.py` file to your existing macros file. Copying the downloaded file into the working folder or into the user home folder would lead to a loss of the existing macros.

Python - script

As an example, here's a link to a sample macros file. When this file is copied to the working folder, and the "`Macro Manager`" is opened, two macros should be listed:

- `MakeCylinderPart` to create a cylindrical solid part; and
- `MakeUsefulMaterials` to create several material models.

1.2.5 Standing assumptions for Python code

Working with arrays and linear algebra in Python is best done with the array module `numpy`. The functions `array`, `dot`, and so on are part of this module, as is the submodule `linalg` for linear-algebra operations. When we present Python code we usually don't state that explicitly, but these objects need to be imported, for instance as

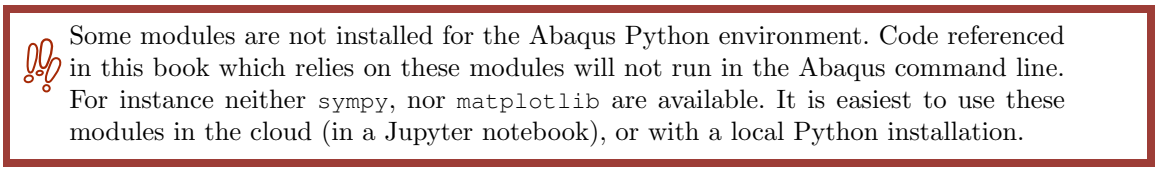
```
from numpy import array, dot
from numpy import linalg
```

Similarly, when we work with the symbolic-math module `sympy`, we import the objects we need from this module

```
from sympy import symbols, simplify, Matrix, diff
```

or, alternatively, we can import the module and then refer to the objects as for instance

```
import sympy
A, B = sympy.symbols('A, B')
```


Some modules are not installed for the Abaqus Python environment. Code referenced in this book which relies on these modules will not run in the Abaqus command line. For instance neither `sympy`, nor `matplotlib` are available. It is easiest to use these modules in the cloud (in a Jupyter notebook), or with a local Python installation.

1.2.6 Python package `pystran`

- URL

The Python package `pystran` implements truss and beam models in two and three dimensions. Instructions for its use may be found at the provided address.

1.3 Units

Abaqus puts the onus of inputting the data in consistent units on the user. It is not alone in this, as typical finite element programs do not allow for explicit provision of physical units. This means that input data is only typed in as numbers, without any indication of the physical units. The program will assume that the numbers that were input by the user were all in consistent units.

In thermomechanical problems one must choose four units, for example for length, mass, time, and temperature. For instance, in SI units for these quantities, we use m for length, kg for mass, s for time and ^{o}K for temperature; then the units for forces will be in N, the stresses and the pressure will be in Pa, the mass density will be in $\mathrm{kg} \cdot \mathrm{m}^{-3}$, the thermal conductivity in $\mathrm{W} \cdot \mathrm{m}^{-1} \cdot {}^o\mathrm{K}^{-1}$, and the specific heat in $\mathrm{J} \cdot \mathrm{kg}^{-1} \cdot {}^o\mathrm{K}^{-1}$.

One should usually choose physical units that make the numerical values of the quantities with which the program calculates (typically the stiffness coefficients, the forces, the displacements) not too small and not too large. In analyses of systems of small spatial extent, a possible choice of units is mm for length, N for force, s for time and ^{o}K for temperature. Then mass density would be supplied in the surprising units of metric tons (tonnes) per millimeter cubed, so that the mass density of steel (let us say 7850 $\mathrm{kg.m}^{-3}$) would need to be input as $7.85 \times 10^{-9} \mathrm{tonne} \cdot \mathrm{mm}^{-3}$ (so actually type in `7.85e-9` for the material Density property). However, elastic moduli, stresses, and pressure would be in the convenient units of MPa. Table 1.1 lists four sets of consistent units.

Many unit conversion utilities are available on the web, for instance at `translatorscafe.com`.

- URL

Quantity	SI	SI (mm)	US Unit (ft)	US Unit (inch)
Length	m	mm	ft	in
Force	N	N	lbf	lbf
Mass	kg	tonne (10^3 kg)	slug	lbf s^2/in
Time	s	s	s	s
Stress	Pa ($\mathrm{N/m}^2$)	MPa ($\mathrm{N/mm}^2$)	$\mathrm{lbf/ft}^2$	psi ($\mathrm{lbf/in}^2$)
Energy	J (N $\times$ m)	mJ (10^{-3} J)	ft lbf	in lbf
Density	$\mathrm{kg/m}^3$	$\mathrm{tonne/mm}^3$	$\mathrm{slug/ft}^3$	lbf $\mathrm{s}^2/\mathrm{in}^4$

Table 1.1. Consistent sets of units (after the Abaqus Analysis User's Guide)

1.4 Abbreviations

Abbreviation	Meaning
B33	Abaqus two-node Bernoulli beam finite element
B31	Abaqus two-node Timoshenko beam finite element
CAE	Complete Abaqus Environment
CTE	Coefficient of Thermal Expansion
DOF	degree of freedom
FE	Finite Element
FEA	Finite Element Analysis
FEM	Finite Element Method
H8	Finite element with eight nodes (brick, solid)
H20	Finite element with 20 nodes (brick, solid)
L2	Finite element with two nodes (curve)
L3	Finite element with three nodes (curve)
Q4	Finite element with four nodes (quadrilateral, surface)
Q8	Finite element with eight nodes (quadrilateral, surface)
S3	Abaqus three-node shell finite element
S4	Abaqus four-node shell finite element
STRI3	Abaqus three-node shell finite element
T3	Finite element with three nodes (triangle, surface)
T6	Finite element with six nodes (triangle, surface)
T10	Finite element with 10 nodes (tetrahedron, solid)
WR	Weighted Residuals
MWR	Method of Weighted Residuals
VW	Virtual Work

Table 1.2. Abbreviations

2

Analysis of structures

This book deals with finite element models for structures that can be modelled with beams and shells. Here we will limit ourselves to an intuitive understanding of what a beam or a shell might be. We shall address the criteria for a simple classification below. This book is quite different from the finite element book [PK1] in that the models discussed here are the so called structural finite elements. The book [PK1] dealt with continuum finite element models. The continuum models tend to be conceptually much more simple than structural finite elements.

Using less material more effectively is constantly on the designer's mind. Figure 2.1 is an early example of an innovative design that used a novel material (wrought iron), relying on a novel concept of a built-up tube. Today we could analyze this structure as a beam, a collection of shells, or some combination thereof. In the past, many approximate models have been developed to deal

Fig. 2.1. The innovative Britannia bridge over the Menai Strait by Stephenson. The original structure stood in this form until 1970, when the bridge was damaged in fire and replaced. The locomotives ran through metal tubes of multi-cellular cross section.

with structures of this complexity. Today that task would fall to a finite element analysis.

The variety of shell structures is huge. For a small sample, refer to Figure 2.2. A common element of the designer's vocabulary is curvature: an eggshell is an excellent example of using little material to achieve great stiffness. Another part of the vocabulary are creases, in the form of plates or shells meeting at an angle (folded-plate roofs are a good example), or in the form of stiffeners added to very thin shells (such as found in solid-fuel rocket boosters).

Many methods (dozens) exist for analyzing specific shell structures. They are typically very narrow in being applicable to one particular type of structure, and not to any other. Here, we are looking for an approach that would be powerful and generic.

When designing structures to be light and economical, it is central to take into consideration the importance of shape. Plastic water bottles can be very light because their surface is bent into a variety of ridges and gullies (a technical term for these is "beads"). These features make it possible

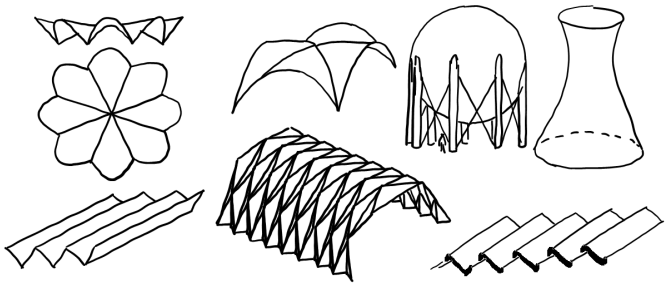


Fig. 2.2. Shells come in great variety of structural forms. The analyst would want to use an analysis method that can deal with any and all of them.

for the material of the bottle to activate stress throughout the thickness of the wall of the bottle in the form of the so called membrane forces (as opposed to mobilizing bending stiffness; refer to Figure 2.3).

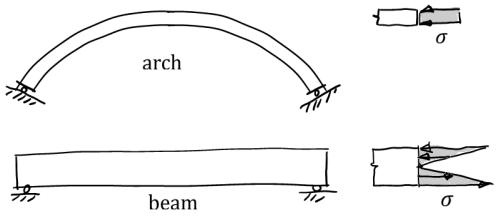


Fig. 2.3. Arch versus a straight beam.

A large field of study has developed around the optimization of shells and other such types of structures (Figure 2.4). In the latter part of the book we shall consider various forms of optimization: sizing and shape optimization of truss structures, topology optimization (generative design), and optimization of shape to reduce stress concentrations.

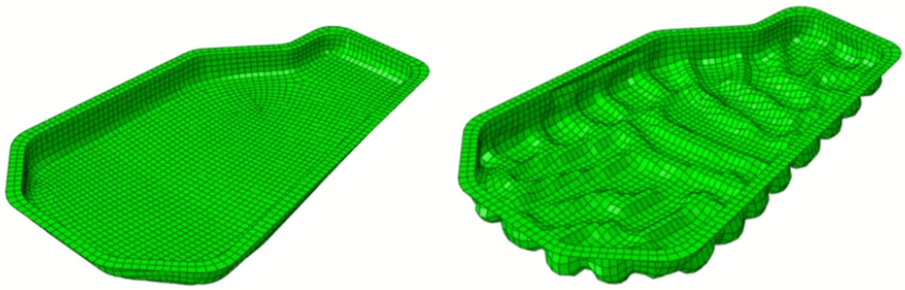


Fig. 2.4. Shell structure can be made much stiffer with not much more material when "beads" are added to introduce curvature in strategically selected locations. Shape can be everything (well, almost everything).

While designers put a lot of effort into using material efficiently, nature conspires against them by introducing localized effects that create bending moments: supports along edges of shells that require for equilibrium shear forces and bending moments to develop; localized loading of shell surfaces; and joints of two or more shell surfaces along folds (Figure 2.5). Since bending moments typically reduce the load-carrying capacity of structures, it is quite important to understand where these effects might occur, and to focus the modelling effort on capturing this behavior.

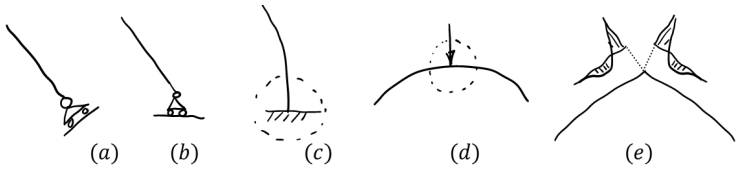


Fig. 2.5. It is easy to design a structure which shows localized bending, and consequently less than optimal use of material.

2.1 Classification of structures

For the purpose of selecting a model, structures may be roughly classified into four categories (Figure 2.6):

(a) In the first category are solids: usually bulky, analyzed in three dimensions, with all three dimensions roughly the same. Structures like this are analyzed with continuum finite element models. We can just as well include models reduced to two dimensions, but still continuum: axially symmetric, plane strain and plane stress [PK1].
(b) In the second category are shells: one dimension significantly smaller than the other two. The model can be expressed in two coordinates.
(c) In the third category are beams, structures whose two dimensions are significantly smaller than the third. They can be analyzed with one-dimensional beam models.
(d) In the fourth category are thin-walled beam-like structures. All three dimensions are really quite different: there is the length, the dimension of the cross section, and the thickness of the wall. These structures are somewhere between the second category, shells, and the third category, the common beams. To address the response of structures in this category, the common beam models must be enhanced to incorporate the various response mechanisms associated with this type of structure (shear lag, warping torsion, crippling, wall buckling, etc.).

The complexity of models progressively increases from category (a) to category (d). The book on the finite element analysis of continuum models [PK1] focused on the category (a). In the present book, we address the remaining three categories of structures.

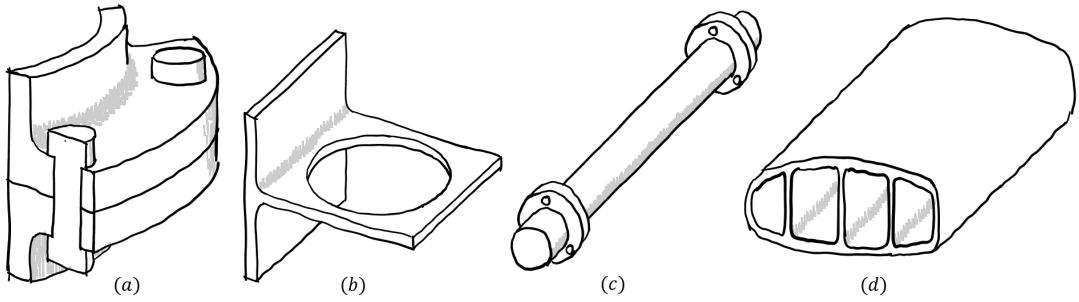


Fig. 2.6. Tentative classification of structures. (a) Solid structure. Analyzed with 3D continuum models. (b) Shell structure. Analyzed with 2D shell models. (c) Beam structure. Analyzed with common 1D models. (d) Thin-walled beam structure. Analyzed with enhanced 1D models.

2.2 Characteristics of models of structures

We are used to manipulating shapes of flexible artifacts (for instance, an ordinary ruler) by applying forces and torques. Figure 2.7 shows an example: the blade of a metal saw is bent into various shapes

by clamping the left hand side to a table, and then by applying forces and torques (i.e. force couples) at the right-hand-side end. In this way we obtain approximations of two of the basis functions that are used for a two-node finite element to represent a beam.

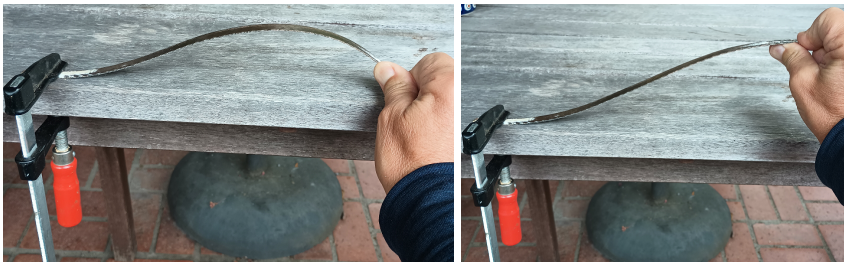


Fig. 2.7. Flexible piece of metal (metal saw blade) forced into two different shapes by applied forces and torques.

Consequently, we will not be surprised to discover that in the structural finite element method the shapes of beams and shells will be described with translations (which are work-conjugate to forces) and rotations (which are work-conjugate to torques). By work-conjugate we simply mean that the force-like quantity performs work on the displacement-like quantity. Figure 2.8 illustrates this for a beam model and for a shell model.

The beam model is represented by a curve (either the centroid axis, or the elastic axis, or an arbitrary reference curve; the word "midline" is often a convenient, if imprecise, stand-in); the beam model becomes a three dimensional body by attaching a cross section to each point of the midline.

The shell model is represented with a surface that is endowed with a thickness. Both models are described by translations and rotations attached to the nodes (in this figure, the beam has two nodes, and the shell has three nodes).

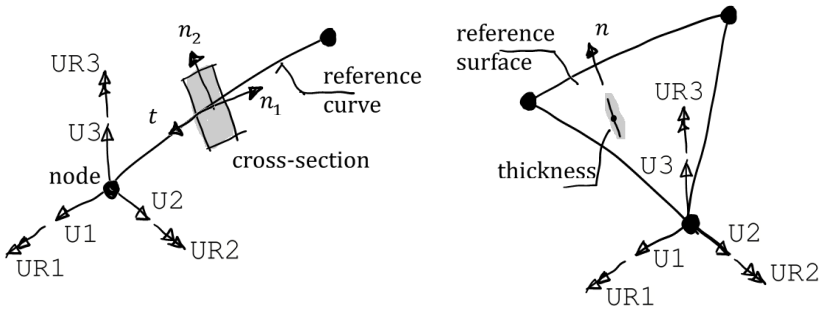


Fig. 2.8. Beam and shell reduced models. Each node has in general six degrees of freedom, three translations (`U1, U2, U3`) and three rotations (`UR1, UR2, UR3`).

In addition to reduced models formulated in a three dimensional space, it is also possible formulate reduced models in a two dimensional space (such as planar frames). Refer to Figure 2.9.

In both figures, Figure 2.8 and Figure 2.9, `U1, U2, U3` refer to displacements along an arbitrary cartesian coordinate system, while `UR1, UR2, UR3` refer to rotations about the axes of that coordinate system.

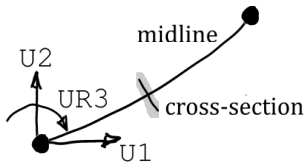


Fig. 2.9. Beam reduced model in the plane. Each node has in general three degrees of freedom, two translations (`U1, U2`) and a rotation (`UR3`).

Consider modelling a single I-beam member with a three dimensional solid finite element model. Let us say 5 m long, the web 0.3 m deep, and the flange 0.2 m wide. Also, for simplicity we shall assume that the web and the flanges are one centimeter thick. So in the cross section we would expect as a minimum 70 brick elements, and overall in the entire beam around 17,500 elements. A shell model of such a beam of comparable accuracy may have 900 elements. However, it is also sometimes plausible that we might model such a beam with a single finite beam element, or perhaps half a dozen. Clearly, going from three to two to one dimension has the potential of reducing the size of the model considerably. However, the question of accuracy is in reality quite complex: theoretically, a three dimensional model should be expected to be more accurate than a one-dimensional one, because it can capture more complexity; in practice we need to employ finite elements, and poorly shaped coarse three dimensional elements may be rightfully suspected of losing a lot of accuracy. On the other hand, a one-dimensional finite element model will not capture the complexity of the deformations faithfully due to the assumptions that go into the formulation of such a model, for instance the assumption that the cross section does not change shape.

3

Shell models of structures

Summary

1. The mathematical models of three-dimensional structural members can be reduced to two independent coordinates for plates and shells.
2. Models are developed for flat membranes and slabs, and boundary conditions are discussed.
3. Models for general shells are only sketched, due to the required mathematical sophistication.
4. Concentrated forces and reactions are discussed.

We assume that readers have at least a fleeting familiarity with the Bernoulli beam model. In what follows, it will prove helpful to mention the parallels of the balance equations for the Bernoulli beam, $M' - V = 0$, and $V' + q = 0$, and the balance equations derived for the shells. Also, when discussing the bending of thin plates, the equation for the deflection of the plate will by readily identifiable as an analogue of the balance equation $EIw'''' - q = 0$ for the Bernoulli beam. Here M, V, q, w refer to the bending moment, shear force, transverse distributed loading, and the deflection of the beam midline respectively.

3.1 Basic equations of shell models

Figure 3.1 shows the fundamental notions concerning the geometry of a shell. The shell is defined by its bottom and top surface, which are separated at precisely the midpoint by the midsurface. The discretization of the shell is associated with the so-called reference surface, sometimes offset from the midsurface, at other times the reference surface and the midsurface may coincide.

Motion.

At any point of the shell reference surface, $\mathbf{p}_0$, we can construct a local Cartesian coordinate system such that the x and y axes are tangent to the surface, and z coordinate is read off from the point $\mathbf{p}_0$ in the direction of the normal $\mathbf{n}$. The points of the shell along the z coordinate form the so-called lineal element. We can think of it as a thin stalk attached to the reference surface at point $\mathbf{p}_0$. The bulk of the shell material can be thought of as formed by an infinitely dense field of such stalks attached to all points of the reference surface.

An arbitrary point of the shell is identified as $\mathbf{p} = \mathbf{p}_0 + z\mathbf{n}$. In other words, we can follow the directions "go to the reference surface, then move by the signed distance z along the normal $\mathbf{n}$". In what follows, we will only consider the so-called geometrically linear models, which means that one only needs to take into account the geometry before deformation: the deformations are so small that they do not affect the form of the equations.

The displacements in the Cartesian coordinate system attached at point $\mathbf{p}_0$ are defined as

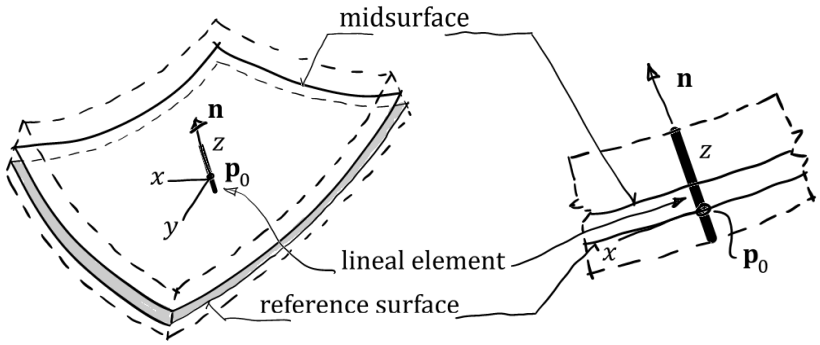


Fig. 3.1. Shell model: geometry. The lineal element is orthogonal to the midsurface of the shell before deformation. The local Cartesian coordinate system is set up so that the z direction is along the normal to the surface.

$$
\begin{aligned}
u(\mathbf{p}_0 + z\mathbf{n}) &= u_0(\mathbf{p}_0) + z\phi_y(\mathbf{p}_0) \\
v(\mathbf{p}_0 + z\mathbf{n}) &= v_0(\mathbf{p}_0) - z\phi_x(\mathbf{p}_0) \\
w(\mathbf{p}_0 + z\mathbf{n}) &= w_0(\mathbf{p}_0)
\end{aligned} \tag{3.1}
$$

The three displacements $u_0(\mathbf{p}_0)$, $v_0(\mathbf{p}_0)$, and $w_0(\mathbf{p}_0)$ describe the motion of the point on the reference surface. In addition, there are two contributions to the displacements due to rotations of the lineal elements, $+z\phi_y(\mathbf{p}_0)$ and $-z\phi_x(\mathbf{p}_0)$. Here ϕ_y and ϕ_x are the rotations of the lineal element about the axes y and x respectively (ϕ_x is a rotation about the x axis, ϕ_y is a rotation about y). The two angles are positive as defined by the right hand side rule (thumb points along the axis, fingers show the rotation direction).

We only consider so-called first-order theories of shells, which assume that the lineal element normal to the mid-surface before deformation moves as a "rigid body" during deformation – it remains straight.

Stress.

Next we shall consider stresses in the local Cartesian coordinate system defined in Figure 3.1. Abaqus refers to the two directions in the tangent plane as 1 and 2. That is also reflected in the names used for the stress components: S11 (direct stress on a cut with the normal in the direction of axis 1) and so on. We can use the stress components to define forces and moments. Figure 3.2 shows the five resultant forces, three in the tangent plane (referred to as membrane forces), and two out of the plane (referred to as transverse shear forces). The formulas for the membrane forces, direct SF1, SF2, and shear SF3, and for the two transverse shear forces SF4 and SF5 read

$$
\{\mathrm{SF1}, \mathrm{SF2}, \mathrm{SF3}, \mathrm{SF4}, \mathrm{SF5}\} = \int_{-h/2-z_0}^{h/2-z_0} \{\sigma_{11}, \sigma_{22}, \sigma_{12}, \sigma_{13}, \sigma_{23}\}\ \mathrm{d}z \tag{3.2}
$$

where by $\{\mathrm{SF1}, \mathrm{SF2}, \mathrm{SF3}, \mathrm{SF4}, \mathrm{SF5}\}$ and $\{\sigma_{11}, \sigma_{22}, \sigma_{12}, \sigma_{13}, \sigma_{23}\}$ we mean that the symbols in the curly braces are in correspondence, one by one. Here the names of the stress components used in the software, S11 etc, and the usual notation used for stress components σ_x, align as

$$
\mathrm{S11} = \sigma_{11} = \sigma_x, \quad \mathrm{S22} = \sigma_{22} = \sigma_y, \quad \ldots, \quad \mathrm{S23} = \sigma_{23} = \sigma_{yz}. \tag{3.3}
$$

Figure 3.3 illustrates the bending and twisting moment resultants: the stress generated in the shell acts on the arm z relative to the reference surface. Integrated throughout the thickness of the shell, this results in bending moments and the twisting moment. The moments are really densities

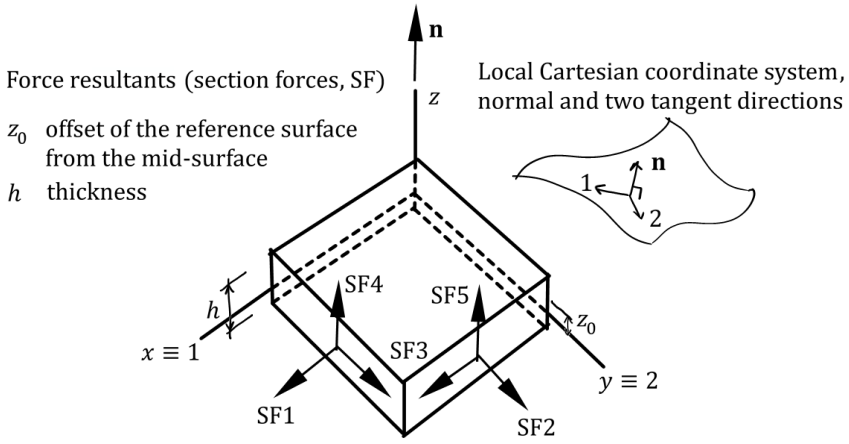


Fig. 3.2. Shell model. The force resultants.

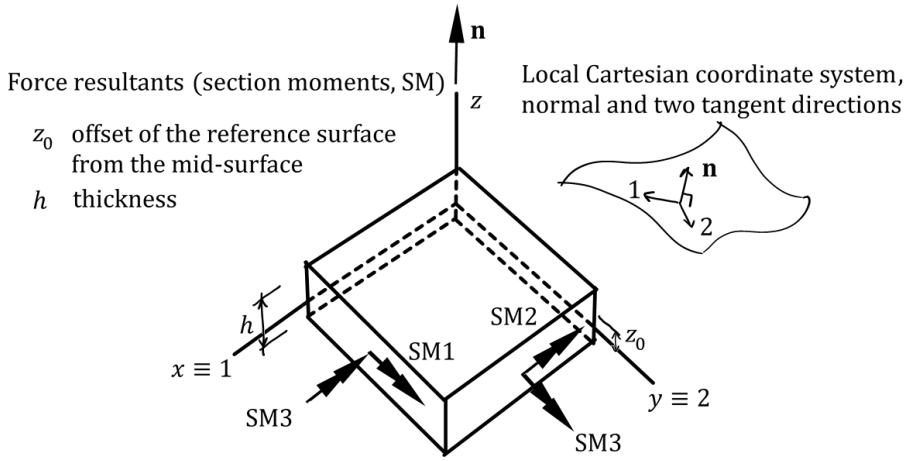


Fig. 3.3. Shell model. The moment resultants.

of moment: moment per unit length of the cut through the shell. The formulas for the bending and twisting moments read

$$\{\mathrm{SM1}, \mathrm{SM2}, \mathrm{SM3}\} = \int_{-h/2-z_0}^{h/2-z_0} z \times \{\sigma_{11}, \sigma_{22}, \sigma_{12}\} \, \mathrm{d}z \tag{3.4}$$

Importantly, the formulas for the resultant forces and moments integrate through the material of the shell. This extends from the bottom surface at $-h/2 - z_0$ to the top surface at $h/2 - z_0$. Figure 3.4 illustrates the geometry of the offset of the reference surface of the shell (where the nodes are located) and the midsurface of the shell. The midsurface of the shell is by definition halfway between the bottom and top surface. The **reference** surface is offset **from** the **midsurface** by z_0, which is negative when it is in the direction opposite to the normal, it is zero when the two surfaces coincide, and it is positive when the reference surface is offset from the midsurface in the direction of the normal. The accuracy of the model may degrade if the offset is too large: when the midsurface is curved, offsetting the reference surface may change significantly the areas that should correspond to each other on those two surfaces.

Balance equations.

Next we derive the balance equations in terms of the force resultants and the moment resultants. Again, these equations will be valid only in the local Cartesian coordinate system: not everywhere

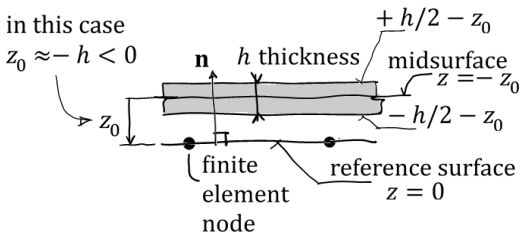


Fig. 3.4. Shell model. The relationship between the reference surface and the midsurface.

along the shell. The starting point is the differential balance equation for the three dimensional stresses [PK1]

$$\begin{bmatrix} \partial/\partial x & 0 & 0 & \partial/\partial y & \partial/\partial z & 0 \\ 0 & \partial/\partial y & 0 & \partial/\partial x & 0 & \partial/\partial z \\ 0 & 0 & \partial/\partial z & 0 & \partial/\partial x & \partial/\partial y \end{bmatrix} \begin{bmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \sigma_{xy} \\ \sigma_{xz} \\ \sigma_{yz} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \sigma_x}{\partial x} + \dfrac{\partial \sigma_{xy}}{\partial y} + \dfrac{\partial \sigma_{xz}}{\partial z} \\ \dfrac{\partial \sigma_{xy}}{\partial x} + \dfrac{\partial \sigma_y}{\partial y} + \dfrac{\partial \sigma_{yz}}{\partial z} \\ \dfrac{\partial \sigma_{xz}}{\partial x} + \dfrac{\partial \sigma_{yz}}{\partial y} + \dfrac{\partial \sigma_z}{\partial z} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \tag{3.5}$$

Here we assume for simplicity that body loads are absent. We take the first equation (3.5), multiply by z, and integrate over the thickness of the shell.

$$\int_{-h/2-z_0}^{+h/2-z_0} z\, \partial\sigma_x/\partial x \, \mathrm{d}z + \int_{-h/2-z_0}^{+h/2-z_0} z\, \partial\sigma_{xy}/\partial y \, \mathrm{d}z + \int_{-h/2-z_0}^{+h/2-z_0} z\, \partial\sigma_{xz}/\partial z \, \mathrm{d}z = 0 \tag{3.6}$$

The first term in (3.6) results in

$$\int_{-h/2-z_0}^{+h/2-z_0} z\, \partial\sigma_x/\partial x \, \mathrm{d}z = \partial \left(\int_{-h/2-z_0}^{+h/2-z_0} z\, \sigma_x \, \mathrm{d}z \right) /\partial x = \partial M_x/\partial x \tag{3.7}$$

Similarly, the second term gives

$$\int_{-h/2-z_0}^{+h/2-z_0} z\, \partial\sigma_{xy}/\partial y \, \mathrm{d}z = \partial \left(\int_{-h/2-z_0}^{+h/2-z_0} z\, \sigma_{xy} \, \mathrm{d}z \right) /\partial y = \partial M_{xy}/\partial y \tag{3.8}$$

The third term in (3.6) is first processed with integration by parts. This splits the term into two contributions, one over the top and bottom surface, and one which remains an integral over the thickness:

$$\int_{-h/2-z_0}^{+h/2-z_0} z\, \partial\sigma_{xz}/\partial z \, \mathrm{d}z = [z\sigma_{xz}]_{-h/2-z_0}^{+h/2-z_0} - \int_{-h/2-z_0}^{+h/2-z_0} \sigma_{xz} \, \mathrm{d}z \tag{3.9}$$

If we assume that the top and bottom surface are without shear tractions, $[z\sigma_{xz}]_{-h/2-z_0}^{+h/2-z_0} = 0$, the first contribution drops out, and we are left with

$$\int_{-h/2-z_0}^{+h/2-z_0} z\, \partial\sigma_{xz}/\partial z \, \mathrm{d}z = -\int_{-h/2-z_0}^{+h/2-z_0} \sigma_{xz} \, \mathrm{d}z = -Q_x \tag{3.10}$$

As a result we have the first balance equation in terms of the force and moment resultants

$$\partial M_x/\partial x + \partial M_{xy}/\partial y - Q_x = 0 \tag{3.11}$$

The above may be compared with the first balance equation for the Bernoulli beam, $M' - V = 0$, obtained in (4.9). Of course the difference is the presence of the twisting moment, which in the beam is absent.

We derive the second equation entirely analogously: we take the second equation (3.5), multiply by z, and integrate over the thickness of the shell.

$$\partial M_{xy}/\partial x + \partial M_y/\partial y - Q_y = 0 \tag{3.12}$$

The third equation (3.5) is not multiplied by z, it is only integrated through the thickness.

$$\int_{-h/2-z_0}^{+h/2-z_0} \partial\sigma_z/\partial z \, \mathrm{d}z + \int_{-h/2-z_0}^{+h/2-z_0} \partial\sigma_{xz}/\partial x \, \mathrm{d}z + \int_{-h/2-z_0}^{+h/2-z_0} \partial\sigma_{yz}/\partial y \, \mathrm{d}z = 0 \tag{3.13}$$

We use the definition of the transverse loading of the shell q

$$q = \int_{-h/2-z_0}^{+h/2-z_0} \partial\sigma_z/\partial z \, \mathrm{d}z = [\sigma_z]_{-h/2-z_0}^{+h/2-z_0} \tag{3.14}$$

and we arrive at the balance equation

$$\partial Q_x/\partial x + \partial Q_y/\partial y + q = 0 \tag{3.15}$$

This might be compared with the Bernoulli-beam second balance equation $V' + q = 0$.

The balance of the membrane force resultants can be derived by integrating the first and second equation (3.5) with respect to the thickness. Here n_x and n_y are the distributed loadings in the plane.

$$\partial N_x/\partial x + \partial N_{xy}/\partial y + n_x = 0 \tag{3.16}$$
$$\partial N_{xy}/\partial x + \partial N_y/\partial y + n_y = 0 \tag{3.17}$$

The balance equations in terms of the moment and force resultants for the flat plate (flat shell), expressed as local expression in Cartesian coordinates aligned with the reference surface, read for the bending, twisting, and transverse shearing

$$\partial M_x/\partial x + \partial M_{xy}/\partial y - Q_x = 0 \tag{3.18}$$
$$\partial M_{xy}/\partial x + \partial M_y/\partial y - Q_y = 0 \tag{3.19}$$
$$\partial Q_x/\partial x + \partial Q_y/\partial y + q = 0 \tag{3.20}$$

and for the membrane forces

$$\partial N_x/\partial x + \partial N_{xy}/\partial y + n_x = 0 \tag{3.21}$$
$$\partial N_{xy}/\partial x + \partial N_y/\partial y + n_y = 0 \tag{3.22}$$

This represents 5 equations in 8 unknown functions. We cannot solve this system as it is, as additional equations are needed.

Constitution of the material.

The next step is to introduce elasticity as a link between the deformations and the stresses (and the resultants, as a consequence). The starting point could be the three dimensional elasticity equation to link the strains to the stresses. We would then introduce the zero σ_z (i.e. thru-the-thickness stress). For an isotropic material with Young's modulus E and Poisson ratio ν, we would then end up essentially with a plane stress assumption plus the transverse shears, which is summarized as follows

$$\begin{aligned}
\partial u/\partial x &= (1/E)\,(\sigma_x - \nu\sigma_y) \\
\partial v/\partial y &= (1/E)\,(-\nu\sigma_x + \sigma_y) \\
\partial u/\partial y + \partial v/\partial x &= \tau_{xy}/G \\
\partial u/\partial z + \partial w/\partial x &= \tau_{xz}/G \\
\partial v/\partial z + \partial w/\partial y &= \tau_{yz}/G
\end{aligned} \tag{3.23}$$

The first two are relations between the membrane stretches and the membranes stresses, the third is the relation between the in-plane shear and the in-plane shear stress, and the last two link the transverse shear strains to the transverse shear stresses.

We take the first equation (3.23), multiply with z and integrate through the thickness (so that on the right we pick up bending and twisting moments), and we substitute on the left from the basic kinematic assumption (3.1)

$$\int_{-h/2-z_0}^{h/2-z_0} z\partial u/\partial x \, \mathrm{d}z = \int_{-h/2-z_0}^{h/2-z_0} \left(z\partial u_0/\partial x + z^2\partial\phi_x/\partial x\right) \, \mathrm{d}z \tag{3.24}$$

$$= \partial u_0/\partial x \int_{-h/2-z_0}^{h/2-z_0} z \, \mathrm{d}z + \partial\phi_y/\partial x \int_{-h/2-z_0}^{h/2-z_0} z^2 \, \mathrm{d}z \tag{3.25}$$

$$= s \, \partial u_0/\partial x + i \, \partial\phi_y/\partial x \tag{3.26}$$

In the last expression we introduce the static moment of area per unit length s and the second moment of area per unit length i.

$$s = \int_{-h/2-z_0}^{h/2-z_0} z \, \mathrm{d}z \, , \quad i = \int_{-h/2-z_0}^{h/2-z_0} z^2 \, \mathrm{d}z \tag{3.27}$$

The result is

$$s \, \partial u_0/\partial x + i \, \partial\phi_y/\partial x = \left(M_x - \nu M_y\right)/E \tag{3.28}$$

Similarly, the second equation (3.23) processed in the same way yields

$$s \, \partial v_0/\partial y - i \, \partial\phi_x/\partial y = \left(-\nu M_x + M_y\right)/E \tag{3.29}$$

The third equation (3.23) gives

$$s \, \left(\partial u_0/\partial y + \partial v_0/\partial x\right) + i \, \left(-\partial\phi_x/\partial x + \partial\phi_y/\partial y\right) = M_{xy}/G \tag{3.30}$$

Next, we continue with the first equation (3.23), but this time we do not multiply with z: Now we want to pick up the membrane forces, not the moments. We integrate through the thickness, and we substitute on the left from the basic kinematic assumption (3.1)

$$h \, \partial u_0/\partial x + s \, \partial\phi_y/\partial x = \left(N_x - \nu N_y\right)/E \tag{3.31}$$

The second equation (3.23) gives

$$h \, \partial v_0/\partial y - s \, \partial\phi_x/\partial y = \left(-\nu N_x + N_y\right)/E \tag{3.32}$$

The third equation (3.23) gives

$$h \, \left(\partial u_0/\partial y + \partial v_0/\partial x\right) + s \, \left(-\partial\phi_x/\partial x + \partial\phi_y/\partial y\right) = N_{xy}/G \tag{3.33}$$

Comparing the equations for the moments and those for the membrane forces we can see how the patterns develop: the cross sectional area properties change according to whether we integrated the equations multiplied with z or not multiplied with z.

Now it remains to integrate the fourth and fifth equation (3.23). Here we anticipate the need to correct the relation between the transverse shear strains and stresses: the shear strains are uniform through the thickness, but the shear stresses cannot be (that wouldn't satisfy the boundary conditions at the bottom and top surface of the of the shell). Hence we introduce a shear correction factor k_s:

$$h(\phi_y + \partial w_0/\partial x) = Q_x/(k_s G) \, , \quad h(-\phi_x + \partial w_0/\partial y) = Q_y/(k_s G) \, , \tag{3.34}$$

We obtain the following as the result of linking the deformations to the resultant moment and forces: on the left, we have five unknown functions, u_0, v_0, w_0, ϕ_x, ϕ_y. On the right, we have eight unknown functions, M_x, M_y, M_{xy}, N_x, N_y, N_{xy}, Q_x, and Q_y.

$$s\,\partial u_0/\partial x + i\,\partial\phi_y/\partial x = (M_x - \nu M_y)/E \tag{3.35}$$
$$s\,\partial v_0/\partial y - i\,\partial\phi_x/\partial y = (-\nu M_x + M_y)/E \tag{3.36}$$
$$s\,(\partial u_0/\partial y + \partial v_0/\partial x) + i\,(-\partial\phi_x/\partial x + \partial\phi_y/\partial y) = M_{xy}/G \tag{3.37}$$
$$h\,\partial u_0/\partial x + s\,\partial\phi_y/\partial x = (N_x - \nu N_y)/E \tag{3.38}$$
$$h\,\partial v_0/\partial y - s\,\partial\phi_x/\partial y = (-\nu N_x + N_y)/E \tag{3.39}$$
$$h\,(\partial u_0/\partial y + \partial v_0/\partial x) + s\,(-\partial\phi_x/\partial x + \partial\phi_y/\partial y) = N_{xy}/G \tag{3.40}$$
$$h(\phi_y + \partial w_0/\partial x) = Q_x/(k_s G) \tag{3.41}$$
$$h(-\phi_x + \partial w_0/\partial y) = Q_y/(k_s G) \tag{3.42}$$

Earlier we constructed a system of five balance equations (3.18) and (3.21). Therefore we can solve for the resultant functions on the right, and substitute them into the balance equations. That will yield five equations for five unknown functions of displacements and rotations.

In order to make things easier for us, we'll make an assumption here that will drastically reduce the complexity of the task: When the offset of the reference surface from the midsurface is zero, the moment of area is zero, $s = 0$. We shall also consider only structures of uniform thickness, $h = \text{const}$, and therefore $i = h^3/12$. The simplified membrane constitutive equations then are

$$h\,\partial u_0/\partial x = (N_x - \nu N_y)/E \tag{3.43}$$
$$h\,\partial v_0/\partial y = (-\nu N_x + N_y)/E \tag{3.44}$$
$$h\,(\partial u_0/\partial y + \partial v_0/\partial x) = N_{xy}/G \tag{3.45}$$

The constitutive equations for the bending and twisting moments read

$$i\,\partial\phi_y/\partial x = (M_x - \nu M_y)/E \tag{3.46}$$
$$-i\,\partial\phi_x/\partial y = (-\nu M_x + M_y)/E \tag{3.47}$$
$$i\,(-\partial\phi_x/\partial x + \partial\phi_y/\partial y) = M_{xy}/G \tag{3.48}$$

And, finally, the two equations for the transverse shear forces

$$h(\phi_y + \partial w_0/\partial x) = Q_x/(k_s G) \tag{3.49}$$
$$h(-\phi_x + \partial w_0/\partial y) = Q_y/(k_s G) \tag{3.50}$$

These relationships can be inverted for the membrane forces as

$$N_x = \frac{Eh}{1-\nu^2}\,(\partial u_0/\partial x + \nu\partial v_0/\partial y) \tag{3.51}$$
$$N_y = \frac{Eh}{1-\nu^2}\,(\nu\partial u_0/\partial x + \partial v_0/\partial y) \tag{3.52}$$
$$N_{xy} = Gh\,(\partial u_0/\partial y + \partial v_0/\partial x) \tag{3.53}$$

for the bending moments as

$$M_x = \frac{Ei}{1-\nu^2}\,(\partial\phi_y/\partial x - \nu\partial\phi_x/\partial y) \tag{3.54}$$
$$M_y = \frac{Ei}{1-\nu^2}\,(\nu\partial\phi_y/\partial x - \partial\phi_x/\partial y) \tag{3.55}$$
$$M_{xy} = Gi\,(-\partial\phi_x/\partial y + \partial\phi_y/\partial x) \tag{3.56}$$

and for the transverse shear forces as

$$Q_x = k_s G h(\phi_y + \partial w_0/\partial x) \tag{3.57}$$
$$Q_y = k_s G h(-\phi_x + \partial w_0/\partial y) \tag{3.58}$$

In the following sections, by introducing additional simplifying assumptions, we will develop a few important models as special cases. In the next section, we shall have a look at membranes (the well known model of plane-stress). Then we turn our attention to flat plates, both without transverse shear strains (Kirchhoff) and with transverse shear strains (Reissner-Mindlin).

3.2 Membranes

If the transverse deflection w_0 and the rotations ϕ_x and ϕ_y are zero, the three relevant constitutive equations are (3.51), which we can substitute in the balance equations (3.21). The result is the classical plane-stress continuum model

$$\frac{Eh}{1-\nu^2}\left(\frac{\partial^2 u_0}{\partial x^2}+\nu\frac{\partial^2 v_0}{\partial x\partial y}\right)+Gh\left(\frac{\partial^2 u_0}{\partial y^2}+\frac{\partial^2 v_0}{\partial x\partial y}\right)+n_x=0 \tag{3.59}$$

$$Gh\left(\frac{\partial^2 v_0}{\partial x^2}+\frac{\partial^2 u_0}{\partial x\partial y}\right)+\frac{Eh}{1-\nu^2}\left(\frac{\partial^2 v_0}{\partial y^2}+\nu\frac{\partial^2 u_0}{\partial x\partial y}\right)+n_y=0 \tag{3.60}$$

The two unknown functions are u_0 and v_0. The two equations are of second order, in two unknown functions: that implies that two boundary conditions are needed at each point of the boundary. At each point of the boundary we can set up a special coordinate system, such that the boundary conditions are easy to describe in this coordinate system (Figure 3.5). As discussed in the continuum-models book [PK1], for each of the coordinate directions we can prescribe either a displacement component, or a traction component. The list below is only for two particularly simple situations:

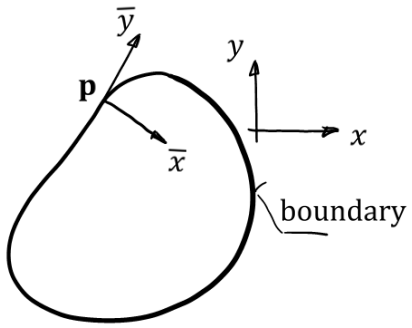


Fig. 3.5. Membrane or plate boundary conditions. The boundary conditions are specified in a coordinate system set up at **p** such that $\overline{y}$ is tangent to the boundary.

Clamped edge. At a clamped edge we prescribe zero displacement in both directions.

$$u_0(\mathbf{p})=0\,,\quad v_0(\mathbf{p})=0 \tag{3.61}$$

The two components u_0, v_0 are on the $\overline{xy}$ coordinate directions; but for the clamped condition the coordinate system does not matter.

Free edge. For an unsupported edge we prescribe zero resultant tractions:

$$N_{\overline{x}}(\mathbf{p})=0\,,\quad N_{\overline{x}\overline{y}}(\mathbf{p})=0 \tag{3.62}$$

Note that the normal component is prescribed, and the shear component is also prescribed. We cannot prescribe $N_{\overline{y}}(\mathbf{p})$.

3.3 Kirchhoff Flat Plates (Slabs)

Kirchhoff was one of the researchers to be eventually credited with developing models of shear-rigid plates. The Kirchhoff theory proceeds by linking the rotations to the slope of the deflection w

$$\phi_x = +\frac{\partial w_0}{\partial y} \tag{3.63}$$

$$\phi_y = -\frac{\partial w_0}{\partial x} \tag{3.64}$$

An immediate consequence is that the transverse shear forces cannot be determined from the deformation since we have

$$h(\phi_y + \partial w/\partial x) = 0 \Longrightarrow Q_x = 0 \tag{3.65}$$

and analogously $Q_y = 0$. This is precisely what we see happen with the Bernoulli beam (it is also rigid in shear).

The Kirchhoff assumption is equivalent to the Bernoulli assumption for beams: the normal to the midsurface remains normal to the surface throughout the deformation. The shear strains vanish, and the shear forces cannot be determined as proportional to the deformation.

If the in-plane displacements u_0 and v_0 are identically zero, the unknowns will become w_0 and the rotations ϕ_x and ϕ_y. Since for the Kirchhoff theory the rotations are linked to the slopes (3.63), the complete solution for the thin plate hinges on finding the function w_0.

By introducing the Kirchhoff assumptions (3.63) into (3.54) we can write

$$M_x = -D\left(\kappa_x + \nu\kappa_y\right) \tag{3.66}$$

$$M_y = -D\left(\nu\kappa_x + \kappa_y\right) \tag{3.67}$$

Here we introduced the flexural stiffness coefficient D

$$D = \frac{Ei}{1-\nu^2} \tag{3.68}$$

and the curvatures

$$\kappa_x = \frac{\partial^2 w_0}{\partial x^2}\ , \quad \kappa_y = \frac{\partial^2 w_0}{\partial y^2}\ . \tag{3.69}$$

The third equation can be directly rewritten in terms of the flexural stiffness coefficient as

$$M_{xy} = \frac{(1-\nu)D}{2}\left(-\frac{\partial \phi_x}{\partial x} + \frac{\partial \phi_y}{\partial y}\right)\ . \tag{3.70}$$

Here we again introduce a curvature (twisting)

$$\kappa_{xy} = \frac{\partial^2 w_0}{\partial x \partial y} \tag{3.71}$$

and we write

$$M_{xy} = -(1-\nu)D\kappa_{xy}\ . \tag{3.72}$$

An equation for the deflection w_0 is now derived with a trick: the first and second balance equations (3.18) are differentiated with respect to x and y respectively

$$\frac{\partial^2 M_x}{\partial x^2} + \frac{\partial^2 M_{xy}}{\partial x \partial y} - \frac{\partial Q_x}{\partial x} = 0 \tag{3.73}$$

$$\frac{\partial^2 M_{xy}}{\partial x \partial y} + \frac{\partial^2 M_y}{\partial y^2} - \frac{\partial Q_y}{\partial y} = 0 \tag{3.74}$$

These equations are then added together, and the third equation (3.18) is used to replace the rate of change of the shear forces with the transverse loading to yield

$$\frac{\partial^2 M_x}{\partial x^2} + 2\frac{\partial^2 M_{xy}}{\partial x \partial y} + \frac{\partial^2 M_y}{\partial y^2} - q = 0 \,. \tag{3.75}$$

Finally we substitute the expressions for the curvatures to obtain

$$\frac{\partial^4 w_0}{\partial x^4} + 2\frac{\partial^4 w_0}{\partial x^2 \partial y^2} + \frac{\partial^4 w_0}{\partial y^4} - q/D = 0 \,. \tag{3.76}$$

This balance equation is analogous to that of the Bernoulli beam $EIw'''' - q = 0$: to see that we simply write

$$w'''' - q/(EI) = 0 \,. \tag{3.77}$$

Here obviously $w'''' = \partial^4 w/\partial x^4$.

Boundary conditions.

The balance equation of the thin plate (3.76) is of the fourth order and it is an equation for a single unknown function, w_0. Therefore, at any point of the boundary of the plate, there need to be two boundary condition values.

For the description of the boundary conditions we use a Cartesian coordinate system set up at $\mathbf{p}$ such that $\overline{y}$ is tangent to the boundary (Figure 3.5). For simplicity we specify the boundary conditions assuming that a zero value is prescribed. (This is easily modified when we wish to prescribe non zero values.)

Clamped edge. At a clamped edge we prescribe zero displacement and zero slope in the direction normal to the boundary. Note that the prescribed zero displacement along the boundary implies zero slope along the boundary, but we cannot prescribe three things, only two.

$$w_0(\mathbf{p}) = 0 \,, \quad \frac{\partial w_0(\mathbf{p})}{\partial \overline{x}} = 0 \,. \tag{3.78}$$

Simply-supported edge. For an edge on a knife-edge support the deflection vanishes, and we can see that the moment about the tangent to the edge is zero. This is very much related to our notions of simple support developed for the Bernoulli beam.

$$w_0(\mathbf{p}) = 0 \,, \quad M_{\overline{x}}(\mathbf{p}) = 0 \,. \tag{3.79}$$

Free edge. For an unsupported edge, i.e. a free edge, it would appear reasonable to prescribe zero bending moment, zero twisting moment, and zero shear force:

$$M_{\overline{x}}(\mathbf{p}) = 0 \,, \quad M_{\overline{x}\overline{y}}(\mathbf{p}) = 0 \,, \quad Q_{\overline{x}}(\mathbf{p}) = 0 \,. \tag{3.80}$$

But that is one condition too many: we can only prescribe two. This problem was solved by Kirchhoff, who introduced the so called effective shear force as

$$Q_{\mathrm{eff},\overline{x}}(\mathbf{p}) = \frac{\partial M_{\overline{x}\overline{y}}(\mathbf{p})}{\partial \overline{y}} + Q_{\overline{x}}(\mathbf{p}) \tag{3.81}$$

The decisive feature of this effective force is that it combines the twisting moment with the shear force – Figure 3.6 provides a visual explanation of how elementary twisting moments are converted to elementary couples which are combined to transverse forces using a Taylor series. In terms of the Kirchhoff effective force and the bending moment, the free edge boundary condition is given as

$$M_{\overline{x}}(\mathbf{p}) = 0 \,, \quad Q_{\mathrm{eff},\overline{x}}(\mathbf{p}) = 0 \,. \tag{3.82}$$

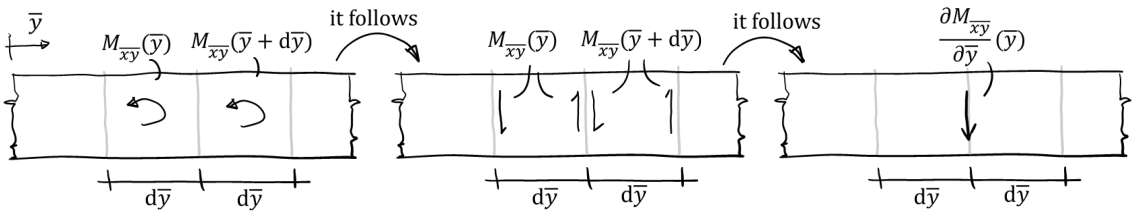


Fig. 3.6. Plate boundary conditions. The contribution of the twisting moment to the Kirchhoff effective shear force.

3.4 Reissner-Mindlin Flat Plates (Slabs)

Reissner and Mindlin formulated similar, but not identical, theories of plates to incorporate transverse shear. These theories have in common that ϕ_x and ϕ_y are independent of the deflection w_0. Thus, there are three independent functions to solve for from a system of balance equations. This is in contrast to the Kirchhoff theory, where there is a single unknown function and, correspondingly, a single balance equation.

The moments from (3.54) and the transverse shear forces from (3.57) are directly substituted in (3.18)

$$D\left(\frac{\partial^2\phi_y}{\partial x^2}-\nu\frac{\partial^2\phi_x}{\partial x\partial y}\right)+\frac{(1-\nu)D}{2}\left(\frac{\partial^2\phi_y}{\partial y^2}-\frac{\partial^2\phi_x}{\partial x\partial y}\right)-hk_sG\left(\phi_y+\frac{\partial w_0}{\partial x}\right)=0 \tag{3.83}$$

$$\frac{(1-\nu)D}{2}\left(\frac{\partial^2\phi_y}{\partial x\partial y}-\frac{\partial^2\phi_x}{\partial x^2}\right)+D\left(\nu\frac{\partial^2\phi_y}{\partial x\partial y}-\frac{\partial^2\phi_x}{\partial y^2}\right)-hk_sG\left(-\phi_x+\frac{\partial w_0}{\partial y}\right)=0 \tag{3.84}$$

$$hk_sG\left(\frac{\partial\phi_y}{\partial x}+\frac{\partial^2 w_0}{\partial x^2}\right)+hk_sG\left(-\frac{\partial\phi_x}{\partial y}+\frac{\partial^2 w_0}{\partial y^2}\right)-q=0 \tag{3.85}$$

This represents three partial differential equations in three unknown functions.

Boundary conditions.

The differential equations are of second order, in contrast to the Kirchhoff equation (3.76). In order to have enough boundary conditions, three boundary conditions must be provided at each point of the boundary of the plate. Recall that having three boundary conditions was a problem for the Kirchhoff plate: not here.

For the description of the boundary conditions we again use a Cartesian coordinate system set up at $\mathbf{p}$ such that $\overline{y}$ is tangent to the boundary (Figure 3.5). And again, for simplicity we specify the boundary conditions assuming that a zero value is prescribed. (This can be modified when we wish to prescribe non zero values.)

Clamped edge. At a clamped edge we prescribe zero displacement, and zero rotations both about the normal to the boundary and about the tangent to the boundary.

$$w_0(\mathbf{p})=0\,,\quad \phi_{\overline{x}}(\mathbf{p})=0\,,\quad \phi_{\overline{y}}(\mathbf{p})=0\,. \tag{3.86}$$

Simply-supported edge. For an edge on a knife-edge support the deflection vanishes, and we can see that the moment about the tangent to the edge is zero. Next, there are two possibilities: either the rotation about the normal to the boundary is zero (the so called hard support)

$$w_0(\mathbf{p})=0\,,\quad M_{\overline{x}}(\mathbf{p})=0\,,\quad \phi_{\overline{x}}(\mathbf{p})=0\,, \tag{3.87}$$

where $\phi_{\overline{x}}(\mathbf{p})$ is the rotation about the local axis $\overline{x}$, or the twisting moment about the normal to the boundary is zero (the so called soft support)

$$w_0(\mathbf{p})=0\,,\quad M_{\overline{x}}(\mathbf{p})=0\,,\quad M_{\overline{xy}}(\mathbf{p})=0\,. \tag{3.88}$$

The soft support is usually more realistic.

Free edge. For an unsupported edge, i.e. a free edge, we prescribe zero bending moment, zero twisting moment, and zero shear force:

$$M_{\overline{x}}(\mathbf{p}) = 0 \,, \quad M_{\overline{x}\overline{y}}(\mathbf{p}) = 0 \,, \quad Q_{\overline{x}}(\mathbf{p}) = 0 \,. \tag{3.89}$$

3.5 Shells

If we focus our attention on a very small piece of a shell (infinitesimal element), its action can be thought of as a superposition of a membrane with a plate. That was in fact the point of the two figures, Figure 3.2 and Figure 3.3: the elastic action of the infinitesimal piece of the shell can be decomposed into the membrane forces, the bending and twisting moments, and the transverse shear forces. Consequently, the model of the shell appears to be composed of the equations (3.59), and either (3.76) to obtain the Kirchhoff-Love shell model, or (3.83) to obtain the Reissner-Mindlin shell model. Plus, of course, the boundary conditions appropriate to the models that we chose.

Curvilinear coordinates.

Alas, things are not so simple. In order to describe the model properly, we have to make it applicable to the entire shell. That will involve working in the so called curvilinear coordinates on the reference surface (Figure 3.7). The mathematics gets somewhat hairy at that point (viz nearly one hundred pages of linear shell theory in [ZTF]). Therefore, we will not endeavour to deal with the general shell model here.

Nevertheless, we *will* need to keep in mind that on any shell that we wish to analyze we have to construct a system of coordinates (i.e. the above-mentioned curvilinear coordinates). It is crucial for being able to talk about stress components and stress resultants in a meaningful way: We will need to compare stresses and forces acting on different parts of the shell, but that will require that the coordinate systems at different points of the shell be somehow coherent: we need to make sure that when we talk of S11 at two different points we are comparing apples to apples.

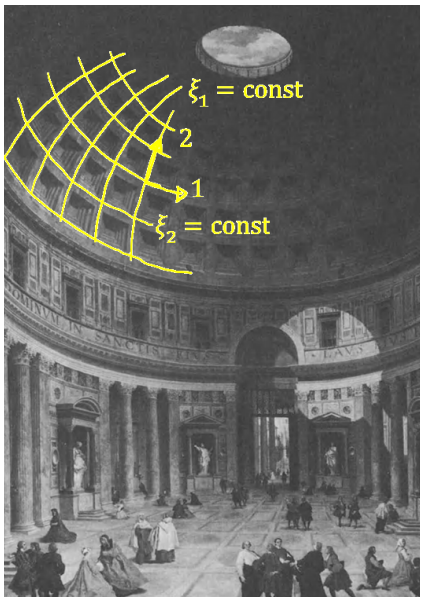


Fig. 3.7. Cartoon of a curvilinear coordinate system on a shell.

When is a shell model appropriate?

The Love of the Kirchhoff-Love researcher duo is known for significant contributions to elasticity, including shells. Love proposed the ratio of the radius of curvature of the shell to the thickness of the shell as a measure of what a shell might be. When this ratio is ≥ 10, the structure is in the range in which the shell theory is applicable; below this value of the ratio the structure should be considered a curved solid. The structure in Figure 3.8 qualifies for a shell treatment. Obviously, this criterion has nothing to say about flat slabs (radius of curvature is infinite). The relevant quantity then is the ratio of the span to the thickness.

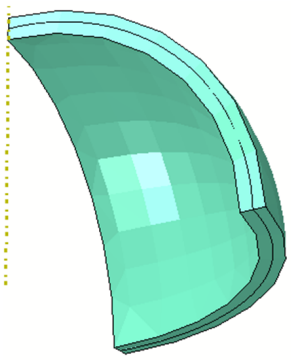


Fig. 3.8. Hollow sphere, mid surface radius 50 units, thickness 5 units: ratio of the radius of curvature of the shell to the thickness of the shell equal to 10.

3.6 Gotchas of plate models

Plates subject to bending and transverse shear are only a subset of the full capabilities of shell models, yet there is plenty of complexity here already. The plate model based on the Kirchhoff assumption has certain shortcomings related to the boundary conditions (refer to the effective shear force), plus it cannot incorporate shear flexibility, of course. The shear-flexible Reissner-Mindlin plate model improves the handling of the boundary conditions, while remaining consistent with the Kirchhoff model for thin plates away from the boundaries.

It is exactly for thin plates where the Reissner-Mindlin plate model adds additional complexity. This model is capable of satisfying boundary conditions that make it possible for the twisting moment to vanish on the boundary: at the free edge, and at the simply-supported edge. However, this happens over a very narrow strip of the plate right next to the boundary: sometimes this is called the *boundary layer*.

Figure 3.9 shows a square plate with a uniform distributed load on top, with aspect ratio 50 (side length divided by the thickness), simply supported around the circumference. Figure 3.9(a) shows the twisting moment for the shear-rigid Kirchhoff plate model, where the rotations around the normal to the boundary are suppressed. Consequently, the twisting moment on the boundary does not vanish (note how the red and blue colors extend all the way to the edge of the plate), which means this model is not consistent with the real 3D plate situation. Figure 3.9(b) shows a good solution with the shear-flexible-model (Reissner-Mindlin): the twisting moment drops off to zero at the circumference of the plate. Notice how it happens over a very short distance from the boundary (roughly proportional to the thickness of the plate). The mesh must be very much refined towards the boundary in order to capture this sharp transition (compare with the mesh of Figure 3.10(a), strongly refined towards the circumference).

Figure 3.9(c) shows another solution obtained with the shear-flexible-model. The resulting picture of the twisting moment look like a mixture of the (a) and (b) solutions. It was obtained with quite

a fine mesh (Figure 3.10(b), uniform mesh), which nevertheless was incapable of capturing the the boundary layer. The twisting moment shows the tendency to decrease to zero towards the boundary, but doesn't quite manage to do that.

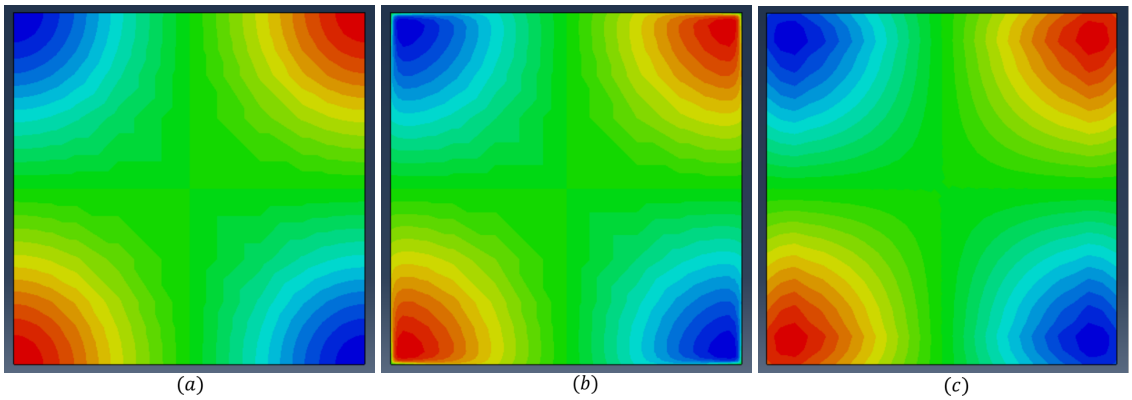


Fig. 3.9. Square plate with uniform loading. Twisting moment $SM3$. (a) Shear-rigid (Kirchhoff) model solution. (b) Shear-flexible (Reissner-Mindlin) model accurate solution with a strongly graded mesh. (c) Shear-flexible (Reissner-Mindlin) model solution with a uniform mesh, which lacks in accuracy.

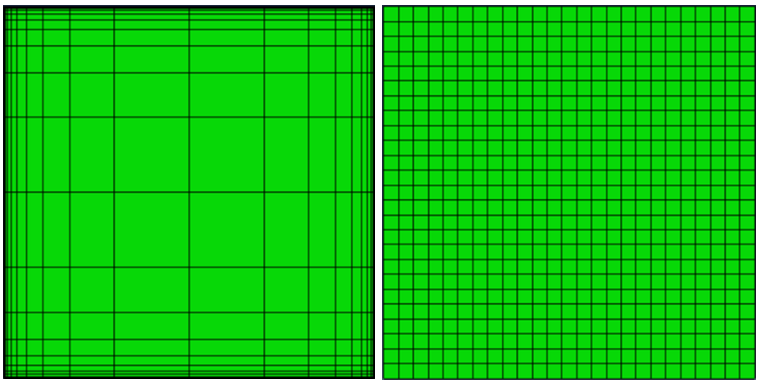
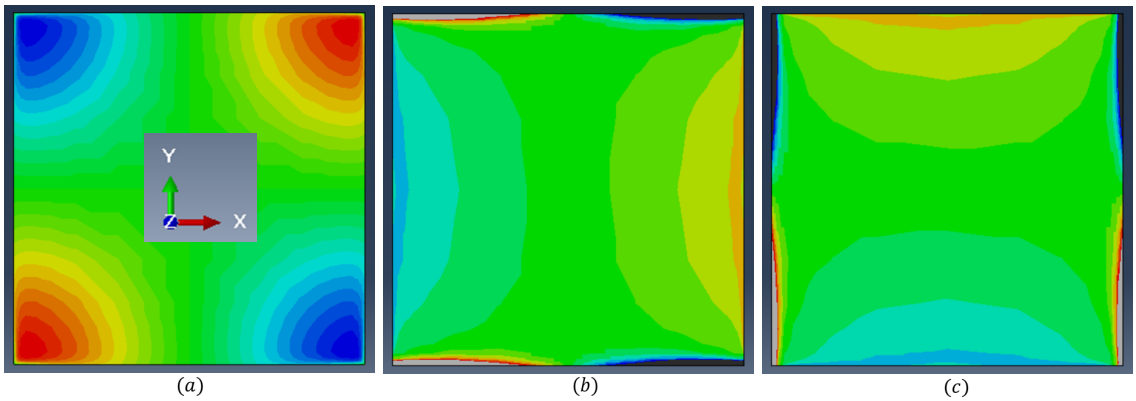


Fig. 3.10. Meshes used for the square plate. (a) Strongly refined towards the boundary in order to capture the boundary layer. (b) Uniform mesh.

That is not all however, the transverse shear forces have boundary layers too. Figure 3.11 shows the shear-flexible plate model solution for the twisting moment in (a), and the solutions for the shear forces (b) $SF4$ and (c) $SF5$. The shear forces have a maximum right next to the boundary, in the places where the twisting moment drops off to zero. Clearly these quantities are coupled together (refer to (3.18): note along the horizontal edges a rapid change – rate– of the twisting moment with respect to Y needs to be balanced by Q_X, and vice versa are along the vertical edges). Note that the shear forces along the horizontal edges represents shear between X and Z, not between Y and Z – the latter would not be consistent with the boundary conditions on the circumferential surface of the plate. Analogously the shear along the vertical edges changes the angle between Y and Z. Figure 3.12 demonstrates the shear stress $S13$ (corresponding to the shear resultant force $SF4$) in a real three-dimensional model of the plate. The hotspot of the shear stress may be compared with the corresponding hotspot of the $SF4$ force in Figure 3.11(b).



(a) (b) (c)

Fig. 3.11. Square plate with uniform loading. Twisting moment $SM3$ (a), and the shear forces (b) $SF4$ (shear between the global coordinate axes X and Z; refer to the inset axis symbol), and (c) $SF5$ (shear between the global coordinate axes Y and Z).

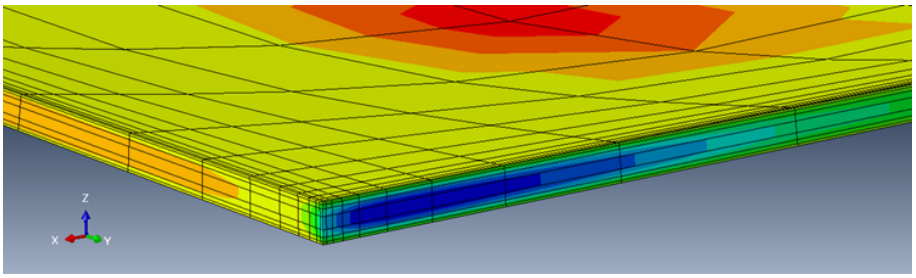


Fig. 3.12. Square plate with uniform loading. Full 3D model, the shear stress $S13$. Compare with the top-right corner of the plate model in Figure 3.11(b).

A note is in order: for thick plates the boundary layers will not be as narrow as for thin plates, and reasonable results can be obtained with uniform meshes. Resolving the boundary layers for thin plates may take a lot of refinement of the meshes towards the boundary.

In summary: The shear-flexible Reissner-Mindlin model treats the boundary conditions in a more realistic manner than the shear-rigid Kirchhoff model, but it turns out that the solutions display boundary layers. Resolving the boundary layers will require care when designing the meshes. Even when the solution doesn't manage to compute a sufficiently accurate picture of the boundary layers, we need to be aware of the possibility that they may be there in order to interpret the force and moment resultant appropriately.

3.7 Concentrated forces on plates and shells

The Kirchhoff-Love and the Reissner-Mindlin models of plates and shells result in different equation types: fourth-order equation for the shear-rigid KL vs. second-order system of equations for the shear-flexible RM. Therefore we should not be surprised the see differences in the circumstances that can be addressed with those different models.

Concentrated forces are an admissible type of load for shear-rigid plates (and shells). Not so for shear-*flexible* plate and shells. The deflection under a nonzero concentrated force is infinite when the shear deformation is taken into account. Analogously, nonzero reaction punches through the

shear-flexible plate/shell like a needle. In other words, concentrated forces (which includes reactions at point supports) are *inadmissible* for RM plates and shells [PKJS].

Avoid concentrated forces and point supports when using the Reissner-Mindlin plate or shell models. Remember that the loading and support conditions could be set up before we select the finite elements to be used in the analysis. It could happen that an inappropriate element type will be selected inadvertently. Always circle back to check that the assumptions made along the way are consistent.

4

Truss and beam models of structures

Summary

1. The mathematical models of structural members can be reduced in dimension to one coordinate for beams and rods. This means they can be very efficient. But: they tend to be more complicated than the mathematical models of the continuum (3d, 2d plane stress and strain, etc.).
2. Mathematical models are here developed for axial bars, beams, and torsion of rods.

4.1 Truss model

Reduced models of three dimensional bodies are more complicated than simple continuum models of such bodies. This may be surprising, given how difficult it is to obtain analytical solutions of three dimensional models. Reduced models in comparison represent substantially less work to solve: For instance to deal with a single-span beam, all we have to do is integrate a simple differential equation and figure out the constants of integration. However, that will not give us a solution of the original three dimensional problem, but only an approximation of it. The complications that we will hear about later come about because of our need to make these approximations progressively better.

As a simple illustrative example, we will consider the truss model. Figure 4.1 shows a planar model of a three dimensional prismatic body (bar). The three dimensional model is reduced by making assumptions as to the displacements and/or stresses. Any textbook we pick up would likely state that the axial displacement u was to be determined, while the displacement component w (and the displacement component v out of the plane of the paper) were identically zero. Furthermore, an assumption would be made that the axial stress was proportional to the axial strain, $\sigma_x = E\epsilon_x$.

Unfortunately, these two assumptions are in conflict. If $w = 0$ and $v = 0$, then the strains $\epsilon_y = 0$, and $\epsilon_z = 0$. We can introduce this into the three dimensional constitutive equations

$$E\epsilon_x = \sigma_x - \nu(\sigma_y + \sigma_z) \ , \quad E\epsilon_y = \sigma_y - \nu(\sigma_x + \sigma_z) \ , \quad E\epsilon_z = \sigma_z - \nu(\sigma_y + \sigma_x) \tag{4.1}$$

from which we can solve for the transverse stresses

$$\sigma_y = \sigma_z = \frac{\nu}{1-\nu}\sigma_x \tag{4.2}$$

and we find them proportional to the axial stress σ_x (not zero!). Solving for the axial stress in terms of the axial strain then gives

$$\sigma_x = E\frac{1-\nu}{(1+\nu)(1-2\nu)}\epsilon_x \ , \tag{4.3}$$

which is different from the assumed $\sigma_x = E\epsilon_x$. For $\nu = 0.3$ the stiffness coefficient is not E, but it is almost 35% percent larger, $1.346E$. The reason is the constraint that was imposed on the cross

sectional area due to our assumption: if the cross section cannot change in size ($\epsilon_y = 0$, and $\epsilon_z = 0$), there must be non zero stresses (reactions) σ_y, σ_z to make it happen. This then increases the stiffness of the bar.

In fact, a better assumption would be $\sigma_y = 0$, $\sigma_z = 0$ (which is actually consistent with the boundary conditions on the cylindrical surface $\widehat{S}$ of the bar). Then we can indeed obtain $\sigma_x = E\epsilon_x$. However, in that case the deflections in a direction orthogonal to the axis of the bar, v and w, are not actually zero, and so the area of the bar must change in response to its axial deformation. That contradicts the usually made assumption that the cross sectional area of the bar is fixed.

The point is to illustrate that when formulating reduced models (such as one-dimensional instead of three-dimensional), assumptions are often made which are not obvious, and which in many cases form contradictions and are mismatched with the conditions of the original three dimensional problem.

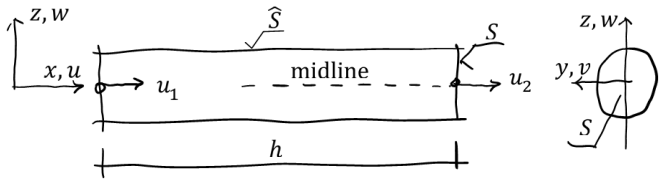


Fig. 4.1. Truss member.

So, to make progress we shall assume that only σ_x matters (the other components of stress will be assumed to be zero, including on the boundary $\widehat{S}$), and also we will have to assume that even though the other displacement components may cause the cross sectional area of the bar to change, we can ignore that change and consider the cross sectional area S fixed. The balance equation of the bar (in the absence of body forces) is

$$S\frac{\partial \sigma_x}{\partial x} = 0 \tag{4.4}$$

where S is the cross sectional area. This readily transforms for a prismatic (i.e. of uniform cross section) homogeneous bar into

$$ES\frac{\partial^2 u}{\partial x^2} = 0\ , \tag{4.5}$$

the pointwise balance of the material of the bar.

Boundary conditions.

The governing equation is of second order in a single variable, and therefore a single boundary condition is needed at each piece of the boundary. Given that the boundary of the bar consists of its two endpoints, there will be two boundary conditions, which is precisely what is needed to supply the integration constants. (Note that the boundary $\widehat{S}$ is traction-free!)

4.2 Beam bending models: General ideas

We would aim to use beam elements to model structures in which one dimension (the length) is significantly greater than the other two dimensions (cross section), and in which the longitudinal stress and shear stresses due to torsion and transverse shear forces are important, whereas the direct stresses normal to the axis of the beam are negligible.

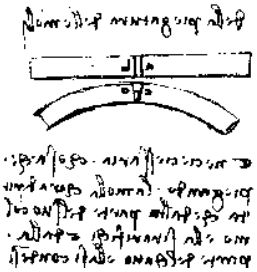


Fig. 4.2. From da Vinci's notebook [Reti]. Da Vinci speaks of compressed and stretched fibres of the beam.

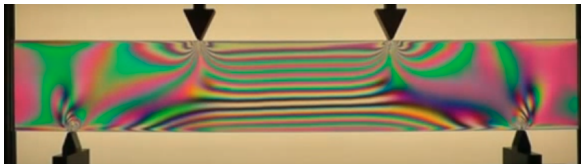


Fig. 4.3. Photoelastic investigation of four-point bending. Fringes in the middle third of the beam correspond to pure bending. Note the disturbed areas where the edges of the test rig push against the bottom and top surface, though. Image credit: Tiedemann Instruments.

Beam theory was historically developed from the experimental observations that the deformation of the structure can be determined as functions of position along the structure's length (Figure 4.2 represents a historical curiosity; real engineering experiments had to wait for the 19th century.)

Beam models tend to produce acceptable results when the dimensions of the cross-section dimensions are less than around 1/10 of the structure's typical axial dimension. We would consider the following as determining the "typical" axial dimension:

1. the distance between supports,
2. the distance between sudden changes in cross-section, and
3. the half-wavelength of the highest vibration mode of interest.

Refer to Figure 4.4 for an illustration.

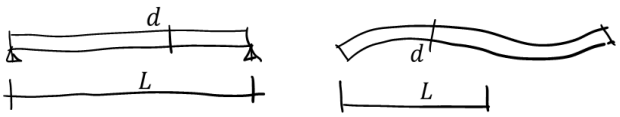


Fig. 4.4. Beam patterns that determine the ratio of the cross-section dimensions to the typical axial dimension.

4.3 Bernoulli beam bending model

We consider here prismatic beams with solid symmetrical sections relative to the $x - z$ plane. Also, the midline is assumed to pass through the centroid of the section. Therefore, the deflection of the beam only occurs in the $x - z$ plane, that is there are displacements w (transverse to the axis of the beam) and u (parallel to the axis of the beam). The following assumptions summarize the Bernoulli-Euler model of transverse beam bending (due to Jacob Bernoulli and Leonhard Euler):

1. The sections of the beam are assumed to remain planar (flat) throughout the deformation.

2. The stresses orthogonal to the midline (i.e. σ_y and σ_z) are assumed to be zero (negligible) so that only the σ_x stress component is of importance. All shear stresses are also neglected.
3. The axial strain only depends on the transverse deflection of the midline.
4. The sections of the beam are assumed to be orthogonal to the midline before and after deformation.

It is well known that the cross section of a beam under bending loads will not in general keep its original shape when deformed. Refer to the readily reproducible example of a bent rubber eraser: see Figure 4.5. As long as the Poisson ratio is not exactly zero, fibres of the beam that are under tension will result in contraction of the corresponding surface of the beam, and vice versa for the fibres under compression.

Fig. 4.5. Bent eraser. Note the anticlastic surface at the top. This is caused by the Poisson effect. The shape of the cross sectional area is no longer a simple rectangle.

Nevertheless, experiments have established that with good engineering accuracy the change in the cross sectional area contributes very little to the response of the beam, with a couple of notable exceptions: (a) curved pipes with thin-walled circular cross section are known to change in stiffness due to the so-called ovaling, (b) thin-walled beams with highly localized loading may experience change in the cross sectional area due to the deformation of the loaded walls.

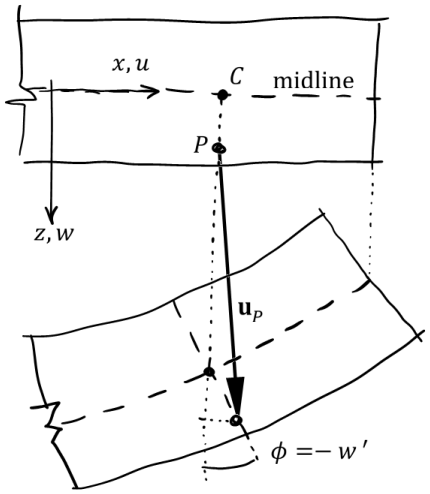


Fig. 4.6. Bernoulli beam. Displacements of point P.

Figure 4.6 illustrates that within any cross section, point P moves axially by the amount

$$u(x_P = x_C, z) = z\phi(x_C) = -zw'(x_C) \tag{4.6}$$

where we introduce rotation ϕ, and note that for the Bernoulli-Euler model the rotation is determined by the derivative w' (in textbooks this is usually stated as a kinematic tie of the rotation to the

slope of the deflection curve). Note that the two points, C (on the midline), and P (in the same cross section, but not on the midline), have the same x coordinate.

The axial strain follows

$$\epsilon_x = u'(x_P = x_C, z) = -zw''(x_C) \tag{4.7}$$

The stress is consequently obtained as, $\sigma_x = E\epsilon_x$, and integration of the couples of the stress through the depth of the beam (in the z direction) yields the bending moment

$$M(x) = -EIw''(x) \tag{4.8}$$

(We dropped the designation x_C, as we now understand that the motion is only described at points on the midline.)

The balance equations may be read off from Figure 4.7. A differential element of the beam needs to be in balance in the transverse direction (sum of the forces), and in the rotation about the y axis (sum of the torques).

$$M' - V = 0, \quad V' + q = 0\ . \tag{4.9}$$

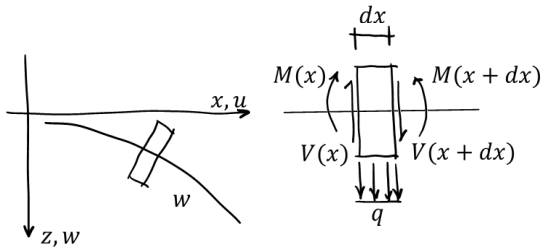


Fig. 4.7. Bernoulli beam. Balance of the differential element.

Since we are considering only prismatic beams, it is possible to obtain the following balance equation by successive substitution in the previous equations

$$EIw'''' - q = 0\ . \tag{4.10}$$

This is a single equation in a single variable. The dependent variable there is differentiated four times, which determines the character of the solution.

The balance equation (4.10) is based on the fourth derivative of the deflection: it is a fourth-order partial differential equation. In that it is rather different from the equations treated in [PK1], where all the balance equations were of second order. This for instance means that in one dimension both endpoints of the interval will need two boundary conditions. For the second-order heat conduction balance equation we had just one boundary condition per endpoint, such as temperature or heat flux.

Boundary conditions.

Next we introduce some boundary conditions. Figure 4.8 shows four combinations of boundary conditions at the right hand side end of the beam. Note that the balance equation is of fourth order for a single quantity (deflection), which necessitates two boundary conditions at each boundary point. In our case, the beam has two endpoints which constitute the boundary of the beam. The quantities that are prescribed are shown with a bar on top.

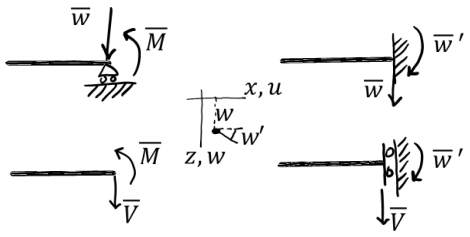


Fig. 4.8. Beam model. The four combinations of the boundary conditions. The barred quantities are prescribed, either zero or non-zero ($\overline{w}$: deflection, $\overline{w}'$: slope (rotation), $\overline{M}$: moment, $\overline{V}$: shear force).

4.4 Timoshenko beam bending model

The Timoshenko beam model is appropriate when the transverse shear forces can cause significant deformations due to the associated shear strains. For homogeneous solid-section beams, such as circular or rectangular cross-section members, the Bernoulli assumption is usually good enough when the ratio of length to depth is greater than 15. The Timoshenko beam can by applied to deeper beams: when the ratio of length to depth is greater than around 8. And, the build-up (and material composition) can make beam-like structures quite soft in shear (for instance, think sandwich beams), and in that case the Timoshenko beam model may be indicated in preference to the shear-rigid Bernoulli-Euler beam model.

Fig. 4.9. Timoshenko on a postage stamp (wikimedia).

Figure 4.10 shows the basic difference: Note that we still assume that the beams considered here are homogeneous and prismatic, that they will bend only in the plane $x - z$. However, the angle of rotation of the cross section is now independent of the deflection curve: ϕ is no longer kinematically tied to w'.

The balance equation for transverse planar deflection is still the same as for the Bernoulli beam, namely

$$M' - V = 0, \quad V' + q = 0\,. \tag{4.11}$$

However, the curvature of the beam is now $\mathrm{d}\phi/\mathrm{d}x = \phi'$ and so we write for the bending moment

$$M = EI\phi' \tag{4.12}$$

In addition to the axial strains, there is now the shear strain between the tangent to the midline and the cross section. We can write this as the difference between the (small) angles ϕ and $-w'$

$$\gamma = \phi - (-w') = \phi + w' \tag{4.13}$$

Therefore, we could now use a constitutive equation to write the uniform shear stress throughout the section as $\tau = G\gamma$. However, the overall shear force should not be simply $V = S\tau$: as shown in Figure 4.11, to maintain the cross section flat (i.e. with uniform shear strain) requires more force

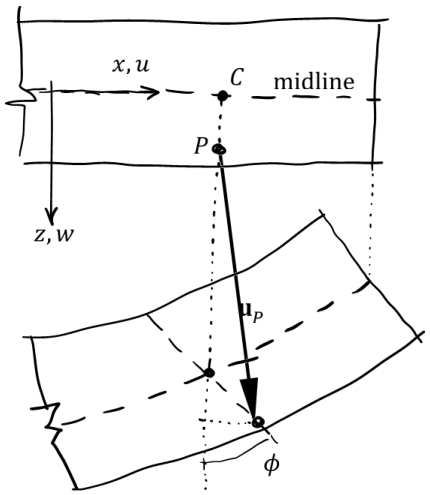


Fig. 4.10. Timoshenko beam. Displacements of point P. In contrast to the Bernoulli beam assumption, the rotation ϕ is independent of the shape of the deflection curve.

than when we allow the section to distort with nonuniform shear strain, and hence with parabolic shear stresses through the depth. In order to compensate for this lack of flexibility inherent in the flat-section assumption, we introduce the shear correction factor $k_s < 1$ and write

$$V = k_s S\tau = k_s GS\gamma = k_s GS(\phi + w') \tag{4.14}$$

(The shear correction factor can be derived by setting the work of the corrected uniform stresses equal to the work of the nonuniform stresses.)

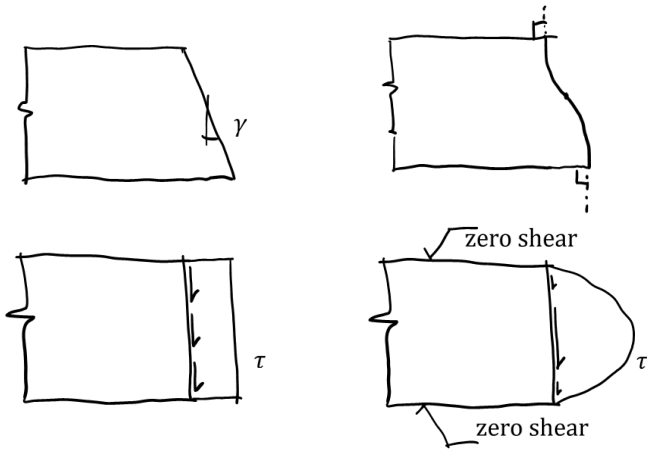


Fig. 4.11. Timoshenko beam. On the left, the uniform shear deformation $\gamma = \phi + w'$ (top left corner) implies a uniform shear stress throughout the depth shown (bottom left). On the right, the true parabolic distribution of the shear stress is produced by a deformation throughout the section that distorts the section so that the conditions of zero-shear are satisfied at the top and at the bottom surface of the beam (i.e. the angle between the section and the free surface remains the right angle). To achieve the deformation on the right requires less force than for the deformation shown on the left. The shear correction factor is intended to compensate for that.

Boundary conditions.

As opposed to the fourth-order equation of balance we obtained for the Bernoulli beam (4.10), we now obtain two second-order equations in two unknown functions ϕ and w (instead of one function, w, for the Bernoulli beam)

$$k_s GS(\phi' + w'') + q = 0 \,, \tag{4.15}$$

$$EI\phi'' - k_s GS(\phi + w') = 0 \,. \tag{4.16}$$

Thus, at each point of the boundary (if it is an interval on the real line, the endpoints are the boundary), we need two boundary conditions (Figure 4.8). This is the same number, four, as for the Bernoulli beam, but for different reasons: for the Bernoulli beam we had a fourth order equation for a single function – two boundary conditions at each point of the boundary. Here the equations are second order, but there are two of them. Hence, the same number of boundary conditions needed.

Recovery of the transverse shear stress.

The transverse shear stress is not uniformly distributed through the thickness, as discussed above. Here we shall briefly sketch how to recover the (more) accurate transverse shear distribution. For simplicity we will focus on a rectangular cross section. This kind of cross section is important not only for beams, but also when discussing plates and shells.

Refer to Figure 4.12. The balance of the differential element of the beam can be written down as

$$\frac{\partial \sigma_x}{\partial x} + \frac{\partial \tau}{\partial z} = 0 \,. \tag{4.17}$$

Here we now use

$$\frac{\partial \sigma_x}{\partial x} = \frac{\partial M}{\partial x}\frac{z}{I} = F\frac{z}{I} \tag{4.18}$$

and we integrate the resulting expression over the area (rectangle) $\overline{S}$ to recover

$$\int_{\overline{S}} \frac{\partial \tau}{\partial z} \mathrm{d}\overline{S} = \int_{\overline{S}} \frac{\partial \tau}{\partial z} b \mathrm{d}z = [\tau]_z^{h/2} = \tau(z) b \tag{4.19}$$

were we assumed that the stress τ was uniform along the width of the section, and

$$\int_{\overline{S}} \frac{\partial \sigma_x}{\partial x} \mathrm{d}\overline{S} = \int_{\overline{S}} F\frac{z}{I} \mathrm{d}\overline{S} = \int_{\overline{S}} F\frac{z}{I} b \mathrm{d}z = \frac{F}{I}\int_{\overline{S}} bz \mathrm{d}z = \frac{F\overline{S}\overline{z}}{I} \tag{4.20}$$

Here $\overline{z}(z) = (h/2 + z)/2$ is the coordinate of the centroid of the rectangle $\overline{S}$, whose area is $\overline{S}(z) = b(h/2 - z)$. Substituting the above two results into (4.17) yields

$$\tau(z) = \frac{F\overline{S}(z)\overline{z}(z)}{bI} \tag{4.21}$$

where we express the dependence on the z coordinate explicitly (known as the Zhuravskii formula). Thus we arrive at

$$\tau(z) = \frac{Fb(h^2/4 - z^2)}{b\frac{1}{12}bh^3} = \frac{F}{bh}\frac{3}{2}\left(1 - \frac{4z^2}{h^2}\right) \,, \tag{4.22}$$

which is the well known result that the transverse shear is distributed parabolically, and its maximum is (3/2) times the uniformly distributed shear stress $F/(bh)$.

Of course, in general cross sections we cannot really assume that the shear stress is uniform along the width cut. Figure 4.13 illustrates that even if the resultant shear force is parallel to z, the shear tractions corresponding to this shear force must at least be tangent to the contour of the cross section (assuming the cylindrical surface is free of shear).

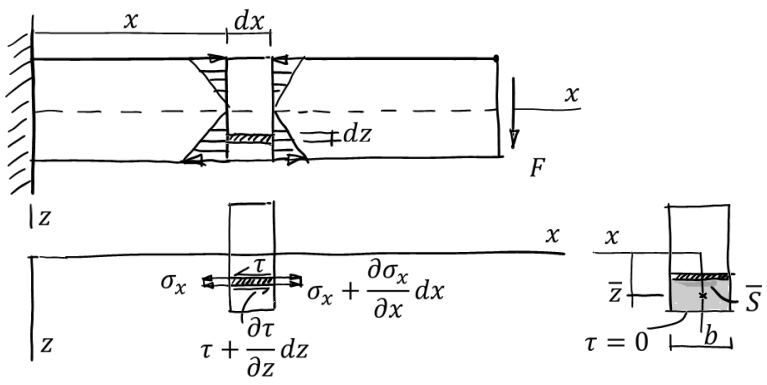


Fig. 4.12. Beam with known shear force and moment. The balance of a differential element and the partial area of the cross section and its centroid are shown.

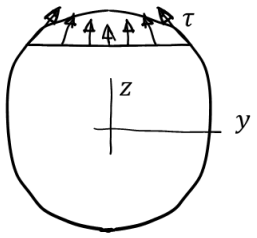


Fig. 4.13. Solid beam profile loaded by a shear force in the z direction. General distribution of shear traction along a width cut: the traction vector needs to be tangent to the contour of the cross section, and it will not be uniformly distributed along the cut.

4.5 Rod torsion model

The subject of the torsion of rods with solid cross section was thoroughly addressed by Saint-Vénant post mid-nineteenth century. A drawing from his experiments with rubber beams is shown in Figure 4.14. Clearly, the assumption of the bending theory that cross sections remain plane throughout deformation needs to be abandoned. In its place comes the warping of the plane of the cross section of the rod. We shall look here at the so called uniform torsion: the rate of twist is constant along the rod. Figure 4.15 shows the forward-facing cross section of the rod to be torqued by the angle ϕ relative to the stationary backward-facing cross section. The rate of twisting ϕ' is uniform along the bar, meaning that one unit length of the rod twists by $\partial\phi/\partial x = \phi' = \phi/L$. The rod is twisted about the axis passing through the point SC, located at (SC_y, SC_z) relative to the center of gravity (centroid) of the cross section.

The observations above lead to the following assumptions about the stresses involved in uniform torsion:

$$\sigma_x = 0\,, \quad \sigma_y = 0\,, \quad \sigma_z = 0\,, \quad \sigma_{yz} = 0\,. \tag{4.23}$$

These assumptions may be inserted into the differential balance equations of 3D elasticity

$$\partial\sigma_{xy}/\partial y + \partial\sigma_{xz}/\partial z = 0\,, \quad \partial\sigma_{xy}/\partial x = 0\,, \quad \partial\sigma_{xz}/\partial x = 0\,, \tag{4.24}$$

4.5.1 Prandtl Stress function

The last two equations simply say that indeed the torsion is uniform along the length of the rod, and the first equation can be addressed using the so called Airy stress function. Prandtl was the first to use this function to write

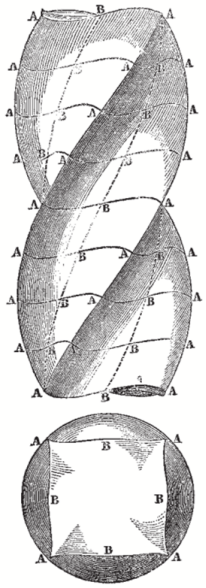


Fig. 4.14. Saint-Vénant's experiments on rubber beams.

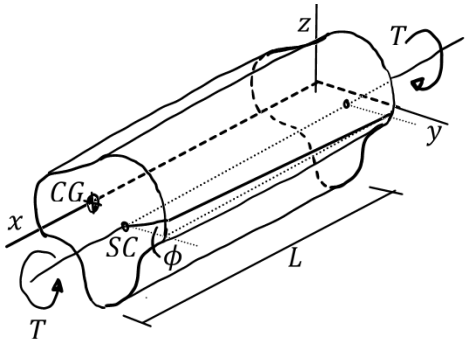


Fig. 4.15. Rod of a solid cross section subject to a torsional moment T. The rotation of the cross sections is about the axis passing through the point SC.

$$\sigma_{xy} = \frac{\partial \Psi(y,z)}{\partial z} \ , \quad \sigma_{xz} = -\frac{\partial \Psi(y,z)}{\partial y} \ . \tag{4.25}$$

This guess satisfies the first balance equation (4.24) identically

$$\frac{\partial^2 \Psi(y,z)}{\partial z \partial y} - \frac{\partial^2 \Psi(y,z)}{\partial y \partial z} = 0 \ . \tag{4.26}$$

The price to pay now is to figure out what the function Ψ is.

We need to link the stresses to the deformation. Expressing the displacements in terms of the uniform twisting reads

$$u = \omega(y,z)\phi' \ , \quad v = -x\phi'(z - SC_z) \ , \quad w = +x\phi'(y - SC_y) \ , \tag{4.27}$$

Here we introduced a function, $\omega(y,z)$, so-far unknown (the warping function). Note that it does not depend on the axial coordinate, x. Correspondingly, the strains follow

$$\epsilon_{xy} = \frac{\partial \omega(y,z)}{\partial y}\phi' - \phi'(z - SC_z) = \left[\frac{\partial \omega(y,z)}{\partial y} - (z - SC_z)\right]\phi' \tag{4.28}$$

$$\epsilon_{xz} = \frac{\partial \omega(y,z)}{\partial z}\phi' + \phi'(y - SC_y) = \left[\frac{\partial \omega(y,z)}{\partial z} + (y - SC_y)\right]\phi' \tag{4.29}$$

Therefore, the stresses corresponding to shearing σ_{xy}, σ_{xz} may be written as

$$\sigma_{xy} = \left[\frac{\partial \omega(y,z)}{\partial y} - (z - SC_z)\right] G\phi' \tag{4.30}$$

$$\sigma_{xz} = \left[\frac{\partial \omega(y,z)}{\partial z} + (y - SC_y)\right] G\phi' \tag{4.31}$$

Matching the stresses from (4.25) with (4.30), we have

$$\frac{\partial \Psi(y,z)}{\partial z} = \left[\frac{\partial \omega(y,z)}{\partial y} - (z - SC_z)\right] G\phi' \tag{4.32}$$

$$-\frac{\partial \Psi(y,z)}{\partial y} = \left[\frac{\partial \omega(y,z)}{\partial z} + (y - SC_y)\right] G\phi' \tag{4.33}$$

What follows will feel like a trick: we first differentiate the first of these relations with respect to z, and the second with respect to y,

$$\frac{\partial^2 \Psi(y,z)}{\partial z^2} = \left(\frac{\partial^2 \omega(y,z)}{\partial y \partial z} - 1\right) G\phi' \tag{4.34}$$

$$-\frac{\partial^2 \Psi(y,z)}{\partial y^2} = \left(\frac{\partial^2 \omega(y,z)}{\partial y \partial z} + 1\right) G\phi' \tag{4.35}$$

and then we subtract the first from the second

$$-\frac{\partial^2 \Psi(y,z)}{\partial z^2} - \frac{\partial^2 \Psi(y,z)}{\partial y^2} = 2G\phi' \tag{4.36}$$

This is precisely the same equation we have been solving in the FEA book [PK1] for the heat conduction problem, with thermal conductivity set equal to 1, and the heat generation rate equal to $2G\phi'$ (Poisson's equation).

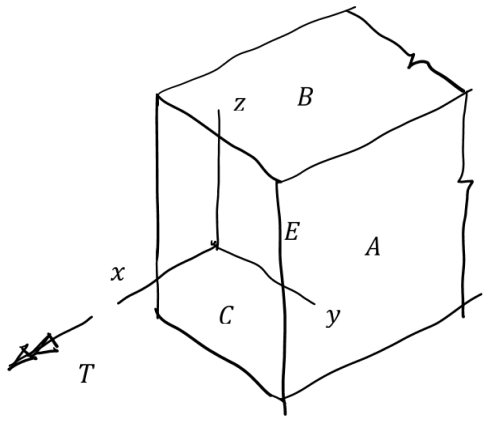


Fig. 4.16. Rod of a rectangular cross section subject to a torsional moment.

The balance equation (4.47) for Ψ is thus available, but in order to obtain a solution we need some boundary condition on the circumference of the cross section. Consider a rod with a rectangular cross section subject to torsion by an externally applied torque T (Figure 4.16). The two faces A and B are traction-free. For the face A this means that the shear stresses σ_{xy} and σ_{yz} must be zero because they are directly equal to the shear components of the traction vector (and that is by assumption zero). For the face B the traction-free condition means that the shear stresses σ_{xz} and σ_{yz} must be zero.

Therefore, we must conclude that along the edge E the shear stress σ_{xy} must be zero, because this stress component is zero in the adjacent face A. We can make similar observations for the other three edges of the face C: σ_{xy} must be zero along the edges parallel to the z coordinate, and σ_{xz} must be zero along the edges parallel to the y coordinate.

Conversely, we can see that the two edges of the face C parallel to the z coordinate allow for the σ_{xz} to be non-zero, and the two edges of the face C parallel to the y coordinate allow for the σ_{xy} to be non-zero.

These observations are generalized in Figure 4.17. The shear traction vector in the interior of the cross section of the rod is composed of the two stress components σ_{xy} and σ_{xz}. However, when we approach a point on the circumference of the cross section, Q, we should express this fact in a special coordinate system, such that $\overline{y}$ is orthogonal to the circumference. In this coordinate system, a traction vector must only consist of $\sigma_{x\overline{z}}$, because the prismatic surface of the road is traction free and therefore $\sigma_{x\overline{y}} = 0$. Using (4.25) results in

$$\frac{\partial \Psi(y_Q, z_Q)}{\partial \overline{z}} = 0 \tag{4.37}$$

where $\overline{z}$ is always tangent to the contour of the cross section.

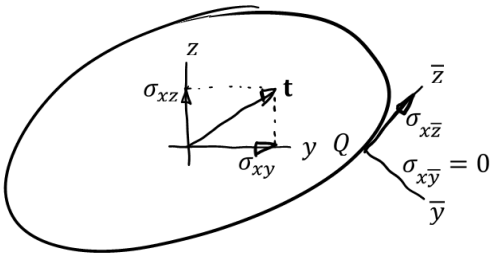


Fig. 4.17. Shear stresses due to a torsional moment. A special coordinate system is used along the circumference.

In words, Ψ is constant along the circumference of the cross section. Since stresses only depend on the derivatives of this function, not its value, we can take $\Psi = 0$ along the circumference. So we solve (4.47) with this boundary condition. Once we have the function, we can differentiate it with respect to the coordinates in the plane of the cross section to obtain the shear stresses.

4.5.2 Membrane analogy

There is a great analogy due to Prandtl: the solutions to the problem of torsion have been in the past solved in the lab. A wire loop was dipped in soap water, and bubble was blown with constant pressure underneath the soap water membrane. The slope of this membrane could be measured, directly delivering the stresses needed. The reason this works is the fact that the deflection of the membrane can be described with the same equation as (4.47).

The contour lines (level curves) of the function Ψ are the lines to which the shear stresses are tangent (lines of "shear stress flow"). The magnitude of the shear stress is given by the gradient of the function Ψ: The largest shear stress is found where the level curves are close (bunched up) together. On the other hand, zero shear stress can be found in "flat" places: In this case, the center of the rectangle, and its corners.

4.5.3 Torsion constant (torsional rigidity)

Solving for the stress function means finding Ψ such that (4.47) is satisfied and $\Psi = 0$ on the boundary of the cross section. That is easily accomplished with the finite element method if a mesh

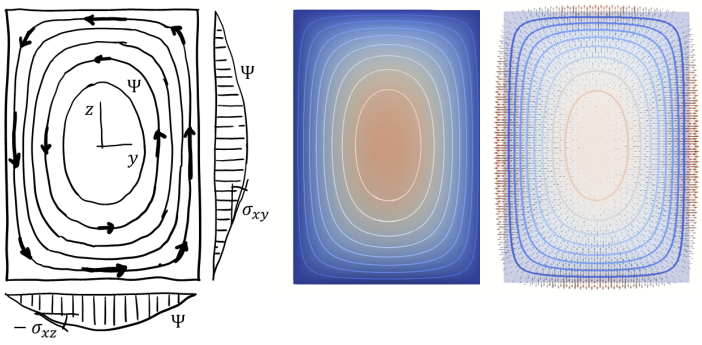


Fig. 4.18. Rod of a rectangular cross section subject to a torsional moment. Lines of shear stress flow, with a few arrows indicating the direction and magnitude of the shear stresses. Stress magnitude is derived by taking the slope of the cuts through the surface at the center of the section. On the right are two images that come from a simulation of the heat-conduction equation, the temperature and the heat flux. The arrows representing the stress come from the heat flux arrows by rotating them about the normal to the screen by 90^o.

of the cross section is available; for some simple shapes such as rectangles or ellipsoids, approximate or exact analytical techniques exist. Now that we have a solution for the stress function Ψ, we can calculate the total applied torque from the shear stress components

$$T = -\int_S \left[(y - SC_y)\sigma_{xz} - (z - SC_z)\sigma_{xy}\right] \, \mathrm{d}S \tag{4.38}$$

$$= -\int_S \left(y\frac{\partial \Psi}{\partial y} + z\frac{\partial \Psi}{\partial z}\right) \, \mathrm{d}S + \int_S SC_y\sigma_{xz} \, \mathrm{d}S - \int_S SC_z\sigma_{xy} \, \mathrm{d}S \tag{4.39}$$

The two expressions $\int_S SC_y\sigma_{xz} \, \mathrm{d}S = SC_y \int_S \sigma_{xz} \, \mathrm{d}S$ and $-\int_S SC_z\sigma_{xy} \, \mathrm{d}S = -SC_z \int_S \sigma_{xy} \, \mathrm{d}S$ are easily identified as the shear forces $\int_S \sigma_{xz} \, \mathrm{d}S$ and $\int_S \sigma_{xy} \, \mathrm{d}S$ producing torque on the arms SC_y and $-SC_z$. But, if the stresses really come from torsion of the rod, then

$$SC_y \int_S \sigma_{xz} \, \mathrm{d}S = SC_y \int_S \frac{\partial \Psi}{\partial y} \, \mathrm{d}S = SC_y \times 0 = 0 \tag{4.40}$$

and

$$-SC_z \int_S \sigma_{xy} \, \mathrm{d}S = SC_z \int_S \frac{\partial \Psi}{\partial z} \, \mathrm{d}S = SC_z \times 0 = 0 \tag{4.41}$$

which, after integration by parts, follows from $\Psi = 0$ on the boundary of the cross section. In other words, if the stresses come from torsion, the shear forces are zero.

Now we have for the torque

$$T = -\int_S \left(y\frac{\partial \Psi}{\partial y} + z\frac{\partial \Psi}{\partial z}\right) \, \mathrm{d}S \tag{4.42}$$

$$= -\left[y\Psi\right]_{\partial S} - \left[z\Psi\right]_{\partial S} + 2\int_S \Psi \, \mathrm{d}S \tag{4.43}$$

To obtain the last line we used integration by parts. The boundary integrals dropped out because $\Psi = 0$ on the boundary ∂S. We are left with

$$T = 2\int_S \Psi \, \mathrm{d}S \tag{4.44}$$

which we can rewrite

$$T = GJ\phi' \, , \tag{4.45}$$

where we defined the torsional constant J as

$$J = \frac{2}{G\phi'} \int_S \Psi \, \mathrm{d}S \tag{4.46}$$

This can be also simplified by introducing a scaled stress function $\widehat{\Psi} = \Psi/(G\phi')$, which we can solve for from the equation

$$-\frac{\partial^2 \widehat{\Psi}(y,z)}{\partial z^2} - \frac{\partial^2 \widehat{\Psi}(y,z)}{\partial y^2} = 2 \,, \quad \widehat{\Psi} = 0 \text{ on } \partial S \tag{4.47}$$

Then

$$J = 2 \int_S \widehat{\Psi} \, \mathrm{d}S \tag{4.48}$$

where J is now purely the property of the shape of the cross section.

The equation of motion of the rod is easily formulated by stating that the torque transmitted by the rod is uniform along the length

$$\frac{\partial T}{\partial x} = 0 \tag{4.49}$$

Substituting from (4.45), we obtain

$$GJ \frac{\partial^2 \phi}{\partial x^2} = 0 \tag{4.50}$$

This is a balance equation of the second order, which requires two boundary conditions, one at each end of the rod, either a prescribed rotation, or a prescribed torque. Very similar to the axial bar!

In the above, we have covered torsion of prismatic rods of solid cross section (first row in Figure 4.19). Of course, we realize that this does not exhaust the complexity of structures that can be modelled as twisted rods: for instance, thin-walled rods are quite common (the second row in Figure 4.19). Special treatment is required, but that will not be addressed in this section, but rather later in the book.

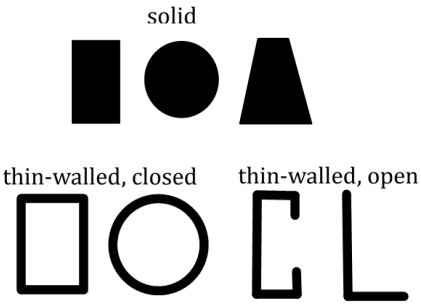


Fig. 4.19. Types of sections.

4.5.4 Shear center and twist center

Imagine now a beam is loaded with transverse shear forces. F_y applied at the point SC produces the shear stresses $\sigma_{xy}(F_y)$ and $\sigma_{xz}(F_y)$ such that the total torque applied on the rod relative to the centroid of the cross section CG may be written as

$$T_{CG} = -F_y SC_z = \int_S \left[y\sigma_{xz}(F_y) - z\sigma_{xy}(F_y) \right] \, \mathrm{d}S \tag{4.51}$$

We use the notation $\sigma_{xy}(F_y)$ and $\sigma_{xz}(F_y)$ to explicitly state that those stresses are computed for the applied force F_y. The equation above yields readily the coordinate SC_z. Analogously, SC_y may be solved for from

$$T_{CG} = F_z SC_y = \int_S [y\sigma_{xz}(F_z) - z\sigma_{xy}(F_z)] \, \mathrm{d}S \tag{4.52}$$

The coordinates SC_y and SC_z give the location of the so called shear center. Note that shear force applied at this point generates zero torque about the axis passing through this point. The shear center is thus identical to the so called twist center. The beam loaded by force passing through the shear center axis will not experience twist (the torque relative to the shear center is zero). Simultaneously, twisting the beam about the twist center will leave the shear centre stationary.

The shear center and the twist center coincide.

5

Modeling considerations with beams

Summary

1. The Bernoulli beam 2D finite element with both axial and transverse (bending) action can be used in the analysis of planar frames.
2. A number of considerations need to be taken into account for practical modelling, and the goal of this chapter is to talk of orientation, section forces, offsets, computation of the resultants, and more.

A word about terminology: In practice, when one says "beam", it is more often than not the case that they mean beam-column or frame. Analytical elements with solely bending action are rarely used.

5.1 Orientation

All beams must be assigned a section and an orientation. The orientation essentially aligns the section in space so that its normal is the tangent vector to the reference line. Then the second vector, $\mathbf{n}_1$, is defined. In two dimensional models in Abaqus, $\mathbf{n}_1$ always points into the plane of the screen. In practice, for three-dimensional beams in Abaqus we choose an approximate direction of the vector $\mathbf{n}_1$, and the software then completes an orthogonal triad because it knows the direction of the tangent vector to the reference line, $\mathbf{t}$.

5.2 Section stresses and forces

The resultant section forces and section moments for a general three dimensional beam are shown in Figure 5.1. In three dimensions the shear force nomenclature (SF3 pointing along $\mathbf{n}_1$, while SF2 points along $\mathbf{n}_2$) is a bit peculiar, but the bending moments are consistent with the naming in shells, as is the twisting moment. The shear force naming was probably chosen so that it also made sense in planar models. In a two dimensional (planar) beam the three resultants are the axial force SF1, the shear force SF2, and the bending moment SM1 about the axis pointing into the screen ($\mathbf{n}_1$).

The section forces (and moments) correspond to the stress components on the $\mathbf{n}_1$, $\mathbf{n}_2$, $\mathbf{t}$ basis vectors as: S11 is the axial stress component, S12 is the shear a stress component along $\mathbf{n}_2$, S13 is the shear a stress component along $\mathbf{n}_1$ (refer to Figure 5.2). In the definitions shown below, A refers to the cross sectional area, and the coordinates in the plane of the section are named S^1 and S^2 (that is the nomenclature of the Abaqus manual). The bending moments are defined relative to the center of gravity of the cross section; the twisting moment is defined relative to the origin of the cross section axes.

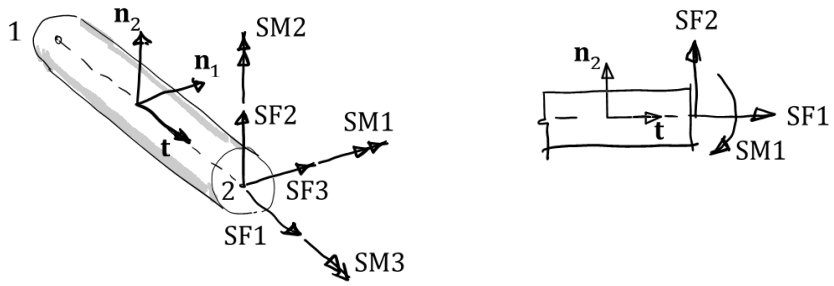


Fig. 5.1. Abaqus beam section forces and moments. three-dimensional beam on the left, planar beam on the right.

$$\mathrm{SF1} = \int_A \mathrm{S11}\,\mathrm{d}A \tag{5.1}$$

$$\mathrm{SF2} = \int_A \mathrm{S12}\,\mathrm{d}A \tag{5.2}$$

$$\mathrm{SF3} = \int_A \mathrm{S13}\,\mathrm{d}A \tag{5.3}$$

$$\mathrm{SM1} = \int_A \mathrm{S11}(S^2 - S_c^2)\,\mathrm{d}A \tag{5.4}$$

$$\mathrm{SM2} = -\int_A \mathrm{S11}(S^1 - S_c^1)\,\mathrm{d}A \tag{5.5}$$

$$\mathrm{SM3} = \int_A \mathrm{S12}\,S^1 - \mathrm{S13}\,S^2\,\mathrm{d}A \tag{5.6}$$

$$\tag{5.7}$$

In order to visualize stresses in beams conveniently, there is a special field output for beams,

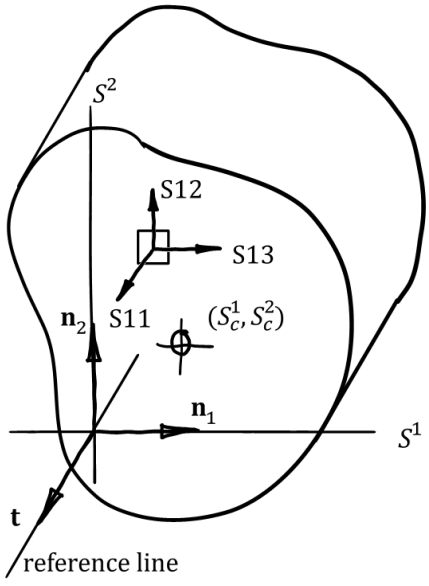


Fig. 5.2. Abaqus beam stress components on the section-attached coordinate system.

"BEAM_STRESS": it is not available for all cross section types, only the main ones (rectangular, circular, I, ...).

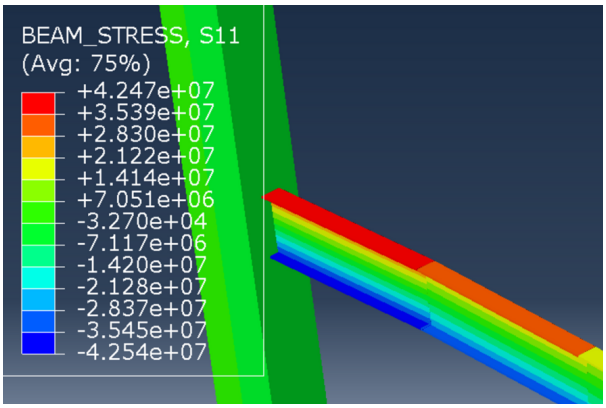


Fig. 5.3. Abaqus "BEAM_STRESS" S11 component visualized on an I beam member.

5.3 Curved beams

The directions of the vectors $\mathbf{n}_1$ and $\mathbf{n}_2$ can be used to describe beam elements with initial curvature: The quadratic B32 and B22 shear-flexible elements and the cubic beams B33 can represent members with initial curvature, both bending and twisting. An unexpected (perhaps?) consequence of initial curvature is the coupling of extension of the member with its bending. This requires significant sophistication in the design of the finite elements for this kind of situation, as a naive implementation would result in elements that are way too stiff to be useful.

5.4 Offsets

It is very important to learn to work with offsets: The beam profiles come with at least three special points (locations), as shown in Figure 5.2:

1. There is the centroid of the profile at the coordinates (S_c^1, S_c^2).
2. There is the center of shear (and of twist: these two coincide); this point is not shown in Figure 5.2.
3. Finally, there is the origin of the coordinate axes referred to as "1" and "2". Notice that there is no requirement that this origin coincide with either one of the first two points. The reference curve of the beam passes through this point.

Further, the directions of the two coordinate axes "1" and "2" do not have to coincide with the principal axes of the cross section (at least, that holds true in three dimensions).

Choosing the location of the reference curve has important implications: the nodes are located along this curve when the beam is meshed. Therefore, loading applied at nodes (vertices), may or may not result in the expected response of the beam: Say the reference curve does not pass through the center of shear, then a transverse force applied to the nodes of the beam will generate torsion in the beam. Another example: the reference curve does not pass through the centroid, and the orientation of the beam (the two coordinate axes "1" and "2") is not aligned with the principal axes of the profile. Transverse loading will then generate bending in two planes.

The center of shear and the two coordinate axes "1" and "2" being the principal axes seems to be cleanest choice: when we do hand calculations, we would be tempted to make it so. Unfortunately,

that doesn't mean that that will be the most convenient choice for modeling the structure: for instance, an L profile used as a stringer on a curved shell will have a complicated orientation as it follows the shape.

5.5 Nonuniform cross section

Beams whose cross sectional properties vary from point to point along the length are not uncommon in practice. Sometimes this effect can be reproduced to good accuracy with multiple prismatic elements of different cross sections employed alone a single member. However, Abaqus does provide tapered beam finite elements with the "general" section definition.

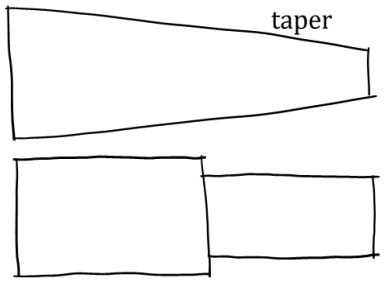


Fig. 5.4. Tapered beams can be at times modeled as a series of prismatic elements.

Taper of the beam cross section sometimes requires that the reference curve be considered inclined: Figure 5.5. Note that the inclined centerline means that nominally transverse loading will generate an axial force in the beam. Refer also to Figure 5.6 with a considered incline: Figure 5.5. Note that this haunched beam is a practical example of design which could take advantage of a quasi-arch action, provided the sideways force reactions could be accommodated.

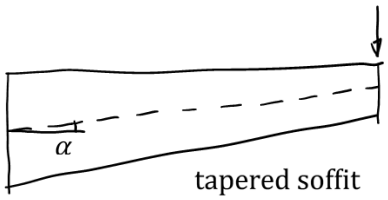


Fig. 5.5. Tapered beam with inclined soffit.

5.6 Joints

Figures 5.7 to 5.9 illustrate the need for a library of devices for connecting nodes (joints) in various ways. Figure 5.7 shows the correct way of connecting a column to a beam, compensating for the distance between the center line of the beam and the endpoint of the column with a rigid connection. Figure 5.8 shows an analogous situation, except that the connection must now allow for differential rotation between the endpoint of the beam and the column, due to the flexibility of the web cleat. Figure 5.9 demonstrates how a complicated joint, with the center lines of the members intersecting at more than one point, may be realized computationally with general rigid links.

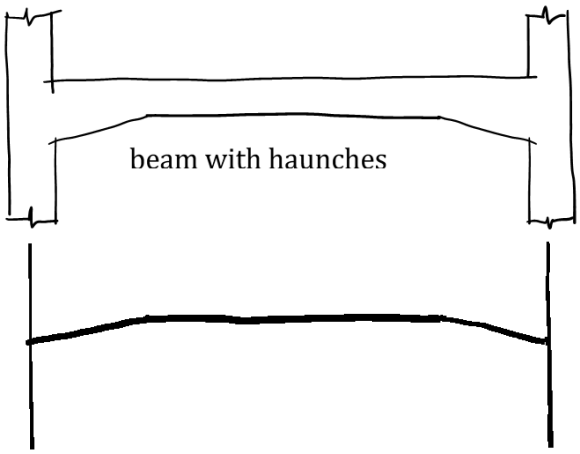


Fig. 5.6. Taper of the beam cross section often means that the reference curve should be considered sloped. In this case, taper creates a form of arch effect.

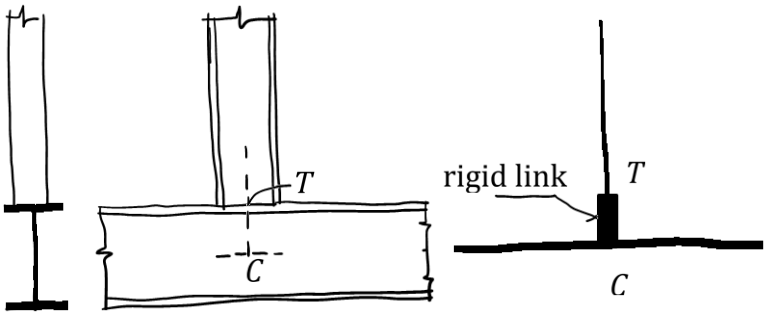


Fig. 5.7. Rigid link used to connect beam with column: column is attached to the flange of the beam.

Connectors.

Abaqus provides a library of connectors that can be used to insert simple or complicated constraints between degrees of freedom. Here we mention the multi-point constraint (MPC) types.

When we wish to tie together two nodes placed at the same location, we have two choices: the pin constraint makes the displacements of the two nodes identical, but each node can rotate independently; the tie constraint makes the displacements and rotations of the two nodes exactly the same. Refer to Figure 5.10. The "Coincident Builder" in the interaction module of Abaqus can be used to create these types of connections efficiently.

When we wish to tie together two nodes placed at different locations, we have the following two choices: the "beam" constraint makes the displacements of the two nodes such as would be expected if those two nodes were part of a single rigid body; the "link" constraint makes the displacements such that the distance between the nodes is fixed, but the two nodes can rotate independently of each other. Refer to Figure 5.11.

Wire features.

The rigid links are represented in the model as wire features (lines), which are endowed with connector sections representing the desired connection MPC type (beam, link, ...). These wire features can be created in the Interaction module. Additional points which are not on any of the geometrical features already present can be created as Attachment points. As an example, consider the need to

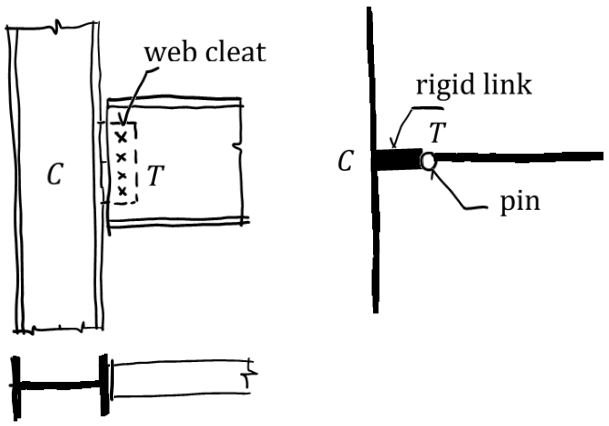


Fig. 5.8. Rigid link used in conjunction with a pin connection: beam is attached to the flange of the beam through a web cleat whose capability to transfer moments is limited. The connection is modelled as pinned, but the offset of the cleat from the center of the column is taken into account with a rigid link.

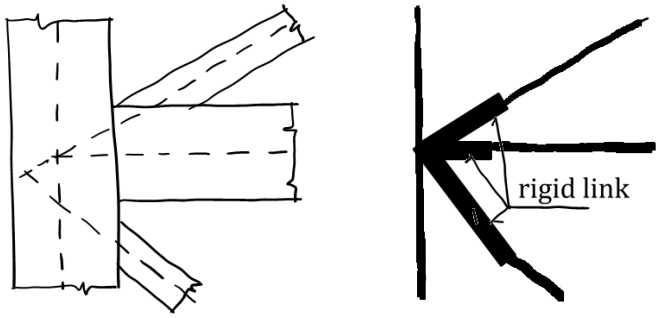


Fig. 5.9. Rigid links used to model a complicated joint with diagonals.

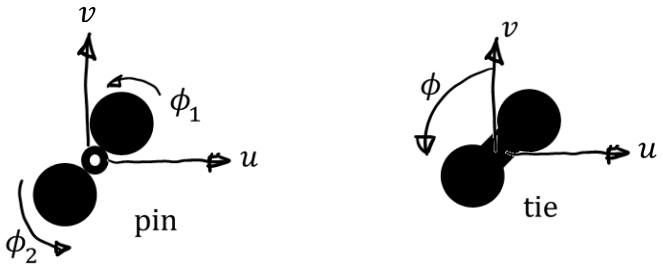


Fig. 5.10. Multi-point constraints (MPC) applicable to nodes at the same location. Types "pin" and "tie". The large dark dots are the nodes. They are really at the same location, but the drawing separates them for clarity.

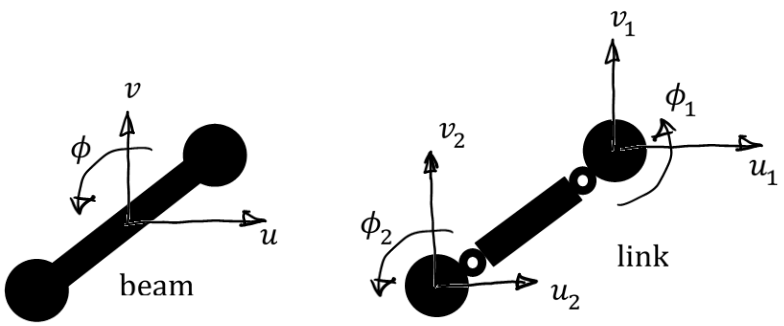


Fig. 5.11. Multi-point constraints (MPC) applicable to nodes at different locations. The large dark dots are the nodes. Types "beam" and "link".

introduce an eccentric force into a horizontal beam (Figure 5.12). The force needs to be applied at P. The centre line of the beam gives us the point C. The distance between these two points needs to be spanned by a rigid link to introduce the loading at P into the beam at C. We can introduce the point P as an attachment point, and then we create a wire feature between P and C.

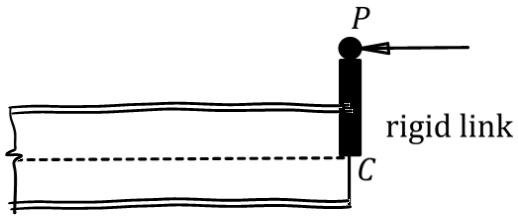


Fig. 5.12. Force applied at P (eccentric force) introduced into the beam at C using a rigid link (wire feature) between the end of the beam at C and the attachment point P.

5.7 Coupling of beams to shells

Beams can be used as stiffeners (spars and ribs) and edge beams. Beams can be tied to a shell using tie constraints, but attention needs to be paid to the relationship between the reference lines of beams and the reference surface of the shell: the nodes of both are located on these entities. Surface-Based Tie Constraint forces each of the nodes on the *secondary* surface to have the same motion and the same value of displacement as the point on the *main* surface to which it is closest. Abaqus forms constraints between the secondary nodes and the nodes on the main surface by determining the tie coefficients that are used to interpolate quantities from the main nodes to the tie point.

Abaqus can use one of two approaches to generate the coefficients: the "surface-to-surface" approach (preferable) or the "node-to-surface" approach (usually used when the first approach does not work). The "surface-to-surface" approach minimizes numerical noise for tied interfaces involving mismatched meshes. The surface-to-surface approach enforces constraints in an average sense over a finite region, rather than at discrete points as in the traditional node-to-surface approach.

5.8 Resultant forces and moments in beams

Classical linear structural analysis takes advantage of the principle of superposition to produce accurate internal resultant force and moment diagrams. The procedure can be explained as shown

in Figure 5.13. Figure 5.13(a) shows the original frame structure, with a distributed loading at the top horizontal member. Now we imagine that we prevent the joints of that member from moving (zero deflections and zero rotation), as indicated in Figure 5.13(b) by the clamping symbols. The member's joints cannot move, but the member can still deform under the applied distributed loading. Therefore, it will experience internal forces and moments. These forces are balanced by the reactions shown in Figure 5.13(c). In other words, these reactions make the joints of the horizontal member immovable – these are the fixed-end forces. Now we convert these reactions to applied forces by simply flipping their sign, $F_k = -R_k$, and the structure is loaded by these forces (and not the distributed loading anymore).

And at this point we put all these observations together: The original structure in Figure 5.13(a) is equivalent to the superposition of Figure 5.13(c) plus Figure 5.13(d). As we can see, each R_k in (c) is neutralized by its corresponding F_k from (d), and what is left is then just the original structure with the distributed loading at the top.

So what we do is:

1. Solve the problem for each loaded member with its joints fixed: The resulting reaction forces at the joints are converted to applied loading.
2. Solve the finite element problem for the entire structure, loaded by the forces at the joints, F_k.
3. Superimpose these two results.

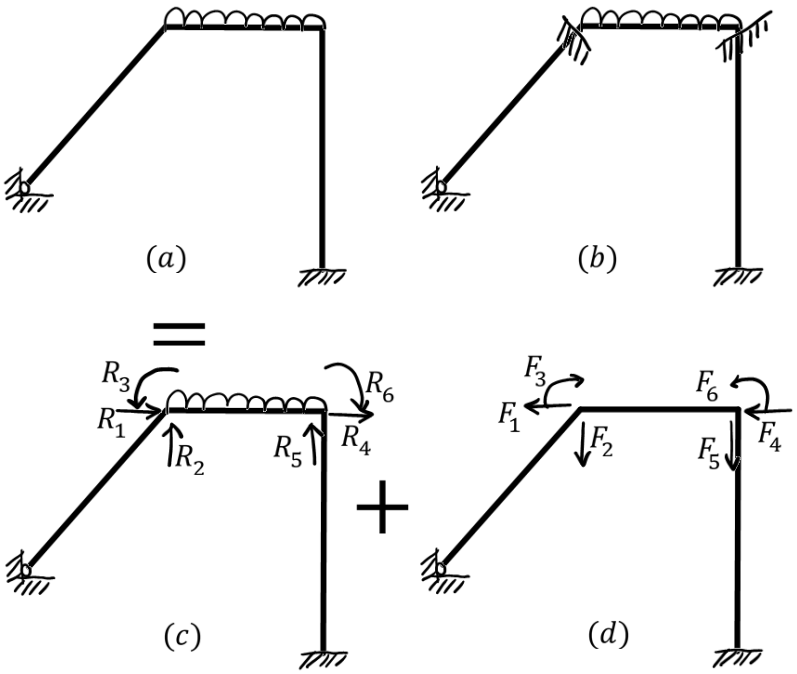


Fig. 5.13. Illustration of the superposition of the structure with distributed loading and fixed joints and the structure loaded at joints.

Abaqus does not use the superposition described in this section to report the resultant forces and moments. It only reports the results for the structure loaded at joints.

Now, Abaqus does not use this principle. As a principle it is only useful for finite elements that are loaded in their interiors, and for which we can determine the response to the loading while their nodes (boundaries) are fixed. This is really only useful for beams (or possibly plates), as most

common elements are incapable of deforming in response to an interior loading, or we simply have no way of solving for the way in which these elements would respond to such loading. In any case

- The superposition is only available for linear static solutions; it is not applicable to dynamics, buckling, or other general procedures.
- We have to be aware that with beam structures we will need more than one element per member when they are loaded with distributed loading. A single finite element per member will not be enough, since the distributed loading will have been converted to joint forces, and the deformation and internal forces due to the distributed loading will not be part of the solution we get from the finite element program.
- If a beam member is loaded with a concentrated force or moment in its interior, it should be broken up into two or more elements with joints inserted at the point of application of these concentrated forces or moments.
- In any case, when we solve the problem only with the loading extracted from the fixed-end forces (i.e. when the structure is loaded with forces at of the nodes consistently derived from the distributed loading in the interior of the elements), the internal resultants will converge with refinement to the correct solution of the original problem.

6

Shell finite elements

Summary

1. The finite element formulations of shell finite elements are involved. We will not get deep into the details here.
2. We will touch upon some properties of finite elements present in Abaqus in an effort to separate what is on offer into "use classes".
3. We will provide a list of desirable characteristics of shell finite elements.

6.1 Tentative classification

In Abaqus the shell elements are classified into two larger groups, conventional shell elements, and continuum shell elements. Figure 6.1 shows with a cartoon the main characteristics of the two classes: conventional shells look like surfaces, continuum shells look like solid (continuum) elements.

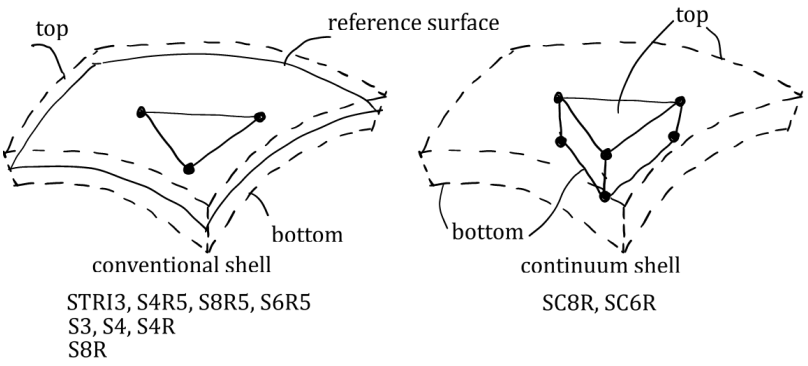


Fig. 6.1. Cartoon of conventional versus continuum shell finite elements. Conventional shells have in general six degrees of freedom per node, three deflections and three rotations. (There may be exceptions: S4, S4R, S3 have only two rotations per node, in the tangent plane to the shell.) Continuum shells have only the deflections at each node.

Further, Figure 6.2 delves into the differences in the degrees of freedom that the element formulation operates in: conventional shells have both translations and rotations as degrees of freedom; continuum shells only operate with translation (deflection) degrees of freedom. In that they are exactly like the solid elements. On the other hand, conventional shells are like beams: both have deflections and rotations as degrees of freedom at the nodes.

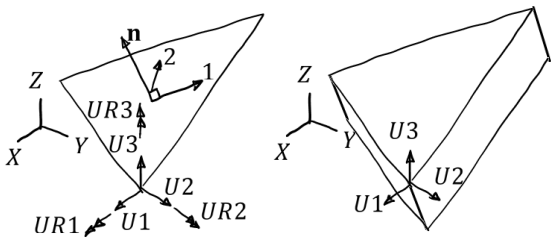


Fig. 6.2. The degrees of freedom of conventional versus continuum shell finite elements. Conventional shell finite elements have three displacements and three rotations at each node (like beams). Continuum shell finite elements have three displacements at each node (like continuum solid elements).

There are other sides to the classification question: for instance, there is a distinction between specialist elements and general-purpose elements. Specialist finite elements can be very effective in their own domain (elements only for thin shells), but they may fail entirely when used outside of that domain (thin-shell specialists should not be used for thick shells, for instance). Finite elements which are general-purpose on the other hand may trade off their wide range of applicability for not being best-in-class anywhere.

6.2 Desirable characteristics of shell finite elements

The list of features that can make or break a finite element model that uses shells is long, and some demands may be contradictory:

1. Represent thick (shear-flexible) structures.
2. Represent very thin structures.
3. Report stress resultants and stress.
4. Handle curved shells.
5. Define offset between the midsurface and the reference surface.
6. Connect to stiffeners and edge beams.
7. Require minimum user input.
8. Do not rely on user-specified parameters.
9. Connect to solids.
10. Represent layered structures.
11. Represent branched shells.
12. Handle dynamics and buckling.
13. Model contact.
14. Be a general tool.
15. Be convergent.
16. Be robust.
17. Be economical.

6.2.1 Represent thick (shear-flexible) structures

Representing very thick shell-like (plate-like) structures is certainly possible with solid elements (also known as continuum elements), in other words hexahedra and tetrahedra. However, it is known that the accuracy of solid elements progressively deteriorates as the aspect ratio increases (Figure 6.3). As the structure becomes increasingly thinner, the accuracy of the model goes down, unless we employ more and more elements in the plane of the structure to keep the aspect ratio reasonable. The cost eventually becomes prohibitive.

Shell elements can counteract this increase in cost, but only if accuracy is sufficient for given purpose. For shell-like structures this means representing bending and transverse shear well. Shell finite

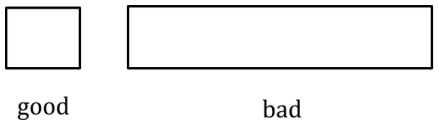


Fig. 6.3. Cartoon of solid elements, demonstrating aspect ratio of width over height. Large aspect ratio is not good for accuracy.

elements usually have no problem with bending, the troublesome stress resultant is the transverse shear. Abaqus has several shell finite elements for shear-flexible structures such as thick homogeneous shells, or shells which due to their composition are exceptionally flexible in shear (e.g. as seen in sandwich structures).

Conventional shells.

The (conventional shell) quadratic quadrilateral with eight nodes, S8R, is specialized to thick shells. It uses reduced integration ("R"), a technique intended to make the finite element more flexible. The quadrilateral conventional shell finite element with four nodes, S4, and its reduced-integration version, S4R, both take shear flexibility into account.

Continuum shells.

Additionally, Abaqus provides the so-called continuum shell elements, SC8R and SC6R, and these elements are very effective for thick shells. Stacks of continuum shell elements can be useful when modeling laminated (layered) shells, including sandwich shells.

6.2.2 Represent very thin structures

One of the points of using complicated structural forms is that it allows the designer to use little material. This leads to thin structures: for instance the Apollo capsules had aluminum skins only 0.3mm thick. The Kirchhoff-Love (shear-rigid) hypothesis is very appropriate for such very thin structures. It is however quite hard to formulate finite elements starting from this hypothesis. In Abaqus, all shell elements, except one (STRI3), start from the Reissner-Mindlin (RM, shear-flexible) theory.

Shear-rigid hypothesis and discrete Kirchhoff.

The Kirchhoff-Love (KL) assumption of zero transverse shear strain is enforced ab initio in the STRI3 element, where the formulation makes the transverse shear strains along the three edges vanish by design. The transverse shears are not necessarily zero anywhere else within the triangle. Nevertheless, this element converges to the results of the KL shell theory.

In the other Abaqus thin-shell elements, S4R5, S8R5, S6R5 (quadrilateral with 4 and 8 nodes, respectively, and triangle with 6 nodes), the KL assumption of zero transverse shear strains is only enforced numerically at preselected points of the shell.

The elements that respect the KL hypothesis of "no transverse shear" in some approximate way are usually referred to as "discrete Kirchhoff" finite elements, or DK triangles (DKT) and DK quadrilaterals (DKQ).

Shear-flexible hypothesis and reduced integration.

In the other camp are finite elements that pursue the RM hypothesis even in the limit of very thin shells: meaning that even as the shell is becoming very thin, the transverse shear is still being calculated from the deformation of the shell. That could be a serious problem, as explained in this book in detail for the Timoshenko beam: the elements would become progressively stiffer with a decrease of the thickness, and that would seriously hamper convergence to the correct solution.

Since the deflections of the affected finite elements are only a fraction of the correct deflections, we refer to the phenomenon as shear *locking*.

The same or similar remedies as for the beam may still be applied here: reduced integration, for instance. Or, as in the S4, S3, S4R elements, the distribution of the transverse shear strains is assumed, instead of being calculated from the deformation of the shell, which allows for the shear strains to be matched to the bending-dominated response of the very thin shells.

The shell element in Abaqus for which no effective remedy of shear locking is available is the S8R. This element should not be used for thin shells.

6.2.3 Report stress resultants and stress

The definition of forces and bending moments in shells (3.2) and (3.4) takes into account the offset of the reference surface from the mid-surface. This makes the user work a little bit harder to interpret forces and bending moments in shells where the offset is non-zero. For instance, consider Figure 6.4 – The two circumstances look superficially similar, but there are significant differences. The applied force F exerts no moment relative to the reference surface ad. (a), while it does in case (b). Abaqus treats the offsets as in (a); case (b) can be simulated in Abaqus too, not using offsets, but using rigid arms.

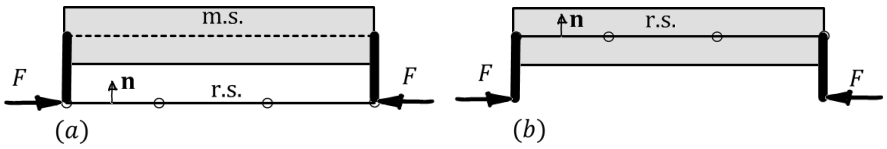


Fig. 6.4. Shell with eccentric compressive loading. (a) The body of the shell is attached to the reference surface with an offset (the offset is approximately $z_0 \approx -h$). The loading is applied directly to the nodes on the reference surface. (b) The body of the shell is centered on the reference surface, so that the reference surface and the midsurface coincide. The loading is applied with an offset using rigid arms. Here m.s. stands for midsurface, and r.s. stands for reference surface.

Material orientation.

While there are circumstances under which no material orientation need to be defined (free vibration or buckling analysis, when we are not interested in looking at the stress), it is good practice to always define explicit material orientations. This is crucial in aiding the interpretation of stress (resultant) results. The orientation for shells is defined by setting up the normal vector to the surfaces, and then by describing the directions of two orthogonal vectors in planes tangent to the shell (directions "1" and "2"). The default definition of the material directions used by the software is by no means guaranteed to be consistent. An example is given in Figure 6.5: the surface shown represents one eighth of a pressure vessel, which consists of a cylindrical surface and spherical caps. where the cylindrical surface and the caps meet, it is important that the coordinate systems used to describe the resultant forces be consistent. The Figure 6.5(a) shows the material orientation coordinate triads which clearly have inconsistent orientations along the junction of the two pieces of the shell. In such a case, speaking of one particular force resultant does not make sense along the junction as two *different* coordinate systems are used. Figure 6.6 shows the SF3 stress resultant force described in two inconsistent coordinate systems along the junction of the cylindrical and spherical piece of the shell: Clearly the display is misleading.

Integration.

As expected, numerical integration is used to evaluate the weighted residual method equations for shell elements. The complicating factor is that integration is needed not only in the surface of the

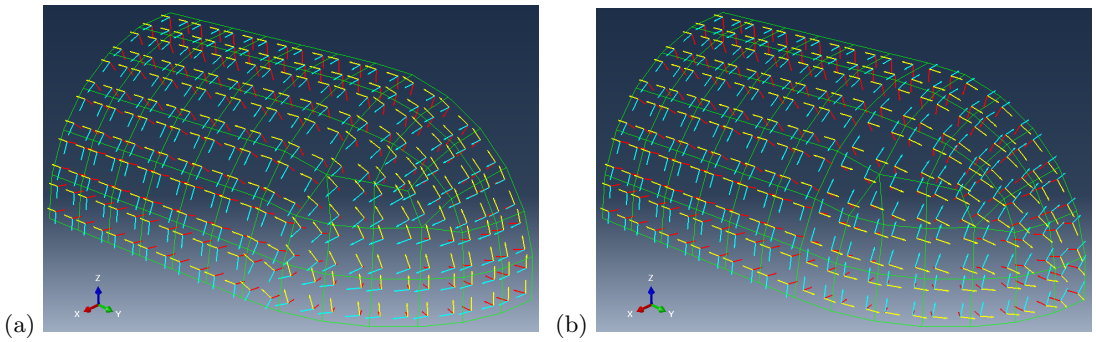


Fig. 6.5. Shell representing one eighth of a cylindrical pressure vessel with spherical caps. Upper image: no material orientations explicitly assigned: the default yields orientations which are inconsistent between the cylindrical part and the spherical part. Lower image: material orientations explicitly assigned to be compatible (the first direction runs circumferentially, the second is along the meridians).

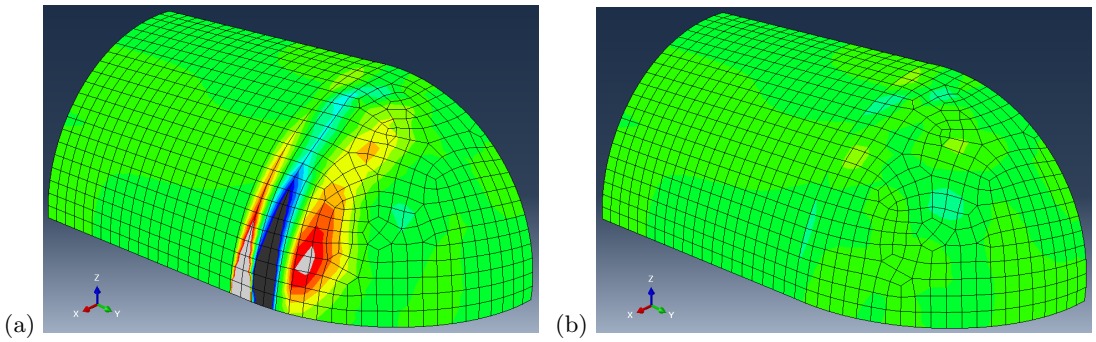


Fig. 6.6. Shell representing one eighth of a cylindrical pressure vessel with spherical caps. The stress force resultant SF3 for the shell with (a) inconsistent material orientation, and (b) consistent material orientation. The color scale has the same limits for both figures.

element, but also through the thickness. The integration through the thickness is carried out at the so called section points, while the locations in the surface are referred to as integration points. In other words, each integration point is associated with a stack of section points (Figure 6.7 and Figure 6.8). The integration through the thickness should be performed at the bottom and top surface (for one thing, the stresses have a minimum or a maximum at the surface): hence the most common rules are Simpson's, which have integration points at the extreme ends of the integration interval. Simpson 5-point rule for homogeneous shells, and 3-point Simpson rule for the layers of laminated shells. In this way, the extreme stresses are captured.

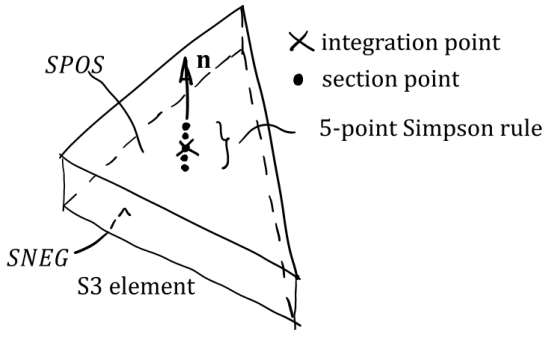


Fig. 6.7. Integration scheme of the Abaqus S3 general-purpose shell finite element.

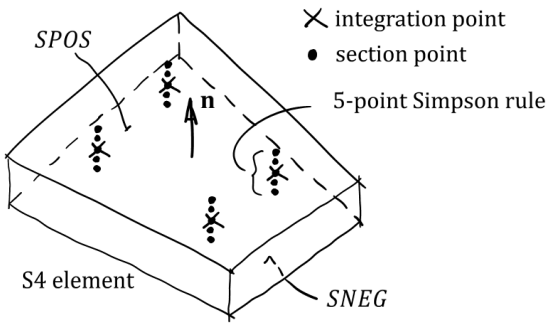


Fig. 6.8. Integration scheme of the Abaqus S4 general-purpose shell finite element.

Transverse shear stress.

The elements that incorporate the KL assumption (STRI3, S4R5, S8R5, S6R5) cannot, by definition, calculate the transverse shear forces from the deformation, and therefore these are not available for output.

Output of the transverse shear stresses is a complicated process: the RM hypothesis implies uniform transverse shear stress through the thickness. That is clearly inconsistent with the boundary conditions at the bottom and top surfaces of the shell. Therefore, there is a post processing step that can output the shear stress so that it approximates the expected parabolic variation through the thickness. (Look up computation of the transverse sheer stresses TSHR13, TSHR23 in the Abaqus documentation.)

6.2.4 Handle curved shells

One of the points of shells is that their shape makes it possible to utilize material more efficiently: stresses distributed uniformly through the thickness (i.e. as membrane forces) are more effective than bending stresses. Curvatures of shell surfaces are important pieces of the vocabulary of design, as they make the transfer of the loading through the membrane forces possible, and it is therefore important to be able to analyze such curved surfaces.

Curved surfaces may be approximated as a collection of flat facets (Figure 6.9). The elements with linear basis functions (both conventional and continuum) in Abaqus provide such a modeling capability. The danger is that the polygonal approximation of the curved surface makes it artificially stiffer.

The quadratic elements in Abaqus can approximate curved shapes better. However, curved shell elements are exposed to the so called membrane locking. Abaqus treats this locking with reduced integration (note that S6R5, S8R5, S8R all rely on reduced integration). Another kind of locking may also present for quadrilateral shapes: when the initial shape of the element is twisted, the element may unphysically stiffen. This effect is successfully controlled in the Abaqus S4-group of elements, however.

6.2.5 Define offset between the midsurface and the reference surface

Oftentimes CAD softwares represent the shell-like structures as surfaces that are the bottom or top surface of the shell, not the midsurface (Figure 6.10). In that case, it is very convenient to have the ability to represent the volume of the shell offset from the reference surface. In Abaqus this ability is provided in the definition of the section. The offset can be uniform along the shell, or it can be defined to be almost arbitrary.

Also, when connecting stiffeners and edge beams with the shell, the nodes of the reinforcing beams can be conveniently placed within the reference surface of the shell (Figure 6.11).

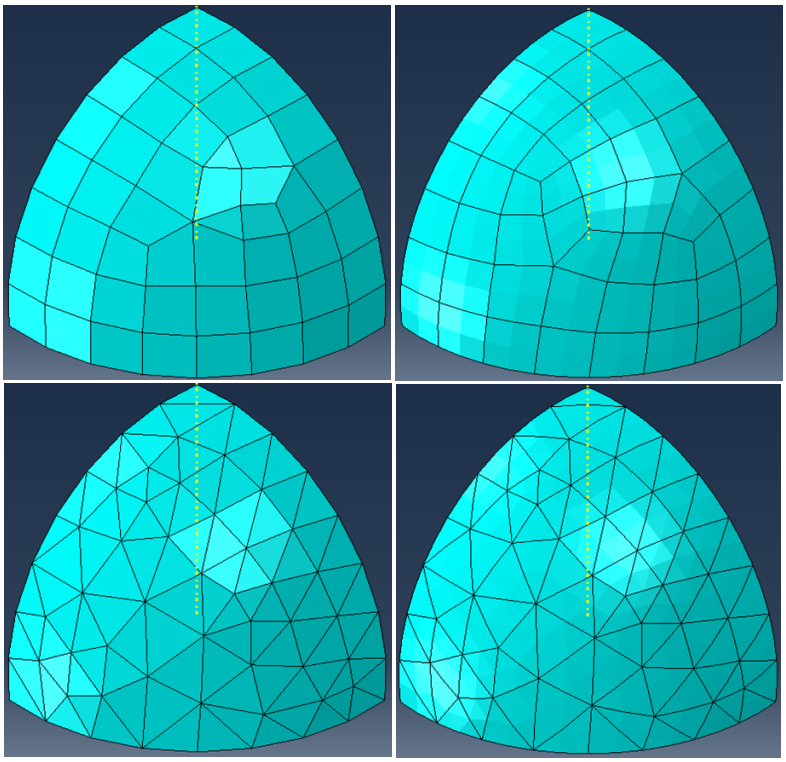


Fig. 6.9. Shell representing one eighth of a spherical surface. On the left: "linear" elements (S3, S4) provide a faceted representation of the surface. On the right: "quadratic" element (S6R5, S8R5, S8R) can provide a smoother approximation of the shape (but not exact!).

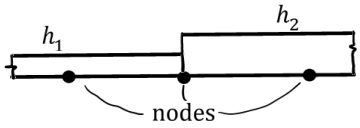


Fig. 6.10. Two shells of different thicknesses are connected together so that a smooth surface from one side is preserved.

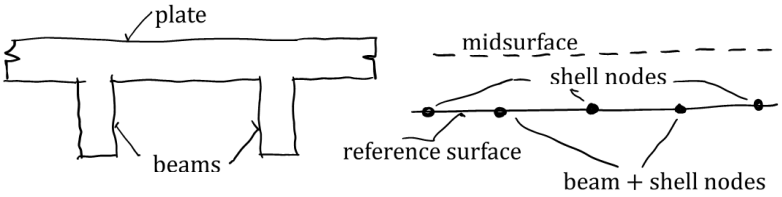


Fig. 6.11. Cartoon of a connection between a plate and beam stiffeners. The reference surface of the plate is offset to pass through the centroids of the beam stiffeners. Then the nodes of the beams lie on the same surface as the nodes of the shell.

Figure 6.12 shows a couple of configurations (single shear lap joint, attachment of stringer to skin) where the ability to define offsets makes the modeling much easier. The eccentricities that need to be taken into account are easier to deal with if the reference surfaces overlap , since then the tie constraints between the shells are easy to setup.

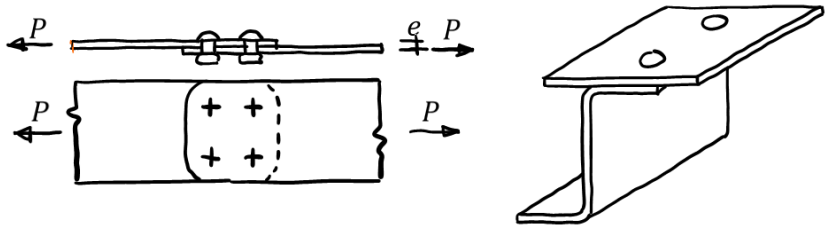


Fig. 6.12. Various joints between shells where the ability to define offsets is of use.

6.2.6 Connect to stiffeners and edge beams

Shells are rarely used by themselves (as a monocoque structure). Most often there are stiffeners (spars and stringers), and often there may be edge beams, frames, or bulkheads (see for instance Figure 6.13). Some elements of such structures may be modeled as beams (typically the stringers), others as shells (the skins and the frame).

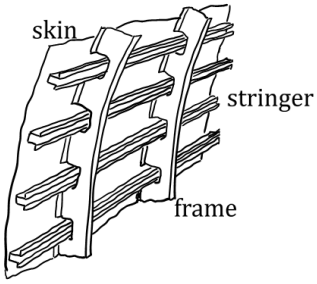


Fig. 6.13. Aircraft body frame, skin, and stringers.

Figure 6.14 shows the example of the front spar as part of the wing box. The typically open profiles of the stiffeners have a limited capacity to transfer torsion moments. (This aspect will be discussed in a separate chapter.) The spar web is often considered only for its capacity to transfer shears, while the chords' action in compression and tension is typically idealized as the main mechanism of transferring a bending moment by the box.

6.2.7 Require minimum user input

We already mentioned that for the user to make sense of the computed stress resultants and the stresses themselves, coordinate systems need to be set up to make interpretation of stresses possible. This usually means that a normal to the shell needs to be defined, which is typically accomplished by referring to the CAD surface that underlies the shell. The more difficult part is definition of the directions in the planes tangent to the shell (directions 1 and 2). If two or more pieces of shell meet along a junction, it would be nice to have all the pieces agree on the coordinate directions. Depending on how complex the geometry is, this may be all but impossible.

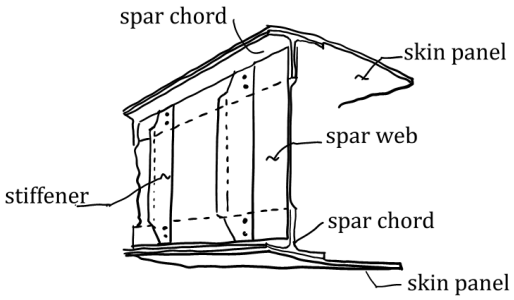


Fig. 6.14. Structure of the front spar of an airplane wing. The skins and the spar web may be modelled as shells, the stiffeners as beams.

The above is complicated further by the reasonable requirement that the effort on the part of the user should be minimal. Sometimes doing a little bit more work upfront can result in significant reduction of time later on: A judicious partitioning of the surface geometry may help with the definition of suitable coordinate directions.

6.2.8 Do not rely on user-specified parameters

A user-specified parameter is usually bad news. The user might be hard-pressed to find a suitable value for this parameter, and a poor choice may mean compromised results.

As an example, consider shell elements used in a flat configuration, meaning that the shell elements lie in the same plane around a node, or are nearly coplanar. Then some shell elements (those that have six degrees of freedom per node) will require an arbitrary stiffness coefficient for the rotation about the normal to this arrangement of elements. An example is the flat triangle STRI3 shown in Figure 6.15.

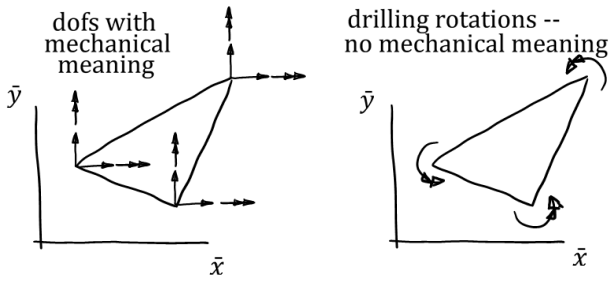


Fig. 6.15. Shell elements in a nearly coplanar configuration. The rotation about the normal to the surface has no mechanical meaning. An arbitrary stiffness coefficient is required.

The rotation about the normal is sometimes referred to as the "drilling rotation". Abaqus can usually guess a reasonable number to use for the artificial drilling rotation stiffness, but when it cannot do so, we may get unphysical results: for instance, in vibration analysis we may find meaningless frequencies corresponding to the drilling rotations. Then, we may have to pick a better value for the drilling stiffness: not necessarily an easy task.

One more note: we should not try to connect the beams to shells in a way that would require transfer of moment from the beam into the drilling rotation. Since the stiffness associated with that degree of freedom on the shell is artificial, we cannot hope to match the expected mechanical behavior.

6.2.9 Connect to solids

On the one hand, Abaqus provides the so-called shell-to-solid coupling constraint for the conventional shells. The shell-to-solid coupling is enforced through automatically created set of distributing coupling constraints between the nodes on the shell edge and the nodes on the solid surface. Each distributing constraint spreads the forces and moments acting at its shell node as forces acting on the related set of coupling solid-surface nodes. Figure 6.16 shows an example: the middle portion of the structure with the hole is represented with a solid mesh, and the remainder is modeled with shells coupled to the solid.

Abaqus
- CAE file

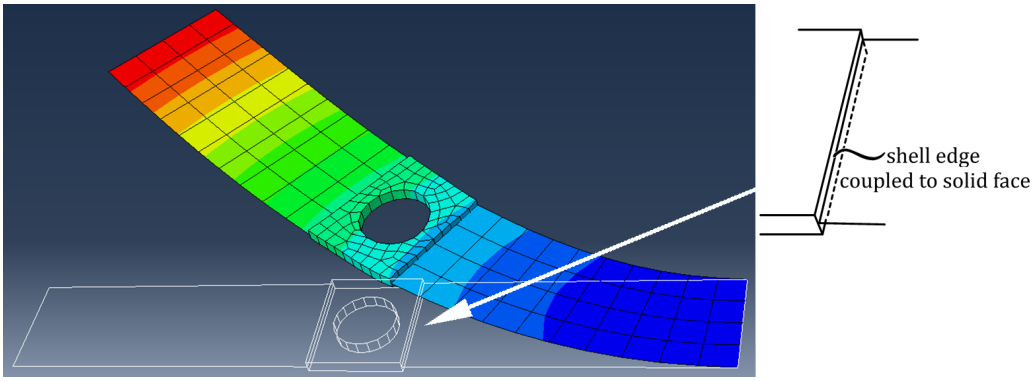


Fig. 6.16. Illustration of the solid-to-shell coupling for conventional shells.

On the other hand, the continuum shells are directly compatible with solids (albeit only with hexahedral meshes, due to the shape of the side faces of the continuum shell elements).

6.2.10 Represent layered structures

Engineering practice today relies increasingly on modern materials. Be they laminated composites or cross laminated timber, or corrugated cardboard sheets, layered structures are an important part of the structural engineer's vocabulary. This topic is picked up in more detail in Chapter 13.

6.2.11 Represent branched shells

The issues are different for the conventional shell and the continuum shell elements. Some conventional shell elements are formulated with just two rotations per node, i.e. five degrees of freedom per node (Figure 6.17). However, where the branches of a shell meet (i.e. at the junction), all three rotations (which means all six degrees of freedom per node) need to be present. Fortunately, this does not fall on the user, the software can handle this automatically.

The meshing of junctions between conventional shells is likely to be easier than the meshing of the junctions between continuum shells, where a 3D mesh is required: Figure 6.18.

6.2.12 Handle dynamics and buckling.

Most elements would have no trouble, but unfortunately the default 4-node element S4R sometimes fails on both counts. The failure is due to the use of reduced integration with stabilization. Their stabilization is just not enough, and hourglassing shows up and pollutes the response.

6.2.13 Model contact.

Linear elements (in the sense that the basis functions inside the element are basically "linear", which includes the quadrilateral with four nodes and certainly the triangle with three nodes) are usually much better at handling contact then higher-order elements (quadratic quadrilaterals and triangles).

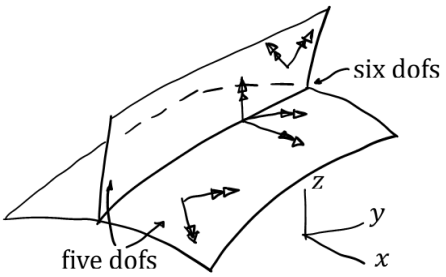


Fig. 6.17. Illustration of the degrees of freedom on the branches of shells and at the junction.

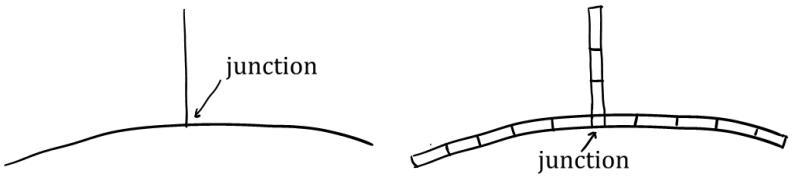


Fig. 6.18. Illustration of the junctions of the branches of shells. Conventional (left) and continuum (right) shell models.

6.2.14 Be a general tool

The Abaqus element lineup features some elements labelled "general" (or general-purpose). The intended meaning is to express how the elements can handle the spectrum from very thick shells to very thin ones. For the thick shells, being able to represent transverse shear strains is important. S4 can do it. For the very thin shells, being able to avoid shear locking is important, and again, S4 is robust for such shells. It is in this sense general-purpose.

Is the reduced-integration S4R a general-purpose element? It certainly displays the same characteristics as S4 relative to the bending response. However, if we take into account its other characteristics, we must exclude it from consideration as a general tool. Because the element uses reduced integration, it needs to be stabilized against hourglassing modes. That is sometimes successful, other times not. It is especially pronounced in membrane response next to partially supported edges.

The continuum shell element SC8R also has claim on being a general purpose tool. It certainly represents transverse shear, and it does not lock for thin shells. However, its accuracy for thin shells is not great. That illustrates an important point: general-purpose shell elements may be easily beaten by specialist elements, when those specialist elements are used properly in their own domain. So, for instance STRI3 will be very accurate for very thin shells, and S8R will be impressively accurate for thick shells. These two elements are definitely not general-purpose (STRI3 ignores shear strains, so it is only suitable for homogeneous thin shells, and S8R locks for thin shells), but they do have their uses.

6.2.15 Be convergent

The analyst should always make sure that the employed model is reliable: Any element used for work to support decision-making must be convergent. In other words, in the limit of the element size approaching zero, the solution delivered by the mesh must be approaching the correct solution of the problem. Alas, even in highly regarded finite element software one can find formulations of finite elements that are at least in some circumstances (geometry, loading, ...) not convergent [PKJS].

Abaqus
- CAE file

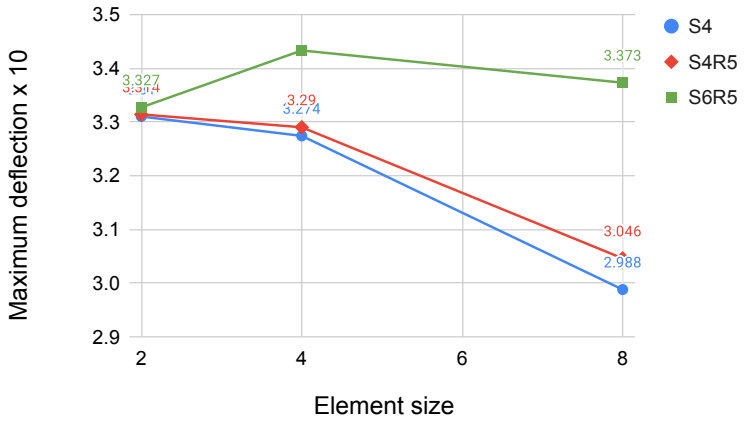


Fig. 6.19. Illustrative example of convergence graphs: Thin equilateral triangle plate with uniform distributed loading. Span to thickness ratio of 1000. The convergence of the S6R5 triangular element is erratic. The generalist S4 element and the thin-shell specialist S4R5 appear to converge well.

6.2.16 Be robust

We could call elements that can be used irrespectively of the thickness of the shell robust. But that is only one aspect of the concept of robustness. Most importantly of all, no element that cannot be shown to be convergent can be called robust!

Sensitivity to the element shape is of great concern to the users of FE software. As an example, S8R's performance degrades significantly when the element departs from a square shape. Other element types can also be sensitive, for instance quadrilaterals whose surface is warped à la hyperbolic paraboloid. Could lead to "locking" and such behavior must be controlled (S4 had been designed to be resistant to this).

Elements that require special tuning parameters to be supplied by the user should probably not be called robust. One aspect discussed above was the the need to control hourglassing in the S4R element, and the handling of the drilling rotations. Both required, at least in some cases, user input.

Elements that are under-integrated can be susceptible to hourglassing — development of unphysical modes of deformation. An example is the "default" shell element, S4R (the R stands for reduced integration). Elements of this nature can perform well in some problems and be economical, but we cannot call them robust.

6.2.17 Be economical

As we just pointed out, the quadrilateral S4R can be quite economical: even though the hourglassing must be controlled, a single integration point means much less computational effort than the usual Gauss 4-point integration rule. But here we can see that the requirement that the element be economical can (and often does) run counter to the requirement that the element be robust.

7

Truss finite element

Summary

1. The mathematical model of the truss member in special orientation is developed for finite element application.
2. The finite element model is then adjusted for a general orientation of the member.

7.1 Truss FE model in basic orientation – basic truss element

Here we assume a homogenous prismatic bar of cross section S: refer to Figure 4.1. We are only interested in a one dimensional model: the deformation only occurs along the x the axis of the bar. We refer to this as the basic orientation of the bar, and hence we call this model the basic truss model.

The balance equation reads

$$\frac{\partial \sigma_x}{\partial x} = 0 \tag{7.1}$$

or, substituting for the stress from the constitutive equation and then expressing the strain through the displacements,

$$E\frac{\partial \epsilon_x}{\partial x} = E\frac{\partial^2 u}{\partial x^2} = Eu'' = 0 \tag{7.2}$$

Here we use the abbreviated notation $u'' = \partial^2 u/\partial x^2$.

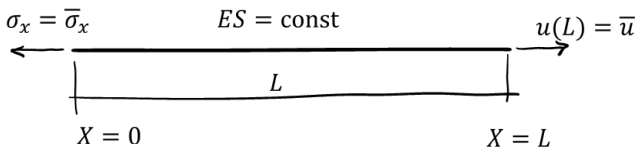


Fig. 7.1. Truss boundary value problem.

Figure 7.1 presents an example of a boundary value problem to be solved with our finite element model: there is a traction boundary condition on the left hand side, and the displacement is prescribed at the right hand side.

The weighted residual statement is obtained by taking the balance equation and multiplying with the test function, and integrating the product over the volume of the bar

$$\int_V \eta E u'' \mathrm{d}V = 0 \,. \tag{7.3}$$

Since the integrand does not depend on the coordinates in the plane of the cross section, this can be simplified to

$$S \int_0^L \eta E u'' \mathrm{d}x = 0 \tag{7.4}$$

Integration by parts can be used to shift one of the derivatives from the trial function to the test function

$$S \int_0^L \eta E u'' \mathrm{d}x = S \int_0^L \left(\eta E u'\right)' \mathrm{d}x - S \int_0^L \eta' E u' \mathrm{d}x = S \left[\eta E u'\right]_0^L - S \int_0^L \eta' E u' \mathrm{d}x = 0 \tag{7.5}$$

There is a direct connection of (7.5) to the so-called virtual work principle. We can write

$$\left[\delta u E S u'\right]_0^L - \int_0^L \delta u' S E u' \mathrm{d}x = 0 \tag{7.6}$$

or, using $N = ESu'$ as the definition of the internal force,

$$\left[\delta u N\right]_0^L - \int_0^L \delta u' N \mathrm{d}x = 0 \tag{7.7}$$

The first term is the virtual work of the external forces on the boundary (i.e. at the endpoints), while the second term is the virtual work of the internal forces. Here we simply call the test function η the virtual displacement δu (which when differentiated becomes the virtual strain $\delta u'$). So since (7.5) was derived from scratch, from the balance equation, we can see that the virtual work principle is really no principle at all. It is just a convenient way of getting to the useful expression (7.5) with little effort.

Now, referring to Figure 7.1 we can see that the test function needs to go to zero at the right hand side, $\eta(x = L)$. At the left hand side, we can replace the stress Eu' with the prescribed stress component. Hence we get

$$S \left(\eta \overline{\sigma}_x\right)\big|_{x=0} - S \int_0^L \eta' E u' \mathrm{d}x = 0 \tag{7.8}$$

Next we introduce the finite element mesh. For this particular element we can use the isoparametric formulation. Both the coordinate and the approximated displacement can be written using the basis functions shown in Figure 11.1

$$x = N_{\underline{1}}(\xi) x_{\underline{1}} + N_{\underline{2}}(\xi) x_{\underline{2}} \,, \quad u = N_{\underline{1}}(\xi) u_{\underline{1}} + N_{\underline{2}}(\xi) u_{\underline{2}} \,. \tag{7.9}$$

The displacement along the axis of the bar finite element is expressed through the displacements of the two end points, $u_{\underline{1}}$ and $u_{\underline{2}}$. Given the linear change of the displacements between the nodes, we arrive at a uniform axial strain, $\epsilon_x = (u_{\underline{2}} - u_{\underline{1}})/h$, where h is the element length. This can be easily verified using the definition of the basis functions and

$$\frac{\partial N_{\underline{j}}}{\partial x} = N'_{\underline{j}} = \frac{\partial N_{\underline{j}}}{\partial \xi} \frac{\partial \xi}{\partial x} \quad \text{where } \frac{\partial \xi}{\partial x} = 2/h \,. \tag{7.10}$$

Hence, $N'_{\underline{1}} = -1/h$ and $N'_{\underline{2}} = +1/h$, and the expression for the axial strain follows.

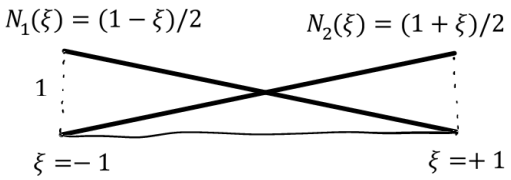


Fig. 7.2. L2 finite element basis functions.

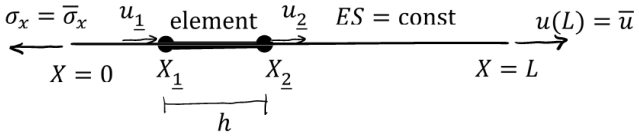


Fig. 7.3. The finite element mesh on the domain from Figure 7.1.

We get the stiffness matrix of the truss finite element as follows: Equation (7.9) is substituted into the weighted residual statement (7.8) for the test function being taken first as $\eta = N_{\underline{1}}$, and then as $\eta = N_{\underline{2}}$. So

$$S\int_0^h N'_{\underline{1}} E\left(N'_{\underline{1}}u_{\underline{1}} + N'_{\underline{2}}u_{\underline{2}}\right)\mathrm{d}x = S\int_0^h (-1/h)E\left[(-1/h)u_{\underline{1}} + (1/h)u_{\underline{2}}\right]\mathrm{d}x = \frac{ES}{h}u_{\underline{1}} - \frac{ES}{h}u_{\underline{2}} \tag{7.11}$$

and

$$S\int_0^h N'_{\underline{2}} E\left(N'_{\underline{1}}u_{\underline{1}} + N'_{\underline{2}}u_{\underline{2}}\right)\mathrm{d}x = S\int_0^h (1/h)E\left[(-1/h)u_{\underline{1}} + (1/h)u_{\underline{2}}\right]\mathrm{d}x = -\frac{ES}{h}u_{\underline{1}} + \frac{ES}{h}u_{\underline{2}} \tag{7.12}$$

The meaning of the first equation is the force that acts on joint $\underline{1}$, and similarly the second equation represents the force that acts on joint $\underline{2}$.

We can write the right-hand side of both of the above statements in one matrix expression as

$$\mathbf{K}^{(e)}\begin{bmatrix} u_{\underline{1}} \\ u_{\underline{2}} \end{bmatrix} \tag{7.13}$$

where the stiffness matrix reads

$$\mathbf{K}^{(e)} = \frac{ES}{h}\begin{bmatrix} +1\ , & -1 \\ -1\ , & +1 \end{bmatrix}\ . \tag{7.14}$$

The above may be expressed using the familiar format of a strain-displacement matrix. We express the axial strain, $\epsilon_x = u'$, as

$$\epsilon_x = \begin{bmatrix} -1/h\ , & +1/h \end{bmatrix}\begin{bmatrix} u_{\underline{1}} \\ u_{\underline{2}} \end{bmatrix} \tag{7.15}$$

The matrix

$$\mathbf{B}^{(e)} = \begin{bmatrix} -1/h\ , & +1/h \end{bmatrix} \tag{7.16}$$

is the strain-displacement matrix for the two-node basic bar finite element. It has a single row, and two columns: two columns because there are two displacement degrees of freedom. Now we write

$$\eta' = \mathbf{B}^{(e)} \begin{bmatrix} \eta_{\underline{1}} \\ \eta_{\underline{2}} \end{bmatrix} \tag{7.17}$$

and then the product $\eta' E u'$ may be stated as

$$\eta' E u' = \begin{bmatrix} \eta_{\underline{1}} \\ \eta_{\underline{2}} \end{bmatrix}^T \mathbf{B}^{(e)^T} E \mathbf{B}^{(e)} \begin{bmatrix} u_{\underline{1}} \\ u_{\underline{2}} \end{bmatrix} \tag{7.18}$$

Now we substitute into (7.8), and it follows that using the strain-displacement matrix, the stiffness matrix of the truss element may be written

$$\mathbf{K}^{(e)} = S \int_0^h \mathbf{B}^{(e)^T} E \mathbf{B}^{(e)} \mathrm{d}x \ . \tag{7.19}$$

This is again the result of Equation (7.14). In particular, the integrand is constant, so the integration is simply the integrand times the length of the element h.

7.2 Truss in the plane

Figure 7.4 shows a schematic of a two dimensional truss structure on the left. This stick figure of course represents a real three dimensional structure, as shown on the right. The domain in which we write down and solve the equations of motion is the union of the volumes of the individual bars. The joints will be in reality some complicated designs (probably with gusset plates and such), but here we simply represent those joints as some form of pins and show them as black dots.

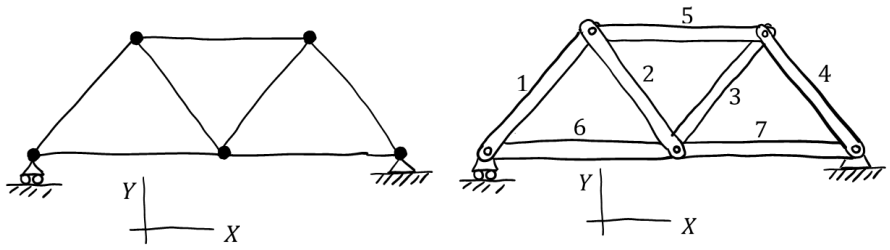


Fig. 7.4. Truss structure: stick schematic on the left, domain of a real three dimensional structure on the right.

In the preceeding Section we formulated the stiffness matrix of a basic bar finite element in one dimension: The domain was interval along the x coordinate, and the degrees of freedom were $u_{\underline{1}}$ and $u_{\underline{2}}$. The structure in Figure 7.4 may be treated with a slight modification of such a model by using a local coordinate system aligned with each bar. Figure 7.5 shows the bars 1, 2, and 6, each with a coordinate system aligned with its axis. The weighted residual equation analogous to (7.5) simply needs to be integrated over the volume of the structure, which here means over each of the seven bars.

Figure 7.5 shows bar 1 isolated from the rest of the structure. The displacements of the two joints to which the bar is connected result in the two displacements $u_{\underline{1}}$ and $u_{\underline{2}}$ that compress or stretch the bar 1, and which the bar can resist with its stiffness matrix. The displacement of the joint in the plane is shown at the top joint, with the components $(\mathbf{U}_{\underline{2}})_X$, $(\mathbf{U}_{\underline{2}})_Y$, in the global Cartesian coordinate system, XY.

We define the direction of the local coordinate system on the bar with a unit vector $\mathbf{e}_x = [(\mathbf{e}_x)_X, (\mathbf{e}_x)_Y]^T$. The local displacement $u_{\underline{j}}$ is obtained by projection of the global joint displacement vector at node $\underline{j}$, $\mathbf{U}_{\underline{j}}$, onto the direction of $\mathbf{e}_x$

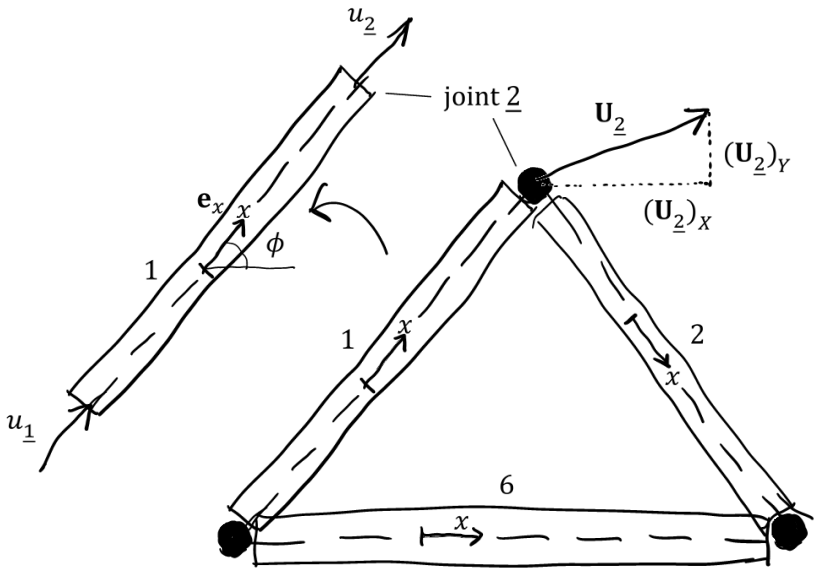


Fig. 7.5. Truss structure: three bars with their own coordinate systems. Bar 1 is shown isolated from the rest of the structure, and the displacements that stretch or compress the bar are shown.

$$u_{\underline{j}} = [(\mathbf{e}_x)_X, (\mathbf{e}_x)_Y] \begin{bmatrix} (\mathbf{U}_{\underline{j}})_X \\ (\mathbf{U}_{\underline{j}})_Y \end{bmatrix} = (\mathbf{e}_x)_X (\mathbf{U}_{\underline{j}})_X + (\mathbf{e}_x)_Y (\mathbf{U}_{\underline{j}})_Y \tag{7.20}$$

Therefore, we can rewrite the strain from (7.15) as

$$\epsilon_x = \begin{bmatrix} -1/h \ , \ +1/h \end{bmatrix} \begin{bmatrix} u_{\underline{1}} \\ u_{\underline{2}} \end{bmatrix} = \begin{bmatrix} -1/h \ , \ +1/h \end{bmatrix} \begin{bmatrix} [(\mathbf{e}_x)_X, (\mathbf{e}_x)_Y] \begin{bmatrix} (\mathbf{U}_{\underline{1}})_X \\ (\mathbf{U}_{\underline{1}})_Y \end{bmatrix} \\ [(\mathbf{e}_x)_X, (\mathbf{e}_x)_Y] \begin{bmatrix} (\mathbf{U}_{\underline{2}})_X \\ (\mathbf{U}_{\underline{2}})_Y \end{bmatrix} \end{bmatrix} \tag{7.21}$$

This we can rearrange as

$$\epsilon_x = \begin{bmatrix} (-1/h)[(\mathbf{e}_x)_X, (\mathbf{e}_x)_Y] \ , \ (+1/h)[(\mathbf{e}_x)_X, (\mathbf{e}_x)_Y] \end{bmatrix} \begin{bmatrix} (\mathbf{U}_{\underline{1}})_X \\ (\mathbf{U}_{\underline{1}})_Y \\ (\mathbf{U}_{\underline{2}})_X \\ (\mathbf{U}_{\underline{2}})_Y \end{bmatrix} . \tag{7.22}$$

The matrix

$$\mathbf{B}^{(e)} = \begin{bmatrix} (-1/h)[(\mathbf{e}_x)_X, (\mathbf{e}_x)_Y] \ , \ (+1/h)[(\mathbf{e}_x)_X, (\mathbf{e}_x)_Y] \end{bmatrix} \tag{7.23}$$

with one row and four columns is the strain-displacement matrix for the two-node bar finite element arbitrarily oriented in the global Cartesian coordinate system XY.

The stiffness matrix of a bar element in the XY plane draws on the same formula as before, namely (7.19), but the strain-displacement matrix is replaced with that of (7.23). The result is a 4×4 stiffness matrix (recall that the stiffness matrix of the basic truss element was 2×2).

It is possible to be more explicit about the form of the strain displacement matrix: consider that $(\mathbf{e}_x)_X = \cos\phi$, and $(\mathbf{e}_x)_Y = \sin\phi$, in terms of the signed angle ϕ in Figure 7.5.

$$\mathbf{B}^{(e)} = \begin{bmatrix} (-1/h)[\cos\phi, \sin\phi] \ , \ (+1/h)[\cos\phi, \sin\phi] \end{bmatrix} = \frac{1}{h} \begin{bmatrix} -\cos\phi \ , \ -\sin\phi \ , \ +\cos\phi \ , \ +\sin\phi \end{bmatrix}$$

(7.24)

So finally we arrive at the stiffness matrix of the truss member as

$$\mathbf{K}^e = S\int_0^h \mathbf{B}^{(e)^T} E\mathbf{B}^e \mathrm{d}x = \frac{ES}{h}\begin{bmatrix} -\cos\phi \\ -\sin\phi \\ \cos\phi \\ \sin\phi \end{bmatrix}\begin{bmatrix} -\cos\phi, -\sin\phi, \cos\phi, \sin\phi \end{bmatrix} \tag{7.25}$$

which is identical to the well-known expression from structural analysis.

The strain-displacement matrices for use with anisotropic materials, where the orientation of the material needs to be taken into account, have been discussed in the book [PK1]. Note that each nodal strain-displacement submatrix $\mathbf{B}_k^e = \frac{\partial N_k}{\partial \overline{x}}\mathbf{e}_{\overline{x}}{}^T$ from (7.23) agrees with the general formula

$$\mathbf{B}_k^e = \mathcal{B}^{(x,X)}\left(N_k(x)\right) = \mathcal{B}^{(x)}\left(N_k(x)\right)\mathbf{R}_m{}^T$$

discussed in [PK1].

7.3 Truss in space

The stiffness matrix of a truss member in space is a straightforward extension of (7.23). The matrix

$$\mathbf{B}^{(e)} = \left[(-1/h)[(\mathbf{e}_x)_X, (\mathbf{e}_x)_Y, (\mathbf{e}_x)_Z]\ ,\ (+1/h)[(\mathbf{e}_x)_X, (\mathbf{e}_x)_Y, (\mathbf{e}_x)_Z]\right] \tag{7.26}$$

with one row and six columns is the strain-displacement matrix for the two-node bar finite element arbitrarily oriented in the three-dimensional global Cartesian coordinate system XYZ. We simply added the Z-component of the basis vector along the length of the bar, $(\mathbf{e}_x)_Z$. The stiffness matrix is therefore

$$\mathbf{K}^e = S\int_0^h \mathbf{B}^{(e)^T} E\mathbf{B}^e \mathrm{d}x \tag{7.27}$$

This matrix has dimension 6×6, and, given that the integrand is constant, evaluating this integral is simple. In numerical implementation, one-point Gauss quadrature rule is adequate.

7.4 Dynamics: mass matrix

As a reminder, here is the problem to be solved in free vibration analysis:

$$-\omega_k^2\mathbf{M}\widetilde{\mathbf{U}}_k + \mathbf{K}\widetilde{\mathbf{U}}_k = \mathbf{0}\ , \tag{7.28}$$

where $\mathbf{M}$ and $\mathbf{K}$ are square $N_\mathrm{f}\times N_\mathrm{f}$ matrices (N_f is the number of free degrees of freedom). The solution is sought as a certain number of pairs ω_k and $\widetilde{\mathbf{U}}_k$, where ω_k is the $k-$th circular frequency of free vibration (also called angular frequency), and $[\phi_\mathbf{k}]$ is the $k-$th eigenmode (also called normal mode, or free vibration shape). By the way, the natural frequency in cycles per second (Hz) is calculated from the angular frequency as

$$f_k = \frac{\omega_k}{2\pi}\ . \tag{7.29}$$

Note that the frequencies are typically ordered by magnitude, from the smallest to the largest. The lowest-magnitude natural frequency is called the fundamental frequency. We talked about the construction of the stiffness matrices above, and now we will turn our attention to the mass matrix.

Here we'll look at the mass matrix of the two-node truss element. It will be good practice to use an alternative to the method of weighted residuals: In dynamic problems the mass matrix can be derived from an expression for the kinetic energy.

The kinetic energy of a vibrating truss element can be written as

$$KE = \frac{1}{2}\int_0^h \rho S\left(\dot{u}^2 + \dot{v}^2\right)\, dx = \frac{1}{2}(h/2)\int_{-1}^{+1} \rho S\left(\dot{u}^2 + \dot{v}^2\right)\, d\xi \tag{7.30}$$

where ρS is the mass density of the bar per unit length (mass density of the material times the cross sectional area), and $\dot{u}$ is the X component of the velocity and $\dot{v}$ is the Y component of the velocity. The integral over the length of the element is converted into an integral over the standard shape (interval -1 to +1) by change of variables using the Jacobian $h/2$.

Now we introduce the finite element expansion of the global components of the velocity using the linear shape functions shown in Figure 11.1

$$\dot{u} = N_{\underline{1}}(\xi)\dot{u}_{\underline{1}} + N_{\underline{2}}(\xi)\dot{u}_{\underline{2}}\ , \quad \dot{v} = N_{\underline{1}}(\xi)\dot{v}_{\underline{1}} + N_{\underline{2}}(\xi)\dot{v}_{\underline{2}}\ . \tag{7.31}$$

Thus we have

$$\dot{u} = \begin{bmatrix} N_{\underline{1}}(\xi) & 0 & N_{\underline{2}}(\xi) & 0 \end{bmatrix} \begin{bmatrix} \dot{u}_{\underline{1}} \\ 0 \\ \dot{u}_{\underline{2}} \\ 0 \end{bmatrix} = \mathbf{N}_u \begin{bmatrix} \dot{u}_{\underline{1}} \\ 0 \\ \dot{u}_{\underline{2}} \\ 0 \end{bmatrix} \ , \quad \dot{v} = \begin{bmatrix} 0 & N_{\underline{1}}(\xi) & 0 & N_{\underline{2}}(\xi) \end{bmatrix} \begin{bmatrix} 0 \\ \dot{v}_{\underline{1}} \\ 0 \\ \dot{v}_{\underline{2}} \end{bmatrix} = \mathbf{N}_v \begin{bmatrix} 0 \\ \dot{v}_{\underline{1}} \\ 0 \\ \dot{v}_{\underline{2}} \end{bmatrix} \tag{7.32}$$

and therefore we can write

$$\dot{u}^2 = \dot{u}^T\dot{u} = \begin{bmatrix} \dot{u}_{\underline{1}} & 0 & \dot{u}_{\underline{2}} & 0 \end{bmatrix} \mathbf{N}_u^T\mathbf{N}_u \begin{bmatrix} \dot{u}_{\underline{1}} \\ 0 \\ \dot{u}_{\underline{2}} \\ 0 \end{bmatrix} \tag{7.33}$$

and

$$\dot{v}^2 = \dot{v}^T\dot{v} = \begin{bmatrix} 0 & \dot{v}_{\underline{1}} & 0 & \dot{v}_{\underline{2}} \end{bmatrix} \mathbf{N}_v^T\mathbf{N}_v \begin{bmatrix} 0 \\ \dot{v}_{\underline{1}} \\ 0 \\ \dot{v}_{\underline{2}} \end{bmatrix} \tag{7.34}$$

Because of the locations of the zeros in these matrices, the sum of the squares may be expressed as

$$\dot{u}^2 + \dot{v}^2 = \begin{bmatrix} \dot{u}_{\underline{1}} & \dot{v}_{\underline{1}} & \dot{u}_{\underline{2}} & \dot{v}_{\underline{2}} \end{bmatrix} \left(\mathbf{N}_u^T\mathbf{N}_u + \mathbf{N}_v^T\mathbf{N}_v\right) \begin{bmatrix} \dot{u}_{\underline{1}} \\ \dot{v}_{\underline{1}} \\ \dot{u}_{\underline{2}} \\ \dot{v}_{\underline{2}} \end{bmatrix} \tag{7.35}$$

In other words, the two components of the velocity do not interact. Substituting (7.35) into the expression for the kinetic energy (7.30) results in

$$KE = \frac{1}{2}\begin{bmatrix} \dot{u}_{\underline{1}} & \dot{v}_{\underline{1}} & \dot{u}_{\underline{2}} & \dot{v}_{\underline{2}} \end{bmatrix} (h/2)\int_{-1}^{+1} \rho S\left(\mathbf{N}_u^T\mathbf{N}_u + \mathbf{N}_v^T\mathbf{N}_v\right)\, d\xi \begin{bmatrix} \dot{u}_{\underline{1}} \\ \dot{v}_{\underline{1}} \\ \dot{u}_{\underline{2}} \\ \dot{v}_{\underline{2}} \end{bmatrix} \ . \tag{7.36}$$

The matrix sandwiched by the vectors of the nodal velocities is the mass matrix of the truss element

$$\mathbf{M} = (h/2)\int_{-1}^{+1} \rho S\left(\mathbf{N}_u^T\mathbf{N}_u + \mathbf{N}_v^T\mathbf{N}_v\right)\, d\xi\ . \tag{7.37}$$

This expression is translated into computer code by using numerical integration with two-point Gauss rule.

7.5 Python implementation of truss finite elements

- URL

The Python package `pystran` implements models of two and three dimensional truss structures, both for static analysis of equilibrium and analysis of free vibration.

8

Bernoulli beam finite element in the plane

Summary

1. We introduce the basic beam bending model.
2. The mathematical model of the Bernoulli basic beam bending member is treated with the weighted residual method to produce a finite element formulation.
3. The finite element model is then adjusted for a general orientation of the bending member in the two dimensional cartesian plane.
4. Lastly, we combine a two dimensional truss model of axial deformation with a two dimensional bending model to get a beam-column (or frame) element representing the complete response.

8.1 Beam in basic orientation – Basic beam bending

The weighted residual equation is obtained from the balance equation (4.10) by multiplying the residual with a test function (η), and integrating over the length of the beam

$$\int_0^L \eta \left(EIw'''' - q\right) \mathrm{d}x = 0 \tag{8.1}$$

Here $\eta(x)$, $w(x)$, and $q(x)$ are functions of x. Note the sign convention in Figure 4.7: the positive deflection is downwards, while the x coordinate is measured left to right. Using the following identity obtained by integration by parts

$$\int_0^L \eta w'''' \mathrm{d}x = \int_0^L \left(\eta w'''\right)' \mathrm{d}x - \int_0^L \eta' w''' \mathrm{d}x \tag{8.2}$$

$$= \int_0^L \left(\eta w'''\right)' \mathrm{d}x - \int_0^L \left(\eta' w''\right)' \mathrm{d}x + \int_0^L \eta'' w'' \mathrm{d}x \tag{8.3}$$

$$= \left[\eta w'''\right]_0^L - \left[\eta' w''\right]_0^L + \int_0^L \eta'' w'' \mathrm{d}x \tag{8.4}$$

where L is the length of the entire beam, we can convert (8.1) into

$$\left[EI\eta w'''\right]_0^L - \left[EI\eta' w''\right]_0^L + \int_0^L EI\eta'' w'' \mathrm{d}x - \int_0^L \eta q \mathrm{d}x = 0 \tag{8.5}$$

The boundary conditions are introduced by utilizing the following relationships

$$
\begin{aligned}
&V(x=0) = -EIw'''(x=0) = \overline{V}_0 \text{ (given)} \\
&V(x=L) = -EIw'''(x=L) = \overline{V}_L \text{ (given)} \\
&M(x=0) = -EIw''(x=0) = \overline{M}_0 \text{ (given)} \\
&M(x=L) = -EIw''(x=L) = \overline{M}_L \text{ (given)}
\end{aligned}
\tag{8.6}
$$

The weighted residual statement is thus recast as

$$
-[\eta V]_0^L - [\eta' M]_0^L + \int_0^L EI\eta'' w'' \mathrm{d}x - \int_0^L \eta q \mathrm{d}x = 0 \tag{8.7}
$$

where the terms on the boundary are simplified for the four boundary conditions shown in Figure 4.8 using the fact that the test function needs to vanish where the deflection is prescribed, and the slope of the test function needs to vanish where the slope of the beam is prescribed.

1. Prescribed $\overline{w}$ at $x = 0$: $V(x = 0)$ is knocked out by $\eta(x = 0) = 0$.
2. Prescribed $\overline{w}'$ at $x = 0$: $M(x = 0)$ is knocked out by $\eta'(x = 0) = 0$.
3. Prescribed $\overline{w}$ at $x = L$: $V(x = L)$ is knocked out by $\eta(x = L) = 0$.
4. Prescribed $\overline{w}'$ at $x = L$: $M(x = L)$ is knocked out by $\eta'(x = L) = 0$.

The reason for requiring the test function to have these properties is that the corresponding shear force and moment are unknown reactions. To have these reactions appear in the weighted residual equation would be an inconvenience.

As usual in the finite element method, we proceed to introduce the nodal degrees of freedom and the basis functions. The degrees of freedom are the transverse displacements of the nodes, W_1, W_3, and the counterclockwise rotations at the nodes, W_2, W_4. See Figure 8.1. Note that the rotations have sense opposite to that of the slope of the beam midline: For instance at node $\underline{1}$, the slope is $w'_{\underline{1}} = -W_{\underline{2}}$.

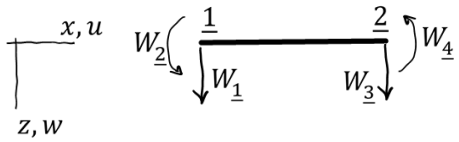


Fig. 8.1. Bernoulli beam degrees of freedom.

In terms of the degrees of freedom, the deflection curve of a beam element may be written as

$$
w(\xi) = N_{\underline{1}}(\xi)W_{\underline{1}} + cN_{\underline{2}}(\xi)W_{\underline{2}} + N_{\underline{3}}(\xi)W_{\underline{3}} + cN_{\underline{4}}(\xi)W_{\underline{4}} \tag{8.8}
$$

using the basis functions expressed in the parametric coordinate ξ (refer to Figure 8.2).

Here we use the so-called Hermite basis functions. These functions allow us to control the curve of the beam not only by the values of the deflection, but also by the slopes (rotations).

We can use the following map between the parametric coordinate ξ and the real physical coordinate x

$$x(\xi) = \frac{x_{\underline{2}} + x_{\underline{1}}}{2} + \xi \frac{x_{\underline{2}} - x_{\underline{1}}}{2} \,. \tag{8.9}$$

Here $h = x_{\underline{2}} - x_{\underline{1}}$ is the length of the element. The Jacobian of this map is $\mathrm{d}x/\mathrm{d}\xi = h/2$.

In equation (8.8) we needed to include a coefficient, c, to compensate for the difference between the physical units: the basis functions themselves are nondimensional, the deflection degrees of freedom have the dimension of length, but rotation degrees of freedom are again nondimensional. In order to be able to add together all those terms, they must have the same physical units.

In order to determine c, we compute the slope from (8.8), for instance at the first node:

$$w'(\xi = -1) = N'_{\underline{1}}(\xi = -1)W_{\underline{1}} + cN'_{\underline{2}}(\xi = -1)W_{\underline{2}} + N'_{\underline{3}}(\xi = -1)W_{\underline{3}} + cN'_{\underline{4}}(\xi = -1)W_{\underline{4}} \tag{8.10}$$

where we will use

$$N'_{\underline{j}}(\xi) = \frac{\mathrm{d}N_{\underline{j}}(\xi)}{\mathrm{d}\xi} \frac{\mathrm{d}\xi}{\mathrm{d}x} \,. \tag{8.11}$$

Here the prime ($N'_{\underline{j}}$) means differentiate with respect to x. The slopes of $N_{\underline{1}}$ and $N_{\underline{3}}$ at the nodes are zero. The slope of $N_{\underline{4}}$ at the left-hand side end is also zero. Therefore we have

$$w'(\xi = -1) = cN'_{\underline{2}}(\xi = -1)W_{\underline{2}} \tag{8.12}$$

and we can evaluate

$$cN'_{\underline{2}}(\xi) = c \frac{\mathrm{d}N_{\underline{2}}(\xi = -1)}{\mathrm{d}\xi} \frac{\mathrm{d}\xi}{\mathrm{d}x} = c \times (-1) \times (2/h) \tag{8.13}$$

We see that it is necessary for the coefficient to be $c = h/2$, so that we get $w'(\xi = -1) = -W_{\underline{2}}$.

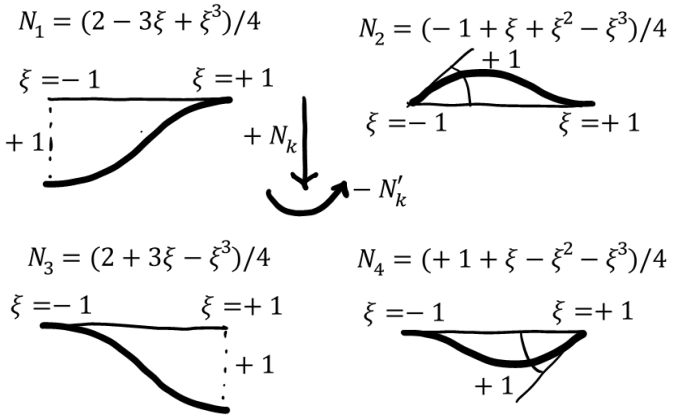


Fig. 8.2. The four Hermite shape functions. Note well that the basis function value is positive downwards, and the counterclockwise rotation is represented as the negative slope of the basis functions.

The slope of the beam may be computed as

$$w'(\xi) = N'_{\underline{1}}(\xi)W_{\underline{1}} + (h/2)N'_{\underline{2}}(\xi)W_{\underline{2}} + N'_{\underline{3}}(\xi)W_{\underline{3}} + (h/2)N'_{\underline{4}}(\xi)W_{\underline{4}} \tag{8.14}$$

$$= (2/h)\left\{ \frac{\mathrm{d}N_{\underline{1}}(\xi)}{\mathrm{d}\xi} W_{\underline{1}} + (h/2)\frac{\mathrm{d}N_{\underline{2}}(\xi)}{\mathrm{d}\xi} W_{\underline{2}} + \frac{\mathrm{d}N_{\underline{3}}(\xi)}{\mathrm{d}\xi} W_{\underline{3}} + (h/2)\frac{\mathrm{d}N_{\underline{4}}(\xi)}{\mathrm{d}\xi} W_{\underline{4}} \right\} \tag{8.15}$$

The curvature of the beam then follows as

$$w''(\xi) = N_{\underline{1}}''(\xi)W_{\underline{1}} + (h/2)N_{\underline{2}}''(\xi)W_{\underline{2}} + N_{\underline{3}}''(\xi)W_{\underline{3}} + (h/2)N_{\underline{4}}''(\xi)W_{\underline{4}} \tag{8.16}$$

$$= (2/h)^2 \left\{ \frac{\mathrm{d}N_{\underline{1}}^2(\xi)}{\mathrm{d}\xi^2}W_{\underline{1}} + (h/2)\frac{\mathrm{d}N_{\underline{2}}^2(\xi)}{\mathrm{d}\xi^2}W_{\underline{2}} + \frac{\mathrm{d}N_{\underline{3}}^2(\xi)}{\mathrm{d}\xi^2}W_{\underline{3}} + (h/2)\frac{\mathrm{d}N_{\underline{4}}^2(\xi)}{\mathrm{d}\xi^2}W_{\underline{4}} \right\} \tag{8.17}$$

The matrix

$$\mathbf{B}^{(e)} = (2/h)^2 \left[\frac{\mathrm{d}N_{\underline{1}}^2(\xi)}{\mathrm{d}\xi^2} \, , \, (h/2)\frac{\mathrm{d}N_{\underline{2}}^2(\xi)}{\mathrm{d}\xi^2} \, , \, \frac{\mathrm{d}N_{\underline{3}}^2(\xi)}{\mathrm{d}\xi^2} \, , \, (h/2)\frac{\mathrm{d}N_{\underline{4}}^2(\xi)}{\mathrm{d}\xi^2} \right] \tag{8.18}$$

or, explicitly,

$$\mathbf{B}^{(e)} = (1/h)^2 \left[6\xi \, , \, (h/2)(2-6\xi) \, , \, -6\xi \, , \, (h/2)(-2-6\xi) \right] \tag{8.19}$$

is the strain-displacement matrix for the two-node Bernoulli beam finite element. It has a single row, and four columns: four columns because there are four displacement degrees of freedom, as each node has a deflection and a rotation degree of freedom. It delivers the curvature of the beam

$$w''(\xi) = \mathbf{B}^{(e)} \begin{bmatrix} W_{\underline{1}} \\ W_{\underline{2}} \\ W_{\underline{3}} \\ W_{\underline{4}} \end{bmatrix} \tag{8.20}$$

With this result we can return to the weighted residual equation (8.7), and for each element evaluate its elementwise stiffness matrix as

$$\mathbf{K}^e = \int_0^h {\mathbf{B}^{(e)}}^T EI \mathbf{B}^{(e)} \mathrm{d}x \tag{8.21}$$

This integral can be converted to

$$\mathbf{K}^e = \int_{-1}^{+1} {\mathbf{B}^{(e)}}^T EI \mathbf{B}^{(e)} (h/2) \mathrm{d}\xi \tag{8.22}$$

by change of variable using the Jacobian $(h/2)$. Since the integrand consists of at most quadratic polynomials in ξ, this integral can be evaluated exactly with a 2-point Gauss numerical integration rule. Analytical integration is also easy, and yields the well-known result

$$\mathbf{K}^e = \frac{EI}{h^3} \begin{bmatrix} 12 \, , & -6h \, , & -12 \, , & -6h \\ -6h \, , & 4h^2 \, , & 6h \, , & 2h^2 \\ -12 \, , & 6h \, , & 12 \, , & 6h \\ -6h \, , & 2h^2 \, , & 6h \, , & 4h^2 \end{bmatrix} \tag{8.23}$$

8.2 Bending of beam in general orientation

So far we have been dealing with a beam element responding with transverse bending, oriented in the special frame as shown in Figure 8.1. A general orientation of the beam is easily accommodated: refer to Figure 8.3. We define a unit vector $\mathbf{e}_x$ along the axis of the beam, pointing from the first to the second node. Then we define another unit vector $\mathbf{e}_z$, orthogonal to $\mathbf{e}_x$, which points along the local deflection degrees of freedom $W_{\underline{1}}$ and $W_{\underline{3}}$. The global degrees of freedom in a Cartesian coordinate system XY (X is horizontal in Figure 8.3) are the translations $(\mathbf{U})_X$ (this is called `U1` in Abaqus) and $(\mathbf{U})_Y$ (`U2`), and the rotation (counterclockwise, referred to as `UR3` in Abaqus). We can see that in order to get $W_{\underline{3}}$ all we have to do is to project the global displacement vector $\mathbf{U}$ into the direction $\mathbf{e}_z$

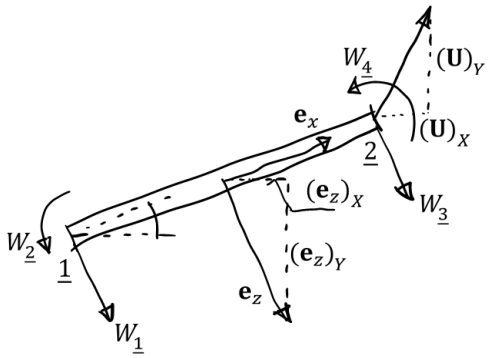


Fig. 8.3. Geometrical relationships for the generally-oriented 2D beam model.

$$W_{\underline{3}} = \mathbf{e}_z \cdot \mathbf{U}_{\underline{2}} = [\mathbf{e}_z]^T \left[\mathbf{U}_{\underline{2}}\right] = \left[\, (e_z)_X, (e_z)_Y \,\right] \begin{bmatrix} (\mathbf{U}_{\underline{2}})_X \\ (\mathbf{U}_{\underline{2}})_Y \end{bmatrix} \tag{8.24}$$

and similarly for $W_{\underline{1}}$. We can substitute this result into the expression for the curvature of the beam

$$w''(\xi) = (2/h)^2 \left[\frac{\mathrm{d}N^2_{\underline{1}}(\xi)}{\mathrm{d}\xi^2} , (h/2)\frac{\mathrm{d}N^2_{\underline{2}}(\xi)}{\mathrm{d}\xi^2} , \frac{\mathrm{d}N^2_{\underline{3}}(\xi)}{\mathrm{d}\xi^2} , (h/2)\frac{\mathrm{d}N^2_{\underline{4}}(\xi)}{\mathrm{d}\xi^2} \right] \begin{bmatrix} W_{\underline{1}} \\ W_{\underline{2}} \\ W_{\underline{3}} \\ W_{\underline{4}} \end{bmatrix} \tag{8.25}$$

$$= (2/h)^2 \left[\frac{\mathrm{d}N^2_{\underline{1}}(\xi)}{\mathrm{d}\xi^2} , (h/2)\frac{\mathrm{d}N^2_{\underline{2}}(\xi)}{\mathrm{d}\xi^2} , \frac{\mathrm{d}N^2_{\underline{3}}(\xi)}{\mathrm{d}\xi^2} , (h/2)\frac{\mathrm{d}N^2_{\underline{4}}(\xi)}{\mathrm{d}\xi^2} \right] \begin{bmatrix} [\mathbf{e}_z]^T \begin{bmatrix} (\mathbf{U}_{\underline{1}})_X \\ (\mathbf{U}_{\underline{1}})_Y \end{bmatrix} \\ W_{\underline{2}} \\ [\mathbf{e}_z]^T \begin{bmatrix} (\mathbf{U}_{\underline{2}})_X \\ (\mathbf{U}_{\underline{2}})_Y \end{bmatrix} \\ W_{\underline{4}} \end{bmatrix} \tag{8.26}$$

$$= (2/h)^2 \left[[\mathbf{e}_z]^T \frac{\mathrm{d}N^2_{\underline{1}}(\xi)}{\mathrm{d}\xi^2} , (h/2)\frac{\mathrm{d}N^2_{\underline{2}}(\xi)}{\mathrm{d}\xi^2} , [\mathbf{e}_z]^T \frac{\mathrm{d}N^2_{\underline{3}}(\xi)}{\mathrm{d}\xi^2} , (h/2)\frac{\mathrm{d}N^2_{\underline{4}}(\xi)}{\mathrm{d}\xi^2} \right] \begin{bmatrix} (\mathbf{U}_{\underline{1}})_X \\ (\mathbf{U}_{\underline{1}})_Y \\ W_{\underline{2}} \\ (\mathbf{U}_{\underline{2}})_X \\ (\mathbf{U}_{\underline{2}})_Y \\ W_{\underline{4}} \end{bmatrix} \tag{8.27}$$

$$\tag{8.28}$$

Note that in this way we have expressed the curvature of the beam in terms of the six degrees of freedom (in the vector on the right), and a strain-displacement matrix with one row and six columns:

$$\mathbf{B}^{(e)} = (2/h)^2 \left[[\mathbf{e}_z]^T \frac{\mathrm{d}N^2_{\underline{1}}(\xi)}{\mathrm{d}\xi^2} , (h/2)\frac{\mathrm{d}N^2_{\underline{2}}(\xi)}{\mathrm{d}\xi^2} , [\mathbf{e}_z]^T \frac{\mathrm{d}N^2_{\underline{3}}(\xi)}{\mathrm{d}\xi^2} , (h/2)\frac{\mathrm{d}N^2_{\underline{4}}(\xi)}{\mathrm{d}\xi^2} \right] . \tag{8.29}$$

The matrix (8.29) when substituted into the expression for the element stiffness matrix (8.22) yields a formula which looks identical

$$\mathbf{K}^e = \int_{-1}^{+1} \mathbf{B}^{(e)^T} EI \mathbf{B}^{(e)} (h/2) \mathrm{d}\xi \,, \tag{8.30}$$

but the stiffness matrix is of dimension 6×6. Each of the strain-displacement matrices is a linear function of ξ. Their product is therefore a quadratic function of ξ. In order to integrate those expressions from -1 to $+1$ numerically, we can employ a step up from a single-point Gauss rule, the two point Gauss rule.

8.3 Beam-column (frame) element

As we mentioned earlier, speaking of a "beam" element usually means that both axial and bending deformation is expected to be accounted for. Here we will equip the bending beam element with the axial response to get a beam-column or frame finite element.

The finite element stiffness matrix (8.30) describes only an element which responds to transverse bending. In order to get a full beam-column response we combine (superimpose) the rod finite element stiffness matrix (7.25) with that of (8.30). Of course, we cannot just add those two matrices together, they have different dimensions, what we mean is we can assemble both of these two matrices into the global structural matrix. Figure 8.4 explains this concept. Consider that the degrees of freedom are numbered as in (8.25). Then the axial stiffness will fill in the 16 entries (elements of the matrix) as shown on the left, because the axial stiffness does not react to rotations, and translations of the joints of a truss do not generate any moments. On the other hand, both translations and rotations of joints generate forces and moments acting on joints, which is why the bending stiffness is fully populated.

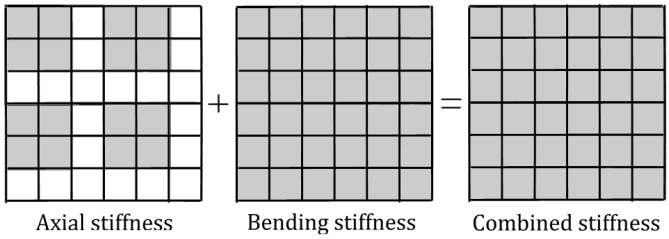


Fig. 8.4. Two dimensional beam model. Composition of the axial stiffness matrix with the bending stiffness matrix.

We are using here the fact that the axial response of the rod element is fully uncoupled from the transverse bending response of the beam element developed above. The uncoupling is due to our use of the centroid axis as the reference axis for the beam bending.

8.4 Dynamics: mass matrix

The mass matrix for the truss element in two dimensions was derived in Section 7.4. Here the treatment must be slightly different. For the truss, the approximation of the axial and transverse displacements could be described by the same linear basis functions. Contrariwise, for the beam-column the axial deformation is described by linear functions, but the transverse displacement is described by the Hermite cubic functions. Therefore, we need to work in the local coordinate system first, and then introduce the global coordinate system by projecting the global displacements onto the local Cartesian basis.

The kinetic energy of a vibrating beam element can be written as

$$KE = \frac{1}{2}\int_0^h \rho S\left(\dot{u}^2 + \dot{w}^2\right)\, dx = \frac{1}{2}(h/2)\int_{-1}^{+1} \rho S\left(\dot{u}^2 + \dot{w}^2\right)\, d\xi \tag{8.31}$$

where S is the cross sectional area of the beam, ρS is the mass density of the bar per unit length (mass density of the material times the cross sectional area), $\dot{u}$ is the x component of the velocity and $\dot{w}$ is the z component of the velocity. Note that the u and w components are given on the basis of the local Cartesian coordinate system, $\mathbf{e}_x$ and $\mathbf{e}_z$. Again, as for the truss, the integral over the

length of the element is converted into an integral over the standard shape (interval -1 to +1) by change of variables using the Jacobian $h/2$.

For the axial vibration, we write $\dot{u} = N_{\underline{1}}\dot{u}_{\underline{1}} + N_{\underline{2}}\dot{u}_{\underline{2}}$, and so the part of the kinetic energy due to axial vibration may be written as

$$KE = \frac{1}{2}(h/2)\int_{-1}^{+1} \rho S \dot{u}^2 \, d\xi = \frac{1}{2}\left[\, \dot{u}_{\underline{1}} \; \dot{u}_{\underline{2}} \,\right](h/2)\int_{-1}^{+1} \rho S \mathbf{N}_u^T \mathbf{N}_u \, d\xi \begin{bmatrix} \dot{u}_{\underline{1}} \\ \dot{u}_{\underline{2}} \end{bmatrix} \tag{8.32}$$

where

$$\mathbf{N}_u = \left[\, N_{\underline{1}} \; N_{\underline{2}} \,\right] \tag{8.33}$$

Since the velocity vector at the two joints is $\dot{\mathbf{U}}_{\underline{k}}$, $k = 1, 2$, the local velocities are calculated as $\dot{u}_{\underline{k}} = \mathbf{e}_x \cdot \dot{\mathbf{U}}_{\underline{k}}$. We can introduce this by expanding the matrix $\mathbf{N}_u$ and writing the kinetic energy of axial vibration (8.32) as

$$KE = \frac{1}{2}\left[\, \dot{\mathbf{U}}_{\underline{1}} \; \dot{\mathbf{U}}_{\underline{2}} \,\right](h/2)\int_{-1}^{+1} \rho S \mathbf{N}_u^T \mathbf{N}_u \, d\xi \begin{bmatrix} \dot{\mathbf{U}}_{\underline{1}} \\ \dot{\mathbf{U}}_{\underline{2}} \end{bmatrix} \tag{8.34}$$

where we recycle the symbol for the $\mathbf{N}_u$ matrix, which now incorporates the projection onto $\mathbf{e}_x$

$$\mathbf{N}_u = \left[\, N_{\underline{1}} \times [\mathbf{e}_x]^T \;,\; N_{\underline{2}} \times [\mathbf{e}_x]^T \,\right] \tag{8.35}$$

The mass matrix corresponding to axial vibration

$$\mathbf{M}_u = (h/2)\int_{-1}^{+1} \rho S \mathbf{N}_u^T \mathbf{N}_u \, d\xi \;, \tag{8.36}$$

is of dimension 4×4. It is possible to incorporate all degrees of freedom at a node, which for the beam includes the rotation, by applying one further expansion to the matrix $\mathbf{N}_u$

$$\mathbf{N}_u = \left[\, N_{\underline{1}} \times [\mathbf{e}_x]^T, \, 0 \;,\; N_{\underline{2}} \times [\mathbf{e}_x]^T, \, 0 \,\right] \tag{8.37}$$

so that it has six columns. Then the matrix (8.36) will have six rows and six columns.

For the transverse vibration of the beam we use an analogy of the equation (8.8),

$$\dot{w} = N_{\underline{1}}\dot{W}_{\underline{1}} + (h/2)N_{\underline{2}}\dot{W}_{\underline{2}} + N_{\underline{3}}\dot{W}_{\underline{3}} + (h/2)N_{\underline{4}}\dot{W}_{\underline{4}} = \left[\, N_{\underline{1}}, \, (h/2)N_{\underline{2}}, \, N_{\underline{3}}, \, (h/2)N_{\underline{4}} \,\right] \begin{bmatrix} \dot{W}_{\underline{1}}, \\ \dot{W}_{\underline{2}}, \\ \dot{W}_{\underline{3}}, \\ \dot{W}_{\underline{4}} \end{bmatrix} \tag{8.38}$$

Note that here $\dot{w} = \dot{w}(\xi)$ is the transverse velocity of vibration, and we use the symbol $N_{\underline{k}} = N_{\underline{k}}(\xi)$ to refer to the Hermite shape functions (refer to Figure 8.2). Recall that the degrees of freedom are defined as shown in Figure 8.1. The degrees of freedom $\dot{W}_{\underline{1}}$ and $\dot{W}_{\underline{3}}$ are transverse translation velocities, in the local coordinate system (in the direction of $\mathbf{e}_z$), and $\dot{W}_{\underline{2}}$ and $\dot{W}_{\underline{4}}$ are rotation velocities. So introducing the matrix

$$\mathbf{N}_w = \left[\, N_{\underline{1}}, \, (h/2)N_{\underline{2}}, \, N_{\underline{3}}, \, (h/2)N_{\underline{4}} \,\right] \tag{8.39}$$

we can write the kinetic energy of transverse vibration of the beam as

$$KE = \frac{1}{2}\left[\, \dot{W}_{\underline{1}}, \, \dot{W}_{\underline{2}}, \, \dot{W}_{\underline{3}}, \, \dot{W}_{\underline{4}} \,\right](h/2)\int_{-1}^{+1} \rho S \mathbf{N}_w^T \mathbf{N}_w \, d\xi \begin{bmatrix} \dot{W}_{\underline{1}}, \\ \dot{W}_{\underline{2}}, \\ \dot{W}_{\underline{3}}, \\ \dot{W}_{\underline{4}} \end{bmatrix} \tag{8.40}$$

As above, we will project the displacements of the nodes $\dot{\mathbf{U}}_{\underline{k}}$ but this time onto $\mathbf{e}_z$. The result is the matrix

$$\mathbf{N}_w = \left[N_{\underline{1}} \times [\mathbf{e}_z]^T,\, (h/2)N_{\underline{2}} \,,\, N_{\underline{2}} \times [\mathbf{e}_z]^T,\, (h/2)N_{\underline{4}} \right] \tag{8.41}$$

which can be finally used to express the contribution of the transverse vibration of the beam to the 6×6 mass matrix

$$\mathbf{M} = (h/2) \int_{-1}^{+1} \rho S \mathbf{N}_w^T \mathbf{N}_w \, d\xi \,. \tag{8.42}$$

The overall mass matrix of the beam-column to include both the axial and the transfers vibration is the sum of two 6×6 matrices, (8.36) and (8.42).

The transverse-vibration mass matrix contains products of one basis function with another basis function, where the basis functions are cubic polynomials of the independent variable ξ. Therefore, a higher order Gauss integration needs to be employed, at a minimum 4-point rule (capable of integrating exactly polynomials of degree $2 \times 4 - 1 = 7$).

8.5 Python implementation of 2D frame finite elements

- URL

The Python package `pystran` implements models of two dimensional beam-column (planar frame) structures. The functions that compute the strain-displacement and stiffness matrices can be readily recognized in the code itself.

9

Torsion rod finite element

Summary

1. The finite element model of the torsion rod is derived by reference to the analogous truss FE.
2. The resulting stiffness matrix may be employed as a constituent part to form a general 3D beam.

9.1 Rod finite element for uniform torsion in basic orientation

The balance equation for the truss was written within the volume of the member, in terms of the stress as shown in (7.1). It is possible to write it in terms of the axial force

$$\frac{\partial N}{\partial x} = S\frac{\partial \sigma_x}{\partial x} = ES\frac{\partial^2 u}{\partial x^2} = 0 \tag{9.1}$$

where $N = S\sigma_x$. Then the residual equation would involve integration over the length of the truss member rather than the volume

$$\int_0^L \eta SEu'' \mathrm{d}x = 0 \tag{9.2}$$

which is precisely the same as (7.4).

The balance equation of the torsion rod (4.50) appears to be analogous to (9.1), and the weighted residual statement is therefore obtained in a similar way by taking the balance equation and multiplying with the test function, and integrating the product over the length of the rod

$$\int_0^L \eta GJ\phi'' \mathrm{d}x = 0 \tag{9.3}$$

Notice the direct analogy with (9.2). Integration by parts yields

$$\int_0^L \eta GJ\phi'' \mathrm{d}x = \int_0^L \left(\eta GJ\phi'\right)' \mathrm{d}x - \int_0^L \eta' GJ\phi' \mathrm{d}x = \left[\eta GJ\phi'\right]_0^L - \int_0^L \eta' GJ\phi' \mathrm{d}x = 0 \tag{9.4}$$

The above may be particularized given certain boundary conditions. At the end where the torque is prescribed, $GJ\phi' = T$ is the applied torque and the test function $\eta = 1$; otherwise, the end is given a particular value of rotation, and the test function vanishes at that end, $\eta = 0$, and the unknown torque is the reaction.

The term $\int_0^L \eta' GJ\phi' \mathrm{d}x$ leads to the stiffness matrix of the rod entirely analogously to the truss member stiffness matrix: The twisting rotation along the rod is written with the finite element basis functions as

$$\phi = N_{\underline{1}}(\xi)\phi_{\underline{1}} + N_{\underline{2}}(\xi)\phi_{\underline{2}} \,. \tag{9.5}$$

where the basis functions are as shown in Figure 11.1.

The above may be expressed using the "strain-displacement" matrix (rate of twist versus joint rotation). We express the rate of twist, ϕ', as

$$\phi' = \begin{bmatrix} -1/h \ , \ +1/h \end{bmatrix} \begin{bmatrix} \phi_{\underline{1}} \\ \phi_{\underline{2}} \end{bmatrix} \tag{9.6}$$

The matrix

$$\mathbf{B}^{(e)} = \begin{bmatrix} -1/h \ , \ +1/h \end{bmatrix} \tag{9.7}$$

is the strain-displacement matrix for the two-node torsion rod finite element in basic orientation. It has a single row, and two columns: two columns because there are two rotation degrees of freedom. Now we write for the rate of the test function

$$\eta' = \mathbf{B}^{(e)} \begin{bmatrix} \eta_{\underline{1}} \\ \eta_{\underline{2}} \end{bmatrix} \tag{9.8}$$

and then the product $\eta' GJ \phi'$ may be stated as

$$\eta' GJ \phi' = \begin{bmatrix} \eta_{\underline{1}} \\ \eta_{\underline{2}} \end{bmatrix}^T {\mathbf{B}^{(e)}}^T GJ \mathbf{B}^{(e)} \begin{bmatrix} \phi_{\underline{1}} \\ \phi_{\underline{2}} \end{bmatrix} \tag{9.9}$$

Using the strain-displacement matrix, the stiffness matrix of the torsion rod basic finite element may be written

$$\mathbf{K}^{(e)} = GJ \int_0^h {\mathbf{B}^{(e)}}^T \mathbf{B}^{(e)} \mathrm{d}x = \frac{GJ}{h} \begin{bmatrix} +1 \ , \ -1 \\ -1 \ , \ +1 \end{bmatrix} \tag{9.10}$$

Compare visually with (7.14).

9.2 Torsion rod in general orientation

We proceed entirely analogously to the Section 7.2: The local axial rotation $\phi_{\underline{j}}$ is obtained by projection of the global joint rotation vector at node $\underline{j}$, $\mathbf{\Phi}_{\underline{j}}$, onto the direction of $\mathbf{e}_x$ aligned with the axis of the rod

$$\phi_{\underline{j}} = [\mathbf{e}_x]^T \begin{bmatrix} (\mathbf{\Phi}_{\underline{j}})_X \\ (\mathbf{\Phi}_{\underline{j}})_Y \end{bmatrix} \tag{9.11}$$

Torsion really makes sense only in three dimensions. The unit vector $\mathbf{e}_x$ will have three components in three dimensions. The matrix

$$\mathbf{B}^{(e)} = \begin{bmatrix} (-1/h)[\mathbf{e}_x]^T \ , \ (+1/h)[\mathbf{e}_x]^T \end{bmatrix} \tag{9.12}$$

will then have one row and six columns. It is the strain-displacement matrix for the two-node torsion rod finite element arbitrarily oriented in the global Cartesian coordinate system XYZ.

The stiffness matrix of a torsion rod element follows from (9.10), and will be a 6×6 matrix. The point of this finite element stiffness matrix is mainly that it can be combined with bending and axial stiffness matrices to form the stiffness matrix of a general beam in three dimensions; occasionally this can also be employed for grids (grillages) or for structures composed entirely of torsion springs.

10

Bernoulli beam finite element in 3D space

Summary

1. We explain the derivation of the space beam from the principle of superposition: the stiffness matrices for four deformation mechanisms will be superimposed.

10.1 Bernoulli beam in three dimensions

Figure 10.1 shows the conventions adopted in this book regarding the directions of the internal resultant forces and moments. Also, the balance equations are displayed. As shown, distributed moment loading is ignored as it is uncommon in practice.

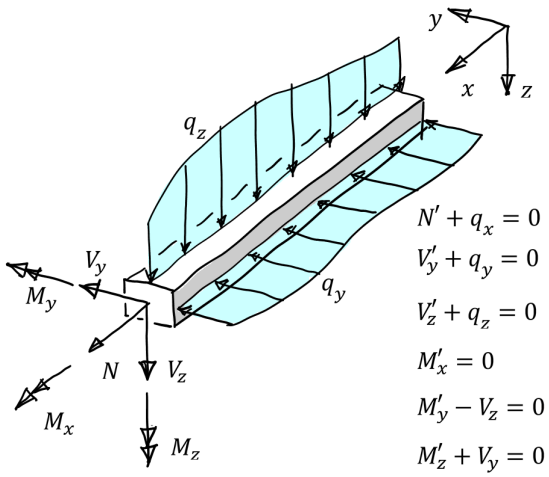


Fig. 10.1. Resultant force and moment conventions for the three dimensional beam model.

A straight beam member has a great advantage in that deformation modes can be (possibly completely) uncoupled. In particular, here we will consider the case of doubly symmetric cross section profile, and only linearly elastic materials. If the axial action (tensile/compressive force), twisting action (torsion), and bending in two principal axes of the profile, are uncoupled, the response of the beam can be considered a superposition of four totally separate deformation states.

First, let us talk about conditions under which the superposition is applicable. We observe:

1. Axial force and the bending will be decoupled when the response is described in a coordinate system placed at the centroid of the cross section.

2. The bending can be composed of two independent planar transverse-bending modes when the coordinate system is aligned with the principal planes of the cross section.
3. The torsion will by decoupled from bending when the transverse loading is applied at the shear center (centre of twist).
4. The torsion of the beam will be decoupled from the axial response when the cross sections can warp freely, or when the warping is relatively minor. This usually holds approximately true for solid cross sections.

We can see that 1 and 3 required simultaneously will be satisfied for a doubly symmetric solid section. For thin-walled sections we should also consider the importance of warping.

When the above conditions are met, the three dimensional beam element representing a straight member can be composed of the four models shown in Figure 10.2 (the notation is in this figure Abaqus-based, for practice). This superposition means that we can separately compute the four stiffness matrices of the constituent models: 6×6 stiffness matrix of the axial bar, and the 6×6 stiffness matrix of the torsion rod; further, two 12×12 stiffness matrices of the planar transverse-bending response for bending in two orthogonal planes (the principal planes of the cross section). Then those four matrices are assembled into single 12×12 stiffness matrix of a three dimensional beam element.

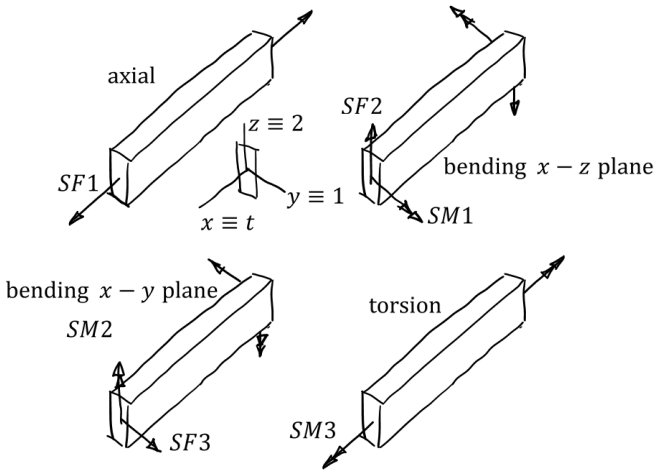


Fig. 10.2. Composition of four simple models, axial, bending in two planes, and torsional, into a single three-dimensional beam model.

First we will consider bending. There is a slight complication in that the relationship between bending moments (rotations) and shear forces (translations) necessitates some sign changes. Consider Figure 10.3, where we show the shapes of a beam bending in the two planes, $x - z$ and $x - y$. We give the beam a shape based on the translation of its endpoints in the positive direction, and the rotation of the endpoints such that they rotate by a positive angle. Note that the positive deflection is downwards for bending in the $x - z$ plane, and vice versa for the bending in the $x - y$ plane. The basis functions N_1 and N_3 are the same (given that the axes were flipped), but the basis functions for the rotations look the same, even though the axes were flipped. Therefore, for bending in the $x - z$ plane we use N_2 and N_4, while for the bending in the $x - y$ we must use $-N_2$ and $-N_4$.

10.1.1 Bending in the $x - z$ plane

For bending of the three dimensional beam in the $x - z$ plane we can essentially reuse the matrix developed for the two dimensional beam in (8.29). However, now we need to take into account the

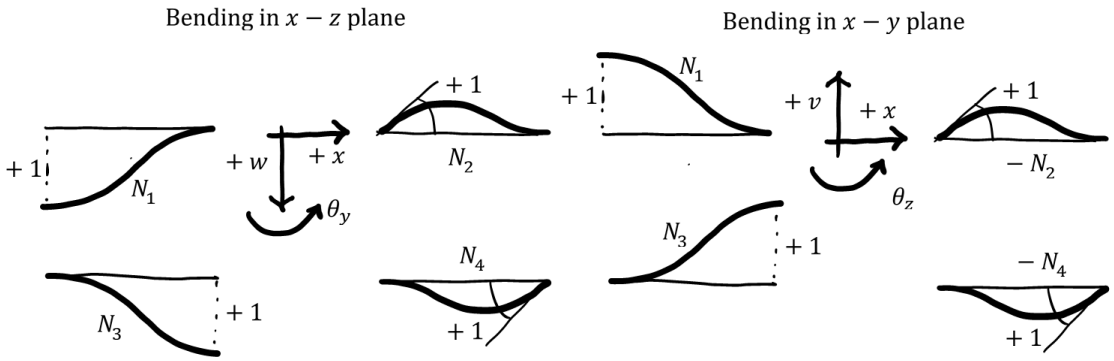


Fig. 10.3. The four beam shape functions for bending in the two principal planes, $x-z$ and $x-y$. Note well that the w deflection is positive downwards (along z), and, contrariwise, the v deflection is positive upwards (along y). In both figures the positive rotation is counterclockwise ("about" the third cartesian axis).

general orientation of this element-local plane in the global three-dimensional Cartesian space: the translation along the local z axis is still obtained by projecting the joint displacement into the direction of $\mathbf{e}_z$ (which will now have three components instead of two). The direction of the rotation in the two dimensional case was given as about the normal to the "paper" (screen), but now we need to explicitly specify that the rotation to which the beam reacts is a projection of the three-dimensional rotation of the joint into the direction $\mathbf{e}_y$ (which is the normal to the plane $x-z$). As a result we obtain the following matrix that produces curvature from the three-dimensional translations and rotations

$$\mathbf{B}^{(e)} = (2/h)^2 \left[[\mathbf{e}_z]^T \frac{\mathrm{d}N_{\underline{1}}^2(\xi)}{\mathrm{d}\xi^2} \, , [\mathbf{e}_y]^T \, (h/2) \frac{\mathrm{d}N_{\underline{2}}^2(\xi)}{\mathrm{d}\xi^2} \, , [\mathbf{e}_z]^T \frac{\mathrm{d}N_{\underline{3}}^2(\xi)}{\mathrm{d}\xi^2} \, , [\mathbf{e}_y]^T \, (h/2) \frac{\mathrm{d}N_{\underline{4}}^2(\xi)}{\mathrm{d}\xi^2} \right] \tag{10.1}$$

Now this is a matrix of one row and twelve columns. The stiffness matrix expression still looks the same as (8.30)

$$\mathbf{K}^e = \int_{-1}^{+1} {\mathbf{B}^{(e)}}^T EI \mathbf{B}^{(e)} (h/2) \mathrm{d}\xi \, , \tag{10.2}$$

but this matrix is 12×12.

10.1.2 Bending in the $x-y$ plane

The curvature-displacement matrix for bending in the $x-y$ plane sees the roles of the basis vectors swapped ($\mathbf{e}_y$ extracts the transverse displacement, and $\mathbf{e}_z$ dotted with the vector of rotations produces the rotation in the local coordinate system). Also note the change of the sign in front of the terms with N_2 and N_4:

$$\mathbf{B}^{(e)} = (2/h)^2 \left[[\mathbf{e}_y]^T \frac{\mathrm{d}N_{\underline{1}}^2(\xi)}{\mathrm{d}\xi^2} \, , - [\mathbf{e}_z]^T \, (h/2) \frac{\mathrm{d}N_{\underline{2}}^2(\xi)}{\mathrm{d}\xi^2} \, , [\mathbf{e}_y]^T \frac{\mathrm{d}N_{\underline{3}}^2(\xi)}{\mathrm{d}\xi^2} \, , - [\mathbf{e}_z]^T \, (h/2) \frac{\mathrm{d}N_{\underline{4}}^2(\xi)}{\mathrm{d}\xi^2} \right] \tag{10.3}$$

The stiffness matrix corresponding to bending in the $x-y$ plane has the form of (10.2) (with a proper substitution of the curvature-displacement matrix).

Both (10.1) and (10.3) contain terms linear in ξ, their outer product therefore produces an integrand which is quadratic in ξ. The numerical integration used for the bending terms therefore needs to be able to handle such polynomials. A two-point Gauss integration rule can deal with it, as it is accurate up to cubic polynomials.

10.1.3 Axial response

The straight three dimensional beam responds to axial shortening or lengthening exactly as an truss member. The axial stiffness is therefore that of (7.27).

10.1.4 Torsional response

The torsional stiffness matrix was derived in Section 9.2.

10.1.5 Superposition

Figure 10.4 illustrates how the four stiffness matrices, axial, torsional, and bending in two planes, overlap. For that purpose, we assume that the degrees of freedom at each joint are ordered as three translations, followed by three rotations, the first joint followed by the second.

Therefore, the axial stiffness 6×6 matrix will be assembled into the 36 colored locations in the four 3×3 submatrices corresponding to the forces acting on the joints when the beam member joints are subjected to translations.

Analogously, the torsional stiffness 6×6 matrix will be assembled into the 36 colored locations in the four 3×3 submatrices corresponding to the moments acting on the joints when the beam member joints are subjected to rotations. Keep in mind that a twisting moment in the bar may result in moment components acting on the joint along all three global Cartesian coordinate axes, depending on the orientation of the beam member.

The two bending actions populate the entire 12×12 matrix, as for a general orientation, all forces and all moments acting on the joints are due to all the translations and rotations.

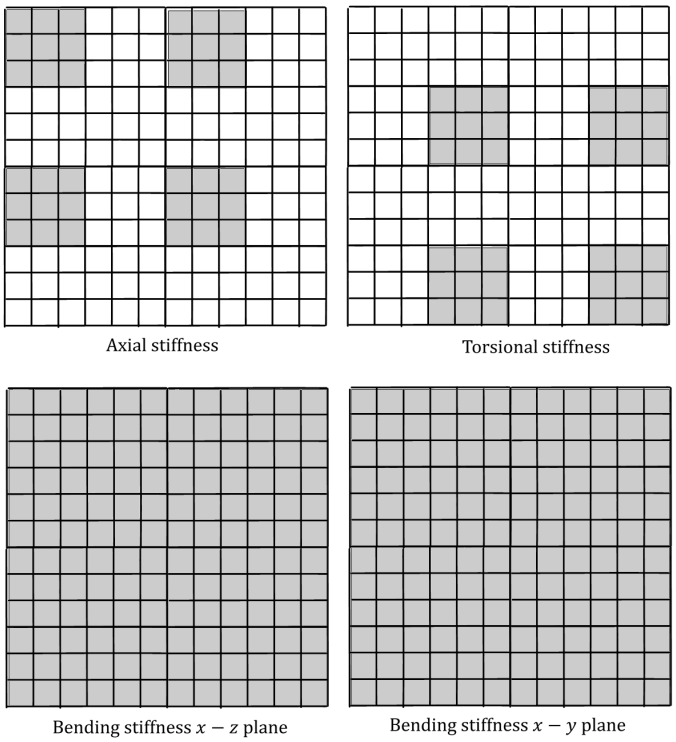


Fig. 10.4. Three dimensional beam model. Composition of the axial stiffness matrix with the bending stiffness matrices and the torsion stiffness matrix. These four 12×12 matrices can be simply added together to form the overall stiffness matrix of the beam (frame) finite element.

Abaqus offers the Bernoulli beam in two varieties: B23 is for planar analysis (the modeling space is two dimensional), and B33 is for three dimensional structures. The last digit (3) refers to the cubic approximation of the deflection line of a single element.

10.2 Coupled deformation

If it is possible to employ uncoupled deformation modes to model 3D beams, why would we need to contemplate coupled models? The reason lies in the convenience and flexibility of being able to define arbitrary cross sections, in arbitrary (non-principal) coordinates, including offsets of the reference line from the centroid. The software will take care of the necessary transformations the formulate a coupled axial/torsional/bending beam finite element.

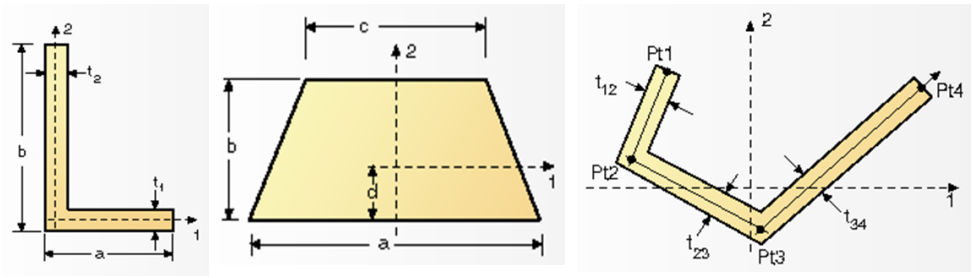


Fig. 10.5. Some cross sections that introduce non-principal, non-centroidal axes. The result is a beam model whose deformation modes are coupled (axial with bending, etc.).

11

Timoshenko beam finite element

Summary

1. The Timoshenko beam finite element model in basic orientation is formulated first.
2. The finite element model is then adjusted for a general orientation of the member.
3. Reduced integration is discussed as a means of eliminating locking.

11.1 Method of weighted residuals

The method of weighted residuals proceeds by multiplying both balance equations (4.15) by test functions (η_w, which would have the meaning of transverse displacement, and η_ϕ with the meaning of rotation), and by integrating over the length of the beam. We also apply integration by parts the shift some of the derivatives from the trial function to the test function. The first equation results as

$$0 = \int_0^L \eta_w k_s GS(\phi' + w'') \,\mathrm{d}x + \int_0^L \eta_w q \,\mathrm{d}x \tag{11.1}$$

$$= -\int_0^L \eta_w' k_s GS(\phi + w') \,\mathrm{d}x \; + [\eta_w k_s GS(\phi + w')]_0^L + \int_0^L \eta_w q \,\mathrm{d}x \tag{11.2}$$

$$= -\int_0^L \eta_w' k_s GS(\phi + w') \,\mathrm{d}x \; + \left[\eta_w \overline{V}\right]_0^L + \int_0^L \eta_w q \,\mathrm{d}x \tag{11.3}$$

We can trivially clean this up by switching the sign as

$$\int_0^L \eta_w' k_s GS(\phi + w') \,\mathrm{d}x \; - \left[\eta_w \overline{V}\right]_0^L - \int_0^L \eta_w q \,\mathrm{d}x = 0 \tag{11.4}$$

The second equation follows as

$$0 = \int_0^L \eta_\phi EI\phi'' \,\mathrm{d}x - \int_0^L \eta_\phi k_s GS(\phi + w') \,\mathrm{d}x \tag{11.5}$$

$$= -\int_0^L \eta_\phi' EI\phi' \,\mathrm{d}x + [\eta_\phi EI\phi']_0^L - \int_0^L \eta_\phi k_s GS(\phi + w') \,\mathrm{d}x \tag{11.6}$$

$$= -\int_0^L \eta_\phi' EI\phi' \,\mathrm{d}x + \left[\eta_\phi \overline{M}\right]_0^L - \int_0^L \eta_\phi k_s GS(\phi + w') \,\mathrm{d}x \tag{11.7}$$

And, cleaned up with a change of the sign the above will yield

$$\int_0^L \eta_\phi' EI\phi' \,\mathrm{d}x + \int_0^L \eta_\phi k_s GS(\phi + w') \,\mathrm{d}x - \left[\eta_\phi \overline{M}\right]_0^L = 0 \tag{11.8}$$

The first term in (11.4) and the first and second term in (11.8) will result in contributions to the stiffness matrix.

Using a finite element approximation, we write for the slope of the deflection curve w'

$$w' = \begin{bmatrix} N'_{\underline{1}} \, , \, 0 \, , \, N'_{\underline{3}} \, , \, 0 \end{bmatrix} \begin{bmatrix} W_{\underline{1}} \\ 0 \\ W_{\underline{3}} \\ 0 \end{bmatrix} = [\mathbf{b}_w] \begin{bmatrix} W_{\underline{1}} \\ 0 \\ W_{\underline{3}} \\ 0 \end{bmatrix} \tag{11.9}$$

for the rotation of the cross section ϕ

$$\phi = \begin{bmatrix} 0 \, , \, N_{\underline{2}} \, , \, 0 \, , \, N_{\underline{4}} \end{bmatrix} \begin{bmatrix} 0 \\ W_{\underline{2}} \\ 0 \\ W_{\underline{4}} \end{bmatrix} = [\mathbf{i}_\phi] \begin{bmatrix} 0 \\ W_{\underline{2}} \\ 0 \\ W_{\underline{4}} \end{bmatrix} \tag{11.10}$$

and for the curvature ϕ'

$$\phi' = \begin{bmatrix} 0 \, , \, N'_{\underline{2}} \, , \, 0 \, , \, N'_{\underline{4}} \end{bmatrix} \begin{bmatrix} 0 \\ W_{\underline{2}} \\ 0 \\ W_{\underline{4}} \end{bmatrix} = [\mathbf{b}_\phi] \begin{bmatrix} 0 \\ W_{\underline{2}} \\ 0 \\ W_{\underline{4}} \end{bmatrix} \tag{11.11}$$

We shall use the equivalence of the test functions and the trial functions, $\eta_w \sim w$, and $\eta_\phi \sim \phi$, to formally write the stiffness matrix as

$$\mathbf{K}^e = \tag{11.12}$$

$$\begin{aligned}
&\int_0^h \mathbf{b}_w^T k_s GS \, \mathbf{i}_\phi \, \mathrm{d}x \qquad \text{[first term from (11.4)]} \\
+ &\int_0^h \mathbf{b}_w^T k_s GS \, \mathbf{b}_w \, \mathrm{d}x \qquad \text{[first term from (11.4)]} \\
+ &\int_0^h \mathbf{b}_\phi^T EI \, \mathbf{b}_\phi \, \mathrm{d}x \qquad \text{[first term from (11.8)]} \\
+ &\int_0^h \mathbf{i}_\phi^T k_s GS \, \mathbf{i}_\phi \, \mathrm{d}x \qquad \text{[second term from (11.8)]} \\
+ &\int_0^h \mathbf{i}_\phi^T k_s GS \, \mathbf{b}_w \, \mathrm{d}x \qquad \text{[second term from (11.8)]}
\end{aligned} \tag{11.13}$$

11.2 Linear Timoshenko beam element

Now we derive the simplest Timoshenko finite element. It has two nodes, and the basis functions are linear (refer to Figure 11.1). The basis functions associated to degrees of freedom $\underline{2}$ and $\underline{4}$ (rotations) are the same as basis functions for the first two degrees of freedom (deflections): $N_{\underline{2}} = N_{\underline{1}}$ and $N_{\underline{4}} = N_{\underline{3}}$ The matrices in (11.12) are in general computed with numerical integration. The highest order polynomial is quadratic (in the fourth term), and a two-point Gauss quadrature rule would be perfectly adequate to integrate everything "exactly". However, as is known today, exact integration here is detrimental. This is illustrated as follows: consider a very thin beam, such that the shear strain (4.13) should be zero. We interpolate w' with (11.9) and the rotation with (11.10) and we set the shear strain to zero:

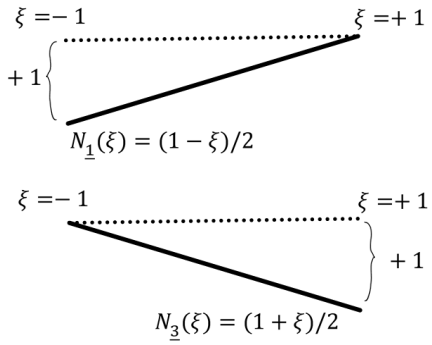


Fig. 11.1. Linear basis functions to be used with the Timoshenko beam element.

$$\phi + w' = N_1' W_1 + N_3' W_3 + N_2 W_2 + N_4 W_4 \tag{11.14}$$
$$= (-1/h) W_1 (1/h) W_3 + (1-\xi)/2 W_2 + (1+\xi)/2 W_4 \tag{11.15}$$
$$= (W_3 - W_1)/h + (W_2 + W_4)/2 + \xi(-W_2 + W_4)/2 \tag{11.16}$$
$$= 0 \tag{11.17}$$

Because the resulting expression is a linear polynomial in ξ, we must require both

$$(W_3 - W_1)/h + (W_2 + W_4)/2 = 0 \tag{11.18}$$
$$-W_2 + W_4 = 0 \tag{11.19}$$

The second equation insists $W_2 = W_4$, in other words the rotation must be the same everywhere within the finite element, which unfortunately means the beam cannot carry any moment (because of (4.12)). Another way of looking at it is by considering the displacement and the rotations as being rigidly linked: Now think of some support condition, either prescribed zero displacement or prescribed zero rotation. The consequence then is that the beam does not deform at all. We say that the beam element *locks*.

The reason of this unfortunate locking behavior is the mismatch between the interpolation of the slopes and the interpolation of the rotations. Note however that at the midpoint of the beam, i.e. at $\xi = 0$, Equation (11.14) would no longer impose that the rotation must be the same, since the term multiplied by ξ would drop out. This is the idea behind using an inexact numerical integration rule to evaluate the stiffness matrix (11.12). A single-point Gauss integration rule cannot deal exactly with all the terms, but that is good, as it manages to avoid locking. The result is

$$\mathbf{K}^e = \frac{k_s G S}{h} \begin{bmatrix} +1\,, & -h/2\,, & -1\,, & -h/2 \\ -h/2\,, & \left(\dfrac{EI}{k_s GS} + \dfrac{h^2}{4}\right), & h/2\,, & \left(-\dfrac{EI}{k_s GS} + \dfrac{h^2}{4}\right) \\ -1\,, & h/2\,, & +1\,, & +h/2 \\ -h/2\,, & \left(-\dfrac{EI}{k_s GS} + \dfrac{h^2}{4}\right), & +h/2\,, & \left(\dfrac{EI}{k_s GS} + \dfrac{h^2}{4}\right) \end{bmatrix} \tag{11.20}$$

We call finite elements that use inexact integration rules "reduced integration" elements. The beam elements in Abaqus with linear approximation of the deflections and rotations are labelled B21 and B31. The "1" refers to the linear approximation functions, whereas the other digit refers to the dimension of the modeling space.

11.3 Quadratic Timoshenko beam element

The Timoshenko beam elements in Abaqus are formulated so that they are efficient for thin beams, where Bernoulli-Euler theory is accurate, as well as for thick beams. The linear approximation

beams are preferable in contact problems, but otherwise the quadratic variety is preferred due to increased accuracy. The quadratic elements can also take initial curvature into account by placing the midpoint off the straight line between the two endpoints (Figure 11.2). The quadratic Timoshenko beam elements (B22, B32) are the most effective beam elements available in Abaqus.

Figure 11.2 illustrates the positions of the nodes and of the integration points along the reference line of the beam. We can see that the Timoshenko beam element is integrated at fewer integration points than expected, given the degree of the basis functions. Compare with the Bernoulli-Euler element, which is integrated at three points.

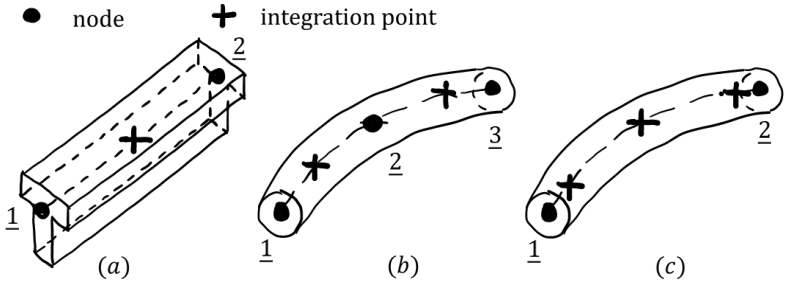


Fig. 11.2. Beam integration points. (a) Timoshenko beam element with two nodes, B31. (b) Timoshenko beam element with three nodes, B32. (c) Bernoulli-Euler beam element, B33.

The Abaqus finite elements that implement the Timoshenko beam model (B21, B22, B31, B32) are all reduced integration elements. They use equal interpolation for the rotations and the transverse deflections, and the reduced integration ensures the resulting elements are accurate by preventing locking.

In three dimensional analysis, the definition of the local coordinate system at each integration point is based on the tangent vector to the reference line (curve), and the are two vectors $\mathbf{n}_1$ and $\mathbf{n}_2$, which initially lie in a cross section plane orthogonal to the reference curve. Figure 11.3 illustrates this concept for a straight and curved beam.

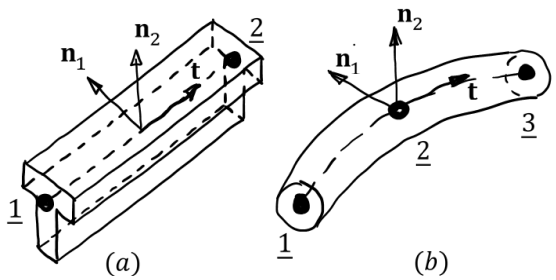


Fig. 11.3. Linear and Quadratic Timoshenko beam element. Beam orientation: (a) linear, (b) quadratic.

11.4 Integration and section points

If a beam remains elastic, its stiffness characteristics can be computed just once during the analysis. On the other hand, in the presence of inelastic deformation, for instance plasticity, the stiffness

properties of the material in different parts of the beam, both along its length and in its cross section, will change with the deformation. Therefore, it is quite important for numerical integration rules to be sufficiently flexible and powerful to track correctly the evolution of the stresses.

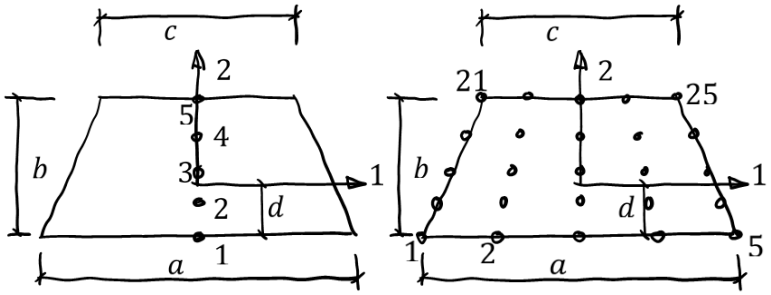


Fig. 11.4. Section points in a trapezoidal profile. 2D beam on the left, 3D beam on the right.

Figure 11.2 shows the nodes and the integration points for the Timoshenko and Bernoulli-Euler beam elements. Each integration point along the length of the beam is then associated with an array of section points: quadrature points at which the properties of the material and the profile are evaluated. As an example, Figure 11.4 illustrates the section points used in two dimensional and three dimensional analysis for a beam with a trapezoidal profile. The output of stress (and other quantities) can then be provided at a selected number of section points. The default choice for the trapezoid profile is output at section points 1, 5, 21, and 25.

The manual documents the positions of section points and output points for other built-in profiles, for instance the common I profile (Figure 11.5).

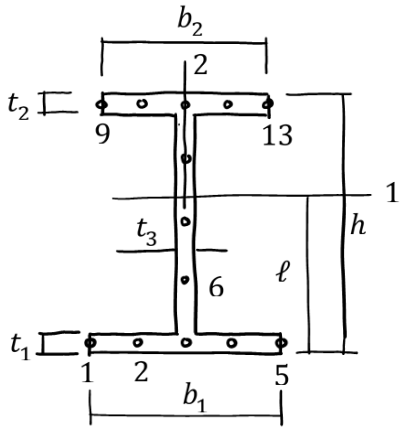


Fig. 11.5. Section points in an I profile. For analysis of a three-dimensional beam.

11.5 Guidelines for selecting beam elements

Several guidelines can be provided for selecting beam elements:

1. First-order, shear-deformable (Timoshenko) beam elements (B21, B31) should be used in any simulation that includes contact.

2. If the transverse shear deformation is important, use Timoshenko beam elements (B21, B22, B31, B32).
3. The Bernoulli-Euler (cubic) beams (B23, B33) available in Abaqus are very accurate for simulations that include distributed loading, such as dynamic vibration analyses.
4. Structures with open, thin-walled cross-sections should be modeled with elements that use open-section warping theory: refer to Chapter 12.

12

Thin-walled beam modeling

Summary

1. Limitations of thin-walled beam theory.
2. Closed-section beam modeling.
3. Open-section beam element, unconstrained and constrained torsion
4. Thin walled structures that cannot be modelled with regular beam elements.
5. We shall not cover the modeling of thin-walled beams exhaustively here. Please refer to [BaCr] for additional details.

12.1 Limitations of thin-walled beam theory

It is known that thin-walled beams are susceptible to local-deformation effects: connections between members, fixtures that transfer loading into the beam, or just buckling of compressed parts of the cross section, may lead to deformations that violate the assumption that the cross section shape is unchanging. Figure 12.1 shows a couple of types of deformation of thin-walled beam cross sections that can occur in reality, but that cannot be represented by the theory implemented in the Abaqus software. (There are some research codes that can handle these types of deformations, but they are not in general use.) If we suspect that a part of or the entire thin-walled beam may experience these types of local deformations of the cross section, we must employ shell models to represent this.

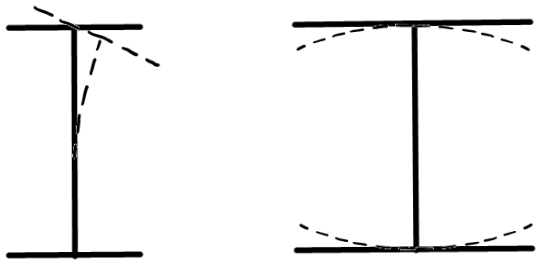


Fig. 12.1. Types of deformation that the beam theory cannot represent.

12.2 Closed-profile thin-walled beam modeling

Closed-profile thin-walled beams behave pretty much the same as solid-profile (rectangular, circular, ...) beams. The axial, torsion, and bending response follows familiar theory. (Local buckling of the

walls can occur and it is an exception to this rule.) In some respect, a rectangular box thin-walled beam under torsional loads is in fact simpler to analyze than a rectangular solid-profile beam: the torsional loading is transferred through a constant shear flow carried by the thin walls (Figure 12.2).

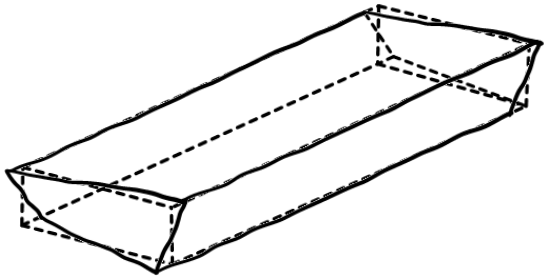


Fig. 12.2. Warping of a box thin-walled beam under torsion loading.

Any Abaqus beam element can be used to model a closed-profile thin-walled beam structure, but: Abaqus cannot handle closed-profile cross-sections that have multiple cells, or open-profile fins. Figure 12.3 shows a couple of examples. The profile on the left has two closed cells; the profile on the right has a single closed cell, but this cell is connected to two protruding fins. In these cases, the user needs to compute the cross section properties (area, second moments of area, torsion constant) offline, and then use the "Generalized" beam profile equipped with the computed properties.

Fig. 12.3. Types of sections that cannot be handled in Abaqus.

12.3 Open-profile thin-walled beam modeling

Figure 12.4 shows a few unsymmetric open-profile cross sections. Note that in it is not uncommon for these profiles to have inclined principal planes. The shear center will also rarely be at the centroid of the section.

The most significant characteristic of open-profile thin-walled beams is their extreme flexibility in torsion. As an example, consider a hollow tube of circular cross section, with D and d its outer and inner diameter, respectively (Figure 12.5). Its torsional constant is

$$J = \frac{\pi(D^4 - d^4)}{32} \tag{12.1}$$

which for a small thickness $t = (D - d)/2 \ll d$ becomes

$$J \approx \frac{\pi(D - t)^3 t}{4} \tag{12.2}$$

If we slit the hollow tube by a longitudinal cut, its torsional constant will drop to

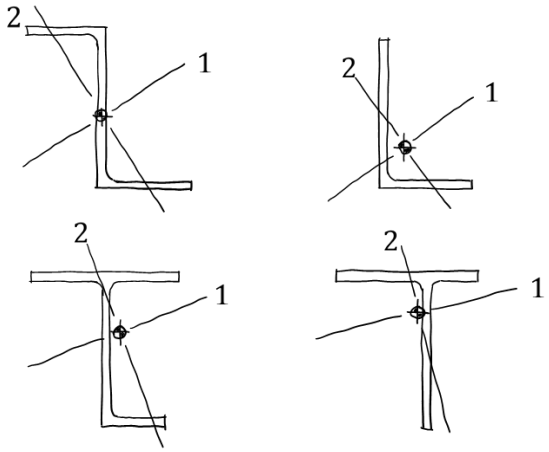


Fig. 12.4. Unsymmetric open-profile cross sections. Note in general these profiles will have inclined principal planes. The shear center is also rarely at the centroid of the section.

$$J = \frac{\pi D t^3}{3} \tag{12.3}$$

For $D = 0.1\text{m}$ and $t = 0.001\text{m}$, the intact tube has a $J = 7.62 \times 10^{-7}\text{m}^4$ and the slit tube has a $J = 1.05 \times 10^{-10}\text{m}^4$. In words, the slit tube is more than a thousand times more flexible than the intact tube.

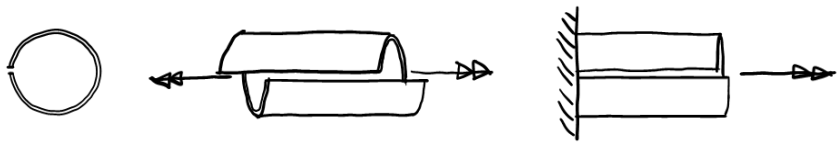


Fig. 12.5. Torsion of slit thin walled cylinder. Unconstrained torsion in the (sketch in the middle) illustrates that the cylinder is quite flexible. Constraining the torsion by clamping one side makes the cylinder stiffer in torsion (sketch on the right).

12.3.1 Unconstrained (Saint-Vénant) torsion

Saint-Vénant's torsion mechanism relies entirely on the activation of the shear stresses within the wall. Since those stresses are distributed linearly through the thickness, they are not very effective for thin walls. The amount of torque carried by this torsion mechanism is usually negligible for open profile thin-walled beams.

This mechanism of carrying twisting torques typically assumes uniform torsion of prismatic beams. In other words, the rate of twist ϕ' is constant along the beam. This is in sharp contrast to the mechanism described in the next section, where the warping of the cross section of the beam is prevented or reduced at some locations along the beam, and consequently the rate of twist changes along the beam.

12.3.2 Constrained torsion

This torsion mechanism is activated when the warping of the cross sections is hampered in some way: for instance, totally prevented by some physical device, such as attachment to a stiff structure.

Under such circumstances, the rate of twist is nonuniform, and axial stresses are activated to partially support the applied torque.

Figures 12.6 and 12.7 explain the salient differences. Figure 12.6 shows an I-beam structure subjected to a torsional moment. As shown schematically, the cross section shall warp. However, since the only stress activated is the shear follow round the thin-walled profile, this beam has low stiffness to activate as a response to an applied torque.

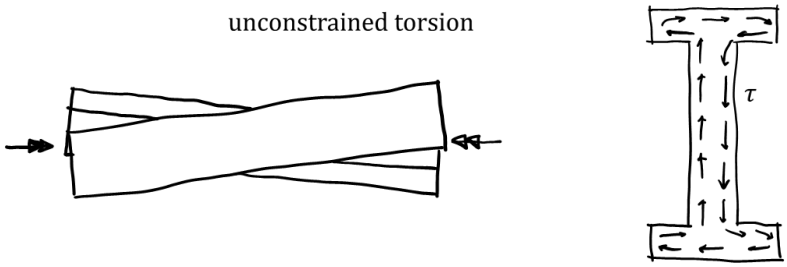


Fig. 12.6. Unconstrained torsion of a thin-walled I beam. The entire torque is carried by the shear stress flow circulating in the cross section. The cross section is not the scale.

On the other hand, as shown in Figure 12.7, when the beam is clamped to a stiff structure at one end, the flanges will have to bend in order to accommodate the torque. This generates axial stresses in the cross section, has shown schematically on the right, in addition to the shear stresses in the flanges. The warping constraint increases the stiffness of the structure significantly.

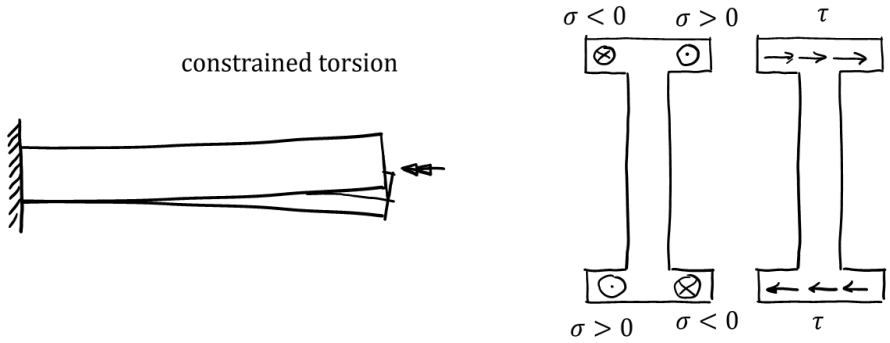


Fig. 12.7. Constrained torsion of a thin-walled I beam. Warping constraint at the left-hand side makes it possible to support a torque by the bending of the flanges, i.e. by generating axial stress. The schematic on the right shows the distribution of the axial stresses, which form a self-equilibrated system, and the distribution of the shear stresses in the flanges. The cross section is not the scale.

Always use specialized beam elements in Abaqus, B31OS, B32OS, to analyse open-profile thin-walled beams. Other beam element types do not consider warping stresses, in other words warping is considered unconstrained and axial stress due to warping is not taken into account; torsional behavior of thin-walled, open-profile beams will not be represented adequately.

Abaqus provides special elements for open-profile thin-walled beams, which capture the effects of constrained torsion by adding a seventh degree of freedom per node, the warping multiplier. The warping function of the profile (as a purely geometrical quantity) is multiplied with the warping multiplier to yield the total amount of axial displacement due to warping. The analyst must think

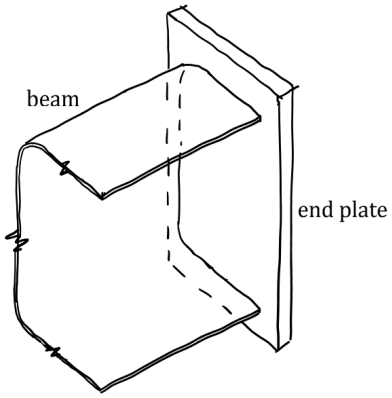


Fig. 12.8. Constrained torsion of a thin-walled I beam. Warping constraint at the left-hand side makes it possible to support a torque by the bending of the flanges, i.e. by generating axial stress. The schematic on the right shows the distribution of the axial stresses, which form a self-equilibrated system, and the distribution of the shear stresses in the flanges. The cross section is not the scale.

carefully about which impediments to warping there might be: reducing the warping is the main torsion-carrying mechanism. For instance, end plates attached to thin-walled beams may effectively stiffen the structure. Refer to Figure 12.8 which shows an end plate attached to a C channel beam by welding. Even though the plate is quite massive relative to the thickness of the beam wall, it is still difficult to imagine that warping will be totally eliminated. Engineering judgment is required.

Beware: It is possible to select the B31OS, B32OS finite elements in the CAE, but it is not possible to set any warping constraints. So to run a simulation from within the CAE means that all open-profile thin-walled only respond with unconstrained torsion, which is probably wrong. Torsion constraints can only be set by using keywords in the input file.

13

Modeling of layered structures

Summary

1. Anisotropic materials
2. Sandwich Structures

Fiber reinforced plastic (FRP) is now frequently used in a large variety of applications, including sports goods such as golf clubs and tennis rackets, load-bearing frames of sports cars, components in commercial and military aircraft, and in the space industry for launch vehicles, satellites, antennae systems, and so on. Structural components in the form of plates, shells, sandwich panels, sandwich shells, beams and struts, are manufactured from FRP materials to deliver a design with high strength-to-weight and stiffness-to-weight ratios. The design goal is to achieve this by using the minimum number of plies of unidirectional material or woven fabric oriented at carefully selected angles to provide a laminate that satisfies the stiffness, strength, thermal distortion, and functional requirements. The thickness, orientation and layup of the plies can produce laminates that exhibit coupling in membrane and bending behaviour, which typically has a significant effect on the response of the laminate to mechanical and thermal loads.

Because of the practical importance of fiber reinforced composites (laminates), we shall focus first on this type of structure. Later we will consider other layered structures, such as sandwich shells and plates, were the individual layers can be, but do not need to be, laminates.

13.1 Lamina materials

Our initial focus is on one lamina (alternatively called a ply). We shall consider material of unidirectional fibers reinforcing a matrix. Perhaps carbon fibers reinforcing an epoxy matrix. We shall consider the lamina initially by itself, but subsequently we will need to consider it in the context of a composite layup. Figure 13.1 shows that we will describe the lamina properties in its own coordinate system, x'_1 and x'_2. In Abaqus graphical displays these two directions are tangent to the surface of the plate or shell and are given the labels "1" and "2".

Most fiber reinforced lamina are described by a version of the orthotopic elasticity, which for the fully three dimensional material is described by the compliance matrix [PK1]

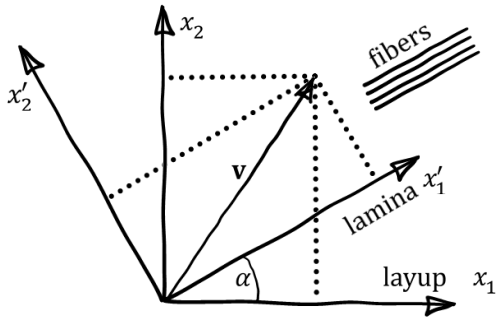


Fig. 13.1. Lamina coordinate system (primed) in relation to the layup (laminate) coordinate system (unprimed). The lamina first direction (x_1') is given by the angle α by which the fibers of the ply are rotated from the layup reference direction (axis x_1) counterclockwise about the normal to the layup.

$$\mathbf{Q'}^{-1} = \begin{bmatrix} {E_1}^{-1}, & -\frac{\nu_{12}}{E_1}, & -\frac{\nu_{13}}{E_1}, & 0, & 0, & 0 \\ -\frac{\nu_{12}}{E_1}, & {E_2}^{-1}, & -\frac{\nu_{23}}{E_2}, & 0, & 0, & 0 \\ -\frac{\nu_{13}}{E_1}, & -\frac{\nu_{23}}{E_2}, & {E_3}^{-1}, & 0, & 0, & 0 \\ 0, & 0, & 0, & {G_{12}}^{-1}, & 0, & 0 \\ 0, & 0, & 0, & 0, & {G_{13}}^{-1}, & 0 \\ 0, & 0, & 0, & 0, & 0, & {G_{23}}^{-1} \end{bmatrix}.$$

All nine coefficients are independent. However, in most cases in practice, the lamina material can be described with a simplification appropriate for the plane stress + transverse shear circumstances of typical lamina: The in-plane strains and stresses are related through this compliance expression

$$\begin{bmatrix} \epsilon_1' \\ \epsilon_2' \\ \gamma_{12}' \end{bmatrix} = \begin{bmatrix} 1/E_1, & -\nu_{12}/E_1, & 0 \\ -\nu_{12}/E_1, & 1/E_2, & 0 \\ 0, & 0, & 1/G_{12} \end{bmatrix} \begin{bmatrix} \sigma_1' \\ \sigma_2' \\ \tau_{12}' \end{bmatrix} \tag{13.1}$$

and the transverse shear strains and transverse shear stresses are related through

$$\begin{bmatrix} \gamma_{13}' \\ \gamma_{23}' \end{bmatrix} = \begin{bmatrix} 1/G_{13}, & 0 \\ 0, & 1/G_{23} \end{bmatrix} \begin{bmatrix} \tau_{13}' \\ \tau_{23}' \end{bmatrix} \tag{13.2}$$

Note that the transverse strains and stresses are sometimes also referred to as intralaminar, because shearing across the fibres is by symmetry of the shears equivalent to shearing along the fibres (13 is equivalent to 31, etc.): refer to Figure 13.2.

Equation (13.1) may be expressed using the plane-stress stiffness matrix as

$$\begin{bmatrix} \epsilon_1' \\ \epsilon_2' \\ \gamma_{12}' \end{bmatrix} = \mathbf{Q'}^{-1} \begin{bmatrix} \sigma_1' \\ \sigma_2' \\ \tau_{12}' \end{bmatrix} \tag{13.3}$$

Here $\mathbf{Q'}$ is the lamina membrane stiffness matrix on the ply coordinate system, and its inverse is the ply compliance matrix.

Analogously, we define the transverse shear stiffness by inverting the compliance relation

$$\begin{bmatrix} \gamma_{13}' \\ \gamma_{23}' \end{bmatrix} = \mathbf{H'}^{-1} \begin{bmatrix} \tau_{13}' \\ \tau_{23}' \end{bmatrix} \tag{13.4}$$

Here $\mathbf{H'}$ is the lamina transverse shear stiffness matrix on the ply coordinate system.

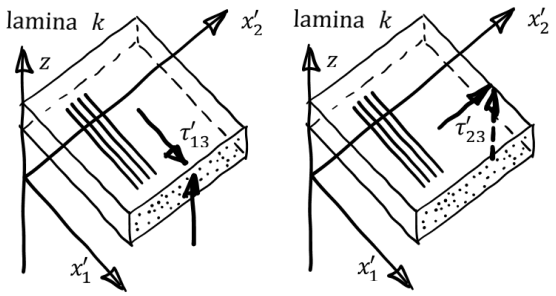


Fig. 13.2. Illustration of intralaminar shears. One of the shear planes is parallel to the midplane of the lamina.

13.2 Laminate

A laminate is a collection of lamina (plies), each of which is described by the equations derived in the previous section. At this point we must develop rules for transitioning from the ply description to the layup description.

The vector in Figure 13.1 can be expressed with components in either coordinate system. From this we get the transformation [PK1]

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \cos(2\alpha) \, , & -\sin(2\alpha) \\ \sin(2\alpha) \, , & \cos(2\alpha) \end{bmatrix} \begin{bmatrix} v_1' \\ v_2' \end{bmatrix} \tag{13.5}$$

The matrix

$$\mathbf{R}_m = \begin{bmatrix} \cos(2\alpha) \, , & -\sin(2\alpha) \\ \sin(2\alpha) \, , & \cos(2\alpha) \end{bmatrix} \tag{13.6}$$

holds as its columns the basis vectors of the primed coordinate system expressed in components of the unprimed coordinate system.

Now we consider the relationship between the traction vector, the stress at a point, expressed with a matrix (tensor), and the normal to a surface through the point, all through the components on the layup coordinate system

$$\begin{bmatrix} t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \sigma_1 & \tau_{12} \\ \tau_{12} & \sigma_2 \end{bmatrix} \begin{bmatrix} n_1 \\ n_2 \end{bmatrix} \, , \tag{13.7}$$

and we contemplate what happens when both vectors are expressed using components in the ply coordinate system. We apply (13.6) as

$$\begin{bmatrix} t_1 \\ t_2 \end{bmatrix} = \mathbf{R}_m \begin{bmatrix} t_1' \\ t_2' \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} n_1 \\ n_2 \end{bmatrix} = \mathbf{R}_m \begin{bmatrix} n_1' \\ n_2' \end{bmatrix} \tag{13.8}$$

so that we can write

$$\begin{bmatrix} t_1 \\ t_2 \end{bmatrix} = \mathbf{R}_m \begin{bmatrix} t_1' \\ t_2' \end{bmatrix} = \begin{bmatrix} \sigma_1 & \tau_{12} \\ \tau_{12} & \sigma_2 \end{bmatrix} \begin{bmatrix} n_1 \\ n_2 \end{bmatrix} = \begin{bmatrix} \sigma_1 & \tau_{12} \\ \tau_{12} & \sigma_2 \end{bmatrix} \mathbf{R}_m \begin{bmatrix} n_1' \\ n_2' \end{bmatrix} \, , \tag{13.9}$$

or

$$\mathbf{R}_m \begin{bmatrix} t_1' \\ t_2' \end{bmatrix} = \begin{bmatrix} \sigma_1 & \tau_{12} \\ \tau_{12} & \sigma_2 \end{bmatrix} \mathbf{R}_m \begin{bmatrix} n_1' \\ n_2' \end{bmatrix} \, , \tag{13.10}$$

and finally we can multiply both sides on the left with $\mathbf{R}_m^{-1} = \mathbf{R}_m^T$ to obtain

$$\begin{bmatrix} t_1' \\ t_2' \end{bmatrix} = \mathbf{R}_m^T \begin{bmatrix} \sigma_1 & \tau_{12} \\ \tau_{12} & \sigma_2 \end{bmatrix} \mathbf{R}_m \begin{bmatrix} n_1' \\ n_2' \end{bmatrix} \, , \tag{13.11}$$

Since one also has

$$\begin{bmatrix} t_1' \\ t_2' \end{bmatrix} = \begin{bmatrix} \sigma_1' & \tau_{12}' \\ \tau_{12}' & \sigma_2' \end{bmatrix} \begin{bmatrix} n_1' \\ n_2' \end{bmatrix} \quad , \tag{13.12}$$

we conclude that the ***stress tensor components transform*** when switching between coordinate systems as

$$\begin{bmatrix} \sigma_1' & \tau_{12}' \\ \tau_{12}' & \sigma_2' \end{bmatrix} = \mathbf{R}_m^T \begin{bmatrix} \sigma_1 & \tau_{12} \\ \tau_{12} & \sigma_2 \end{bmatrix} \mathbf{R}_m \tag{13.13}$$

Of course, this transformation works back and forth so that

$$\begin{bmatrix} \sigma_1 & \tau_{12} \\ \tau_{12} & \sigma_2 \end{bmatrix} = \mathbf{R}_m \begin{bmatrix} \sigma_1' & \tau_{12}' \\ \tau_{12}' & \sigma_2' \end{bmatrix} \mathbf{R}_m^T \, . \tag{13.14}$$

If we multiply (13.14) through, we get this matrix

$$\begin{bmatrix} \sigma_1 & \tau_{12} \\ \tau_{12} & \sigma_2 \end{bmatrix} = \begin{bmatrix} -\tau_{12}' \sin(2\alpha) + \sigma_1' \cos^2(\alpha) + \sigma_2' \sin^2(\alpha) \, , \, \tau_{12}' \cos(2\alpha) + \dfrac{\sigma_1' \sin(2\alpha)}{2} - \dfrac{\sigma_2' \sin(2\alpha)}{2} \\ \tau_{12}' \cos(2\alpha) + \dfrac{\sigma_1' \sin(2\alpha)}{2} - \dfrac{\sigma_2' \sin(2\alpha)}{2} \, , \quad \tau_{12}' \sin(2\alpha) + \sigma_1' \sin^2(\alpha) + \sigma_2' \cos^2(\alpha) \end{bmatrix} \tag{13.15}$$

This may be rewritten as

$$\begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \tau_{12} \end{bmatrix} = \begin{bmatrix} -\tau_{12}' \sin(2\alpha) + \sigma_1' \cos^2(\alpha) + \sigma_2' \sin^2(\alpha) \\ \tau_{12}' \sin(2\alpha) + \sigma_1' \sin^2(\alpha) + \sigma_2' \cos^2(\alpha) \\ \tau_{12}' \cos(2\alpha) + \dfrac{\sigma_1' \sin(2\alpha)}{2} - \dfrac{\sigma_2' \sin(2\alpha)}{2} \end{bmatrix} \tag{13.16}$$

$$= \begin{bmatrix} \cos^2(\alpha) \, , & \sin^2(\alpha) \, , & -\sin(2\alpha) \\ \sin^2(\alpha) \, , & \cos^2(\alpha) \, , & \sin(2\alpha) \\ \dfrac{\sin(2\alpha)}{2} \, , & -\dfrac{\sin(2\alpha)}{2} \, , & \cos(2\alpha) \end{bmatrix} \begin{bmatrix} \sigma_1' \\ \sigma_2' \\ \tau_{12}' \end{bmatrix} \tag{13.17}$$

$$= \mathbf{T}_m^{-1} \begin{bmatrix} \sigma_1' \\ \sigma_2' \\ \tau_{12}' \end{bmatrix} \tag{13.18}$$

Therefore, we may express

$$\begin{bmatrix} \sigma_1' \\ \sigma_2' \\ \tau_{12}' \end{bmatrix} = \mathbf{T}_m \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \tau_{12} \end{bmatrix} \tag{13.19}$$

To transform strains, we take advantage of the equality of the work of stress on the strains:

$$\begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \tau_{12} \end{bmatrix}^T \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \gamma_{12} \end{bmatrix} = \begin{bmatrix} \sigma_1' \\ \sigma_2' \\ \tau_{12}' \end{bmatrix}^T \begin{bmatrix} \epsilon_1' \\ \epsilon_2' \\ \gamma_{12}' \end{bmatrix} \tag{13.20}$$

We substitute from (13.19), and we assume the transformation of the strains as

$$\begin{bmatrix} \epsilon_1' \\ \epsilon_2' \\ \gamma_{12}' \end{bmatrix} = \overline{\mathbf{T}}_m \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \gamma_{12} \end{bmatrix} \tag{13.21}$$

$$\tag{13.22}$$

where the matrix $\overline{\mathbf{T}}_m$ is to be determined. Hence, we write

$$\begin{bmatrix}\sigma_1\\ \sigma_2\\ \tau_{12}\end{bmatrix}^T \begin{bmatrix}\epsilon_1\\ \epsilon_2\\ \gamma_{12}\end{bmatrix} = \left(\mathbf{T}_m \begin{bmatrix}\sigma_1\\ \sigma_2\\ \tau_{12}\end{bmatrix}\right)^T \overline{\mathbf{T}}_m \begin{bmatrix}\epsilon_1\\ \epsilon_2\\ \gamma_{12}\end{bmatrix} = \begin{bmatrix}\sigma_1\\ \sigma_2\\ \tau_{12}\end{bmatrix}^T \mathbf{T}_m^T \overline{\mathbf{T}}_m \begin{bmatrix}\epsilon_1\\ \epsilon_2\\ \gamma_{12}\end{bmatrix} \tag{13.23}$$

And therefore we must conclude

$$\mathbf{T}_m^T \overline{\mathbf{T}}_m = \mathbf{1} \tag{13.24}$$

The definition of the transformation matrix for the strains $\overline{\mathbf{T}}_m$ is therefore

$$\overline{\mathbf{T}}_m = \mathbf{T}_m^{-T} \,, \quad \text{or, alternatively,} \quad \overline{\mathbf{T}}_m^T = \mathbf{T}_m^{-1} \,. \tag{13.25}$$

(We already know what $\mathbf{T}_m^{-1}$ is from (13.16).)

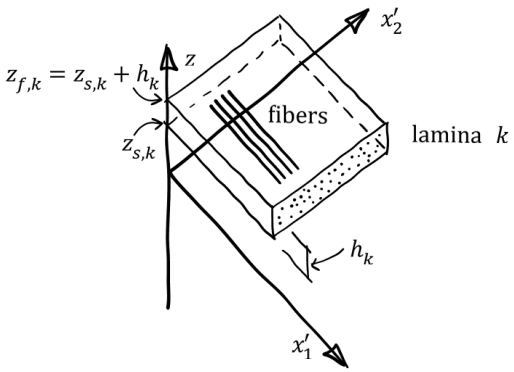


Fig. 13.3. Lamina material layer shown in the lamina coordinate system. The lamina (ply) k starts at $z_{s,k}$, and has a thickness of h_k. The lamina material extends from $z_{s,k}$ to $z_{f,k} = z_{s,k} + h_k$ in the z direction.

Stiffness.

Figure 13.3 shows one particular lamina, number k, offset from the reference plane by $z_{s,k}$, of thickness h_k. We can express the strains experienced by the material within this lamina $\epsilon_1', \epsilon_2', \gamma_{12}'$, using the basic assumption of the shell model, expressed in Equation (3.1), written down so

$$\begin{bmatrix}\epsilon_1'\\ \epsilon_2'\\ \gamma_{12}'\end{bmatrix} = \begin{bmatrix}\epsilon_{1,0}'\\ \epsilon_{2,0}'\\ \gamma_{12,0}'\end{bmatrix} + z \begin{bmatrix}\kappa_1'\\ \kappa_2'\\ \kappa_{12}'\end{bmatrix} \tag{13.26}$$

in terms of the $\epsilon_{1,0}'$, $\epsilon_{2,0}'$, $\gamma_{12,0}'$ which are the membrane strains within the reference plane, and of the curvatures defined as $\kappa_1' = \partial\phi_y'/\partial x_1'$, $\kappa_2' = -\partial\phi_x'/\partial x_2'$, and $\kappa_{12}' = \partial\phi_y'/\partial x_2' - \partial\phi_x'/\partial x_1'$. All components here are expressed on the ply coordinate system.

Now we use the constitutive equation (13.3) and integrate the resulting stress through the thickness of the lamina to obtain the membrane forces in the k-th lamina

$$\begin{bmatrix}N_{1,k}'\\ N_{2,k}'\\ N_{12,k}'\end{bmatrix} = \mathbf{A}_k' \begin{bmatrix}\epsilon_{1,0}'\\ \epsilon_{2,0}'\\ \gamma_{12,0}'\end{bmatrix} + \mathbf{B}_k' \begin{bmatrix}\kappa_1'\\ \kappa_2'\\ \kappa_{12}'\end{bmatrix} \tag{13.27}$$

and we integrate the resulting stress multiplied by z to obtain the bending and twisting moments

$$\begin{bmatrix}M_{1,k}'\\ M_{2,k}'\\ M_{12,k}'\end{bmatrix} = \mathbf{B}_k' \begin{bmatrix}\epsilon_{1,0}'\\ \epsilon_{2,0}'\\ \gamma_{12,0}'\end{bmatrix} + \mathbf{D}_k' \begin{bmatrix}\kappa_1'\\ \kappa_2'\\ \kappa_{12}'\end{bmatrix} \tag{13.28}$$

The matrices $\mathbf{A}'_k$, $\mathbf{B}'_k$, $\mathbf{D}'_k$ relate the (generalized) strains to the stress resultants, all on the ply coordinate system. The matrix $\mathbf{A}'_k$ relates the stretches and shears of the reference surface to the membrane forces:

$$\mathbf{A}_k = \mathbf{Q}'_k \left(z_{f,k} - z_{s,k}\right) \tag{13.29}$$

The matrix $\mathbf{D}'_k$ relates the curvatures of the reference surface to the bending and twisting moments:

$$\mathbf{D}'_k = \mathbf{Q}'_k \left(z^3_{f,k} - z^3_{s,k}\right)/3 \tag{13.30}$$

The so-called coupling matrix $\mathbf{B}'_k$ relates the stretches and shears of the reference surface to the bending and twisting moments, and also conversely the curvatures of the reference surface to the membrane forces:

$$\mathbf{B}'_k = \mathbf{Q}'_k \left(z^2_{f,k} - z^2_{s,k}\right)/2 \tag{13.31}$$

Note that the effect of the coupling matrix $\mathbf{B}'_k$ vanishes only if the lamina (ply) is positioned symmetrically with respect to $z = 0$; otherwise it is always present.

Stiffness matrix of the layup.

For each lamina we have the constitutive equations (13.27) and (13.28). For an entire stack of lamina (i.e. a laminate), we need to add the matrices of the individual lamina together. But first we need to transform the lamina matrices into the layup coordinate system.

Because the stresses are transform according to (13.19), and the strains according to (13.21), the force and moment resultants and the generalized strains transform in the same way. therefore we can write

$$\begin{bmatrix} N'_{1,k} \\ N'_{2,k} \\ N'_{12,k} \end{bmatrix} = \mathbf{A}'_k \begin{bmatrix} \epsilon'_{1,0} \\ \epsilon'_{2,0} \\ \gamma'_{12,0} \end{bmatrix} + \mathbf{B}'_k \begin{bmatrix} \kappa'_1 \\ \kappa'_2 \\ \kappa'_{12} \end{bmatrix} \tag{13.32}$$

$$= \mathbf{T}_{m,k} \begin{bmatrix} N_{1,k} \\ N_{2,k} \\ N_{12,k} \end{bmatrix} = \mathbf{A}'_k \overline{\mathbf{T}}_{m,k} \begin{bmatrix} \epsilon_{1,0} \\ \epsilon_{2,0} \\ \gamma_{12,0} \end{bmatrix} + \mathbf{B}'_k \overline{\mathbf{T}}_{m,k} \begin{bmatrix} \kappa_1 \\ \kappa_2 \\ \kappa_{12} \end{bmatrix} \tag{13.33}$$

In the above, the generalized strains and the membrane forces are in components on the layup coordinate system. This may be rewritten into the form

$$\begin{bmatrix} N_{1,k} \\ N_{2,k} \\ N_{12,k} \end{bmatrix} = \mathbf{T}^{-1}_{m,k} \mathbf{A}'_k \overline{\mathbf{T}}_{m,k} \begin{bmatrix} \epsilon_{1,0} \\ \epsilon_{2,0} \\ \gamma_{12,0} \end{bmatrix} + \mathbf{T}^{-1}_{m,k} \mathbf{B}'_k \overline{\mathbf{T}}_{m,k} \begin{bmatrix} \kappa_1 \\ \kappa_2 \\ \kappa_{12} \end{bmatrix} \tag{13.34}$$

Thus on the layup coordinate system, we have the matrices

$$\mathbf{A}_k = \mathbf{T}^{-1}_{m,k} \mathbf{A}'_k \overline{\mathbf{T}}_{m,k} = \overline{\mathbf{T}}^T_{m,k} \mathbf{A}'_k \overline{\mathbf{T}}_{m,k} \tag{13.35}$$

$$\mathbf{B}_k = \mathbf{T}^{-1}_{m,k} \mathbf{B}'_k \overline{\mathbf{T}}_{m,k} = \overline{\mathbf{T}}^T_{m,k} \mathbf{B}'_k \overline{\mathbf{T}}_{m,k} \tag{13.36}$$

where we used (13.25). Similarly we obtain an analogous expression for $\mathbf{D}_k$. The overall layup matrices are simply obtained by summing over all the plies (lamina)

$$\mathbf{A} = \sum_k \mathbf{A}_k \tag{13.37}$$

etc. for the other two matrices.

Note that the coupling matrix will be zero for a symmetric layup, as long as the offset between the reference plane and the midplane is zero. Not all coupling is necessarily due to the $\mathbf{B}$ matrix. Figure 13.4 shows various kinds of coupling: bending-stretching in a [0/90] laminate, thermal expansion-bending giving a saddle shape for a [0/90] laminate, torsion-extension coupling in a [θ, $-\theta$] laminate, and shear-extension coupling in a [$(\theta)_2$] laminate.

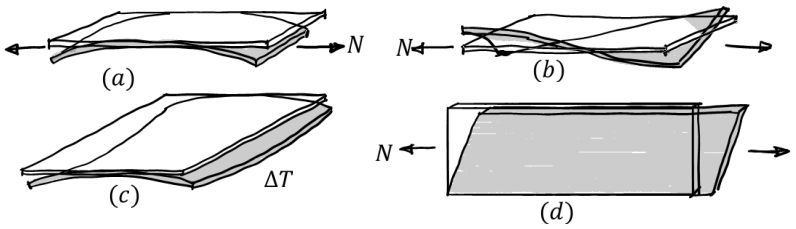


Fig. 13.4. Coupling of various modes of deformation. (a) [90/0] laminate: coupling of bending to stretching, (b) torsion and extension are coupled in a $[\theta, -\theta]$ laminate, (c) thermal expansion of a [90/0] laminate induces bending,, and (d) shear and extension are coupled in a $[(\theta)_2]$ laminate.

The transverse shear stiffness matrix of the composite layup is built up of the matrices of the plies, but a correction needs to be introduced to account for the parabolic distribution of the shear stress [ViSi]

$$\mathbf{H} = \sum_k \mathbf{H}_k \tag{13.38}$$

where the $\mathbf{H}_k$ are expressed as

$$\mathbf{H}_k = \frac{5}{4}\mathbf{R}^T_{m,k}\mathbf{H}'_k\mathbf{R}_{m,k}\left[h_k - \frac{4}{\sum_j h_j}\left(h_k(z_{f,k}+z_{s,k})^2 + \frac{h_k^3}{12}\right)\right] . \tag{13.39}$$

The shear-flexible plate and shell theory requires shear correction factors, for it is known that in composite laminated beams and plates the transverse shear stresses vary at least quadratically within each layer. (Recall Section 4.4 on the Timoshenko beam.) The shear correction factors are difficult to determine for arbitrarily laminated composite plate structures, and Abaqus provides a complicated heuristic for this purpose.

[discuss the hygrothermal behavior of laminates]

General shell section.

In Abaqus it is possible to input the stiffness matrices directly. The General Stiffness input encompasses the upper triangle of the **A** and **D** and the entire matrix **B** of the layup, in the layup coordinate system. Further, in the Advanced tab, the upper triangle of the **H** maybe input (the coefficients are called K11, ...).

13.3 Composite layup

Composite *layups* in Abaqus are able to manage a large number of plies in a typical composite model. This mechanism has a number of advantages over using composite *sections* for the same purpose.

The procedure for creating a composite layup mimics creating a real composite part: start with a basic shape, and then add plies of different materials and thicknesses to selected regions, and orient the plies to provide the greatest strength in a particular direction. Plies may be arbitrarily added or deleted, the regions to which they are applied can be interactively selected. The plies may be made up of any number of materials. It is crucial to specify the correct orientation of the fibers in a ply. Reference orientations may be defined for the layup or for the individual plies.

The first ply in the ply table represents the bottom ply in the layup.

One of the most useful ways of orienting plies is the so-called "discrete orientation". Figure 13.5 demonstrates the setup of a ply orientation: the surface itself defines the normal direction ("n"), and then one picks the direction of an edge which defines the so-called primary direction. The local coordinate system direction "1" is parallel to this direction, but, if in addition the ply is at an angle, the local coordinate system is rotated about the normal (in this case by 30°) from the primary direction.

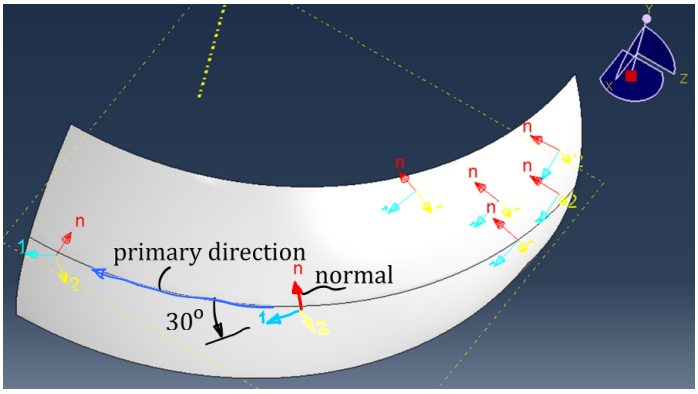


Fig. 13.5. Curved surface with ply orientation.

Continuum shell and solid composite layups are expected to have a single element through the entire thickness. Each single element through the thickness contains the plies defined in the ply table. If the region contains multiple elements through the thickness, each element will contain all of the plies in the ply table!

13.4 Composites and finite elements

Metals are relatively stiff in shear, as $G = E/2/(1+\nu) \approx E/3$, and thus the intralaminar strains are generally negligible. In addition, the shear strength of metals is comparable to their tensile strength, and thus the interlaminar shear stress does not need to be checked make for metallic plates or shells.

The circumstances are quite different in laminated composite materials. The intralaminar shear moduli G_{23} and G_{13} are relatively small when compared with the in-plane modulus E_1. Therefore, the intralaminar strains (i.e. the transverse shear strains) are generally small, but for laminated composites they are likely not negligible. Even more importantly, the intralaminar shear stresses need to be supported by a matrix that has limited shear strength values. This makes it is necessary to evaluate the intralaminar shear stress: Figure 13.6 illustrates the reciprocity of shearing stresses: intralaminar shearing stresses are present, as well as the stresses acting upon the cross section orthogonally to the midsurface (Figure 13.6(a)). The intralaminar stresses may be inferred by considering the two layers resting one on top of the other without friction (Figure 13.6(b)). If the lower layer is supported at its ends and the upper layer is loaded with a distributed force q, then each layer will bend, freely sliding past each other, but otherwise being in contact. At the interface between the two layers, we detect dilation in the upper layer and contraction in the lower one. Now we bond the two layers with adhesive, and the bonded interface will hamper the dilation in the upper layer and the contraction in the lower layer by interlaminar shear stresses (Figure 13.6(c)). In laminate composite materials, the failure of delamination is caused by the shear stresses exceeding the bonding resistance.

Modeling of laminated composites differs from modeling homogenous isotropic structures in at least three aspects.

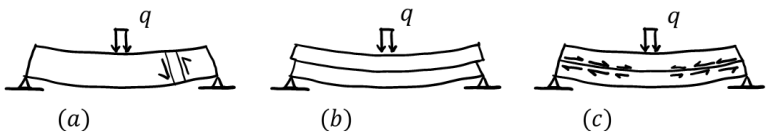


Fig. 13.6. Intralaminar shear in a two-layer composite plate.

1. The constitutive equations of the laminar material are typically not isotropic.
2. The model should enable the evaluation of intralaminar shear stresses.
3. Material symmetry needs to be considered in conjunction with the geometry and the boundary conditions. Only lamina with planes of symmetry at 0^o and 90^o relative to the purported symmetry plane will allow the overall structure to claim symmetry with respect to that plane.

Intralaminar shear.

Abaqus uses a fairly complicated formula to determine the intralaminar shear stresses so that they are consistent with (i) zero shear tractions on the bottom and top face, (ii) continuous shear stresses at the interfaces between the lamina. Abaqus computes the transverse shear stiffness by matching the shear response of the shell bending about one axis, using a parabolic variation of transverse shear stress in each layer. The approach generally provides a reasonable estimate of the shear flexibility of the shell. It also enables estimates of interlaminar shear stresses in composite shells. However, it is still an approximation, which will not be great for unsymmetric layups.

In order to visualize these shear stresses, the variables $TSHR13$ and $TSHR23$ (for conventional shells, S3, S4, S8R) or the variables $CTSHR13$ and $CTSHR23$ (for continuum shells, SC6R and SC8R) need to be added to a field output request.

Abaqus by default outputs the intralaminar shears at the surfaces of the shell, but there they are zero by definition. Select the section points in the middle of the lamina instead. okays

Section points in plies.

The numerical integration rule used for layered structures is typically a 3-point Simpson through the thickness. Which means that at any integration point within the surface of the element, there is a stack of section (integration) points, $3 \times m$ in total, where m is the number of plies. The section points are numbered from the bottom, 1, 2, 3 for ply 1, followed by 4, 5, 6 for ply 2, and se on.

13.5 Sandwich structures

Layered shells with very compliant interior layers (so-called "sandwich" composites – Figure 13.7) have very low transverse shear stiffness and should almost always be modeled with "thick"-shell specialist finite elements (S4, S4R, S3/S3R, S8R).

Figure 13.8 illustrates that the assumption that the section of a sandwich shell will remain planar after deformation cannot be quite right. The interlaminar shear between the layers must be continuous across the interface, but since the core has a very low shear modulus, it must activate the corresponding value of the shear stress by experiencing a much larger shear strain than the skins.

Thickness modulus.

A note on the through-the-thickness modulus: The default modulus of elasticity in the direction along the normal (referred to as the "thickness modulus" in the Abaqus documentation) is deduced from the shear modulus, and it will be typically way too low. Most core materials of the honeycomb

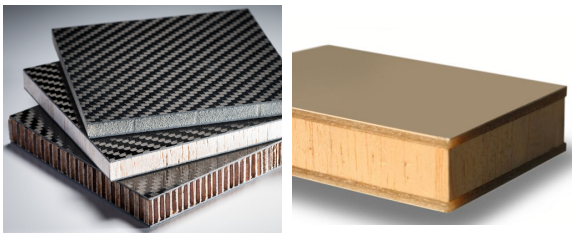


Fig. 13.7. Illustration of sandwich panel materials. Carbon fiber laminate skins combined with foams or honeycomb cores on the left. Laminate skins combined with balsa wood core on the right.

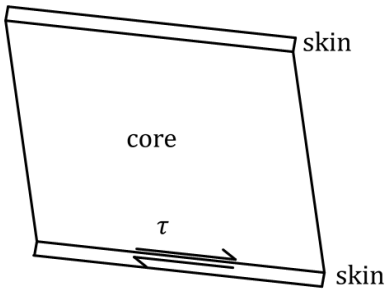


Fig. 13.8. Interlaminar shear in a sandwich panel. This shear needs to be continuous between the layers.

type have a low shear modulus, but quite high modulus in the thickness direction. And that is good, otherwise we would see the so-called pinching effect: lateral loads would press the skin into the core, substantially changing the thickness of the sandwich. The definition of the core material should be supplemented by a sufficiently realistic thickness modulus.

Discretization options.

There are several possible options for modeling sandwich structures. Figure 13.9(a) shows a typical configuration of a sandwich plate with a thick core and thin faces. Figure 13.9(b) documents division of the domain into a single layer of continuum-shell elements, which can be assigned a composite layup consisting of three plies. Figure 13.9(c) shows an alternative which meshes the partitioned domain into three layers (face, core, and another face). Each layer is then given appropriate material properties. Note that the same composite layup used in (b) cannot be assigned to all three layers of the elements in (c): each layer of elements would repeat the same laminate stacking sequence, which is not what we want. Figure 13.9(d) shows another possibility: the core is modelled with a 3D mesh of continuum shell elements, with the faces modeled with reinforcing skins. The skins are conventional shells, and they are shown by their midsurface. Their reference surfaces must coincide with the outer boundary of the core, otherwise the material of the core and the faces would overlap. The possibility that is not shown in Figure 13.9 is to use conventional shell models for the sandwich plate.

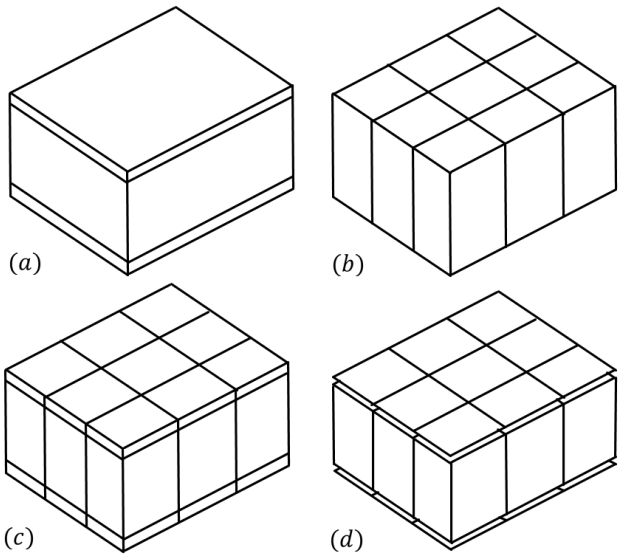


Fig. 13.9. Sandwich panel and its possible discretizations into finite elements. (a) Shows the division of the geometry into two faces and the core. (b) Shows the mesh composed of a single layer of elements. (c) Shows the mesh composed of a three layers of elements. (d) Shows the mesh of the core composed of a single layer of elements and the surface meshes of the skins.

14

Buckling

Summary

1. The basic notions of buckling analysis are developed on the model of a truss structure.
2. The general equation for buckling analysis (eigenvalue problem) is developed.
3. Local versus global buckling is discussed.

Analysis of the stability of structures is part of a much larger topic: failure of structures. Failures of many engineering structures fall into one of two general categories: (i) material failure, and (ii) structural instability.

The first type of failure, often amenable to treatment in introductory courses on the strength of materials, can be usually adequately analyzed on the basis of equilibrium conditions written for the initial, undeformed, configuration of the structure.

On the other hand, the prediction of structural instability necessitates the knowledge of the equations of equilibrium in the deformed configuration of the structure. Since the deformed configuration depends on the initially unknown deflections, the problem is in principle nonlinear.

Structural failures caused by the material inadequacy can be studied in the simplest cases from the viewpoint of the value of the material strength or yield limit: These quantities can be considered independent of structural geometry and size. This may be contrasted with instability: the load at which a structure becomes unstable can be, at least in the first approximation, regarded as independent of the material strength or yield limit; the load is found to depend on structural geometry and size (e.g. slenderness), and is governed primarily by the stiffness of the structure. The primary cause of the failure of an elastic structure due to structural instability can be traced to geometric effects: the deformation introduces geometric nonlinearities that amplify the stresses calculated in the undeformed configuration of the structure.

14.1 Introduction to buckling: truss structures

In order to introduce the subject we will consider the simple case of two dimensional truss structures. Consider two structures, or perhaps rather two states of one and the same structure (Figure 14.1): The structure (a) does not experience any internal forces because it is not loaded by external applied forces at all; the structure (b) is loaded by some externally applied forces F_j and therefore the truss members must carry internal forces $N_j \neq 0$ to support the loading. We will introduce buckling by considering these two sets of circumstances. We will show that by taking into account the effect of the internal forces we can construct two kinds of stiffness matrices: the first one is the usual elastic stiffness, and it is the same for both structures. The second stiffness depends on the internal forces, and is therefore usually labeled the "initial-stress" stiffness, and it vanishes for the structure (a) (because all the internal forces are zero), but it is non zero for structure (b).

These two kinds of stiffness matrix can be separated, and by introducing a scaling factor of the loading (usually called the loading factor or load multiplier) we can formulate the so-called

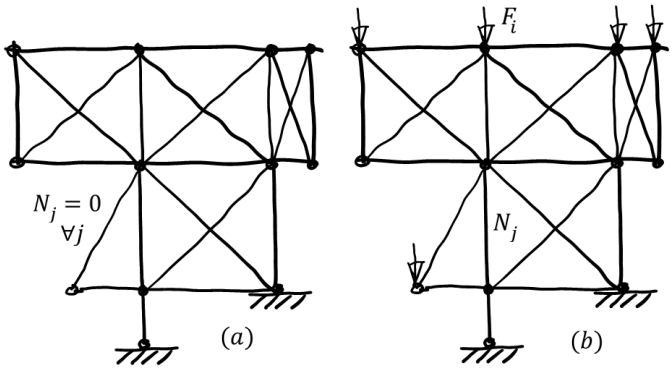


Fig. 14.1. Two configurations of the same structure: (a) without any applied loading, and hence without any internal forces ($N_j = 0$), and (b) with applied loading F_j, which is resisted by non-zero internal forces N_j.

buckling problem. The solution of the buckling problem yields the critical load factor and the buckling shape. These two quantities are representative of the onset of instability (alternatively, of the loss of stability) of the structure.

14.1.1 Stiffness matrix of structure without initial stress

Here we deal with structure (a) from Figure 14.1. Figure 14.2 illustrates the classical way of deriving the elastic stiffness matrix of a two node truss member: The truss is considered in the special orientation (horizontal), but its joints can move both horizontally by u_j, and vertically by v_j. We consider the end forces that ensure static equilibrium of the member, horizontal force H_j, and vertical force V_j. Figure 14.2(a) shows the bar without any deformation, its length being h, and the axial force in the bar being zero. Therefore also the forces at the joints required to maintain the truss in this state are all zero.

In Figure 14.2(b) the left hand side joint is moved to the right by $u_1 = 1$.

A quick note: Even though we enforce a unit displacement, the tacit assumption is that such displacements are so small that the changes to the geometry do not have to be considered! In particular, the displacement does not change the reference length of the truss member, which is still considered as h.

Then the engineering strain in the truss results as $-u_1/h$, and the motion therefore generates the force $N = ES \times (-u_1/h)$, or, $N = k \times (-u_1)$, where we employ the stiffness of the truss member as if it was a spring with the spring constant $k = ES/h$. The joint forces required to maintain this shortening of the truss change from zero to $H_1 = ku_1$ and $H_2 = -ku_1$. This change can be recorded in the stiffness matrix. The displacement u_1 multiplies the first column, and since the vertical forces are unaffected, we get

$$\mathbf{K} = \begin{bmatrix} k\,, & ?\,, ?\,, ? \\ 0\,, & ?\,, ?\,, ? \\ -k\,, & ?\,, ?\,, ? \\ 0\,, & ?\,, ?\,, ? \end{bmatrix} \tag{14.1}$$

where ? indicate quantities which have not been determined yet. Note that the joint deflections are ordered as u_1, v_1, u_2, v_2, and the multiplication of the stiffness matrix with the vector of the joint displacements gives the forces H_1, V_1, H_2, V_2 as the entries of the force vector.

In Figure 14.2(c) the right hand side joint is moved to the right by $u_2 = 1$. This again does not change the vertical forces necessary for equilibrium (they are still all zero), but the horizontal

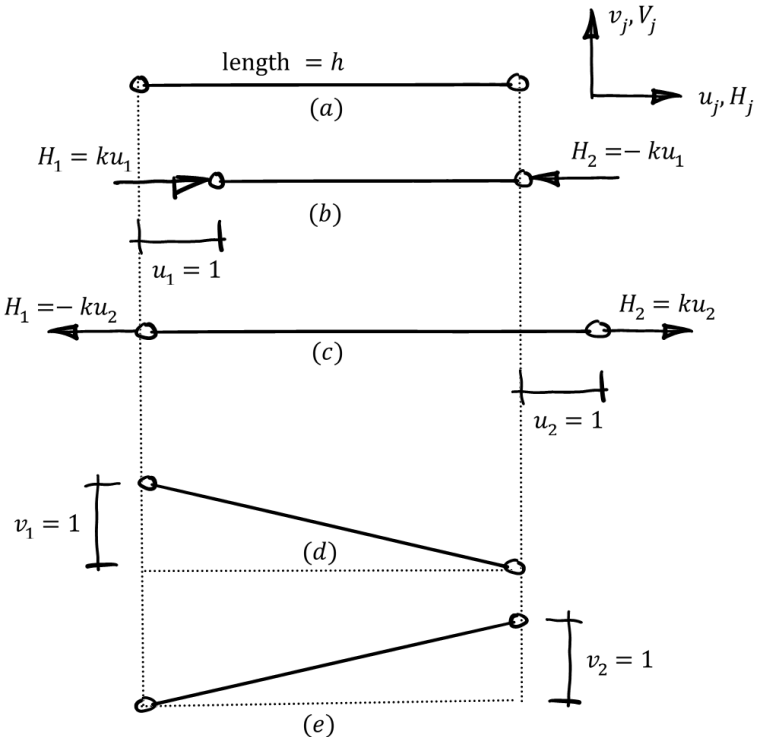


Fig. 14.2. Derivation of the stiffness matrix of a truss member which is initially free of any applied forces.

forces change by the requirement to now support a tensile force in the truss, to $H_1 = -ku_2$ and $H_2 = ku_2$. This can be recorded as the third column of the stiffness matrix, as that corresponds to the displacement u_2:

$$\mathbf{K} = \begin{bmatrix} k\,, & ?\,, & -k\,, & ? \\ 0\,, & ?\,, & 0\,, & ? \\ -k\,, & ?\,, & k\,, & ? \\ 0\,, & ?\,, & 0\,, & ? \end{bmatrix} \tag{14.2}$$

As the vertical displacements of the joints in In Figure 14.2(d)-(e) do not require any forces to exist for equilibrium, we can complete the stiffness matrix by replacing the question marks with zeroes:

$$\mathbf{K} = \begin{bmatrix} k\,, & 0\,, & -k\,, & 0 \\ 0\,, & 0\,, & 0\,, & 0 \\ -k\,, & 0\,, & k\,, & 0 \\ 0\,, & 0\,, & 0\,, & 0 \end{bmatrix} \tag{14.3}$$

A bar in a general orientation in the plane will have a stiffness matrix that can be obtained by transforming from the global displacements $\mathbf{U}_j$ to the local ones (u_j, v_j) exactly as in (14.4):

$$u_j = [(\mathbf{e}_x)_X, (\mathbf{e}_x)_Y] \begin{bmatrix} (\mathbf{U}_j)_X \\ (\mathbf{U}_j)_Y \end{bmatrix}, \quad v_j = [-(\mathbf{e}_x)_Y, (\mathbf{e}_x)_X] \begin{bmatrix} (\mathbf{U}_j)_X \\ (\mathbf{U}_j)_Y \end{bmatrix} \tag{14.4}$$

Using the abbreviations $c = \Delta X/h$, $s = \Delta Y/h$ we can write $(\mathbf{e}_x)_X = c$, and $(\mathbf{e}_x)_Y = s$, and the stiffness matrix transformed into global coordinates reads

$$\mathbf{K} = \begin{bmatrix} c\,, & -s\,, & 0\,, & 0 \\ s\,, & c\,, & 0\,, & 0 \\ 0\,, & 0\,, & c\,, & -s \\ 0\,, & 0\,, & s\,, & c \end{bmatrix} \begin{bmatrix} k\,, & 0\,, & -k\,, & 0 \\ 0\,, & 0\,, & 0\,, & 0 \\ -k\,, & 0\,, & k\,, & 0 \\ 0\,, & 0\,, & 0\,, & 0 \end{bmatrix} \begin{bmatrix} c\,, & s\,, & 0\,, & 0 \\ -s\,, & c\,, & 0\,, & 0 \\ 0\,, & 0\,, & c\,, & s \\ 0\,, & 0\,, & -s\,, & c \end{bmatrix} \tag{14.5}$$

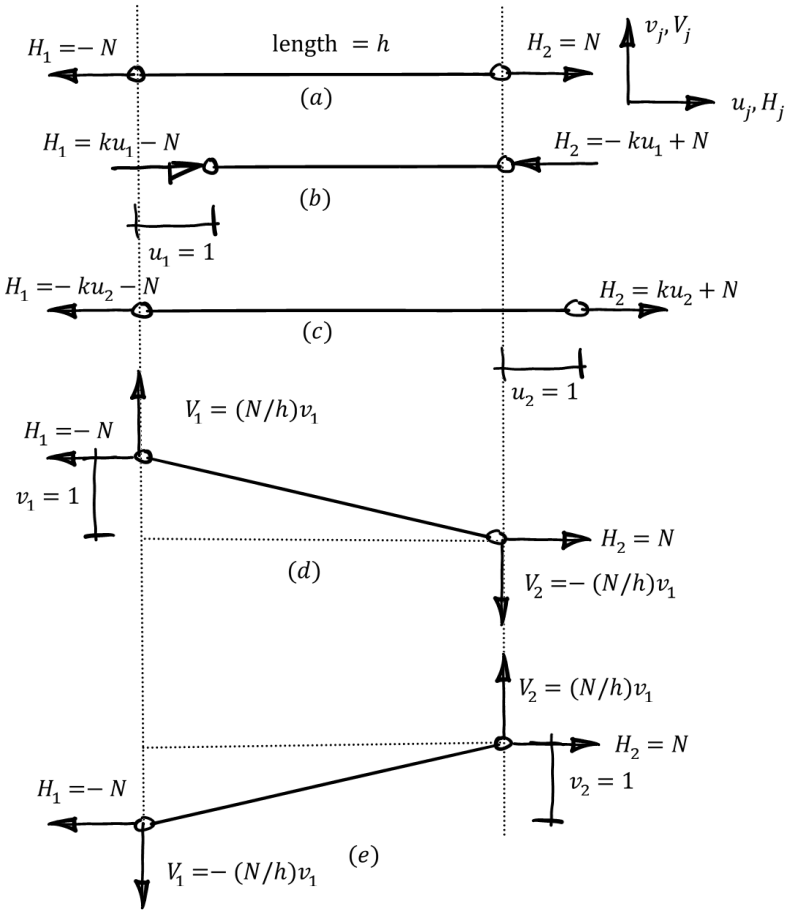


Fig. 14.3. Derivation of the stiffness matrix of a truss member which is under equilibrium conditions with an initial axial force N.

This expression is found to be identical to the one derived earlier using the strain-displacement matrix (7.25). It suffices to subdivide the matrix into 2×2 submatrices and to realize that

$$\begin{bmatrix} c\,, & -s \\ s\,, & c \end{bmatrix} \begin{bmatrix} k\,, & 0 \\ 0\,, & 0 \end{bmatrix} \begin{bmatrix} c\,, & s \\ -s\,, & c \end{bmatrix} = \begin{bmatrix} c \\ s \end{bmatrix} k \begin{bmatrix} c\,, & s \end{bmatrix} \tag{14.6}$$

and so on.

14.1.2 Stiffness matrix of structure with initial stress

Here we deal with structure (b) from Figure 14.1. Figure 14.3(a) shows the configuration of the truss member when it is initially experiencing an axial force N ($N > 0$ corresponds to a tensile force). Under these circumstances, the forces acting on the joints are $H_1 = -N$, $V_1 = 0$, $H_2 = N$, $V_2 = 0$. Now we subject the truss member to enforced displacements. Figure 14.3(b) shows how the forces required to hold the truss member in equilibrium change when we move the joint 1 to the right by $u_1 = 1$. The force $H_1 = -N$ changes to $H_1 = ku_1 - N$, $H_2 = N$ changes to $H_2 = -ku_1 + N$, and the vertical forces do not change away from zero. Analogous change can be detected in Figure 14.3(c) when we move the joint 2 horizontally by $u_2 = 1$.

$$\mathbf{K} = \begin{bmatrix} k\,, & ?\,, & -k\,, & ? \\ 0\,, & ?\,, & 0\,, & ? \\ -k\,, & ?\,, & k\,, & ? \\ 0\,, & ?\,, & 0\,, & ? \end{bmatrix} \tag{14.7}$$

Figure 14.3(d) is where it gets interesting. The joint 1 is moved vertically upward by $v_1 = 1$. Again, we pretend that this deflection is actually very small compared to the length of the member, so that the angle by which the member rotates can be approximated by v_1/h (i.e. the tangent). We see that in order to get equilibrium, joint 1 needs a vertical force pointing upwards, and joint 2 needs a vertical force of equal magnitude pointing downwards: To determine this we can use for instance the balance of moments equation written for a pivot placed at joint 2 ($Nv_1 = V_1 h$), and then the sum of the forces in the vertical direction being zero. The horizontal forces do not change, but the vertical forces become $V_1 = (N/h)v_1$ and $V_2 = -(N/h)v_1$. Therefore, we can update the stiffness matrix as

$$\mathbf{K} = \begin{bmatrix} k\,, & 0\,, & -k\,, & ? \\ 0\,, & N/h\,, & 0\,, & ? \\ -k\,, & 0\,, & k\,, & ? \\ 0\,, & -N/h\,, & 0\,, & ? \end{bmatrix} \tag{14.8}$$

Similarly, in Figure 14.3(e) we can see that the horizontal forces do not change in response to moving joint 2 upwards by $v_2 = 1$, but the vertical forces become $V_1 = -(N/h)v_1$ and $V_2 = (N/h)v_1$. Therefore, we can update the stiffness matrix as

$$\mathbf{K} = \begin{bmatrix} k\,, & 0\,, & -k\,, & 0 \\ 0\,, & N/h\,, & 0\,, & -N/h \\ -k\,, & 0\,, & k\,, & 0 \\ 0\,, & -N/h\,, & 0\,, & N/h \end{bmatrix} \tag{14.9}$$

We can separate the matrix in (14.9) into two parts, the elastic part

$$\mathbf{K}_E = \begin{bmatrix} k\,, & 0\,, & -k\,, & 0 \\ 0\,, & 0\,, & 0\,, & 0 \\ -k\,, & 0\,, & k\,, & 0 \\ 0\,, & 0\,, & 0\,, & 0 \end{bmatrix} \tag{14.10}$$

and the "initial-stress" part (aka geometric stiffness)

$$\mathbf{K}_G = \begin{bmatrix} 0\,, & 0\,, & 0\,, & 0 \\ 0\,, & N/h\,, & 0\,, & -N/h \\ 0\,, & 0\,, & 0\,, & 0 \\ 0\,, & -N/h\,, & 0\,, & N/h \end{bmatrix} \tag{14.11}$$

Both the elastic and the initial stress matrix can again be expressed in the global Cartesian coordinate system, using the same approach as that discussed in the previous section.

14.1.3 Structural stability

The equations of static structural equilibrium can be written in terms of the global stiffness matrix (the elastic stiffness matrix), the global displacement vector, and the global load vector

$$\mathbf{K}_E \mathbf{U}_{\text{st}} = \mathbf{F} \tag{14.12}$$

This will yield the static solution $\mathbf{U}_{\text{st}}$. We can substitute this solution into the calculation of the internal forces, and we can then obtain the initial-stress stiffness matrix $\mathbf{K}_G(\mathbf{U}_{\text{st}})$. Here by this notation we mean that the initial-stress stiffness matrix was calculated for the internal forces that the structure experiences under the loading $\mathbf{F}$.

Now, we understand that the response of the structure which already experiences the loading $\mathbf{F}$ can be described more accurately by including in the stiffness matrix the effect of initial stress. Thus, if the loading increases by $\Delta\mathbf{F}$, the response may be described by

$$[\mathbf{K}_E + \mathbf{K}_G(\mathbf{U}_{\text{st}})]\,\Delta\mathbf{U} = \Delta\mathbf{F} \tag{14.13}$$

Provided the stiffness matrix $[\mathbf{K}_E + \mathbf{K}_G(\mathbf{U}_{\text{st}})]$ is not singular, the additional displacement $\Delta\mathbf{U}$ can be solved for and will be unique. On the other hand, when the matrix $[\mathbf{K}_E + \mathbf{K}_G(\mathbf{U}_{\text{st}})]$ *is singular*, the additional displacement $\Delta\mathbf{U}$ may or may not be solved for, and it will not be unique. We call the case of the singular stiffness matrix ***instability***. The structure becomes unstable: the structure experiences ***buckling*** (the structure buckles).

The elastic stiffness matrix $\mathbf{K}_E$ is presumably not singular by itself. What could make the sum of the elastic and some multiple of the initial-stress matrix singular is some particular value of the multiplier. The question is, what level of loading (what value of the multiplier) can make the overall stiffness matrix singular?

For this purpose we introduce the loading parameter (loading multiplier) λ and we consider loading the structure with a multiple of the original loading, $\lambda\mathbf{F}$. Because (14.12) is linear and the principle of superposition holds, we immediately know that the response to this changed loading will be

$$\mathbf{K}_E\lambda\mathbf{U}_{\text{st}} = \lambda\mathbf{F} \tag{14.14}$$

The initial-stress stiffness matrix for this displacement is

$$\mathbf{K}_G(\lambda\mathbf{U}_{\text{st}}) = \lambda\mathbf{K}_G(\mathbf{U}_{\text{st}}) \tag{14.15}$$

Thus, we can ask for which loading level will the overall stiffness matrix be singular, which is equivalent to the *buckling problem.*

Buckling problem.

The buckling problem is stated as: find a value of the loading parameter λ_{crit} such that the overall stiffness matrix becomes singular, which mathematically means there is a solution to

$$(\mathbf{K}_E + \lambda\mathbf{K}_G(\mathbf{U}_{\text{st}}))\,\mathbf{U}_{\text{crit}} = \mathbf{0} \tag{14.16}$$

The displacement vector $\mathbf{U}_{\text{crit}}$ is the buckling shape. It represents (qualitatively) the shape the structure would assume upon buckling. The shape is not entirely quantitative because its magnitude is indeterminate. The buckling problem is an eigenvalue problem: the critical value of the loading parameter λ_{crit} is an eigenvalue.

The solution proceeds in three steps:

1. For a given load vector $\mathbf{F}$, find the solution $\mathbf{U}_{\text{st}}$ of the static equilibrium (14.12).
2. Compute the initial-stress matrix $\mathbf{K}_G(\mathbf{U}_{\text{st}})$.
3. Solve the eigenvalue problem (14.16).

14.1.4 A simple example: a single truss member on a spring foundation

In this example we consider a single truss member mounted on grounded springs, is subjected to an internal force N, which is introduced into the member by horizontal forces acting at the joints (Figure 14.4).

The elastic resistance of the built-up structure is described starting with the matrix for a single truss member (14.10), augmented with the stiffness coefficients of the three springs to ground

$$\mathbf{K}_E = \begin{bmatrix} k + k_{\text{ax}}\,, & 0\,, & -k\,, & 0 \\ 0\,, & k_{\text{tr}}\,, & 0\,, & 0 \\ -k\,, & 0\,, & k\,, & 0 \\ 0\,, & 0\,, & 0\,, & k_{\text{tr}} \end{bmatrix} \tag{14.17}$$

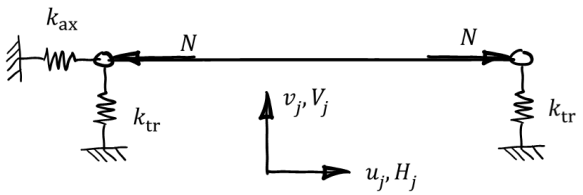


Fig. 14.4. A single truss member mounted on grounded springs, subjected to an internal force N. If the forces act on the joints as shown with the arrows, N is a tensile force.

For definiteness, we will assume that the applied loading generates an internal force of magnitude $N > 0$. The initial stress matrix is then

$$\mathbf{K}_G = \begin{bmatrix} 0\,, & 0\,, & 0\,, & 0 \\ 0\,, & N/h\,, & 0\,, & -N/h \\ 0\,, & 0\,, & 0\,, & 0 \\ 0\,, & -N/h\,, & 0\,, & N/h \end{bmatrix} \tag{14.18}$$

The buckling problem is therefore set up as

$$\left(\begin{bmatrix} k + k_{\text{ax}}\,, & 0\,, & -k\,, & 0 \\ 0\,, & k_{\text{tr}}\,, & 0\,, & 0 \\ -k\,, & 0\,, & k\,, & 0 \\ 0\,, & 0\,, & 0\,, & k_{\text{tr}} \end{bmatrix} + \lambda \begin{bmatrix} 0\,, & 0\,, & 0\,, & 0 \\ 0\,, & N/h\,, & 0\,, & -N/h \\ 0\,, & 0\,, & 0\,, & 0 \\ 0\,, & -N/h\,, & 0\,, & N/h \end{bmatrix} \right) \mathbf{U}_{\text{crit}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{14.19}$$

Python
- script

and can be solved with a simple Python script, in a symbolic form. The eigenvalues of the matrix problem result as

```
{k + kax/2 - sqrt(4*k**2 + kax**2)/2: 1,
 k + kax/2 + sqrt(4*k**2 + kax**2)/2: 1,
 ktr: 1,
 (2*N*lambd + h*ktr)/h: 1}
```

in the form of a list of pairs `eigenvalue: multiplicity`. We can see all eigenvalues have the multiplicity of one, and only one depends on the buckling factor λ. Setting the expression `2*N*lambd + h*ktr` to zero gives an equation that can be solved for the value of λ that will give a zero eigenvalue, $\lambda = -k_{\text{tr}}h/(2N)$. Note that buckling factor $\lambda < 0$ for $N > 0$: this is simply an indication that a compressive force is necessary for instability; the structure will not become unstable for a tensile force in the truss.

The buckling factor can come out of the calculation as negative. That simply means that the direction of loading needs to be reversed in order to give buckling. If the loading can legitimately switch direction, the fundamental buckling factor should be chosen by looking at the absolute value: the smallest eigenvalue in absolute value determines the instability; otherwise, if the direction of the loading is fixed and cannot be flipped, the instability is determined by the smallest nonnegative eigenvalue.

14.2 General buckling considerations

The effect of compressive stress in the members of a truss structure is to cause instability by generating the initial-stress matrix. This matrix essentially cancels some ways in which the structure can mobilize elastic resistance to deformation. If the elastic resistance is canceled by the sideways action of the initial-stress matrix, it becomes possible for the structure the deform without resistance: it buckles.

The same principle is at play whenever there is a compressive stress within the structure. For instance, a floor plank under bending loads from the top will experience compressive axial stresses from the neutral axis towards the top of the cross section. These compressive stresses generate an initial-stress matrix that weakens the lateral resistance of the beam. If it isn't braced, under a sufficiently large bending moment the floor plank may buckle laterally (sometimes referred to as "tippling").

If a structure experiences compressive stresses, it *can* buckle. Whether that actually happens depends on the magnitude of the buckling factor, of course. Compressive stresses make structures more flexible; on the other hand, if a structure is subject only to tensile stresses, their effect will be to make the structure stiffer.

14.3 Local versus global buckling

Abaqus
- CAE file

Figure 14.5 shows a planar truss strut studied for instance in [RoSc]. The Abaqus model is available for download. The pin-jointed bars are of solid circular cross section, but here we consider bars of increased slenderness: cross sectional areas are 1.0 in^2 for the diagonals, and 2.0 in^2 for all the other bars. The longest bar is 26 in long. The material is assumed to have a Young's modulus of 10^7 psi.

The buckling load multiplier can be readily established for the structure with a "Buckle" analysis procedure as 3.582×10^5 (a unit load, 1 lb, was applied). However, the analyst must view this number with a grain of salt: when the Euler buckling load is calculated for each of the longitudinal bars, we find that the Euler force is exceeded by the force in the bars by a considerable margin:

$$N_{\text{Euler}} = \frac{\pi^2 EI}{L^2} = \frac{\pi^2 10^7 \frac{\pi r^4}{4}}{26^2} = 4.64733 \times 10^4 \text{ lb} \tag{14.20}$$

versus $(1/2) \times 3.582 \times 10^5 = 1.791 \times 10^5$ lb. In other words, the truss would in reality fail by a *local* buckling of one of the longitudinal members rather than by the global buckling of the entire structure! The failing of the "Buckle" analysis is that the buckling of individual members is not considered in the model. If local buckling is a possibility, not having this phenomenon in the model will lead to the wrong results [Hart].

Fig. 14.5. Thompson-Hunt truss strut.

14.4 Composites and buckling

Composites have some characteristics that increase the likelihood that their mode of failure will be buckling:

- Bending deflections depend on the product of the elasticity modulus and the second moment of area. Since it is expensive and technically difficult to increase the modulus of elasticity of fiber reinforced composites, composite structures typically have large-envelope cross sections.

- Composites are quite strong, and consequently it is possible to carry large loads with a small area. Often this translates into components with small thickness (since we want a large a second moment of area – viz. above).
- Composites are relatively expensive compared to conventional materials. This dictates using as little material as possible, and, again, this translates to using small thickness.

These driving forces towards cross section profiles that are large and spindly mean that buckling could be of concern. And since we can relatively successfully increase the second moment of area (and hence the global buckling load), we must worry about local buckling of the thin walls of composite components.

Composite beams modeled with beam elements will not be able to capture the buckling modes that are local – distortion of the cross section, and buckles within the walls – and these buckling modes will be missed. The solution may be to use shell models for buckling analyses of composite beams.

Further, due to the heterogeneous and anisotropic nature of composites there is also the possibility of material-level buckling: the compressed fibres of the reinforcement can buckle due to an insufficient restraint of the fibres by the matrix. At the intermediate level composites can also experience buckling within the composite laminate itself, where individual lamina buckle and separate (delaminate) from their neighbors.

It is important to realize that the individual finite elements representing composite structures will not be able to reflect the fiber buckling and delamination and the corresponding buckling load factors will not be computed during buckling analysis (unless the finite elements explicitly incorporate these failure mechanisms, and very few do).

15

Unconstrained Optimization

Summary

1. A number of basic techniques in structural analysis rely on results from the area of optimization. Main idea: Equilibrium of structures and minimization of potential functions are intimately tied. Equilibrium equations are the conditions of the minimum.
2. Stability of structures is connected to the classification of the stiffness matrix. Main idea: positive definite matrices correspond to stable structures.
3. The line search is a basic tool in minimization. Main idea: Monitor the gradient of the objective function. Minimum (extremum) is indicated when the gradient becomes orthogonal to the line search direction.

15.1 Basic ideas

We shall start to explore the subject of ***unconstrained optimization*** on problems of static equilibrium of bar structures. The optimization problems are unconstrained in the sense that the minimum or maximum is sought for variables (displacements) that can take on any value; we say they are not constrained. An example of constrained optimization – equilibrium problem with a contact condition – is treated in a subsequent chapter.

The optimization can search either for a minimum or maximum of the so-called objective function (or as it is sometimes called, the cost function). Without any loss of generality we can assume that the objective function is always minimized. It is possible to switch between minimization and maximization by this trick: Let us assume the goal is to minimize the function $f(x)$ by seeking the location of the minimum as

$$\text{Find } x^* \text{ such that } f(x^*) \leq f(x) \text{ for all } x \,. \tag{15.1}$$

This can be easily changed into a maximization task by flipping the objective function about the horizontal axis (i.e. changing its sign) and seeking the maximum as

$$\text{Find } x^* \text{ such that } -f(x^*) \geq -f(x) \text{ for all } x \,. \tag{15.2}$$

15.1.1 Two degrees of freedom static equilibrium: unstable structure

Consider the simple static system in Figure 15.1. The stretch of the spring can be expressed as

$$s = x_1 \cos 30^o + x_2 \sin 30^o \,.$$

The energy stored in the spring (the energy of deformation) is

$$DE = \frac{1}{2} k s^2 \, .$$

Using a matrix expression (for reasons that will become clear later), the stretch of the spring can be expressed as

$$s = \begin{bmatrix} \cos 30^o & \sin 30^o \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \, .$$

The energy stored in the spring can also be written as

$$DE = \frac{1}{2} k s^T s \, ,$$

where by s^T we mean the transpose (never mind that the transpose of a scalar doesn't do anything). Substituting for the stretch we obtain

$$DE = \frac{1}{2} k \left(\begin{bmatrix} \cos 30^o & \sin 30^o \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \right)^T \begin{bmatrix} \cos 30^o & \sin 30^o \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \, ,$$

which gives in short order

$$DE = \frac{1}{2} k \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} \cos 30^o \\ \sin 30^o \end{bmatrix} \begin{bmatrix} \cos 30^o & \sin 30^o \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \, .$$

If we define the matrix

$$\mathbf{K} = k \begin{bmatrix} \cos 30^o \\ \sin 30^o \end{bmatrix} \begin{bmatrix} \cos 30^o & \sin 30^o \end{bmatrix} = k \begin{bmatrix} \cos 30^o \cos 30^o \, , & \cos 30^o \sin 30^o \\ \sin 30^o \cos 30^o \, , & \sin 30^o \sin 30^o \end{bmatrix} \, , \tag{15.3}$$

we can write the energy stored in the spring as

$$DE = \frac{1}{2} \mathbf{x}^T \mathbf{K} \mathbf{x} \, , \tag{15.4}$$

where we write for convenience

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \, .$$

The matrix $\mathbf{K}$ is the stiffness matrix.

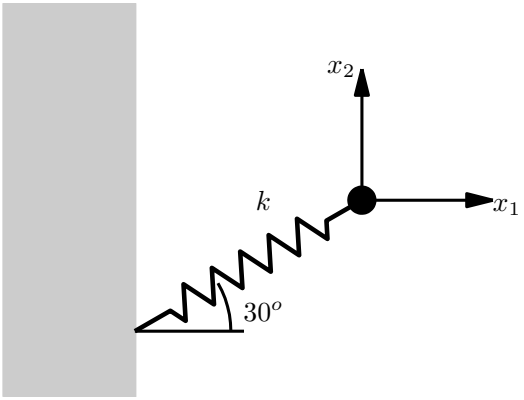


Fig. 15.1. Static equilibrium of particle suspended on a spring.

The energy DE is a quadratic function of the displacements x_1, x_2. Expressed in the form of the matrix expression (15.4), it is called a ***quadratic form***. In the optimization arena the energy

function DE would be referred to as the ***objective function***, and the displacements that minimize this function would be sought as the solution to the optimization problem.

As a scalar function of two variables, x_1, x_2, the energy DE may be visualized as a surface raised above the plane x_1, x_2. Figure 15.2 shows the surface of the deformation energy in two views: from the top, and isometric. We can see that the surface is a trough, with the bottom indicated by the thick white level curve at $DE = 0$. This level curve appears to run in the direction

$$\begin{bmatrix} -\sin 30^o \\ \cos 30^o \end{bmatrix} .$$

Taking the displacement as

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \alpha \begin{bmatrix} -\sin 30^o \\ \cos 30^o \end{bmatrix} \tag{15.5}$$

we can compute the stretch in the spring as

$$s = \begin{bmatrix} \cos 30^o & \sin 30^o \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} \cos 30^o & \sin 30^o \end{bmatrix} \alpha \begin{bmatrix} -\sin 30^o \\ \cos 30^o \end{bmatrix} = 0 .$$

This confirms that for the displacements (15.5) the energy stored in the spring is equal to zero. This property is encountered in structures which are mechanisms: they can move in some ways without deformation, that is without the need to store energy. Such structures are unstable.

Furthermore, we can see that for the displacements (15.5) we get

$$\mathbf{K}\mathbf{x} = \mathbf{0} .$$

Thus we see that the matrix $\mathbf{K}$ of (15.3) is singular. Clearly, the fact that the matrix is singular and the fact that the deformation energy may be zero for some nonzero displacement are related.

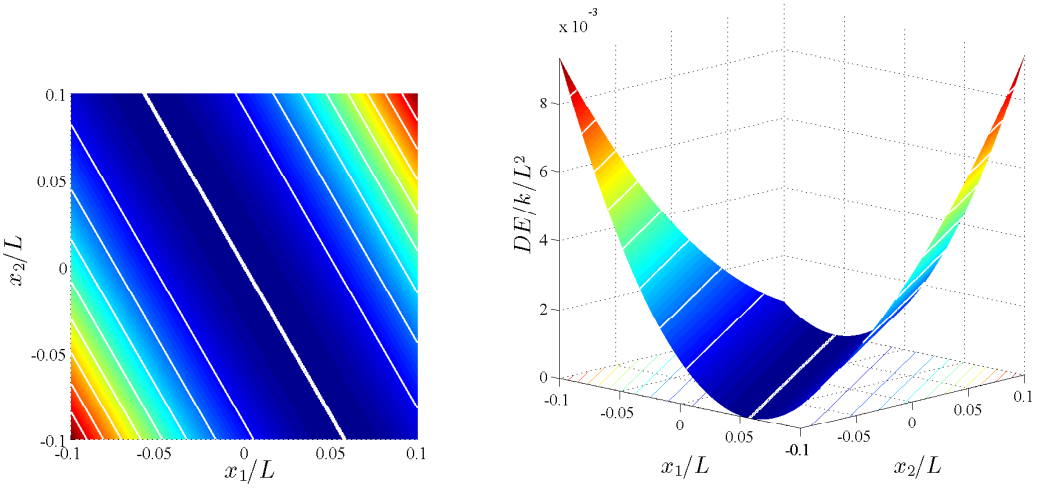


Fig. 15.2. Static equilibrium of particle suspended on a spring. The surface of the deformation energy.

Modify the code below to display the surface in Figure 15.2. The second and the last line need to be modified to reflect a particular objective function. The last line is supposed to draw arrows representing the gradient.

15.1.2 Two degrees of freedom static equilibrium: stable structure

Figure 15.3 shows a static equilibrium system with two degrees of freedom as before, but this time with two springs. The stiffness matrix consists of two contributions, one from each spring

$$\begin{aligned}\mathbf{K} &= k\begin{bmatrix}\cos 30^o\\ \sin 30^o\end{bmatrix}\begin{bmatrix}\cos 30^o & \sin 30^o\end{bmatrix} + (k/2)\begin{bmatrix}\cos -30^o\\ \sin -30^o\end{bmatrix}\begin{bmatrix}\cos -30^o & \sin -30^o\end{bmatrix}\\ &= k\begin{bmatrix}\cos 30^o\cos 30^o\,, & \cos 30^o\sin 30^o\\ \sin 30^o\cos 30^o\,, & \sin 30^o\sin 30^o\end{bmatrix}\\ &\quad +(k/2)\begin{bmatrix}\cos -30^o\cos -30^o\,, & \cos -30^o\sin -30^o\\ \sin -30^o\cos -30^o\,, & \sin -30^o\sin -30^o\end{bmatrix}\,. \end{aligned} \tag{15.6}$$

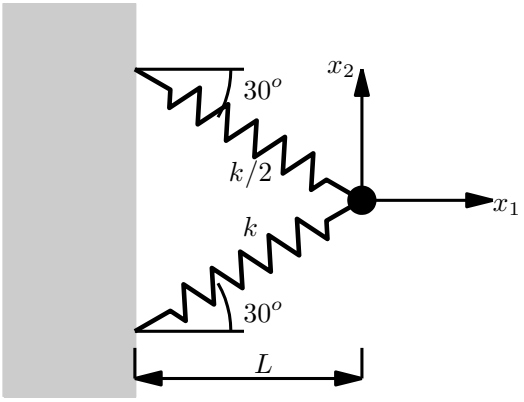


Fig. 15.3. Static equilibrium of particle suspended on two springs.

Figure 15.4 shows the variation of the deformation energy as a function of x_1, x_2: the only point where the DE assumes the value of zero is at $x_1 = 0, x_2 = 0$. Everywhere else the deformation energy is positive. This means that whenever the displacements are different from zero, the springs will store nonzero energy. This is the hallmark of stable structures.

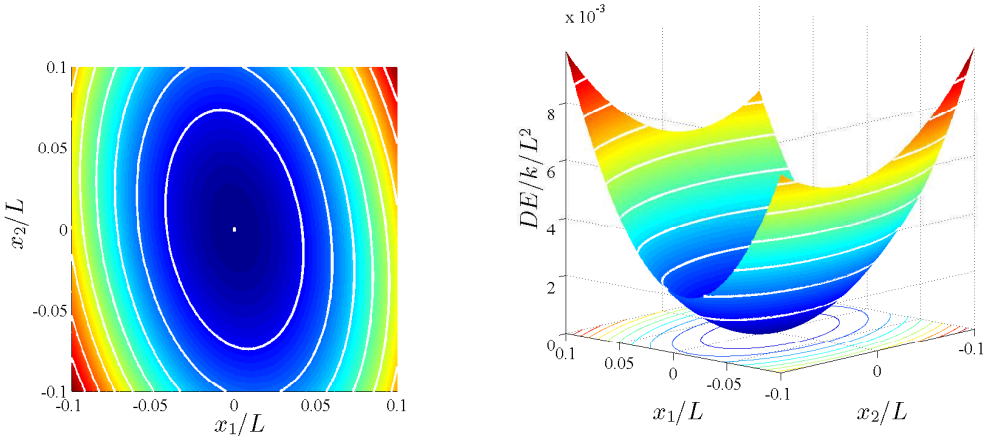


Fig. 15.4. Static equilibrium of particle suspended on a spring. The surface of the deformation energy.

Matrices $\boldsymbol{A}$ that have the property

$$\mathbf{x}^T\mathbf{A}\mathbf{x} > 0$$

for all $\mathbf{x} \neq \mathbf{0}$, and for which

$$\mathbf{x}^T\mathbf{A}\mathbf{x} = 0$$

only for $\mathbf{x} = \mathbf{0}$, are called ***positive definite***. Stable structures have positive definite stiffness matrices. Positive definite matrices are nonsingular (they are regular). This is a fact well worth retaining.

Note that the stiffness matrix is symmetric. An important property of the quadratic forms is that only symmetric matrices contribute to the value of the quadratic form. We can show that as follows: For the moment assume that $\mathbf{A}$ is in general unsymmetric. The quadratic form is a scalar (real number), and as such it is equal to its transpose

$$\mathbf{x}^T\mathbf{A}\mathbf{x} = \left(\mathbf{x}^T\mathbf{A}\mathbf{x}\right)^T \ .$$

Therefore, we can write

$$\mathbf{x}^T\mathbf{A}\mathbf{x} = \mathbf{x}^T\mathbf{A}^T\mathbf{x}$$

or

$$\mathbf{x}^T\mathbf{A}\mathbf{x} - \mathbf{x}^T\mathbf{A}^T\mathbf{x} = \mathbf{x}^T\left(\mathbf{A} - \mathbf{A}^T\right)\mathbf{x} = 0 \ . \tag{15.7}$$

The general matrix $\mathbf{A}$ may be written as a sum of a symmetric matrix and a skew-symmetric (anti-symmetric) matrix

$$\mathbf{A} = \frac{1}{2}\left(\mathbf{A} + \mathbf{A}^T\right) + \frac{1}{2}\left(\mathbf{A} - \mathbf{A}^T\right) \ .$$

In the expression (15.7) we recognize the anti-symmetric part of $\mathbf{A}$. Therefore, we conclude that the anti-symmetric part does not contribute to the quadratic form, only the symmetric part does. Therefore, normally we work only with symmetric matrices in quadratic forms.

15.2 Simple examples of equilibrium

In this section we consider a handful of really simple structural systems. The novelty is the added consideration of the loading. We also investigate the existence of equilibrium.

15.2.1 Two degrees of freedom system

Now consider the system from Figure 15.1 with the stiffness matrix (15.3) loaded by the forces

$$\mathbf{L} = \begin{bmatrix} L_1 \\ L_2 \end{bmatrix} \ .$$

The solution of the static equilibrium problem is (presumably) available from the equilibrium equations

$$\mathbf{K}\mathbf{x} = \mathbf{L} \ .$$

We recall that the matrix (15.3) was singular. This means it is not invertible, and it isn't possible to obtain a solution for *just any* load $\mathbf{L}$ since we cannot write

$$\mathbf{x} = \mathbf{K}^{-1}\mathbf{L} \ .$$

Some particular sets of forces *may* lead to solutions: namely all such loads that are proportional to the eigenvectors of $\mathbf{K}$. Recall that the eigenproblem is written as

$$\mathbf{K}\mathbf{x} = \lambda\mathbf{x}$$

so that setting $\mathbf{L} = \beta\lambda_j\mathbf{x}_j$ where $\mathbf{x}_j$ is an eigenvector, will lead to a solution of the equilibrium problem in the form

$$\mathbf{x} = \beta \mathbf{x}_j \ .$$

The stiffness matrix (15.3) has the eigenvalues $\lambda_1 = 0, \lambda_2 = k$. The corresponding eigenvectors are

$$\mathbf{x}_1 = \begin{bmatrix} \sin 30^o \\ -\cos 30^o \end{bmatrix} \ , \quad \mathbf{x}_2 = \begin{bmatrix} -\cos 30^o \\ -\sin 30^o \end{bmatrix} \ .$$

Evidently, for the zero eigenvalue, no nonzero load is admissible ($\mathbf{L} = \beta \lambda_1 \mathbf{x}_1 = \beta \times 0 \times \mathbf{x}_1 = \mathbf{0}$). For the nonzero eigenvalue, the load is seen to be in the direction of the spring.

15.2.2 One degree of freedom total energy minimization example

Consider a one degree of freedom system (particle on a grounded spring). The deformation energy (elastic energy stored in the spring)

$$DE = \frac{1}{2} x K x \ ,$$

where K is the stiffness constant of the spring. The potential energy of the applied forces is defined as

$$W = -Lx \ .$$

The total energy is defined as

$$TE = DE + W \ . \tag{15.8}$$

The solution for the equilibrium displacement is determined by the principle of minimum total energy: for the equilibrium displacement x^* the total energy assumes the smallest possible value

$$x^* = \arg\min TE \ . \tag{15.9}$$

(It should be read: find x^* as such argument that minimizes TE.) This is an ***unconstrained minimization*** problem: Any x may be considered as a candidate for a solution. The minimum of the total energy is distinguished by the condition that the slope at the minimum is zero:

$$\frac{dTE}{dx} = \frac{d}{dx}\left(\frac{1}{2} x K x - L x\right) = Kx - L = 0 \ .$$

This condition is seen to be simply the equation of equilibrium, whose solution indeed is the equilibrium displacement.

The meaning of equation (15.8) and of the minimization problem (15.9) is illustrated in Figure 15.5. The deformation energy is represented by a parabolic arc (dashed line), which attains zero value (that is its minimum) at zero displacement. The potential energy of the external force is represented by the straight dashed line. The sum of the deformation energy and the energy of the external force tilts the dashed parabola into the solid line parabola, the total energy. That shifts the original minimum on the dashed parabola into the new minimum on the solid parabola (negative value) at x^*. The minimum is easily seen to be

$$\min TE = \frac{1}{2} x^* K x^* - L x^* = \frac{1}{2} x^* K x^* - K x^* x^* = -\frac{1}{2} x^* K x^* = -\frac{1}{2} x^* L \ .$$

15.2.3 Two degrees of freedom total energy minimization example

Now we shall consider again the system of Figure 15.3. The potential energy of the applied forces $\boldsymbol{L}$ is expressed as

$$W = -\boldsymbol{L}^T \boldsymbol{x} \ .$$

The effect of this term on the parabolic surface in Figure 15.4 is very similar to that of Figure 15.5, except now it is in more than one variable: the parabolic surface of the deformation energy is tilted into the parabolic surface of the total energy (TE). This surface is shown in Figure 15.6. The red cross at the bottom represents the solution of the static equilibrium equations.

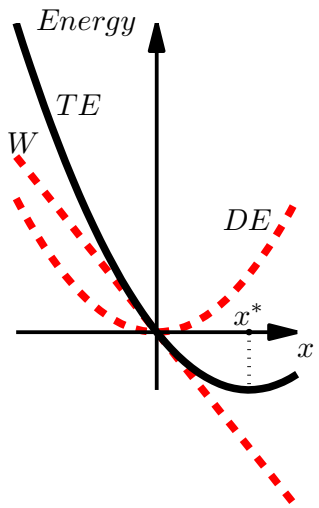


Fig. 15.5. Total energy minimization diagram.

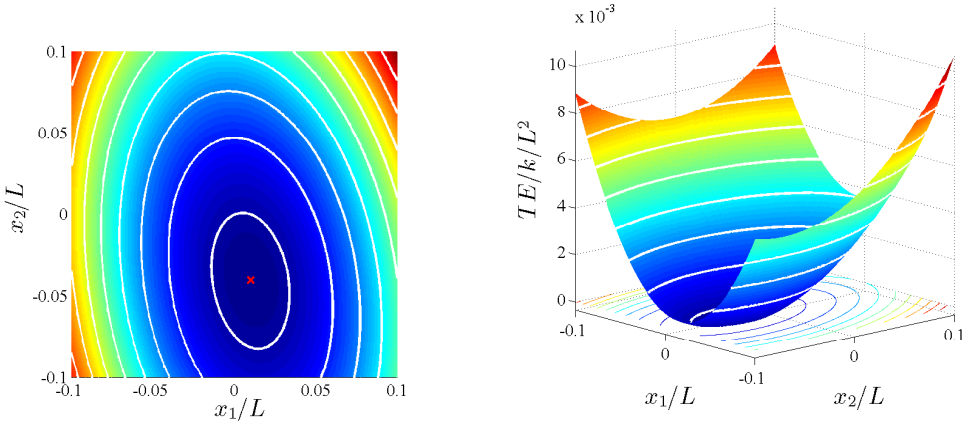


Fig. 15.6. Static equilibrium of particle suspended on two springs. The surface of total energy.

15.3 Application of the minimization of the total potential energy

We have seen above that to find the minimum of the total potential energy TE is equivalent to solving a system of linear equations (the equilibrium equations). By finding the solution to one of these problems, one has automatically solved the other. The application we have in mind here is to solve the system of linear equations

$$\mathbf{Kx} = \mathbf{L}$$

by minimizing the energy

$$TE = \frac{1}{2}\mathbf{x}^T\mathbf{Kx} - \mathbf{L}^T\mathbf{x} \,. \tag{15.10}$$

This is a classical optimization problem. We have an objective function, the total potential energy TE, and our goal is to find the displacement $\mathbf{x}^*$ such that the objective function attains a minimum for that displacement

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} TE \,. \tag{15.11}$$

Since all candidate displacements $\mathbf{x}$ may be considered in the minimization without any restrictions, the minimization problem is called "unconstrained".

15.4 Line search

A commonly used technique for these kinds of problems is the so-called ***line search*** method. It works as follows: start at a point. Then repeat as many times as necessary: pick a direction, and find along this direction a location where the objective function has a lower value than at the start point. Make this the new start point, and go back to picking a direction.

The algorithm is seen to be a sort of walkabout on the surface of the objective function. The goal is to reach the lowest point. Two issues need to be addressed: how to choose the direction, and how to choose where to stop when moving along this direction from the starting point. One particular strategy for addressing the first issue is to choose the direction of the negative gradient at the starting point. Since this direction leads to the steepest decrease of the objective function out of all the directions as the starting point, this strategy is called the ***steepest descent***. For general objective functions, the second issue is difficult to address. To know when to stop when moving from the starting point along the chosen direction could be expensive to compute. Compare with Figure 15.7: the objective function appears to be rather complex, the minimum in the middle is only a local one, not a global minimum: following the drop-off of the objective function in either of the descending corners would lead to further decrease. Fortunately, our present objective function (15.10) is much simpler, and hence much nicer to work with.

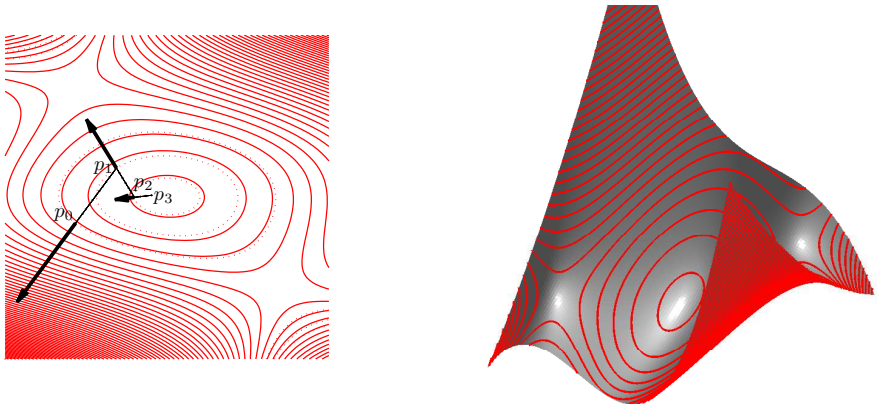


Fig. 15.7. Walk towards the minimum of the objective function. Starting point is p_0, the walk proceeds against the direction of the gradient.

16

Constrained Optimization

Summary

1. Constraints are used to eliminate some values of the optimization variables from consideration (i.e. such values are inadmissible as solutions).
2. Candidate solutions are sought in the so-called feasible regions.
3. Constraints often make the optimization problem much harder, especially because the optima may be located on the boundary of the feasible region.
4. Method of feasible directions is an important tool in constrained optimization.

16.1 Basic ideas

The optimization variables are often Cartesian coordinates of some vector space. To add constraints to the minimization problem means that not all points of the vector space can be solutions. For instance, zero or negative thickness of a metal sheet could be an inadmissible value of the thickness optimization variable. The set from which candidate solutions to the minimization problem may come is called the ***feasible region***. The set from which values of the solution variables cannot be taken is called the *infeasible region*. The addition of constraints could make the minimization problem easier or it could make it harder. It could become easier if the constraints make a lot of the potential minima (or maxima) infeasible which means they don't have to be considered: see Figure 16.1. The

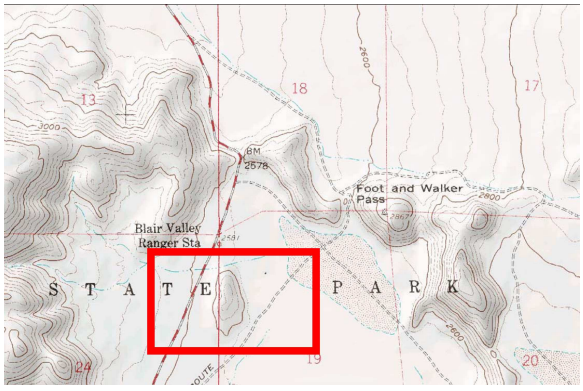


Fig. 16.1. Maximization problem: Climb as high as possible. Much easier when the search is limited to the red rectangle rather than the entire map.

introduction of a constraint could also make the minimization problem harder. For instance, for a convex smooth function we know that there is one and only one minimum point. If we introduce a

constraint as shown in Figure 16.2, the single minimum point (the red cross) suddenly becomes a *set* of minima (i.e. an infinite number of solution points): much harder to deal with computationally.

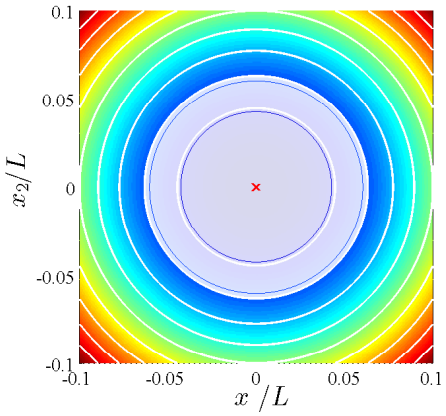


Fig. 16.2. Descend as low as possible. The objective function surface is an axially symmetric paraboloid $f = x_1^2 + x_2^2$, where blue is low and red is high. The feasible region is the outside of the gray circle. Quite hard to compute the solution, since there are an infinite number of points at the same height.

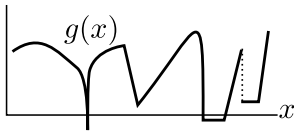


Fig. 16.3. Non-smooth minimization problem in one dimension: minimum may occur locally at the corner, the slope at the minimum may be ill-defined (infinite, non-unique, or nonexistent).

16.2 Locating the minima of smooth objective functions

We will be focusing here on smooth objective functions: functions that have a sufficient number of derivatives, especially at a minimum. This is in contradistinction to the non-smooth minimization problems where the minimum may occur at a "corner" so that in order to make any sort of judgment about the existence of a minimum one has to inspect the entire neighborhood of the suspected minimum point (example in one dimension is shown in Figure **??**). For smooth problems this can be decided by looking at the derivatives of the objective function. As an example of a nonsmooth minimum point consider the surface in Figure **??**. The minimum is reached at a single point, where the gradient (the first derivatives) of the surface is not defined. Clearly, the intuitive notion of the minimum as a point where the first derivative becomes zero needs to be revised for non-smooth optimization problems.

16.2.1 Unconstrained minimization

First we will briefly recall the problem of unconstrained minimization.

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} f \ .$$

The function f has a minimum at the point $\mathbf{x}^*$ if the objective function value at points reached from $\mathbf{x}^*$ in all possible directions $\mathbf{d}$ by steps of length $s \geq 0$ is greater than or equal to the objective function value at $\mathbf{x}^*$:

$$f(\mathbf{x}^* + s\mathbf{d}) \geq f(\mathbf{x}^*) \, .$$

We can express this in terms of the first derivatives (the gradient) as follows: Using the Taylor series, just one term and the remainder in Lagrange form, moving from point $\mathbf{x}^*$ in the unit direction $\mathbf{d}$ with the step length measured by s, we write for the function value at the points $\mathbf{x}^* + s\mathbf{d}$ in the neighborhood of $\mathbf{x}^*$

$$f(\mathbf{x}^* + s\mathbf{d}) = f(\mathbf{x}^*) + \nabla f(\mathbf{x}^* + \theta s\mathbf{d}) s\mathbf{d} \, .$$

Thus we have in the limit of a very short step $s \to 0$

$$\lim_{s \to 0} f(\mathbf{x}^* + s\mathbf{d}) - f(\mathbf{x}^*) = \lim_{s \to 0} \nabla f(\mathbf{x}^* + \theta s\mathbf{d}) s\mathbf{d} \, .$$

In order for the minimum to exist we must have

$$\lim_{s \to 0} f(\mathbf{x}^* + s\mathbf{d}) - f(\mathbf{x}^*) \geq 0$$

so that

$$\lim_{s \to 0} \nabla f(\mathbf{x}^* + \theta s\mathbf{d}) \mathbf{d} \geq 0$$

follows. Since we have

$$\lim_{s \to 0} \nabla f(\mathbf{x}^* + \theta s\mathbf{d}) = \nabla f(\mathbf{x}^*)$$

we see that we must have perforce $\nabla f(\mathbf{x}^*) = \mathbf{0}$ so that

$$\nabla f(\mathbf{x}^*) \mathbf{d} \geq 0$$

could be true for an arbitrary vector $\mathbf{d}$ (in this case, with the equals sign). In fact, a *necessary condition* for the existence of a minimum may be written down in the unconstrained case as

$$\nabla f(\mathbf{x}^*) \mathbf{d} = 0 \, , \quad \forall \mathbf{d} \neq \mathbf{0} \, .$$

16.2.2 Constrained minimization

Now we will consider minima in problems with constraints. As an example consider the following structural optimization task (adapted from Papalambros and Wilde, 2000): design the thickness t and the diameter d of a steel pipe so that its weight is minimized (Figure 16.4). The pipe is a simply supported beam, prestressed by a large tensile force. The optimization is subject to the following constraints: (a) the deflection of the beam δ must be less than a fraction of the span L: $\delta \leq 0.001L$; (b) the tensile stress σ due to the applied force P must be below the yield stress $\sigma \leq S_y$; and (c) the thickness of the pipe is limited from below, $t \geq 0.05$ in. The data: $E = 30 \times 10^6$ psi, $L = 100$ in, $\rho = 0.283$ lb/in^3, $S_y = 36$ kpsi, and $P = 10^4$ lb. The objective function minimization with the three inequality constraints may be simplified to an expression in the two variables t, d

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} f = 88.9td \, , \quad \mathbf{w} = \begin{bmatrix} t \\ d \end{bmatrix}$$

subject to

$$td - 0.0885 \geq 0 \, ,$$
$$d - 0.994 \geq 0 \, ,$$
$$t - 0.05 \geq 0 \, .$$

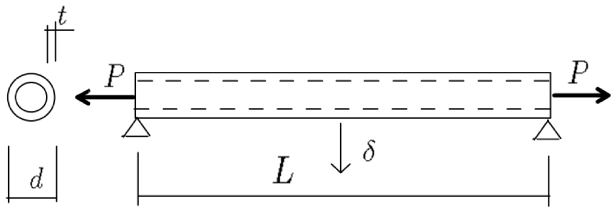


Fig. 16.4. Example of constrained minimization. Optimization of a steel pipe carrying a tensile load, and subject to its own weight.

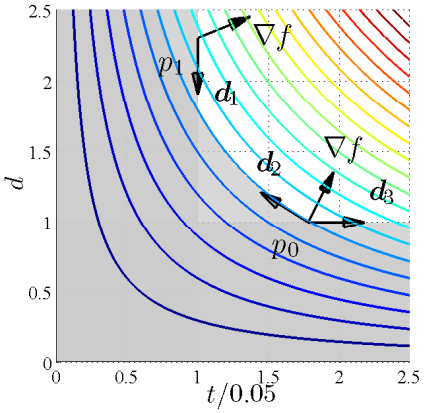


Fig. 16.5. Constrained minimization. The unconstrained minimum of the objective function lies outside of the feasible region. Up to two constraints may be active at any point of the solution set.

Figure 16.5 shows the geometry of the feasible and infeasible regions for the above constrained minimization. The constraints are indicated as overlapping gray regions. The feasible points are located outside of the union of the gray regions. Since the objective function increases towards the upper right corner, we can see that the minimum is obtained at any point of the level curve $td = 0.0885$ between the second and third constraint. The vectors $\mathbf{d}_1$, $\mathbf{d}_2$, $\mathbf{d}_3$ are representatives of the so-called ***limiting feasible directions***. They are tangent to the boundaries of the constraint regions. Their usefulness derives from the fact that they can be viewed as starting directions of the so-called ***feasible sequences***: a sequence of feasible points originating from the tail of and along the feasible direction arrow. Let us now look at some such feasible sequences. Starting from point p_0 along $\mathbf{d}_3$ we see that the objective function value increases. Starting from point p_0 along $\mathbf{d}_2$ we see that there are feasible sequences (along the contour) such that the value of the objective function stays the same. There are of course other sequences starting from the same point in the same direction such that the objective function increases. This information is visually available from the gradient of the objective function at point p_0. If we have

$$\nabla f(p_0)\mathbf{d}_3 > 0\ ,$$

we conclude that the objective function in the direction $\mathbf{d}_3$ increases. The gradient is orthogonal to $\mathbf{d}_2$, and therefore

$$\nabla f(p_0)\mathbf{d}_2 = 0$$

and our observation is that starting along paths tangent to $\mathbf{d}_2$ does not lead to either increase or decrease of the objective function (to first order in the distance).

On the other hand, we see that starting from point p_1 along $\mathbf{d}_1$ produces a decrease in the value of the objective function since

$$\nabla f(p_1)\mathbf{d}_1 < 0 \ .$$

With these observations at hand, we can formulate the conditions from which one can decide whether a given point is a constrained minimum: If at a point $\mathbf{x}^*$ we have for all feasible directions $\mathbf{d}$

$$\nabla f(\mathbf{x}^*)\mathbf{d} \geq 0 \ , \tag{16.1}$$

then such a point is a (local) minimum of the objective function. On the other hand if for any feasible direction we have $\nabla f(\mathbf{x}^*)\mathbf{d} < 0$ the point $\mathbf{x}^*$ is not a minimum, and the location of the minimum may be improved by moving along $\mathbf{d}$. The direction $\mathbf{d}$ is called ***feasible descent direction***.

It follows from (16.1) that minimum does not necessarily require $\nabla f(\mathbf{x}^*) = \mathbf{0}$ because *not all* directions $\mathbf{d}$ are allowed. For directions that *are* allowed (feasible), the gradient of the function may be nonzero as long as (16.1) is satisfied.

16.3 Method of feasible directions

The above observations can be put to use in the so-called method of feasible directions. The method of feasible directions (in one of its many implementations) has important applications in engineering optimal design. It always produces a feasible design: if the initial design is feasible, so are all subsequent designs. The design is improved in each iteration, so that the solution process may be stopped at any time with the result being an improvement on the initial design. Methods of feasible directions involve only a first-order derivatives of the objective function and of the constraints.

The general algorithm can be given as: Start from an initial guess of the location of the minimum.

1. Find a feasible descent direction. If no direction can be found, a minimum has been located.
2. Locate a minimum along the feasible descent direction (one-dimensional constrained minimum line search).
3. Repeat previous steps until a suitable convergence criterion is satisfied.

We see that the basic constituent of the method of feasible descent directions is the search for a constrained minimum along a direction: so-called one-dimensional constrained line search.

16.3.1 One-dimensional constrained line search

The characteristics of the one-dimensional constrained line search are shown in Figure 16.6. The minimum is sought within an interval of the real line, let us say $a \leq x \leq b$. A *local minimum* may obtain at an interior point of the interval (the point x^*), or at the boundary (the point b). The theory of feasible directions may also be illustrated by reference to Figure 16.6. The feasible direction at a is $\mathbf{d}_a$ and a feasible descent sequence is possible from a in the direction of $\mathbf{d}_a$: the point a is therefore not a local minimum. The feasible direction at b is $\mathbf{d}_b$ and there is no feasible descent sequence from b: the point b is therefore a local minimum.

16.4 Static equilibrium with contact

Here we consider a variation on the static equilibrium system of Section 15.1.2. As shown in Figure 16.7, we consider the introduction of an obstacle. There is a gap between the joint and the obstacle of magnitude $g > 0$. The joint connected by the springs cannot move in response to the applied forces arbitrarily. It may be stopped by the obstacle (let us say for a downward directed force of sufficient magnitude).

In terms of the minimization of the total potential energy of the system, the required change will be the introduction of a constraint equation (actually, an inequality). The constraint inequality will evaluate to a non-negative (positive or zero) value for the so-called admissible displacements of the structure. Inadmissible displacements (in optimization lingo they would be called infeasible) would

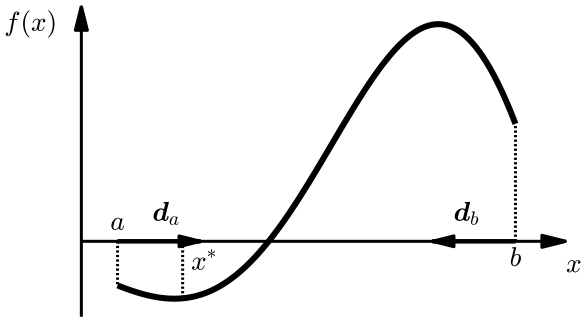


Fig. 16.6. Constrained one-dimensional minimization problem.

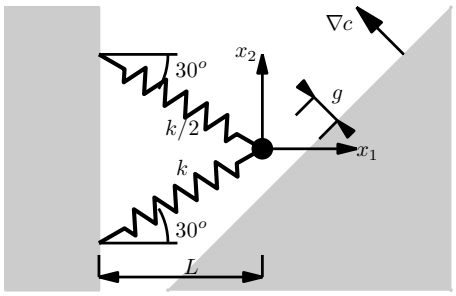


Fig. 16.7. Constrained static two-degree of freedom system.

be associated with negative values of the constraint. In the present situation, the constraint would be written as

$$c(\mathbf{x}) = g + \mathbf{n}^T\mathbf{x} \geq 0 \,,$$

where the normal to the obstacle surface is the gradient of the constraint function, $\mathbf{n}^T = \nabla c$. The originally unconstrained minimization problem (15.11) is now changed into the so-called ***constrained minimization problem***

$$\begin{aligned} &\mathbf{x}^* = \arg\min_{\mathbf{x}} TE \\ &\text{subject to} \quad c(\mathbf{x}) \geq 0 \,. \end{aligned} \tag{16.2}$$

Figure 16.8(a) shows the objective function in the presence of a contact constraint that is sufficiently removed from the loaded joint so that the applied loads do not bring the joint into contact with the obstacle. The minimum of the total potential energy is located in the feasible region. We say that the constraint is inactive. In Figure 16.8(b) we see the situation in which the loads are sufficiently large to push the free joint against the obstacle. The minimum of the total potential energy occurs in the infeasible region, and a contact constraint is active. In other words, the constraint prevents the displacements from reaching the minimum of the original unconstrained total potential energy of the structure.

16.4.1 Truss cantilever example

Figure 16.9 shows the definition of a truss structure, the joint 1 and 6 are pinned, and an obstacle is placed underneath the joints 4 and 5 at various distances $U_{\max}$ (the obstacle is shown schematically with the black rectangle).

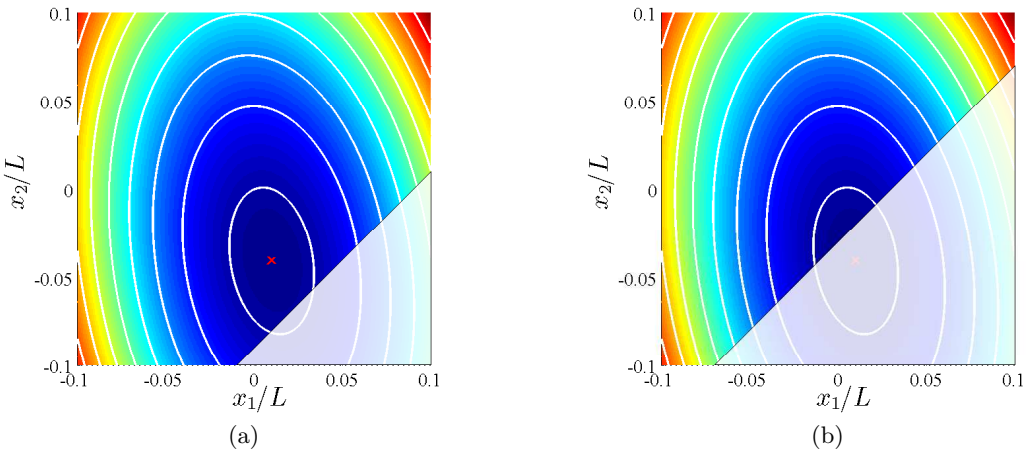


Fig. 16.8. Constrained minimization for the contact problem. (a) The unconstrained minimum of the objective function lies inside of the feasible region. In this case the constraint is inactive. (b) The unconstrained minimum of the objective function lies outside of the feasible region. In this case the joint will be stopped by the obstacle, as the constraint is active since we cannot reach the "unconstrained" minimum without violating the constraint.

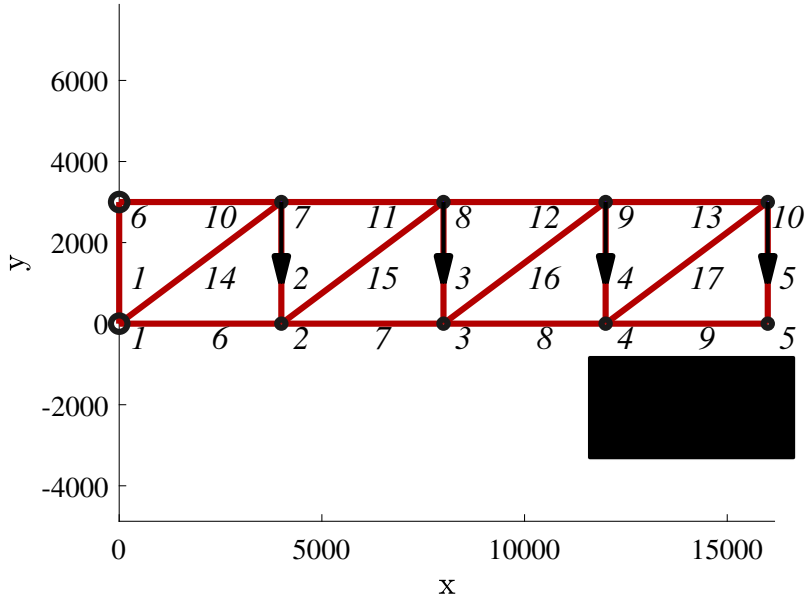


Fig. 16.9. Constrained minimization for the contact problem. Definition of the truss structure. Dimensions are shown in millimeters.

The constrained optimization problem may be formulated as: minimize the total potential energy of the system, where the deflections of the joints are the design variables, subject to the constraints that joints 4 and 5 can only deflect as far as the surface of the obstacle, not more:

$$\begin{aligned} &\mathbf{U}^* = \arg\min_{\mathbf{U}} TE \\ &\text{subject to} \quad -U_{(4,2)} \le U_{\max} \text{ , and } -U_{(5,2)} \le U_{\max} \text{ .} \end{aligned} \tag{16.3}$$

Here $U_{(4,2)}$ is the deflection of the joint 4 in the direction 2 (vertical), and similarly for $U_{(5,2)}$.

Python
- script

The problem is solved in the script `truss_contact.py` that utilizes the Python package `pystran`. The constrained optimization function `scipy.optimize.minimize` was used to find the equilibrium of the truss, subject to the contact constraints.

- URL

The solutions qualitatively vary depending on the distance of the joints from the obstacle. Figure 16.10 shows the optimal solution for four distances of the obstacle: 3, 10, 25, and 100 mm. Figure 16.10 shows the deflections magnified by a factor of 50 (the distance of the obstacle is magnified by the same factor).

Figure 16.10(a): For the smallest distance, $U_{\max} = 3$ mm, both joints 4 and 5 are pushed onto the surface of the obstacle. The minimum of the total potential energy (TE) is constrained, as both constraints are active. The maximum downward vertical deflection actually results at joint 3.

Figure 16.10(b): For the distance $U_{\max} = 10$ mm, both joints 4 and 5 are pushed onto the surface of the obstacle, which is maximum downward vertical deflection. Figure 16.10(c): For the distance $U_{\max} = 25$ mm, only joint 5 is in contact with the surface of the obstacle, which is maximum downward vertical deflection. So there is just one active constraint. And, finally for the distance $U_{\max} = 100$ mm, no joint is in contact with the surface of the obstacle; the maximum downward vertical deflection is only 86 mm. No constraints are active, and the deflection is the same solution as if we were solving the unconstrained problem of total potential energy.

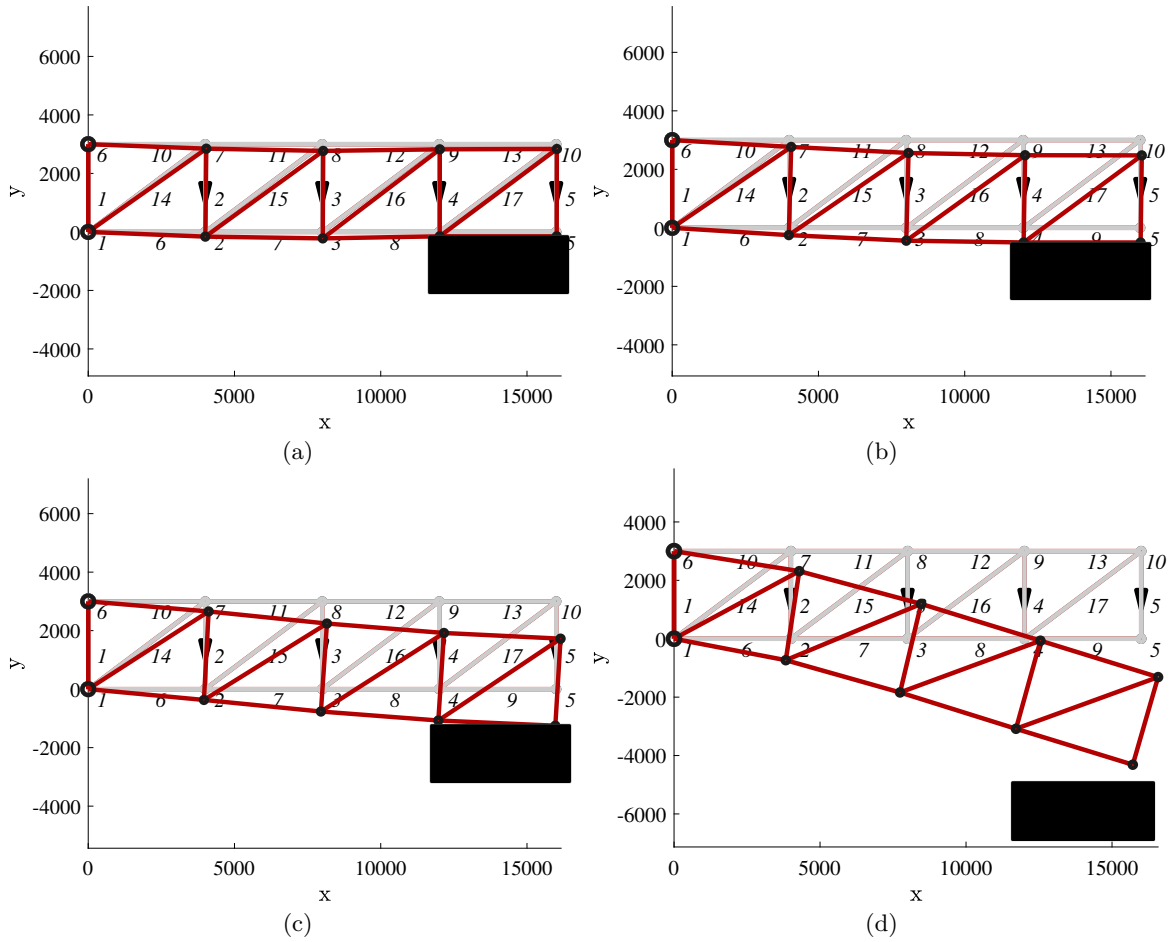


Fig. 16.10. Constrained minimization for the contact problem. (a) The joints are 3 mm from the obstacle before deformation. (b) The joints are 10 mm from the obstacle before deformation. (d) The joints are 25 mm from the obstacle before deformation. (d) The joints are 100 mm from the obstacle before deformation. Note that the distance of the obstacle and the deflections are visually magnified by a factor of 50.

17

Design optimization of truss structures

Summary

1. Design optimization of truss structures is introduced on an example.
2. Minimization of the mass of the structure is carried out by sizing the members and by changing the shape of the structure.
3. Minimization of compliance is introduced as the final example.
4. In anticipation of the need for this terminology in the subsequent chapters, some terms from Abaqus optimization literature are introduced.

17.1 Example structure

We begin with an example. The goal is to improve the performance of the truss cantilever structure shown in Figure 17.1. We wish to use less material while also maintaining constraints: the static deflection due to vertical live loads at joints 1 and 2 must be less than 10 mm in magnitude, and the lowest frequency of free vibration must be at least 13 Hz. The static and dynamic configurations are shown in Figure 17.1. Hence, the objective function (OF) is the mass of the structure, and the design

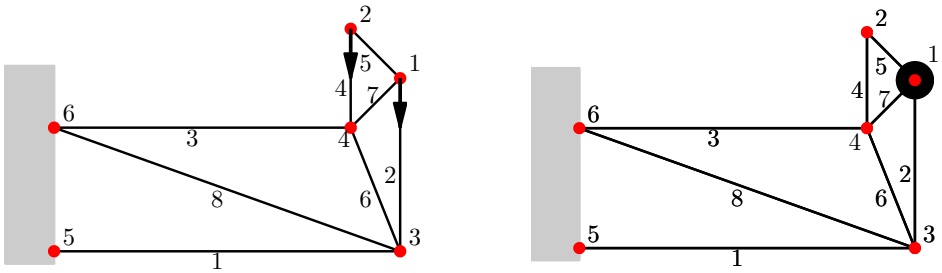


Fig. 17.1. Truss cantilever. Static configuration showing the live loads. Dynamic configuration showing the added mass.

variables depend on which aspect of the structure we subject to variation. We begin by allowing the structure to change shape.

The Python package `pystran` implements models of two-dimensional truss structures, which can be profitably turned to demonstrate use in optimization problems. We use it to compute the

- URL

response of the structure (so-called design response) so that we can evaluate the objective function and the constraints. The actual optimization is carried out by the Python constrained-minimization function `scipy.optimize.minimize`.

17.1.1 Shape optimization

The objective function OF is minimized subject to the constraints on the static and dynamic response:

$$\min OF(DV) \qquad \text{s.t. } \max|U_i| \leq 10 \text{ mm and } f \geq 13 \text{ Hz} \tag{17.1}$$

where the design variables are $DV = [X_3, Y_3, X_4, Y_4]$, that is locations of the joints 3 and 4. (Refer to the script `tcant_shape.py`.)

Python - script

The optimized truss cantilever structure is shown in Figure 17.2. The largest deflection under the live load just under 10.0 mm, and the lowest frequency is approximately 14.40 Hz. The structure volume with respect to the reference configuration is reduced by roughly 25.0%.

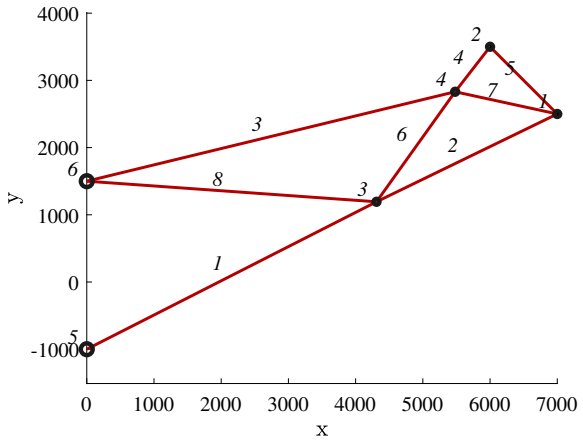


Fig. 17.2. Truss cantilever. Shape optimized to respect the static deflection constraint and also to satisfy the frequency constraint.

17.1.2 Sizing optimization

Next, we will optimize the structure by changing the cross sectional areas of the bars. In the present case, the design parameters are the cross sectional areas of all the members. The data of the problem and the solver are as shown in the linked file (the script `tcant_sizing.py`.) It is worth mentioning that we also employed lower-bound constraints on the cross sectional areas (equal to 1/10 of the initial cross sectional area) – it would not make sense for the cross section area to drop to or below zero, for instance.

Python - script

The original design of the truss does not satisfy the constraints. The truss cantilever structure is shown in Figure 17.1. The largest deflection under the live load is ≈ 15.39 mm, and the lowest frequency is ≈ 10.617 Hz.

As before, the objective function is to be minimized subject to the constraints:

$$\min OF(DV) \qquad \text{s.t. } \max|U_i| \leq 10 \text{ mm and } f \geq 13 \text{ Hz and } DV_i > \text{ initial area } /10 \tag{17.2}$$

where the member cross-section areas are DV_i. Note that we added the lower bound on the design variables.

Note on normalization: in actual code the objective function, the constraints, and indeed the design variables themselves, are best normalized to be close to 1.0 in magnitude. Normalized quantities have the huge advantage that the computation does not depend on the physical units. Also, being close to 1.0 considerably increases numerical robustness of the minimization. As shown in the Python scripts, normalization has been applied throughout.

The computation is implemented in the script `tcant_sizing.py`. The optimized truss cantilever structure is shown in Figure 17.3: the cross sectional areas of the bars are visualized with lines of varying thickness (the line thicknesses don't represent the dimensions literally; rather they show the relative proportions). The largest deflection under the live load is just under 10.0 mm, and the lowest frequency is just over 13.0 Hz. Clearly, in the sizing optimization, the instructions provide enough flexibility so that the optimization process can find a good solution by satisfying the constraints nearly exactly. The structure volume with respect to the reference configuration is reduced by approximately 13.0%.

Python
- script

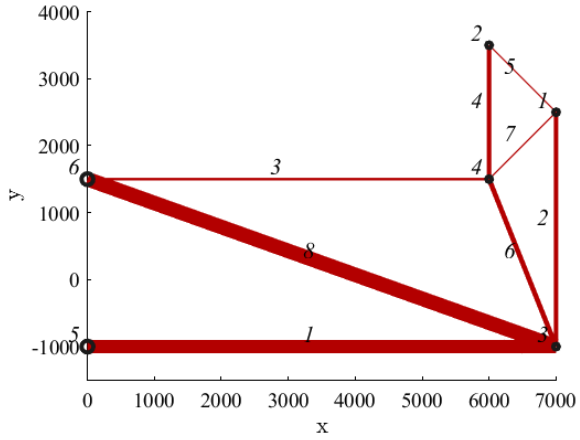


Fig. 17.3. Truss cantilever. Sizing optimization: Cross sectional areas of the truss members are optimized so as to achieve minimum mass while respecting a static deflection constraint and also satisfying a frequency constraint).

17.1.3 Sizing optimization for minimum compliance

Here, we will still optimize the structure by changing the cross sectional areas of the bars, and the design parameters are the cross sectional areas of all the members. Again, lower-bound constraints on the cross sectional areas are employed. Exactly as in the previous section.

However, the optimization problem is now formulated differently. We fix the amount of material to be given by the initial cross sections, and we try to construct a structure that is as stiff as possible, under the constraint of using just the given amount of material. This type of optimization needs to minimize some quantity, and it may be deduced that in order to make a structure as stiff as possible (i.e. to maximize stiffness), the quantity to be minimized is the compliance of such a structure. The more compliant a structure, the more work the external forces do on the displacements that the structure needs to attain equilibrium. Hence, compliance will be minimized when the work of the external forces on the equilibrium displacements is minimized.

As before, the objective function OF (compliance) is to be minimized subject to the constraints as

$$\min OF(DV) = \mathbf{F} \cdot \mathbf{U}(DV) \text{ s.t. } V(DV) \leq V_0 = V(DV_0) \text{ and } (DV)_i > \text{ initial area } /100 \quad (17.3)$$

where the cross-section area of member i is $(DV)_i$, $V(DV)$ and $V(DV_0)$ are the current volume and the initial volume of the structure, and $\mathbf{F}$ and $\mathbf{U}(DV)$ are the vector of the external forces and the vector of displacements of the joints of the structure for the values of the designed variables DV.

Python
- script

The computation is implemented in the function script `tcant_compl.py`. The optimized truss cantilever structure is similar to that in Figure 17.3: the cross sectional areas of the bars are visualized with lines of varying thickness (Figure 17.4). The compliance of the structure decreased to 58% of the original compliance. Since the structure consists of the same volume of material in the optimal configuration as initially, the material was simply reshuffled, making some of the bars stiffer and some more flexible.

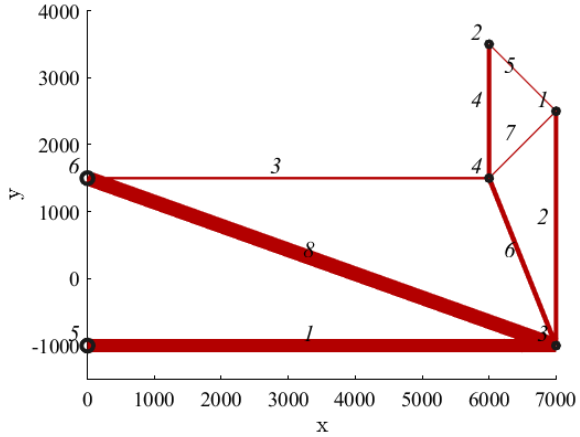


Fig. 17.4. Truss cantilever. Compliance minimization: Cross sectional areas of the truss members optimized to increase stiffness (minimize compliance), while using the same amount of material as initially.

Figure 17.4 is not very different from Figure 17.3. However, close inspection of the optimal areas of the bars reveals some differences. In the present case, the cross sectional areas of the bars are as shown in the left column, whereas previously, when the goal was to minimize the volume of the structure, the areas of the bars were as shown in the right column:

Member	Area (minimize compliance)	Area (minimize mass)
1	2338.66310	2013.57890
2	450.71886	384.90307
3	179.8443	210.23087
4	448.38225	408.42616
5	131.96463	131.95823
6	484.43192	455.24664
7	131.96004	132.25253
8	2677.3727	2307.2853

When the compliance is minimized, we use more material (i.e. the initial volume), whereas when minimizing mass we are allowed to use less material, and we do.

17.2 Some terminology

Here we review what we learned in the example above while firming up the terminology.

17.2.1 Objective function

There is a large variety of possible objective functions: total volume or mass, time to failure, financial cost, ... Here we consider only a single objective, but in practice there are sometimes multiple objectives.

Only minimization of the objective function is considered here, but a combination of minimization and maximization of objectives is also possible (minimize volume of the structure, while maximizing the number of loading cycles to failure).

In our truss examples we consider two kinds of objective function: minimization of mass (or volume), and maximization of stiffness (minimization of compliance). Other objective functions are possible.

17.2.2 Design variables

The things we can change to optimize the structure are call design variables. They can be continuous or discrete. Examples: continuously varying thickness of the sheet of metal, or pick one of the thicknesses provided by the manufacturer.

17.2.3 Design responses

Anything that we can use to measure the performance of the structure, in other words quantities that can be used to formulate the objective function and/or the constraints, can be called design responses: height of the truss, effective stress, volume, ...

17.2.4 Constraints

In real-world applications there are always constraints that the structure must satisfy in order to be acceptable as an optimal solution. Maximum displacement, minimum or maximum natural frequency, minimum buckling load, collapse load, etc.

Constraints which are negative are satisfied; constraints which are positive are violated. In between we have the active constraints.

17.2.5 Active constraint

Constraint which is just about to be violated (in other words, it is negative, both just barely because it is extremely close to zero). An active constraint prevents further optimization of the structure: the optimization process is "pushing" against this constraint, and cannot cross it, since that would lead to a violation of the constraint.

17.2.6 Upper and lower bounds on design variables

Often the design variables cannot just take on any value at all, only some specific ones. For instance, some values do not make sense, such as negative areas or negative Young's modulus. There may be a lower bound, such as the minimum thickness of sheet metal that a manufacturer can produce, or there may be an upper bound, such as the maximum dimension of a truss member that can be made to fit within a joint. These so called box constraints are typically handled much more efficiently when they are separated from the other constraints, and therefore they are not part of the constraint array constructed from all the other constraints in the problem.

17.2.7 Existence of solution

It is certainly possible to overconstrain the optimization problem so that there are no solutions whatsoever. For instance, imagine we wanted to design a truss such that the points of application of external forces would not move at all. We know beforehand that no solution can exist.

If we just required very small displacements, the solutions might or might not exist. If we were allowed to use a lot of material, we could possibly design such a stiff structure to satisfy the constraints of very small displacements; otherwise, the optimization process would fail.

18

Topology Optimization

Summary

1. Principles of topology optimization.
2. Controller-based algorithm.
3. Mesh dependence of the optimization.

In Abaqus optimization is provided by a separate software component called Tosca. Here we will consider the simple variant of topology optimization implemented in Tosca, which is based on the so-called controller-based algorithm. This type of optimization is referred to as "topology" because the structure evolves its shape, including any voids or holes, during the process of optimization: nothing is fixed beforehand (except the boundary conditions, loads and supports).

The controller-based optimization is not very flexible, it only accepts as the objective function the compliance of the structure which is to be minimized, and its single constraint is the amount of material allowed to achieved the optimum distribution of material.

The optimization problem is formulated as follows: The existence of a finite element mesh is a crucial component. The original domain, which we can assume for the moment to be homogenous and isotropic (Figure 18.1(a)), is subdivided into finite elements (Figure 18.1(b)), but we only wish to use a certain fraction of the domain volume, let us say $\alpha < 1$. Therefore, we will divide the finite elements from the mesh into two classes: one which has the full stiffness given by the elastic modulus $\overline{E}$ of the original material (shown in gray), and the other which has no stiffness at all (visualized as empty box). Predictably, only the elements which actually have physical stiffness count towards the volume of the material used, the zero-stiffness elements do not count. So, we can describe the stiffness properties of the elements with the field E such that $E = \overline{E} \neq 0$ in some elements (the gray ones), and $E = 0$ all in the others (empty boxes). Therefore, the optimization may look for such distribution of this field that will minimize the compliance of the structure (i.e. maximize its stiffness) such that we only use a fraction α of the material:

$$E = \arg\min \mathbf{F}^T\mathbf{U} \tag{18.1}$$

$$\text{s.t.} \quad \mathbf{K}(E)\mathbf{U} = \mathbf{F} \quad \text{and} \quad \sum_{e \text{ such that } E \neq 0 \text{ in } e} V^{(e)} \leq \alpha V \tag{18.2}$$

The compliance is measured by the work of the applied loading, $\mathbf{F}^T\mathbf{U}$. The single explicit constraint is that we can only use αV of the volume. The volume actually used is computed as a sum over all the elements with nonzero stiffness. The vector $\mathbf{F}$ represents the applied loading, and the vector of the displacements $\mathbf{U}$ is the result of the solution of the equilibrium equations $\mathbf{K}(E)\mathbf{U} = \mathbf{F}$, which may therefore be considered a form of constraint. We indicate by $\mathbf{K}(E)$ that the stiffness of the structure depends on which of the elements are stiff and which are not.

During the optimization process, some finite elements will actually have an intermediate stiffness, somewhere between the full stiffness and the zero stiffness. The algorithm considers this by assigning the elements an imaginary density, and then making the stiffness depend on this density.

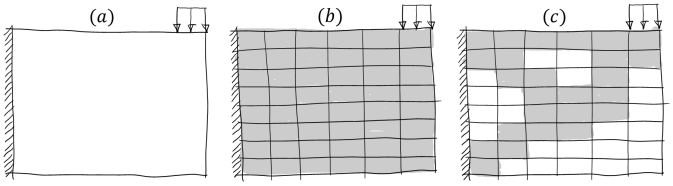


Fig. 18.1. Cantilever – topology optimization. The goal is to distribute the material within the design area (the rectangle) so that (i) we use only half of the available area, and (ii) the structure is as stiff as possible given the limited available material. (a) design area with boundary conditions, (b) the mesh before optimization: all elements are used, (c) the mesh after optimization: only around 50% of elements are used.

The optimization favours then either elements which have zero stiffness or elements which have full stiffness. The SIMP (Simple Isotropic Material With Penalization) approach is to write

$$E(\rho) = \overline{E}\rho^p \quad \text{where } 0 \leq \rho \leq 1 \text{ and } p \geq 3. \tag{18.3}$$

Here $\overline{E}$ is the full stiffness of the original material.

Tosca also provides the so-called sensitivity-based topology optimization, which can accommodate a variety of design responses for the objective function and the constraints. As this usually becomes somewhat involved due to its flexibility, and tends to require substantial knowhow for successful application, it will not be mentioned further in this book.

Do the results depend on the mesh?

The topology optimization is run on a particular mesh: the geometry of the design area is subdivided into finite elements, and the elements are the basis of the decision making of which material to keep and which to give up. Figure 18.2 shows optimization results obtained with a coarse mesh and with a fine mesh. Some features are recognizable in both results, but the finer the mesh, the more detail there will be. However, the optimization may take a long time with a fine mesh. Note the partially filled (gray) elements: the final result will have removed these elements producing a binary, black and white, classification of the finite elements.

Fig. 18.2. Topology optimization. Mesh with 30×30 quadrilateral elements on the left, and mesh with 120×120 elements on the right.

19

Shape Optimization

Summary

1. The principles of the shape optimization are illustrated on an example.
2. The fully-stressed design criterion this explained.
3. The design area, and the smoothing region, are introduced.

We can explain the concepts of shape optimization on an example. Consider a rectangular sheet of aluminum, punctured with a circular hole in the middle (Figure 19.1). A biaxial loading is applied (tensile horizontally and compressive vertically). The circular hole acts as a stress concentrator: the applied loading results in stress distribution in the plate that is strongly magnified relative to the applied tractions near the surface of the hole.

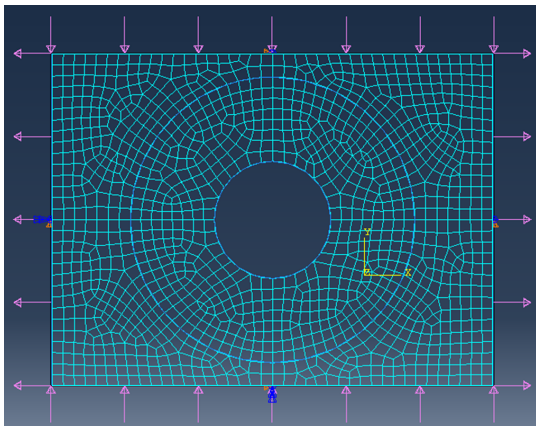


Fig. 19.1. Circular hole under biaxial loading. The horizontal loading is tensile, and it is of double the magnitude of the vertical loading, which is compressive.

The optimization problem can be formulated as: find the locations $\mathbf{x}_{\text{mov}}$ of some "movable" nodes such that the maximum von Mises stress is minimized, subject to the condition that only a certain amount of volume of the original structure V_0 is used.

$$\mathbf{x}_{\text{mov}} = \arg\min \sigma_{\text{vm}} \left(\mathbf{U}(\mathbf{x}_{\text{mov}})\right) \tag{19.1}$$

$$\text{s.t.} \quad \mathbf{K}(\mathbf{x}_{\text{mov}})\mathbf{U} = \mathbf{F} \quad \text{and} \quad V(\mathbf{x}_{\text{mov}}) \leq \alpha V_0 \tag{19.2}$$

The stress in the optimized structure is computed from the displacements obtained from the equilibrium equations, where the stiffness matrix is recomputed to reflect the movement of the nodes. Typically $\alpha = 1$ is used (the amount of material available for the structure is fixed). The structure

before optimization had a circular hole (Figure 19.3(a)), and the maximum stress was ≈ 451 MPa; the structure after optimization has a "rectangular" hole with rounded off corners, and the area of the hole is the same as the area of the circle, meaning the same amount of material is used, while the stress maximum dropped to ≈ 319 MPa (Figure 19.3(b)).

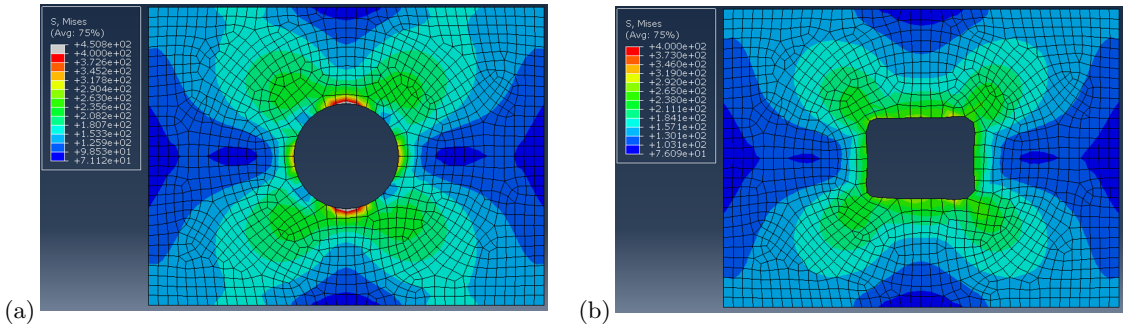


Fig. 19.2. Circular hole under biaxial loading. Von Mises stress before optimization (a), and after optimization (b).

The controller algorithm in Tosca Structure is very effective for this sort of optimization. It can be explained on the basis of "fully stressed design": if some material experiences lower stress than the limit, that material is probably not needed in its entirety (a fraction will do); otherwise, if the material experiences stress over the limit, more material at that location is needed. The algorithm inspects the stress alongside the surface of the structure, which is where structures typically experience the extremes of stress, and applies the fully-stressed criterion. The shape of the hole is changed by moving the nodes on its surface. But moving only these nodes would be very limiting: moving the nodes into the void could lead to very stretched elements, and hence very poor accuracy, while moving the nodes into the material could eventually lead to entanglement of the elements. Therefore, the movement of the nodes is allowed within a certain volume (smoothing volume), not just along the surface. The nodes on the surface dictate the shape, and the nodes in the interior follow along to smooth out the motion of the surface nodes so that the mesh remains reasonably well-shaped. In the present example, the motion of the nodes is allowed within the circular ring around the hole (Figure 19.3). Both objective function and the constraint are also evaluated within this region: this is the design area (as indicated graphically by the markers).

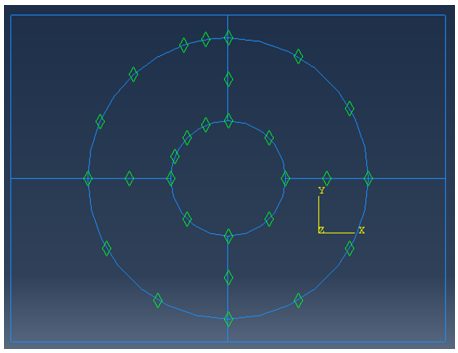


Fig. 19.3. Circular hole under biaxial loading. Design area is the circular ring around the hole. All the nodes within this area are movable. The shape of the hole is changed by moving the nodes on its surface, and the nodes in the interior move also in order to make the mesh well-shaped.

Annotated bibliography

[Barb] Barbero, Ever J. Finite Element Analysis of Composite Materials using Abaqus, CRC Press; 1st edition (June 5, 2013). ucsd.edu permalink
Selection of examples of composite analysis with Abaqus.

[BaCr] Bauchau, O. A. & Craig, J. I. Structural Analysis: With Applications to Aerospace Structures (Springer Science & Business Media, 2009). ucsd.edu permalink
A unified treatment of structural-analysis topics. Exhaustive coverage (943 pages!) with lots of hands-on examples.

[BaCe] Bažant, Z. P., Cedolin, L. (1991). Stability of Structures: Elastic, Inelastic, Fracture and Damage Theories, World Scientific Publishing Co, 57 Shelton Street, London, WC2H 9HE, UK. 2010. 1011pp. Illustrated. ISBN 978-981-4317-03-0.
Insightful and comprehensive coverage of instability; engineering theories of all sorts of failure, including buckling. Includes hundreds of solved exercises. One of the best books on this topic so far.

[Fle] R. Fletcher, Practical methods of optimization, second edition, John Wiley and sons, 2000.
Lucid presentation of the basics of unconstrained optimization.

[Hart] Almost a Tragedy: The Collapse of the Hartford Civic Center, downloaded from.

[PKJS] Petr Krysl and Jiun-Shyan Chen. Benchmarking computational shell models. *Archives of Computational Methods in Engineering*, 30(1):301–315, 2023. Full text.
Proposed suite of benchmarks for linear shell analysis. Discussion of the various aspects of verification of solutions, including Richardson extrapolation.

[PaWi] P.Y. Papalambros, D. J. Wilde, Principles of optimal design, second edition, Cambridge University press, 2000.
Useful and practical discussion of constrained optimization.

[Reti] Ladislao Reti. *The Unknown Leonardo*. Abradale S. Harry N. Abrams, New York, NY, July 1990.
Beautifully illustrated story of Leonardo da Vinci's inventions.

[RoSc] Rosen, A. and Schmit, L.A., Jr. (1979), Design-oriented analysis of imperfect truss structures—part I—accurate analysis. Int. J. Numer. Meth. Engng., 14: 1309-1321.
https://doi.org/10.1002/nme.1620140905

[Stra] G. Strang, Linear Algebra and Its Applications, Brooks Cole; 4th edition, 2005.
Great reference for the least-squares methodology.

[Vand] G.N. Vanderplaats, Numerical optimization techniques for engineering design: with applications, McGraw-Hill Series in Mechanical Engineering, McGraw-Hill Ryerson, Limited, 1984.
A gem of a book for the practicing engineer. It has aged, but gracefully. Well worth reading.

[ViSi] Vinson, J. R. and Sierakowski, R. L. The Behavior of Structures Composed of Composite Materials. Martinus Nijhoff, The Netherlands, (1987).
A go-to reference for composite materials.

[ZTF] O C Zienkiewicz, R L Taylor, and David Fox. General problems in solid mechanics and nonlinearity. In *The Finite Element Method for Solid and Structural Mechanics*, pages 297–392.

Elsevier, 2014.
Great reference for all aspects of FEM. A thorough treatment of linear shell theory. Not an easy read, though.