Calculus of Variations and Finite Elements

Andreas Stahel

April 22, 2021

Contents

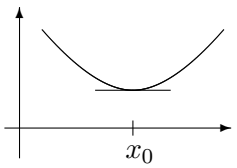
Chapter 1

Extrema of functions of one or multiple variables

This chapter serves as a short repetition of results about functions in one and multiple variables. We will present some facts about extrema of functions.

1.1 Necessary condition for an extremum

As a result from the calculus in one variable we know that if a smooth function $f : \mathbb{R} \to \mathbb{R}$ it to attain a maximal or minimal value at $x = x_0$ then we need $f'(x_0) = 0$. This result implies that the tangent line at an extremal value has to be horizontal. Please observe that this is only a necessary condition, but not a sufficient condition.



This fundamental result also applies to functions of multiple variables. If a function $f : \mathbb{R}^n \to \mathbb{R}$ of multiple variables x_i $(i = 1, 2, \ldots, n)$ attains an extremum at a point $\vec{x}_c \in \mathbb{R}^n$, then the partial derivatives with respect to all variable have to vanish. Thus the gradient has to be the zero vector, in short

$$f(\vec{x}) \text{ extremal at } \vec{x} = \vec{x}_c \qquad \Longrightarrow \qquad \vec{\nabla} f(\vec{x}_c) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \vec{0}$$

If the gradient vanishes at $\vec{x}_c$ then the point is called a **critical point**.

1–1 Example : Consider the function

$$\begin{aligned} f(x,y) &= 2\,x^2 + 2\,x\,y + 2\,y^2 - \frac{5}{2}\,x - \frac{5}{2}\,y + 1.1 \\ &= \frac{1}{2}\,\langle \begin{pmatrix} x \\ y \end{pmatrix}, \begin{bmatrix} 4 & 2 \\ 2 & 4 \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \rangle + \langle \begin{pmatrix} x \\ y \end{pmatrix}, \begin{pmatrix} -5/2 \\ -5/2 \end{pmatrix} \rangle + 1.1 \end{aligned}$$

The corresponding surface and level curves are shown in Figure 1.1. The minimum of this function is characterized by vanishing derivatives in both coordinate directions.

$$\begin{aligned} \frac{\partial\, f}{dx} &= 4\,x + 2\,y - \frac{5}{2} \\ \frac{\partial\, f}{dx} &= 2\,x + 4\,y - \frac{5}{2} \end{aligned}$$

This can also be written in the form

$$\begin{bmatrix} 4 & 2 \\ 2 & 4 \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} -5/2 \\ -5/2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

and the solution is given by $(x, y) = (5/12\,,\, 5/12)$.

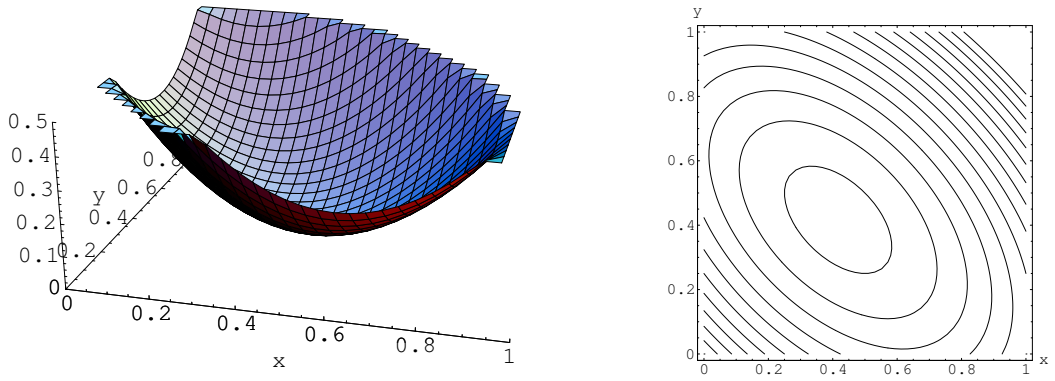


Figure 1.1: The surface of a quadratic function and its level curves

The eigenvalues and eigenvectors of the matrix are given by

$$\lambda_1 = 2 \quad , \quad \vec{v}_1 = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad \text{and} \quad \lambda_1 = 6 \quad , \quad \vec{v}_1 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Thus the function $f(x, y)$ behaves like a parabola $g(t) = t^2$ in the direction $(1\,,\, -1)^T$ and like a parabola $g(t) = 3\,t^2$ in the direction $(1\,,\, 1)$. This is confirmed by Figure 1.1. ♢

1–2 Example : The smooth function

$$f\,(\vec{x}) = f\,(x_1, x_2, x_3) = x_1^2 + 3\,x_2^2 + 5\,x_3^2 + 2\,x_1\,x_2 - 5\,x_2\,x_3 - 2\,x_1 - 3\,x_3 + 17$$

depends on three variables. To find a possible extremum we have the three necessary conditions

$$\begin{array}{lllll} \frac{\partial f}{\partial x_1} = 2\,x_1 & +2\,x_2 & & -2 = & 0 \\ \frac{\partial f}{\partial x_2} = 2\,x_1 & +6\,x_2 & -5\,x_3 & = & 0 \\ \frac{\partial f}{\partial x_3} = & -5\,x_2 & +10\,x_3 & -3 = & 0 \end{array}$$

Since the original function is a polynomial of degree 2, the partial derivatives are linear functions and we have a particularly simple situation. This system of linear equation can be written as

$$\begin{bmatrix} 2 & 2 & 0 \\ 2 & 6 & -5 \\ 0 & -5 & 10 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 2 \\ 0 \\ 3 \end{pmatrix}$$

and with some calculations we arrive at the unique solution

$$x_1 = \frac{4}{3} \quad , \quad x_1 = \frac{-1}{3} \quad , \quad x_1 = \frac{2}{15}$$

Thus there is exactly one critical point. If the function has a minimum, then we found its location. Later we will see that this function has in fact a minimum. ♢

Now we consider a special type of function, which turns out to be very important in the context of the Finite Element Method. We start with a function of three variables.

1–3 Example : Let $\mathbf{A}$ be a symmetric 3×3 matrix

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \quad \text{with} \quad \begin{aligned} a_{12} &= a_{21} \\ a_{13} &= a_{31} \\ a_{23} &= a_{32} \end{aligned}$$

Now consider the function

$$f(\vec{x}) = \frac{1}{2}\langle \vec{x}\,,\, \mathbf{A}\,\vec{x}\rangle + \langle \vec{x}\,,\, \vec{b}\rangle$$

where $\vec{b} \in \mathbb{R}^3$ is given. Writing all the components leads to the expression below. All computations are elementary but tedious. The last transformation is valid, since the matrix $\mathbf{A}$ is symmetric.

$$\begin{aligned} f(\vec{x}) &= \frac{1}{2}\langle \vec{x}\,,\, \mathbf{A}\,\vec{x}\rangle + \langle \vec{x}\,,\, \vec{b}\rangle \\ &= \frac{1}{2}\left\langle \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}, \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \right\rangle + \left\langle \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}, \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \right\rangle \\ &= \frac{1}{2}(x_1, x_2, x_3) \cdot \begin{pmatrix} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 \end{pmatrix} + (x_1, x_2, x_3) \cdot \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \\ &= \frac{1}{2}\big(a_{11}x_1^2 + a_{12}x_1x_2 + a_{13}x_1x_3 \;+\; a_{21}x_1x_2 + a_{22}x_2^2 + a_{23}x_2x_3 \\ &\qquad + a_{31}x_1x_3 + a_{32}x_2x_3 + a_{33}x_3^2\big) + x_1\,b_1 + x_2\,b_2 + x_3\,b_3 \\ &= \frac{1}{2}\big(a_{11}x_1^2 + a_{22}x_2^2 + a_{33}x_3^2\big) + a_{21}x_1x_2 + a_{13}x_1x_3 + a_{23}x_2x_3 + x_1\,b_1 + x_2\,b_2 + x_3\,b_3 \end{aligned}$$

Now it is rather simple to find the three components of the gradient.

$$\begin{aligned} \frac{\partial f}{\partial x_1} &= a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + b_1 \\ \frac{\partial f}{\partial x_2} &= a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + b_2 \\ \frac{\partial f}{\partial x_3} &= a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + b_3 \end{aligned}$$

Thus the point $\vec{x} \in \mathbb{R}^3$ is a critical point of the function f if and only if

$$\vec{\nabla} f(\vec{x}) = \mathbf{A}\,\vec{x} + \vec{b} = \vec{0}$$

If the matrix $\mathbf{A}$ is invertible there is exactly one critical point, given by

$$\vec{x}_c = -\mathbf{A}^{-1}\,\vec{b}$$

Thus for functions of three variables of the above type finding the critical point is equivalent to solving a linear system of equations. ♢

The observation in the above example can be generalised to functions of n variables and we arrive at the following result. The proof of this theorem is entirely based on the idea of the previous example.

1–4 Theorem : *Let* $\mathbf{A}$ *be a symmetric, real* $n \times n$ *matrix and* $\vec{b} \in \mathbb{R}^n$. *Consider the function*

$$f(\vec{x}) = \frac{1}{2}\langle \vec{x}\,,\, \mathbf{A}\,\vec{x}\rangle + \langle \vec{x}\,,\, \vec{b}\rangle$$

Then we have

$$\vec{x}_c \text{ is a critical point of } f \qquad \Longleftrightarrow \qquad \mathbf{A}\,\vec{x}_c + \vec{b} = \vec{0}$$

◇

The i-th component of the equation $\mathbf{A}\,\vec{x}_c + \vec{b} = \vec{0}$ results from the derivative of $f(\vec{x})$ with respect to x_i to be equals zero. A more detailed computation is shown in Exercise 1–3 .

Many finite element problems will lead to functions f of the above type to be minimised, i.e. linear systems to be solved.

1.2 Sufficient conditions for minima

For a function f of one variable and $f'(x_0) = 0$ then the additional condition $f''(x_0) > 0$ is sufficient to determine a local minimum. The situation for functions of multiple variables is not quite as simple. The idea is again to use Taylor's approximation. We consider the function f for arguments close to the critical point $\vec{x}_c$, the deviation from $\vec{x}_c$ is denoted by $\vec{x}$. We have

$$f(\vec{x}_c + \vec{x}) = f(\vec{x}_c) + \operatorname{grad} f(\vec{x}_c)\cdot\vec{x} + \frac{1}{2}\,\vec{x}^T\cdot H_f\cdot\vec{x} + O(\|\vec{x}\|^3)$$

where H_f is the **Hessian** matrix of second derivatives.

$$H_f = \begin{bmatrix} \frac{\partial^2 f(\vec{x}_c)}{\partial^2 x_1 x_1} & \frac{\partial^2 f(\vec{x}_c)}{\partial x_1 x_2} & \cdots & \frac{\partial^2 f(\vec{x}_c)}{\partial x_1 x_n} \\ \frac{\partial^2 f(\vec{x}_c)}{\partial^2 x_2 x_1} & \frac{\partial^2 f(\vec{x}_c)}{\partial x_2 x_2} & \cdots & \frac{\partial^2 f(\vec{x}_c)}{\partial x_2 x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(\vec{x}_c)}{\partial^2 x_n x_1} & \frac{\partial^2 f(\vec{x}_c)}{\partial x_n x_2} & \cdots & \frac{\partial^2 f(\vec{x}_c)}{\partial x_n x_n} \end{bmatrix}$$

This matrix has to be symmetric. Now we use that $\vec{x}_c$ is a critical point and thus

$$f(\vec{x}_c + \vec{x}) \approx f(\vec{x}_c) + \frac{1}{2}\,\langle \vec{x}^T\,,\, H_f\cdot\vec{x}\rangle$$

By moving the coordinate origin to the critical point and ignoring the fixed value $f(\vec{x}_c)$ we have to examine functions

$$f(\vec{x}) = \frac{1}{2}\,\langle \vec{x}, \mathbf{A}\vec{x}\rangle$$

for symmetric matrices $\mathbf{A}$. Now we examine the special situation of an eigenvector $\vec{e}_k$ with eigenvalue λ_k. It is know that all eigenvalues are real and the corresponding eigenvectors can be chosen to have length 1.

$$\begin{aligned} f(\vec{e}_k) &= \frac{1}{2}\,\langle \vec{e}_k, \mathbf{A}\vec{e}_k\rangle = \frac{1}{2}\,\langle \vec{e}_k, \lambda_k\vec{e}_k\rangle = \frac{1}{2}\,\lambda_k\,\|\vec{e}_k\| = \frac{1}{2}\,\lambda_k \\ f(t\vec{e}_k) &= \frac{1}{2}\,\langle t\vec{e}_k\,,\, t\,\mathbf{A}\vec{e}_k\rangle = \frac{1}{2}\,t^2\,\lambda_k \end{aligned}$$

Thus the eigenvalues λ_k are important.

- If $\lambda_k > 0$ then the function f grows in the corresponding direction (given by $\vec{e}_k$) like the function $h(t) = \frac{1}{2}\,\lambda_k\,t^2 \geq 0$.
- If $\lambda_k < 0$ then the function f decays in the corresponding direction (given by $\vec{e}_k$) like the function $h(t) = \frac{1}{2}\,\lambda_k\,t^2 \leq 0$.

Such a matrix can be diagonalised (see Appendix A.1.3), i.e. written in the form

$$\mathbf{A} = \mathbf{R}\cdot\mathbf{D}\cdot\mathbf{R}^T \quad \text{or equivalently} \quad \mathbf{D} = \mathbf{R}^T\cdot\mathbf{A}\cdot\mathbf{R}$$

where the diagonal matrix $\mathbf{D}$ has the eigenvalues λ_i of $\mathbf{A}$ as entries along the diagonal. With the substitution $\vec{y} = \mathbf{R}^T\vec{x}$ (and thus $\vec{x} = \mathbf{R}\,\vec{y}$) we have

$$\begin{aligned} f(\vec{x}) &= \frac{1}{2}\,\langle\vec{x}, \mathbf{A}\vec{x}\rangle = \frac{1}{2}\,\langle\mathbf{R}\vec{y}, \mathbf{A}\mathbf{R}\vec{y}\rangle \\ &= \frac{1}{2}\,\langle\vec{y}, \mathbf{R}^T\mathbf{A}\mathbf{R}\vec{y}\rangle = \frac{1}{2}\,\langle\vec{y}, \mathbf{D}\vec{y}\rangle \\ &= \frac{1}{2}\sum_{k=1}^{n}\lambda_k\,y_k^2 \end{aligned}$$

Multiplying the matrix $\mathbf{R}$ by the column $\vec{y}$ from the right corresponds to a linear combination of the columns of $\mathbf{R}$, where the factors are given by the entries in $\vec{y}$. Since the columns of $\mathbf{R}$ contain the normalised eigenvectors of $\mathbf{A}$ we have

$$\vec{x} = [\vec{e}_1, \vec{e}_2, \ldots, \vec{e}_n]\cdot\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = y_1\,\vec{e}_1 + y_2\,\vec{e}_2 + \ldots + y_n\vec{e}_n$$

Thus the substitution $\vec{y} = \mathbf{R}\,\vec{x}$ corresponds to a rewriting $\vec{x}$ as a linear combination of the eigenvectors $\vec{e}_k$.

From this we can draw some conclusion on the behaviour of the function $f(\vec{x})$. We examine this function along straight line through the origin in the direction of an eigenvector by using the parametrisation

$$\vec{x}(t) = t\,\vec{e}_k \quad \text{for} \quad t \in \mathbb{R}$$

then we have

$$f(\vec{x}(t)) = \frac{1}{2}\,\lambda_k\,t^2 \quad \text{for} \quad t \in \mathbb{R}$$

The restricted function behaves like a parabola, the sign of the eigenvalue λ_k determines whether the function has a local minimum or maximum for $t = 0$. This motivates the following result.

1–5 Result : *Consider a function*

$$f(\vec{x}) = \frac{1}{2}\,\langle\vec{x}, \mathbf{A}\vec{x}\rangle$$

for symmetric matrices $\mathbf{A}$ *with eigenvalues* λ_k.

- *If all eigenvalues are strictly positive then the function has a local minimum at* $\vec{x}_c = \vec{0}$.
- *If all eigenvalues are strictly negative then the function has a local maximum at* $\vec{x}_c = \vec{0}$.
- *If some of the eigenvalues are positive and some negative then the critical point* $\vec{x}_c = \vec{0}$ *is a saddle point.*

◇

If all eigenvalues are positive and we use the substitution $\vec{y} = \mathbf{R}^T \vec{x}$ then we have

$$f(\vec{x}) = \frac{1}{2} \langle A\vec{x}, \vec{x} \rangle = \frac{1}{2} \sum_{k=1}^{n} \lambda_k \, y_k^2 > 0 \quad \text{if} \quad \vec{x} \neq \vec{0}$$

This leads to a definition.

1–6 Definition : A symmetric, real matrix **A** is called **positive definite** if and only if

$$\langle \mathbf{A} \cdot \vec{x} \, , \, \vec{x} \rangle = \langle \vec{x} \, , \, \mathbf{A} \cdot \vec{x} \rangle > 0 \quad \text{for all} \quad \vec{x} \neq \vec{0}$$

It is obvious that a positive definite matrix **A** implies that the function $f(\vec{x})$ examined above will have a minimal value of 0 at $\vec{x} = \vec{0}$. Functions of the form

$$f(\vec{x}) = \frac{1}{2} \langle \vec{x} \, , \, \mathbf{A} \, \vec{x} \rangle + \langle \vec{x} \, , \, \vec{b} \rangle$$

will have a unique minimum. Matrices of this type occur very often in finite element problems and they will be examined more carefully in chapter 11.

1.3 Exercises

• Exercise 1–1:
Consider the function

$$f(x_1, x_2) = x_1^2 + 4\, x_1\, x_2 - 2\, x_2^2 + 3\, x_1 + 6\, x_2$$

(a) Write the function in the form

$$f(\vec{x}) = \frac{1}{2} \langle \vec{x} \, , \, \mathbf{A} \, \vec{x} \rangle + \langle \vec{x} \, , \, \vec{b} \rangle$$

(b) Find the critical point of this function by solving a linear system of equations.

(c) Use the eigenvalues and eigenvectors of the Matrix **A** to describe the type of critical point.

(d) Use appropriate software to examine the graph of this function and verify your results.

• Exercise 1–2:
Examine the function

$$f(x, y) = \frac{1}{2} \langle \begin{pmatrix} x \\ y \end{pmatrix} , \begin{bmatrix} 8 & -2 \\ -2 & 6 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} \rangle + \langle \begin{pmatrix} x \\ y \end{pmatrix} , \begin{pmatrix} b \\ -2\, b \end{pmatrix} \rangle$$

For each value of the parameter $b \in \mathbb{R}$ this function has a unique minimum. Determine the location of this minimum as a function of b.

You can use the code below to illustrate that the graph of the function does not change its shape as b varies, but the location of the minimum is moving. Verify that the minimum moves along a straight line.

Mathematica

```
f[x_,y_,b_]= 4x^2 + 3*y^2 - 2*x*y + b*x - 2*b*y;
limit=3;
makeplot[b_] :=
   Plot3D[f[x,y,b],{x,-limit,limit},{y,-5,5},
         PlotRange->{{-limit,limit},{-limit,limit},{-8,15}},
         PlotPoints -> 30,
         ClipFill ->None];
Animation[Table[makeplot[b],{b,-5,5,0.25}]];
```

• Exercise 1–3:

To compute the partial derivative with respect to u_p we write the expression to be examined as a sum of terms involving u_p and a remainder R, which is independent on u_p

$$\begin{aligned} f(\vec{u}) &= \frac{1}{2}\,\langle \vec{u}\,,\, \mathbf{A}\,\vec{u}\rangle + \langle \vec{u}\,,\, \vec{b}\rangle \\ &= \frac{1}{2}\sum_{i=1}^{n} u_i \left(\sum_{j=1}^{n} a_{i,j}\,u_j\right) + \sum_{i=1}^{n} b_i\,u_i \\ &= \frac{1}{2}\,u_p\,a_{p,p}\,u_p + \frac{1}{2}\sum_{j=1,j\neq p}^{n} u_p\,a_{p,j}\,u_j + \frac{1}{2}\sum_{j=1,j\neq p}^{n} u_j\,a_{j,p}\,u_p + b_p\,u_p + R \end{aligned}$$

This leads to

$$\begin{aligned} \frac{\partial}{\partial u_p}\,f(\vec{u}) &= a_{p,p}\,u_p + \frac{1}{2}\sum_{j=1,j\neq p}^{n} a_{p,j}\,u_j + \frac{1}{2}\sum_{j=1,j\neq p}^{n} u_j\,a_{j,p} + b_p + 0 \\ &= \sum_{j=1,}^{n} a_{p,j}\,u_j + b_p \end{aligned}$$

where we used the symmetry $a_{j,p} = a_{p,j}$. This equals row p of the expression $\mathbf{A}\,\vec{u} + \vec{b}$.

Chapter 2

Two introductionary problems

In this chapter we consider two elementary elasticity problems and determine a numerical approximation to their solution. Many of the ideas presented will carry over to a finite element approach for general problems.

2.1 Finite element solution for a system of trusses

2.1.1 Description of the situation

In Figure 2.1 find a simple system of five trusses, named (a, b, c, d, e). The trusses support only loads in the direction of the trusses, i.e. no bending moments are allowed. They are flexibly connected at four points. All trusses have length L, cross section A and are made of a material with modulus of elasticity E. The point 2 is fixed and point 1 can only move in the x direction. An external load of 100 N is applied to point 4 in vertical direction.

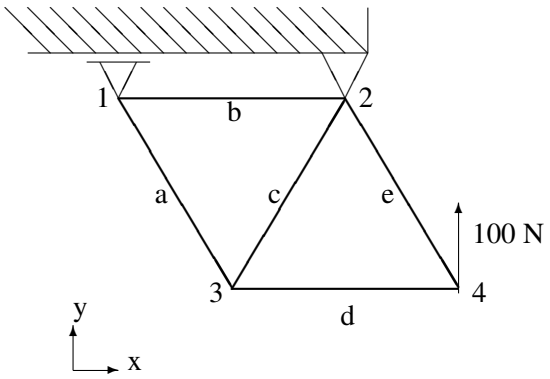


Figure 2.1: Simple system of trusses

2.1.2 Element stiffness matrix

Consider an element e, connecting points i and n, forming an angle α with the vertical axis. Figure 2.2 shows the typical situation.

The displacements of the points i and n out of the initial positions are given by the vectors

$$\begin{pmatrix} u_i \\ v_i \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} u_n \\ v_n \end{pmatrix}$$

Now we express the forces $\vec{F}_{e,i}$ and $\vec{F}_{e,n}$ at both ends in terms of the displacement vectors. **Here we consider the forces applied to the truss, and not the forces the truss is applying to the nodes**. The difference is a sign. In the complete structure these forces have to be produced by the other trusses, the supports or the applied external forces. The observations in section 2.1.3 show that the forces due to displacement in one element are computed by multiplying the vector of displacements $\vec{a}_e$ by the **element stiffness matrix** $\mathbf{K}_e$.

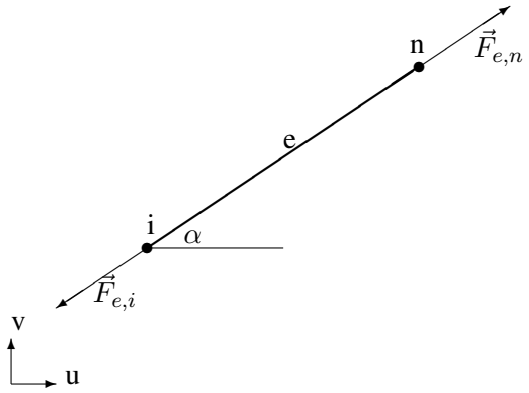


Figure 2.2: An isolated element e of the structure with the forces applied to it

$$\vec{F}_e = \begin{pmatrix} \vec{F}_{e,i} \\ \vec{F}_{e,n} \end{pmatrix} = \begin{pmatrix} U_i \\ V_i \\ U_n \\ V_n \end{pmatrix} = \tag{2.1}$$

$$= \frac{E\,A}{L} \begin{bmatrix} \cos^2\alpha & \sin\alpha\cos\alpha & -\cos^2\alpha & -\sin\alpha\cos\alpha \\ \sin\alpha\cos\alpha & \sin^2\alpha & -\sin\alpha\cos\alpha & -\sin^2\alpha \\ -\cos^2\alpha & -\sin\alpha\cos\alpha & \cos^2\alpha & \sin\alpha\cos\alpha \\ -\sin\alpha\cos\alpha & -\sin^2\alpha & \sin\alpha\cos\alpha & \sin^2\alpha \end{bmatrix} \begin{pmatrix} u_i \\ v_i \\ u_n \\ v_n \end{pmatrix}$$

A short notation for this is

$$\vec{F}_e = \mathbf{K_e}\ \vec{a}_e$$

If the displacements $\vec{a}_e$ are small then the angle α will change only very little. We ignore those changes. If the displacements would be large then the change of the angle would have to be taken into account by recomputing the element stiffness matrices. We want to ignore this problem and thus work under the general assumption:

> The displacements of the nodes are small compared to the dimensions of the trusses. We consider the angles as constant.

2.1.3 Derivation of the element stiffness matrix

We give one of the possible derivations for these matrices. The calculations are based on matrix operations representing rotations in the plane and Hooke's law. One may safely skip this section and accept equation (2.1) as a fact.

Consider a truss of length L, area of cross section A of a material with modulus of elasticity E. If this truss is stretched by length ΔL then the elastic force is given by

$$F = \frac{E\,A}{L}\,\Delta L$$

This is **Hooke's law**. For a horizontal truss we find $|\Delta L| = |u_i - u_n|$ and thus

$$\begin{aligned} U_i &= \frac{E\,A}{L}\,(u_i - u_n) \\ V_i &= 0 \\ U_n &= \frac{E\,A}{L}\,(-u_i + u_n) \\ V_n &= 0 \end{aligned}$$

This can also be rewritten with the help of a 4×4 matrix by

$$\begin{pmatrix} U_i \\ V_i \\ U_n \\ V_n \end{pmatrix} = \frac{E\,A}{L} \begin{pmatrix} u_i - u_n \\ 0 \\ -u_i + u_n \\ 0 \end{pmatrix} = \frac{E\,A}{L} \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{pmatrix} u_i \\ v_i \\ u_n \\ v_n \end{pmatrix}$$

This is the element stiffness matrix in equation (2.1) for the special case of $\alpha = 0$. We base the general situation on this formula and rotations in the plane. With similar calculation one can compute the elastic energy U stored in the truss, due to the deformation[1].

When rotating a vector $(u\,,\,v)^T$ by an angle α (in the positive sense) we obtain the new vector

$$\begin{pmatrix} \cos\alpha\,u - \sin\alpha\,v \\ \sin\alpha\,u + \cos\alpha\,v \end{pmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \cdot \begin{pmatrix} u \\ v \end{pmatrix}$$

Rotating by the opposite angle $-\alpha$ leads to the inverse matrix

$$\begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}^{-1} = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix}$$

If a truss has angle α then we first rotate the displacement vectors $(u_i\,,\,v_i)^T$ and $(u_n\,,\,v_n)^T$ by $-\alpha$ and obtain a truss in horizontal position. The forces $\vec{F}_{e,i}$ and $\vec{F}_{e,n}$ have to be rotated by the same angle $-\alpha$. This

[1] As the change of length s varies from 0 to ΔL the force is given by $F(s) = \frac{E\,A}{L}\,s$ and thus the work needed to stretch the truss by ΔL is given by

$$U = \int_0^{\Delta L} \frac{E\,A}{L}\,s\,ds = \frac{E\,A}{2\,L}\,(\Delta L)^2$$

Since $(\Delta L)^2 = (u_i - u_n)^2$ this can be rewritten as

$$U = \frac{1}{2}\,\langle \begin{pmatrix} u_i \\ v_i \\ u_n \\ v_n \end{pmatrix}\,,\,\frac{E\,A}{L} \begin{pmatrix} u_i - u_n \\ 0 \\ -u_i + u_n \\ 0 \end{pmatrix} \rangle = \frac{1}{2}\,\langle \begin{pmatrix} u_i \\ v_i \\ u_n \\ v_n \end{pmatrix}\,,\,\frac{E\,A}{L} \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{pmatrix} u_i \\ v_i \\ u_n \\ v_n \end{pmatrix} \rangle$$

leads to the condition

$$\begin{bmatrix} \cos\alpha & \sin\alpha & 0 & 0 \\ -\sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & \cos\alpha & \sin\alpha \\ 0 & 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \cdot \begin{pmatrix} U_i \\ V_i \\ U_n \\ V_n \end{pmatrix} =$$

$$= \frac{E\,A}{L} \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \cos\alpha & \sin\alpha & 0 & 0 \\ -\sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & \cos\alpha & \sin\alpha \\ 0 & 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \cdot \begin{pmatrix} u_i \\ v_i \\ u_n \\ v_n \end{pmatrix}$$

The element stiffness matrix is given by

$$\mathbf{K_e} = \frac{E\,A}{L} \begin{bmatrix} \cos\alpha & \sin\alpha & 0 & 0 \\ -\sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & \cos\alpha & \sin\alpha \\ 0 & 0 & -\sin\alpha & \cos\alpha \end{bmatrix}^{-1} \cdot$$

$$\begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \cos\alpha & \sin\alpha & 0 & 0 \\ -\sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & \cos\alpha & \sin\alpha \\ 0 & 0 & -\sin\alpha & \cos\alpha \end{bmatrix}$$

Carrying out the multiplications we arrive at

$$\mathbf{K_e} = \frac{E\,A}{L} \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 & 0 \\ \sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & \cos\alpha & -\sin\alpha \\ 0 & 0 & \sin\alpha & \cos\alpha \end{bmatrix} \cdot \begin{bmatrix} \cos\alpha & \sin\alpha & -\cos\alpha & -\sin\alpha \\ 0 & 0 & 0 & 0 \\ -\cos\alpha & -\sin\alpha & \cos\alpha & \sin\alpha \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

and thus

$$\mathbf{K_e} = \frac{E\,A}{L} \begin{bmatrix} \cos^2\alpha & \sin\alpha\cos\alpha & -\cos^2\alpha & -\sin\alpha\cos\alpha \\ \sin\alpha\cos\alpha & \sin^2\alpha & -\sin\alpha\cos\alpha & -\sin^2\alpha \\ -\cos^2\alpha & -\sin\alpha\cos\alpha & \cos^2\alpha & \sin\alpha\cos\alpha \\ -\sin\alpha\cos\alpha & -\sin^2\alpha & \sin\alpha\cos\alpha & \sin^2\alpha \end{bmatrix}$$

The matrix

$$\mathbf{T}(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \cdot \begin{bmatrix} \cos\alpha & \sin\alpha \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \cos^2\alpha & \sin\alpha\cos\alpha \\ \sin\alpha\cos\alpha & \sin^2\alpha \end{bmatrix}$$

is called a **transformation matrix** and we can write

$$\mathbf{K_e} = \frac{E\,A}{L} \begin{bmatrix} \mathbf{T}(\alpha) & -\mathbf{T}(\alpha) \\ -\mathbf{T}(\alpha) & \mathbf{T}(\alpha) \end{bmatrix}$$

The internal elastic energy of a truss at angle α can be computed by

$$U_e(\alpha) = \frac{1}{2} \langle \vec{a}_e \, , \, \mathbf{K_e} \cdot \vec{a}_e \rangle \tag{2.2}$$

The computations are similar to the above arguments[2].

2.1.4 Explicit calculations for all five element stiffness matrices

The results of the previous section will be carried out for all five trusses of the structure in Figure 2.1.

The truss connecting points 1 and 3

For this element we find the angle $\alpha = -60^\circ = -\pi/3$ and thus

$$\cos\alpha = \frac{1}{2} = 0.5 \quad \text{and} \quad \sin\alpha = -\frac{\sqrt{3}}{2} \approx -0.866$$

The transformation matrix is

$$\mathbf{T}\,(-\frac{\pi}{3}) = \begin{bmatrix} 0.25 & -0.433 \\ -0.433 & 0.75 \end{bmatrix}$$

and thus

$$\mathbf{K_a} = \frac{E\ A}{L} \begin{bmatrix} 0.25 & -0.433 & -0.25 & 0.433 \\ -0.433 & 0.75 & 0.433 & -0.75 \\ -0.25 & 0.433 & 0.25 & -0.433 \\ 0.433 & -0.75 & -0.433 & 0.75 \end{bmatrix}$$

The truss connecting points 1 and 2

For this element we find the angle $\alpha = 0$ and thus

$$\cos\alpha = 1 \quad \text{and} \quad \sin\alpha = 0$$

The transformation matrix is

$$\mathbf{T}\,(-\frac{\pi}{3}) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

and thus

$$\mathbf{K_b} = \frac{E\ A}{L} \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

---

[2]

$$U_e(\alpha) = \frac{1}{2} \langle \begin{bmatrix} \cos\alpha & \sin\alpha & 0 & 0 \\ -\sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & \cos\alpha & \sin\alpha \\ 0 & 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \cdot \begin{pmatrix} u_i \\ v_i \\ u_n \\ v_n \end{pmatrix} ,$$

$$\frac{E\ A}{L} \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} \cos\alpha & \sin\alpha & 0 & 0 \\ -\sin\alpha & \cos\alpha & 0 & 0 \\ 0 & 0 & \cos\alpha & \sin\alpha \\ 0 & 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \cdot \begin{pmatrix} u_i \\ v_i \\ u_n \\ v_n \end{pmatrix} \rangle = \frac{1}{2} \langle \vec{a}_e \, , \, \mathbf{K_e} \cdot \vec{a}_e \rangle$$

The truss connecting points 2 and 3

For this element we find the angle $\alpha = -120^\circ = -2\pi/3$ and thus

$$\cos\alpha = -\frac{1}{2} = -0.5 \quad \text{and} \quad \sin\alpha = -\frac{\sqrt{3}}{2} \approx -0.866$$

The transformation matrix is

$$\mathbf{T}\,(-\frac{2\pi}{3}) = \begin{bmatrix} 0.25 & 0.433 \\ 0.433 & 0.75 \end{bmatrix}$$

and thus

$$\mathbf{K_c} = \frac{E\ A}{L} \begin{bmatrix} 0.25 & 0.433 & -0.25 & -0.433 \\ 0.433 & 0.75 & -0.433 & -0.75 \\ -0.25 & -0.433 & 0.25 & 0.433 \\ -0.433 & -0.75 & 0.433 & 0.75 \end{bmatrix}$$

The truss connecting points 3 and 4

For this element we find the angle $\alpha = 0$ and thus

$$\cos\alpha = 1 \quad \text{and} \quad \sin\alpha = 0$$

The transformation matrix is

$$\mathbf{T}\,(0) = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}$$

and thus

$$\mathbf{K_d} = \frac{E\ A}{L} \begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The truss connecting points 2 and 4

For this element we find the angle $\alpha = -60^\circ = -\pi/3$ and thus

$$\cos\alpha = \frac{1}{2} = 0.5 \quad \text{and} \quad \sin\alpha = -\frac{\sqrt{3}}{2} \approx -0.866$$

The transformation matrix is

$$\mathbf{T}\,(-\frac{\pi}{3}) = \begin{bmatrix} 0.25 & -0.433 \\ -0.433 & 0.75 \end{bmatrix}$$

and thus

$$\mathbf{K_e} = \frac{E\ A}{L} \begin{bmatrix} 0.25 & -0.433 & -0.25 & 0.433 \\ -0.433 & 0.75 & 0.433 & -0.75 \\ -0.25 & 0.433 & 0.25 & -0.433 \\ 0.433 & -0.75 & -0.433 & 0.75 \end{bmatrix}$$

2.1.5 The global stiffness matrix

For the sake of slightly simpler notations we use the condition

$$\frac{E\,A}{L} = 1$$

for the next two subsections.

The goal is to combine the five element stiffness matrices of the previous subsection to one single system of equations, representing the fact that at each node we should have a balance of forces. Each of the node has two degrees of freedom and we have (at first) eight unknown displacements. For the moment we ignore the constraints.

We start out with an 8×8 matrix filled with 0 and a vector containing the known and unknown external forces.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{pmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{pmatrix} = \begin{pmatrix} 0 \\ f_{1,y} \\ f_{2,x} \\ f_{2,y} \\ 0 \\ 0 \\ 0 \\ 100 \end{pmatrix}$$

The first row of this system of linear equation will represent that fact that all horizontal forces at point 1 have to add up to zero. The second row corresponds to the vertical forces at point 1. The third and fourth row represent the balance of forces at point 2. The bottom four row contain the equations for the points 3 and 4. The force vector on the RHS contains the known external force of 100 N and the yet unknown support forces at the points 1 and 2. Now we have to include all force contributions by the elements, element by element.

At first we consider the forces generated by the elements a and d. The intermediate result is shown below, dots representing zeros.

$$\begin{bmatrix} 0.2500 & -0.4330 & \cdot & \cdot & -0.2500 & 0.4330 & \cdot & \cdot \\ -0.4330 & 0.7500 & \cdot & \cdot & 0.4330 & -0.7500 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ -0.2500 & 0.4330 & \cdot & \cdot & 1.2500 & -0.4330 & -1.0000 & \cdot \\ 0.4330 & -0.7500 & \cdot & \cdot & -0.4330 & 0.7500 & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & -1.0000 & \cdot & 1.0000 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \cdot \begin{pmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{pmatrix} = \begin{pmatrix} 0 \\ f_{1,y} \\ f_{2,x} \\ f_{2,y} \\ 0 \\ 0 \\ 0 \\ 100 \end{pmatrix}$$

When considering contribution from the elements b, c and e too we obtain finally the system

$$\begin{bmatrix} 1.250 & -0.433 & -1.000 & 0.000 & -0.250 & 0.433 & \cdot & \cdot \\ -0.433 & 0.750 & 0.000 & 0.000 & 0.433 & -0.750 & \cdot & \cdot \\ -1.000 & 0.000 & 1.500 & -0.000 & -0.250 & -0.433 & -0.250 & 0.433 \\ 0.000 & 0.000 & -0.000 & 1.500 & -0.433 & -0.750 & 0.433 & -0.750 \\ -0.250 & 0.433 & -0.250 & -0.433 & 1.500 & -0.000 & -1.000 & 0.000 \\ 0.433 & -0.750 & -0.433 & -0.750 & -0.000 & 1.500 & 0.000 & 0.000 \\ \cdot & \cdot & -0.250 & 0.433 & -1.000 & 0.000 & 1.250 & -0.433 \\ \cdot & \cdot & 0.433 & -0.750 & 0.000 & 0.000 & -0.433 & 0.750 \end{bmatrix} \cdot \begin{pmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{pmatrix} = \begin{pmatrix} 0 \\ f_{1,y} \\ f_{2,x} \\ f_{2,y} \\ 0 \\ 0 \\ 0 \\ 100 \end{pmatrix}$$

With suitable abbreviations this can be rewritten as

$$\mathbf{K} \cdot \vec{a} = \vec{F}_{ext}$$

At first sight we find 8 equations but 11 unknowns, the 8 displacements and the 3 unknown support forces. But we have not yet used the 3 support constraints $v_1 = u_2 = v_2 = 0$. The **global stiffness matrix K** shows a few properties that will be used to solve the system:

- **K** is symmetric
- Some of the entries are 0. Since point 1 has no connection truss with point 4 the 2×2 in the upper right and lower left corner contain 0. For larger structures this will turn out to be important as it leads to sparse or banded matrices, i.e. most of the entries are 0.
- The matrix **K** is **positive definite**, i.e. all eigenvalues are greater or equals 0.

Since for each element (i.e. truss) the internal elastic energy is given by $U_e = \frac{1}{2} \langle \vec{a}_e \, , \, \mathbf{K_e} \cdot \vec{a}_e \rangle$ (see equation (2.2)) is is not to difficult to verify that the internal energy of the system is given by

$$U_{elast} = \frac{1}{2} \langle \vec{a} \, , \, \mathbf{K} \cdot \vec{a} \rangle$$

2.1.6 Using the constraints and solving the system of equations

In Figure 2.1 we find $v_1 = u_2 = v_2 = 0$. Due to this constraint all contribution coming from columns 2, 3 and 4 will vanish. Thus we remove those columns and the variables $v_1 = u_2 = v_2 = 0$ from the system. By removing equations (rows) 2, 3 and 4 we du not yet try to solve for the unknown support forces $f_{1,y}$, $f_{2,x}$ and $f_{2,y}$. With these simplifications we obtain a system of five liner equations for five unknowns

$$\begin{bmatrix} 1.250 & -0.250 & 0.433 & \cdot & \cdot \\ -0.250 & 1.500 & -0.000 & -1.000 & 0.000 \\ 0.433 & -0.000 & 1.500 & 0.000 & 0.000 \\ \cdot & -1.000 & 0.000 & 1.250 & -0.433 \\ \cdot & 0.000 & 0.000 & -0.433 & 0.750 \end{bmatrix} \cdot \begin{pmatrix} u_1 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 100 \end{pmatrix} \tag{2.3}$$

This new, smaller matrix $\hat{\mathbf{K}}$ inherits a few properties of **K**.

- **K** is symmetric
- Some of the entries are 0.
- The matrix **K** is **strictly positive definite**, i.e. all eigenvalues are greater than 0.

This linear system admits a unique solution given by

$$\begin{pmatrix} u_1 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{pmatrix} = \begin{pmatrix} 28.868 \\ 129.904 \\ -8.3333 \\ 187.639 \\ 241.667 \end{pmatrix}$$

and thus we know the displacement vector $\vec{a}$ of the system

$$\vec{a} = \begin{pmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{pmatrix} = \begin{pmatrix} 28.868 \\ 0 \\ 0 \\ 0 \\ 129.904 \\ -8.3333 \\ 187.639 \\ 241.667 \end{pmatrix}$$

Our simplifying assumption $\frac{E\,A}{L} = 1$ is in practical situations certainly false and the result $\vec{a}$ has to be multiplied by the factor $\frac{L}{E\,A}$. If we are to compute the resulting forces the we are short a factor $\frac{E\,A}{L}$ and the forces would have correct values[3].

The external force F at the point cane be represented by an external energy contribution $U_{ext} = -v_4 \cdot F$. Thus we arrive at the total energy

$$\begin{aligned} U &= U_{elast} + U_{ext} \\ &= \frac{1}{2} \left\langle \begin{pmatrix} u_1 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{pmatrix}, \begin{bmatrix} 1.250 & -0.250 & 0.433 & \cdot & \cdot \\ -0.250 & 1.500 & -0.000 & -1.000 & 0.000 \\ 0.433 & -0.000 & 1.500 & 0.000 & 0.000 \\ \cdot & -1.000 & 0.000 & 1.250 & -0.433 \\ \cdot & 0.000 & 0.000 & -0.433 & 0.750 \end{bmatrix} \cdot \begin{pmatrix} u_1 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{pmatrix} \right\rangle + \\ &\quad + \left\langle \begin{pmatrix} u_1 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -100 \end{pmatrix} \right\rangle \end{aligned}$$

Minimizing this total energy of the system leads to a problem of the type in Theorem 1–4 (page 4). The optimal value of the displacements is given by the solutions of equation (2.3). For this example we have thus verified a very important physical **principle of least energy**:

> Finding the displacement vector for a system of truss is equivalent to finding the minimum of the total energy (elastic and external).

[3] In this situation twice ignoring the factor cancels in fact the error

2.1.7 Interpretation of the results

The result for the displacements $\vec{a}$ can now be used in $\mathbf{K}\vec{a} = \vec{F}_{ext}$ to find all external forces

$$\vec{F}_{ext} = \begin{pmatrix} -3.323e-15 \\ 5.000e+01 \\ -1.310e-14 \\ -1.500e+02 \\ 2.842e-14 \\ -2.665e-15 \\ -1.532e-14 \\ 1.000e+02 \end{pmatrix} = \begin{pmatrix} 0 \\ 50 \\ 0 \\ -150 \\ 0 \\ 0 \\ 0 \\ 100 \end{pmatrix}$$

We determined the support forces at the points 1 and 2, e.g. point 1 applies a force of 50 N (upwards) to the structure.

With the help of the element stiffness matrices we can compute forces on each element. As an example consider element c. We find

$$\mathbf{K_c} \begin{pmatrix} u_2 \\ v_2 \\ u_3 \\ v_3 \end{pmatrix} = \vec{F}_c$$

and thus

$$\begin{bmatrix} 0.25 & 0.433 & -0.25 & -0.433 \\ 0.433 & 0.75 & -0.433 & -0.75 \\ -0.25 & -0.433 & 0.25 & 0.433 \\ -0.433 & -0.75 & 0.433 & 0.75 \end{bmatrix} \cdot \begin{pmatrix} 0 \\ 0 \\ 129.904 \\ -8.3333 \end{pmatrix} = \begin{pmatrix} -28.87 \\ -50.00 \\ 28.87 \\ 50.00 \end{pmatrix} = \begin{pmatrix} U_2 \\ V_2 \\ U_3 \\ V_3 \end{pmatrix}$$

Thus this element is compressed by a total force with strength $\sqrt{28.87^2 + 50^2}\, N = 57.74\, N$.

2.1.8 *Octave*–code for problems of this type

Even for small problems it is not very convenient to perform the calculations in the previous section by hand. It is a simple programming exercise to implement the method in any reasonable language. As an example we show an implementation in *Octave*[4]. Computations are split up in a few subroutines.

Generating an element stiffness matrix

The file `StabElementMatrix.m` contains code for a function to compute the element stiffness matrix of a truss with given data.

Octave

```
function elmat = BarElementMatrix(Coeff,Angle)
%  BarElementMatrix(Coeff,Angle)
%      generates the 4x4 Element stiffnes matrix, where
%      Coeff= E A /L
%      Angle = Angle of the truss with respect to the horizontal
```

[4] *Octave* is comparable to `MATLAB`, each has its own strong and weak points.

```
c = cos(Angle); s=sin(Angle);
T = Coeff* [ c^2  s*c
             s*c s^2] ;
elmat = zeros(4);
elmat(1:2 , 1:2 ) = T ;
elmat(3:4 , 1:2 ) = -T ;
elmat(1:2 , 3:4 ) = -T ;
elmat(3:4 , 3:4 ) = T ;
endfunction
```

Inserting an element stiffness matrix in the global stiffness matrix

The file `assemble.m` contains the corresponding code. As parameters we have to give the numbers of the nodes and the 4×4 element stiffness matrix, generated by `BarElementMatrix()`. The global variable `StMat` is the global stiffness matrix.

Octave

```
function assemble(k,n,Mat)
%  The element stiffness matrix belonging to the  connection from
%  point k to point n is included in the global stiffness matrix

global StMat
k2 = 2*k ; n2 = 2*n;

StMat((k2-1):k2,(k2-1):k2) = StMat((k2-1):k2,(k2-1):k2)  + Mat(1:2,1:2);
StMat((k2-1):k2,(n2-1):n2) = StMat((k2-1):k2,(n2-1):n2)  + Mat(1:2,3:4);
StMat((n2-1):n2,(k2-1):k2) = StMat((n2-1):n2,(k2-1):k2)  + Mat(3:4,1:2);
StMat((n2-1):n2,(n2-1):n2) = StMat((n2-1):n2,(n2-1):n2)  + Mat(3:4,3:4);
endfunction
```

Generating and solving the system of linear equations

With the help of the above two functions we may now solve the problem considered in the previous sections. The code is stored in the file `TrussSystem.m`.

Octave

```
% TrussSystem
output_precision = 4 ;

% find the element stiffness matrices
Ka = BarElementMatrix(1,-pi/3);
Kb = BarElementMatrix(1,0);
Kc = BarElementMatrix(1,-2*pi/3);
Kd = BarElementMatrix(1,0);
Ke = BarElementMatrix(1,-pi/3);

% Global stiffness matrix of the correct size
global StMat = zeros(8);

% taking care of all the elements
assemble(1,3,Ka); assemble(1,2,Kb); assemble(2,3,Kc);
assemble(3,4,Kd); assemble(2,4,Ke);

% Now StMat contains the global stiffness matrix
```

```
% remove 2nd, 3rd and 4th rows and columns
mat = StMat([1 5 6 7 8],[1 5 6 7 8]);

% the vector sol is to contain the unknown displacements of the points
% and is produced by solving the correct system of equations
sol =mat\[0 0 0 0 100]' ;

% Now compute the actual vector with all displacements
a = [sol(1)  0 0 0 sol(2:5)']'

% and find all the forces by a simple matrix multiplication
Fext = StMat * a
```

The result generated by the above code is given by

Octave

```
a =
    28.8675
     0.0000
     0.0000
     0.0000
   129.9038
    -8.3333
   187.6388
   241.6667

Fext =
   -3.323e-15
    5.000e+01
   -1.310e-14
   -1.500e+02
    2.842e-14
   -2.665e-15
   -1.532e-14
    1.000e+02
```

Interpretation of results

Each isolated element can now be examined by

Octave

```
Kc*[ 0 0 a(5:6)']'
```

leading to the forces on element c

Octave

```
ans =
  -28.87
  -50.00
   28.87
   50.00
```

2.2 Finite element method for a horizontal truss with variable cross section

2.2.1 Hooke's law and the energy of a stretched truss

Consider a truss of original length L and constant cross section A, see Figure 2.3. We place the truss along the x axis with $0 \leq x \leq L$ and keep the left endpoint at $x = 0$ fixed. At the right endpoint we apply a force F. Due to this force the truss will stretch and the new endpoint will be at $x = L + \Delta L$. Hooke's law gives

$$\frac{\Delta L}{L} = \frac{F}{E\,A}$$

where E is **Young's modulus** of elasticity, a constant determined by the type of material. The term F/A gives the force per unit area and is called **stress*** σ. The stress could depend on x but in this case we find it to be a constant.

Spannung
tension

Figure 2.3: An elementary horizontal truss

Obviously not only the right endpoint of the truss will move, but all parts of the truss will be moved. The point x will end up at $x + u(x)$ where $u(x)$ is the **displacement***. We know that $u(0) = 0$ and $u(L) = \Delta L$. It is reasonable to assume that the displacement depends linearly on x, i.e.

Verschiebung
déplacement

$$u\,(x) = \frac{\Delta L}{L}\,x$$

Now we can introduce the **strain*** ε as

Verzerrung
déformation

$$\varepsilon\,(x) = \lim_{h\to 0} \frac{u(x+h) - u(x)}{h} = \frac{d\,u(x)}{dx}$$

In the above example we find a constant strain of $\varepsilon(x) = \Delta L/L$. Hooke's law can now be reformulated as

$$\sigma(x) = E\,\varepsilon(x)$$

The truss can also be considered as a simple spring with a spring constant $k = \frac{E\,A}{L}$. The total energy U stored in a spring is known to be

$$U = \frac{1}{2}\,k\,(\Delta L)^2 = \frac{1}{2}\,\frac{E\,A}{L}\,(\Delta L)^2$$

As the energy is evenly distributed over the truss we find the energy density $e(x)$ (units: $\frac{\mathrm{J}}{\mathrm{m}^3}$) by

$$\begin{aligned} e\,A\,L &= U = \frac{1}{2}\,\frac{E\,A}{L}\,(\Delta L)^2 \\ e &= \frac{E}{2}\left(\frac{\Delta L}{L}\right)^2 = \frac{E}{2}\,\varepsilon^2 \\ &= \frac{1}{2}\,\sigma\,\varepsilon \end{aligned}$$

This result remains correct in more general situations, we have the basic formula

$$\text{density of energy } = e(x) = \frac{1}{2}\,\sigma(x)\cdot\varepsilon(x) \tag{2.4}$$

This will allow us to compute the elastic energy stored in a stretched (or compressed) truss. In Exercise 2–1 a truss with variable cross section is considered.

2.2.2 A truss with variable cross section, a finite element approach

The problem in this section in this section is taken from the book [MullGrot97]. It will serve as a first example on how to use finite elements to find an approximate solution to a problem. It is possible to find an exact solution to this question by using Exercise 2–1. As a guideline we use the following path of calculations:

- state the problem
- discretize the problem by dividing the truss in three simpler (constant cross section) elements
- introduce 4 well chosen variables
- compute the elastic energy stored in each of the elements
- give the formula for the total energy of the truss
- use the fact that the total energy has to be at a minimum for the physically stable situation
- solve the system of linear equations to find an approximate solution
- interpretation of the results, compare with the exact solution

2.2.3 Formulation of the special problem

Consider the truss shown in Figure 2.4. We know the following facts about the horizontal truss

Length	$L = 100$ cm
Area of cross section	$A = (10 - 0.09\,x)$ cm^2
Left edge	fixed
Force at right edge	$F = 2 \cdot 10^4$ N
Modulus of elasticity	$E = 3 \cdot 10^6 \frac{\text{N}}{\text{cm}^2}$

The goal is to compute the stress $\sigma(x)$ and the displacement $u(x)$ for this situation.

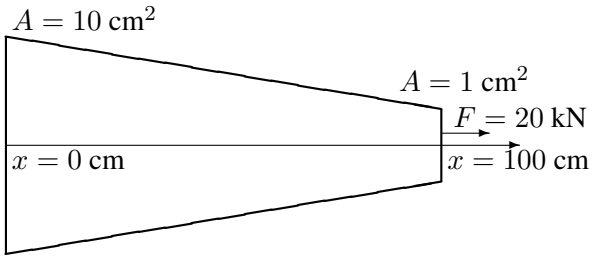


Figure 2.4: truss with variable cross section

2.2.4 Division in three elements

The truss in Figure 2.4 will be divided in three simpler trusses, each with constant cross section, see Figure 2.4. Within each of the tree truss we find the situation of the previous section. Thus the displacement will be an affine function (straight line), except at the connections. If the displacement is known at those

four special point, it can be computed at all points by a piecewise linear interpolation. We have the unknown displacements

$$u_0 = u(0) \quad , \quad u_1 = u(\frac{100}{3}) \quad , \quad u_2 = u(\frac{200}{3}) \quad \text{and} \quad u_1 = u(100)$$

Within each of the elements the strain σ will be constant.

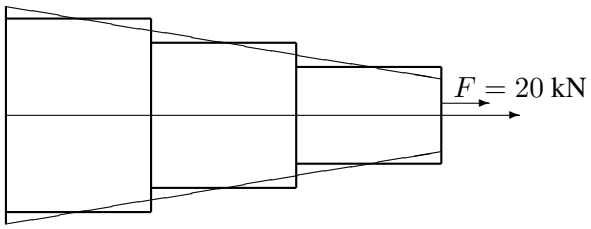


Figure 2.5: Truss divided in three elements, each with constant cross section

We arrive at three elements with the following data:

	Element 1	Element 2	Element 3
Domain of x	$0 \le x \le \frac{100}{3}$	$\frac{100}{3} \le x \le \frac{200}{3}$	$\frac{200}{3} \le x \le 100$
Length	$L_1 = \frac{100}{3}$	$L_2 = \frac{100}{3}$	$L_3 = \frac{100}{3}$
Cross section	$A_1 = 8.5$	$A_2 = 5.5$	$A_1 = 2.5$
Left displacement	u_0	u_1	u_2
Right displacement	u_1	u_2	u_3
Strain	$\varepsilon_1 = \frac{u_1 - u_0}{L_1}$	$\varepsilon_2 = \frac{u_2 - u_1}{L_2}$	$\varepsilon_3 = \frac{u_3 - u_2}{L_3}$

2.2.5 Elastic energy in the elements

Elastic energy in the first element

Since the strain in this element is given by $\varepsilon_1 = \frac{u_1 - u_0}{L_1}$ we find the stress σ_1 by Hooke's law

$$\sigma_1 = E\,\varepsilon_1 = E\,\frac{u_1 - u_0}{L_1}$$

Thus the energy density is given by equation (2.4) and thus

$$e_1 = \frac{1}{2}\,\sigma_1 \cdot \varepsilon_1 = \frac{E}{2}\,\frac{u_1 - u_0}{L_1}\,\frac{u_1 - u_0}{L_1} = \frac{E}{2\,L_1^2}\,\left(u_1^2 - 2\,u_1\,u_0 + u_0^2\right)$$

A simple multiplication leads to the elastic energy E_1 stored in the first element

$$E_1 = e_1\,L_1\,A_1 = \frac{E\,A_1}{2\,L_1}\,\left(u_1^2 - 2\,u_1\,u_0 + u_0^2\right) = \frac{E\,A_1}{2\,L_1}\,\langle \begin{pmatrix} u_0 \\ u_1 \end{pmatrix}, \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{pmatrix} u_0 \\ u_1 \end{pmatrix} \rangle$$

The last transformation of the expression is based on the idea used in Exercise 1–1. This leads to the definition of the **element stiffness matrix $\mathbf{K}_1$** as

$$\mathbf{K}_1 = \begin{bmatrix} \frac{E\,A_1}{L_1} & \frac{-E\,A_1}{L_1} \\ \frac{-E\,A_1}{L_1} & \frac{E\,A_1}{L_1} \end{bmatrix}$$

The energy can now be written as

$$E_1 = \langle \begin{pmatrix} u_0 \\ u_1 \end{pmatrix}, \mathbf{K}_1 \cdot \begin{pmatrix} u_0 \\ u_1 \end{pmatrix} \rangle$$

Elastic energy in the other elements

Similarly we obtain

$$E_2 = \frac{1}{2} \langle \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, \mathbf{K}_2 \cdot \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \rangle \quad \text{where} \quad \mathbf{K}_2 = \begin{bmatrix} \frac{E\,A_2}{L_2} & \frac{-E\,A_2}{L_2} \\ \frac{-E\,A_2}{L_2} & \frac{E\,A_2}{L_2} \end{bmatrix}$$

and

$$E_3 = \frac{1}{2} \langle \begin{pmatrix} u_2 \\ u_3 \end{pmatrix}, \mathbf{K}_3 \cdot \begin{pmatrix} u_2 \\ u_3 \end{pmatrix} \rangle \quad \text{where} \quad \mathbf{K}_3 = \begin{bmatrix} \frac{E\,A_3}{L_3} & \frac{-E\,A_3}{L_3} \\ \frac{-E\,A_3}{L_3} & \frac{E\,A_3}{L_3} \end{bmatrix}$$

Total elastic energy stored in the stretched truss

Now it is an easy exercise to compute the total internal energy in the truss by adding the three contributions. By using the notation of matrices we arrive at

$$U_{elast} = E_1 + E_2 + E_3 = \frac{1}{2} \langle \vec{u}\,, \mathbf{K} \cdot \vec{u} \rangle$$

with the vector of displacements $\vec{u} = (u_0, u_1, u_2, u_3)^T$ and the **total stiffness matrix K**

$$\mathbf{K} = \begin{bmatrix} \frac{E\,A_1}{L_1} & -\frac{E\,A_1}{L_1} & 0 & 0 \\ -\frac{E\,A_1}{L_1} & \frac{E\,A_1}{L_1} + \frac{E\,A_2}{L_2} & -\frac{E\,A_2}{L_2} & 0 \\ 0 & -\frac{E\,A_2}{L_2} & \frac{E\,A_2}{L_2} + \frac{E\,A_3}{L_3} & -\frac{E\,A_3}{L_3} \\ 0 & 0 & -\frac{E\,A_3}{L_3} & \frac{E\,A_3}{L_3} \end{bmatrix}$$

In our case we know that $u_0 = 0$ since the left end point is fixed. Using this we can eliminate the first row and column in the matrix **K** and arrive at the total elastic energy

$$U_{elast} = \frac{1}{2} \langle \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}, \begin{bmatrix} \frac{E\,A_1}{L_1} + \frac{E\,A_2}{L_2} & -\frac{E\,A_2}{L_2} & 0 \\ -\frac{E\,A_2}{L_2} & \frac{E\,A_2}{L_2} + \frac{E\,A_3}{L_3} & -\frac{E\,A_3}{L_3} \\ 0 & -\frac{E\,A_3}{L_3} & \frac{E\,A_3}{L_3} \end{bmatrix} \cdot \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \rangle$$

The external force written as potential energy

The right end point is subjected to a force of strength F. Thus if the displacement u_3 increases the corresponding potential energy has to decrease. A potential of

$$U_{ext} = -F \cdot u_3$$

will lead to the correct force $F = -\frac{d}{dx} U$. This can be rewritten as

$$U_{ext} = \langle \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ -F \end{pmatrix} \rangle$$

2.2.6 Combining the elastic energy of the elements and the external energy

Now it is easy to combine the two energies.

total potential energy = internal elastic energy + potential energy due to external forces

U = Uelast + Uext

This leads to

$$U(\vec{u}) = \frac{1}{2}\left\langle \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}, \begin{bmatrix} \frac{E\,A_1}{L_1}+\frac{E\,A_2}{L_2} & -\frac{E\,A_2}{L_2} & 0 \\ -\frac{E\,A_2}{L_2} & \frac{E\,A_2}{L_2}+\frac{E\,A_3}{L_3} & -\frac{E\,A_3}{L_3} \\ 0 & -\frac{E\,A_3}{L_3} & \frac{E\,A_3}{L_3} \end{bmatrix} \cdot \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \right\rangle + \left\langle \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ -F \end{pmatrix} \right\rangle$$

To solve the problem we have to use the **principle of least energy**:

> The system is at rest if the total potential energy is at a minimum.

Using Theorem 1–4 (page 4) this leads to a system of linear equations for the unknown displacement vector.

$$\begin{bmatrix} \frac{E\,A_1}{L_1}+\frac{E\,A_2}{L_2} & -\frac{E\,A_2}{L_2} & 0 \\ -\frac{E\,A_2}{L_2} & \frac{E\,A_2}{L_2}+\frac{E\,A_3}{L_3} & -\frac{E\,A_3}{L_3} \\ 0 & -\frac{E\,A_3}{L_3} & \frac{E\,A_3}{L_3} \end{bmatrix} \cdot \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ F \end{pmatrix}$$

Now we can apply all the known values and arrive at

$$10^6 \cdot \begin{bmatrix} 1.26 & -0.495 & 0 \\ -0.495 & 0.72 & -0.225 \\ 0 & -0.225 & 0.225 \end{bmatrix} \cdot \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = 10^6 \cdot \begin{pmatrix} 0 \\ 0 \\ 0.02 \end{pmatrix}$$

with the solution

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} 0.026 \\ 0.067 \\ 0.155 \end{pmatrix}$$

Since all the eigenvalues of the symmetric matrix are strictly positive we have in fact a minimum.

Now we can plot the solution in Figure 2.6. The dashed line corresponds to the exact solution. Since the strain ε is given as the derivative of the displacement. It will be a constant on each of the three elements. This is clearly visible in the right half of Figure 2.6. The stress σ is proportional to the strain.

2.2.7 How to improve the accuracy of the solution

Since we replaced the truss with variable cross section by a combination of three trusses with constant cross section we can not expect to obtain exact solutions. There are different ways to improve the accuracy of the solution.

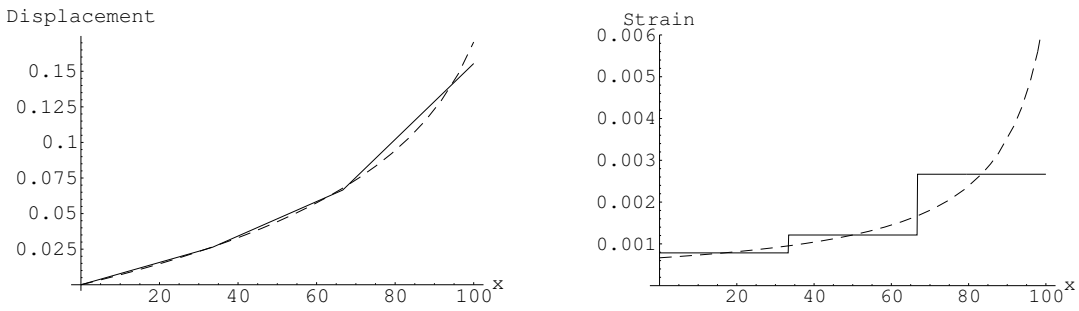


Figure 2.6: Displacement and strain in a truss with three elements

- Choose more elements. In Figure 2.7 the result for 10 elements of equal length is shown. Observe that the approximation of the displacement is good, while the strain still shows rather large errors in the thin parts of the truss.
- Move the location of the internal nodes, such that there are more elements in areas of large stress (see Exercise 2–2).
- Develop better elements to deliver a better approximation (see section 4.7.1). The elements used in this section assume that the strain is constant within one element. This can be changed. A quadratic function for the displacement can be used.

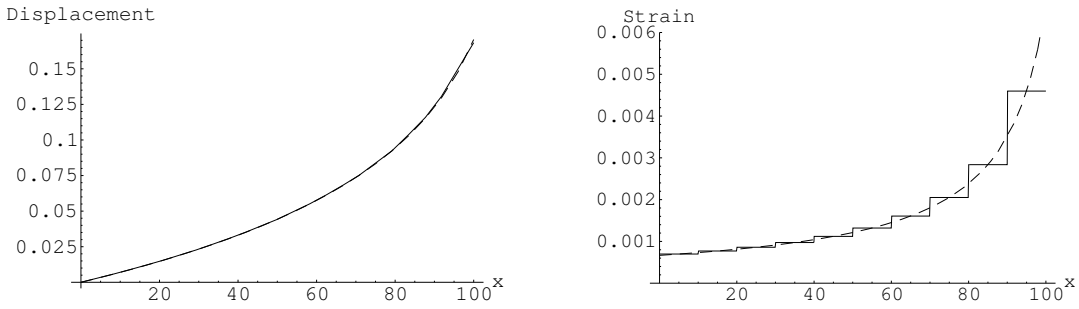


Figure 2.7: Displacement and strain in a truss with 10 elements

2.2.8 An afterthought

In this chapter we considered displacement functions u defined for $0 \leq x \leq 100$ which are piecewise linear and continuous with corners at the interior points x_1 and x_2. On each of the elements the energy density $e(x)$ is given by (Hooke's law: $\sigma = E\,\varepsilon$)

$$e(x) = \frac{1}{2}\,\sigma(x) \cdot \varepsilon(x) = \frac{1}{2}\,E\,u'(x)^2$$

In addition $u(x_0) = u(0) = 0$ is required. Amongst all those functions we tried to minimize the total energy

$$E(u) = \int_0^{100} A(x)\,e(x)\,dx - F \cdot u(100) = \frac{E}{2}\int_0^{100} A(x)\,u'(x)^2\,dx - F \cdot u(100)$$

The total energy $E(u)$ is a function of the displacement function $u(x)$, i.e. a functional. In the next chapter we will look for the minimal energy amongst *all* permissable function, without the restriction *piecewise linear*. This will lead to the calculus of variations.

2.2.9 Exercises

• Exercise 2–1:
Consider a truss of length L ($0 \le x \le L$) with variable cross section $A(x)$. Keep the left endpoint at $x = 0$ fixed and apply a force of strength F at the right end point.

(a) Compute the stress $\sigma(x)$ using the fact that the total force applied to a cross section can not depend on x.

(b) Use Hooke's law to compute the strain $\varepsilon(x)$ and then compute the total elastic energy in the truss.

(c) Find the total change ΔL of the length L by using the strain $\varepsilon(x)$. The result is an integral containing the function $A(x)$.

(d) Show that the spring constant for this truss is given by

$$k = \frac{\text{force}}{\text{change of length}} = \left(\int_0^L \frac{1}{E\,A(s)}\, ds \right)^{-1}$$

• Exercise 2–2:
Search and load the *Mathematica* notebook `Truss.nb`.

(a) Read and try to understand the given code in the section with the approximation by three elements.

(b) Modify the location of the internal nodes to obtain a better approximation.

(c) Modify the code to generate an approximation by 4 elements.

• Exercise 2–3:
Write a *Mathematica* notebook to solve the problem of a truss with variable cross section and arbitrary number and length of elements. The input below should generate the results in this section.

Mathematica

```
nodes={0,100/3,200/3,100};
A[x_]  := 10-0.09*x;
Em      = 3*10^6;
Force   = 2*10^4;
```

The next code should generate Figure 2.7.

Mathematica

```
nodes=Table[x,{x,0,100,10}]
A[x_] := 10-0.09*x;
Em     = 3*10^6;
Force  = 2*10^4;
```

Chapter 3

Calculus of variations for functions of one independent variable

3.1 The Euler Lagrange equation

In this section the most important result will carefully be developed. We start with a technical, but important result, then consider a simple example and aim for the general Euler Lagrange equation.

If the functions $u(x)$ and $f(x, u, u')$ are given then the definite integral

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

is well defined. The main idea is now to examine the behaviour of $F(u)$ if we choose different functions u. We will search for special functions $u(x)$ which will minimise the value of $F(u)$. Some of the ideas and results from the previous chapter will be reused, but some essential new concepts will be used too.

3.1.1 The fundamental lemma of the calculus of variations

3–1 Lemma : *If $u(x)$ is a continuous function for $a \le x \le b$ and*

$$\int_a^b u(x) \cdot \phi(x)\ dx = 0$$

for all differentiable functions ϕ with $\phi(a) = \phi(b) = 0$ then

$$u(x) = 0 \quad \textit{for all} \quad a \le x \le b$$

$\diamond$

Proof : We proceed by contradiction. Assume that for some x_0 between a and b we we have $u(x_0) > 0$. Since the function $u(x)$ is continuous we know that $u(x) > 0$ on a (possibly small) interval $x_1 < x < x_2$. Now we choose

$$\phi(x) = \begin{cases} 0 & \text{for} \quad x \le x_1 \\ (x - x_1)^2 (x - x_2)^2 & \text{for} \quad x_1 \le x \le x_2 \\ 0 & \text{for} \quad x_2 \le x \end{cases}$$

Then we have $u(x)\,\phi(x) \ge 0$ for all $a \le x \le b$ and $u(x_0)\,\phi(x_0) > 0$ and thus

$$\int_a^b u(x) \cdot \phi(x)\ dx = \int_{x_1}^{x_2} u(x) \cdot \phi(x)\ dx > 0$$

This is a contradiction to the condition in the Lemma. Thus we have $u(x) = 0$ for $a < x < b$. As the function u is continuous we also have $u(a) = u(b) = 0$. $\square$

With a few more mathematical ideas the above result can be improved and we obtain an important result for the calculus of variations.

3–2 Theorem : ***Fundamental lemma of the calculus of variations***
If $u(x)$ is a continuous function for $a \leq x \leq b$ and

$$\int_a^b u(x) \cdot \phi(x)\, dx = 0$$

for all infinitely often differentiable functions $\phi(x)$ with $\phi(a) = \phi(b) = 0$ then

$$u(x) = 0 \quad \textit{for all} \quad a \leq x \leq b$$

◇

If the integral condition is formulated with the derivative $\phi'(x)$ of the test function $\phi(x)$ then we arrive at a modified result

3–3 Lemma : *If $u(x)$ is a continuous function for $a \leq x \leq b$ and*

$$\int_a^b u(x) \cdot \phi'(x)\, dx = 0$$

for all infinitely often differentiable functions $\phi(x)$ then

$$u'(x) = 0 \quad \textit{for all} \quad a \leq x \leq b \quad \textit{and} \quad u(a) \cdot \phi(a) = u(b) \cdot \phi(b) = 0$$

◇

Proof : We use integration by parts

$$\begin{aligned} 0 &= \int_a^b u(x) \cdot \phi'(x)\, dx \\ &= u(b) \cdot \phi(b) - u(a) \cdot \phi(a) - \int_a^b u'(x) \cdot \phi(x)\, dx \end{aligned}$$

Considering all test function $\phi(x)$ with $\phi(a) = \phi(b) = 0$ leads to the condition $u'(x) = 0$. We are free to choose test functions with arbitrary values at the end points a and b, thus we arrive at $u(a) \cdot \phi(a) = u(b) \cdot \phi(b) = 0$. □

3.1.2 Shortest connection between two given points

As an first problem to the calculus of variations of one variable, we consider the question of finding the shortest connection between two points. The solution is obviously given by a straight line. In figure 3.1 the optimal and a lesser solution are shown. Now we use the fundamental lemmas to find the optimal solution in a systematic fashion.

For two given points (a, y_1) and (b, y_2) in the xy–plane we are looking for the function $y = u(x)$ which gives the shortest connection between the two points. The length L of the curve is given by the integral

$$L(u) = \int_a^b \sqrt{1 + (u'(x))^2}\, dx$$

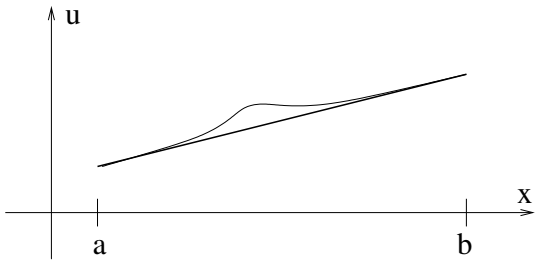


Figure 3.1: Shortest connection between two points

Assuming the optimal solution $u(x)$ is given, we perturb it by a smooth function $\varepsilon\eta(x)$ with $\eta(a) = \eta(b) = 0$ and $\varepsilon \in \mathbb{R}$ small. Thus we have the boundary conditions

$$u(a) = u(a) + \varepsilon\eta(a) = y_1 \quad \text{and} \quad u(b) = u(b) + \varepsilon\eta(b) = y_2$$

Now we examine the function

$$F(\varepsilon) = L(u + \varepsilon\eta) = \int_a^b \sqrt{1 + (u'(x) + \varepsilon\eta'(x))^2}\, dx$$

If $u(x)$ is in fact the optimal solution then the function $F(\varepsilon)$ needs to be minimal at $\varepsilon = 0$. Thus we have the necessary condition $\frac{d}{d\varepsilon} F(0) = 0$. This leads to

$$\begin{aligned}
\frac{d}{d\varepsilon} F(\varepsilon) &= \frac{d}{d\varepsilon} \int_a^b \sqrt{1 + (u'(x) + \varepsilon\eta'(x))^2}\, dx \\
&= \int_a^b \frac{d}{d\varepsilon} \sqrt{1 + (u'(x) + \varepsilon\eta'(x))^2}\, dx \\
&= \int_a^b \frac{(u'(x) + \varepsilon\eta'(x))}{\sqrt{1 + (u'(x) + \varepsilon\eta'(x))^2}}\, \eta'(x)\, dx
\end{aligned}$$

If we set $\varepsilon = 0$ then this integral has to vanish and we obtain

$$\int_a^b \frac{u'(x)}{\sqrt{1 + (u'(x))^2}}\, \eta'(x)\, dx = 0 \quad \text{for all smooth functions } \eta(x) \text{ with } \quad \eta(a) = \eta(b) = 0$$

With the help of the fundamental Lemma 3–3 we conclude

$$\frac{d}{dx} \frac{u'(x)}{\sqrt{1 + (u'(x))^2}} = 0$$

If the derivative of an expression vanishes everywhere, then it has to be a constant, thus

$$\frac{u'(x)}{\sqrt{1 + (u'(x))^2}} = c$$

Subsequently $u'(x)$ has to be constant, thus the optimal solution is a straight line. The two boundary conditions $u(a) = y_1$ and $u(b) = y_2$ will determine the precise form of the line.

We had to find an extremum of a functional

$$F(u) = \int_a^b f(x, u, u')\, dx \quad \text{with} \quad f(x, u, u') = \sqrt{1 + (u')^2}$$

Most of the problems of the calculus of variations in one variable fit into a very similar framework and we will look at this type of problem with some more care.

3–4 Definition : If a mapping is defined for a set of functions X and returns a number as a result then it is called a **functional** on the function space X.

Thus a functional is nothing but a function with a a set of functions as domain of definition. It might help to compare typical functions and functionals.

	domain of definition	range
function	interval $[a, b]$	numbers $\mathbb{R}$
functional	continuous functions defined on $[a, b]$, i.e. $C([a, b], \mathbb{R})$	numbers $\mathbb{R}$

Here are a few examples of functionals

$$
\begin{array}{lllll}
F(u) = \int_0^\pi a(x)\, u^2\, dx & \text{defined on} & C([0,1],\mathbb{R}) & \text{with} & u(0) = u(\pi) = 1 \\
F(u) = \int_0^1 \sqrt{1 + u'(x)^2}\, dx & \text{defined on} & C^1([0,1],\mathbb{R}) & \text{with} & u(0) = 1\ ,\ u(1) = \pi \\
F(u) = \int_0^1 (u'(x)^2 - 1)^2 + u(x)^2\, dx & \text{defined on} & C^1([0,1],\mathbb{R}) & \text{with} & u(0) = u(1) = 0 \\
F(u) = \int_0^1 a(x)\, u''(x)^2\, dx & \text{defined on} & C^2([0,1],\mathbb{R}) &&
\end{array}
$$

The principal goal of the calculus of variations is to find extrema of functionals.

3.1.3 Critical values of functionals of the form $\int f(x, u(x))\, dx$

For a given function f we try to find a function $u(x)$ such that the functional

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

has a critical value for the function u. The critical value can be a maximal value, a minimal value or a generalised saddle point. We require that

$$\frac{d}{d\varepsilon}\, F(u + \varepsilon\eta)\Big|_{\varepsilon=0} = 0 \quad \text{for „all“ functions} \quad \eta$$

To find the equations to be satisfied by the solution $u(x)$ we use linear approximations. For small values of Δu we have

$$
\begin{aligned}
f\,(x, u + \Delta u) &\approx f(x,u) + \frac{\partial\, f(x,u)}{\partial u}\, \Delta u \\
&= f(x,u) + f_u\,(x,u)\, \Delta u \\
f\,(x, u(x) + \varepsilon\eta\,(x)) &\approx f\,(x, u(x)) + \varepsilon\, f_u(x, u(x))\, \eta\,(x)
\end{aligned}
$$

Now examine the functional in question

$$
\begin{aligned}
F\,(u) &= \int_a^b f\,(x, u(x))\, dx \\
F\,(u + \varepsilon\,\eta) &= \int_a^b f\,(x, u(x) + \varepsilon\eta\,(x))\, dx \\
&\approx \int_a^b f\,(x, u(x)) + \varepsilon\, f_u(x, u(x))\, \eta\,(x)\, dx \\
&= F(u) + \varepsilon \int_a^b f_u(x, u(x))\, \eta\,(x)\, dx
\end{aligned}
$$

If u leads to a critical value then the factor of ε has to vanish, i.e.

$$\int_a^b f_u(x,u(x))\,\eta\,(x)\,dx = 0 \quad \text{for all function} \quad \eta$$

Another way to derive this condition is to differentiate with respect to the parameter ε, and then set $\varepsilon = 0$. The result has to be the same.

$$\frac{d}{d\varepsilon}\,F\,(u+\varepsilon\,\eta)\,\Big|_{\varepsilon=0} = \frac{d}{d\varepsilon}\,\left(\int_a^b f\,(x,u(x)+\varepsilon\eta\,(x))\,dx\right)\,\Big|_{\varepsilon=0} = \int_a^b f_u(x,u(x))\,\eta\,(x)\,dx$$

If this integral has to vanish for all function $\eta(x)$ then we use the fundamental lemma to obtain the necessary condition for a critical value

$$\int_a^b f\,(x,u(x))\,dx \text{ critical for } u \qquad \Longrightarrow \qquad f_u(x,u(x)) = 0 \quad \text{for} \quad a < x < b$$

3–5 Example : If the function $u(x)$ leads to a critical value of the functional

$$F(u) = \int_0^\pi x^2 + u^2(x)\,dx$$

then we have to consider

$$\begin{aligned} f(x,u) &= x^2+u^2 \\ f_u(x,u) &= 2\,u \\ 0 &= f_u(x,u(x)) = 2\,u(x) \end{aligned}$$

This leads to the obvious solution $u(x) = 0$. ♢

3–6 Example : If the function $u(x)$ leads to a critical value of the functional

$$F(u) = \int_{-1}^1 \cosh(x+u(x))\,dx$$

then we have to consider

$$\begin{aligned} f(x,u) &= \cosh(x+u) \\ f_u(x,u) &= \sinh(x+u) \\ 0 &= f_u(x,u(x)) = \sinh(x+u(x)) \\ u(x) &= -x \end{aligned}$$

This leads to the solution $u(x) = -x$. By inspection the original problem carefully we can verify that this is in fact a correct solution. ♢

3.1.4 Critical values of functionals of the form $\int f(x,u(x),u'(x))\,dx$

For a given function f we try to find a function $u(x)$ such that the functional

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

has a critical value for the function u. We require that

$$\frac{d}{d\varepsilon}\,F(u+\varepsilon\eta)\,\Big|_{\varepsilon=0} = 0 \quad \text{for „all“ functions} \quad \eta$$

To find the equations to be satisfied by the solution $u(x)$ we use linear approximations. For small values of Δu we have

$$\begin{aligned} f\left(x, u+\Delta u, u'+\Delta u'\right) &\approx f(x,u) + \frac{\partial\, f(x,u,u')}{\partial u}\,\Delta u + \frac{\partial\, f(x,u,u')}{\partial u'}\,\Delta u' \\ &= f(x,u) + f_u\left(x,u,u'\right)\,\Delta u + f_{u'}\left(x,u,u'\right)\,\Delta u' \\ f\left(x, u(x)+\varepsilon\eta\left(x\right), u'(x)+\varepsilon\eta'(x)\right) &\approx f\left(x,u(x)\right) + \varepsilon\, f_u(x,u(x),u')\,\eta\left(x\right) + \\ &\quad +\varepsilon\, f_{u'}(x,u(x),u'(x))\,\eta'\left(x\right) \end{aligned}$$

Now we examine the functional in question

$$\begin{aligned} F\left(u\right) &= \int_a^b f\left(x,u(x),u'(x)\right)\,dx \\ F\left(u+\varepsilon\,\eta\right) &= \int_a^b f\left(x,u(x)+\varepsilon\eta\left(x\right),u'(x)+\varepsilon\eta'\left(x\right)\right)\,dx \\ &\approx \int_a^b f\left(x,u(x),u'(x)\right) + \varepsilon\, f_u(x,u(x),u'(x))\,\eta\left(x\right) + \varepsilon\, f_{u'}(x,u(x),u'(x))\,\eta'\left(x\right)\,dx \\ &= F(u) + \varepsilon\int_a^b f_u(x,u(x),u'(x))\,\eta\left(x\right) + f_{u'}(x,u(x),u'(x))\,\eta'\left(x\right)\,dx \end{aligned}$$

or

$$\frac{d}{d\varepsilon}\,F\left(u+\varepsilon\,\eta\right)\Big|_{\varepsilon=0} = \int_a^b f_u(x,u(x),u'(x))\,\eta\left(x\right) + f_{u'}(x,u(x),u'(x))\,\eta'\left(x\right)\,dx$$

If this integral has to vanish for all function $\eta(x)$ the we have a necessary condition. An integration by parts leads to

$$\begin{aligned} 0 &= \int_a^b f_u(x,u(x),u'(x))\,\eta\left(x\right) + f_{u'}(x,u(x),u'(x))\,\eta'\left(x\right)\,dx \\ &= f_{u'}(x,u(x),u'(x))\,\eta\left(x\right)\Big|_{x=a}^{b} \\ &\qquad + \int_a^b \left(f_u(x,u(x),u'(x)) - \frac{d}{dx}f_{u'}(x,u(x),u'(x))\right)\,\eta\left(x\right)\,dx \end{aligned}$$

If this expression is to vanish for all function $\eta(x)$ we need

$$\int_a^b f\left(x,u(x),u'(x)\right)\,dx \text{ extremal} \qquad \Longrightarrow \qquad \left\{\begin{array}{rcl} \frac{d}{dx}f_{u'}(x,u(x),u'(x)) &=& f_u(x,u(x)) \\ f_{u'}(x,u(a),u'(a))\,\eta\left(a\right) &=& 0 \\ f_{u'}(x,u(b),u'(b))\,\eta\left(b\right) &=& 0 \end{array}\right.$$

The first condition is the **Euler Lagrange** equation, the second and third condition are **boundary conditions**. If the value $u(a)$ is given and we are not free to choose it, then we need $\eta(a) = 0$ and the first boundary condition is automatically satisfied. If we are free to choose $u(a)$ then $\eta(a)$ need not vanish and we have the condition

$$f_{u'}(x,u(a),u'(a)) = 0$$

A similar argument applies at the other endpoint $x = b$.

Now we have the central result for the calculus of variations in one variable.

3–7 Theorem : ***Euler Lagrange equation***
If a smooth function $u(x)$ leads to a critical value of the functional

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

the differential equation

$$\frac{d}{dx}\, f_{u'}(x, u(x), u'(x)) = f_u(x, u(x))$$

has to be satisfied for $a < x < b$. This is usually a second order differential equation.

- *If it is a critical value amongst all functions with prescribed boundary values $u(a)$ and $u(b)$, use these to solve the differential equation.*
- *If you are free to choose the values of $u(a)$ and $u(b)$, then the* ***natural boundary conditions***

$$f_{u'}(x, u(a), u'(a)) = 0 \quad \textit{and} \quad f_{u'}(x, u(b), u'(b)) = 0$$

can be used.

◇

3–8 Example : The problem of the shortest connection between two given points (see section 3.1.2) leads to the question of finding a function $u(x)$ such that

$$F(u) = \int_a^b \sqrt{1 + (u'(x))^2}\, dx \quad \text{minimal, and} \quad u(a) = y_1 \quad , \quad u(b) = y_2$$

Thus we have

$$\begin{aligned} f(x, u, u') &= \sqrt{1 + (u')^2} \\ f_u(x, u, u') &= 0 \\ f_{u'}(x, u, u') &= \frac{u'}{\sqrt{1 + (u')^2}} \end{aligned}$$

The Euler Lagrange equation for this example is

$$\begin{aligned} 0 = \frac{d}{dx}\, f_{u'}(x, u, u') &= \frac{d}{dx}\, \frac{u'}{\sqrt{1 + (u')^2}} = \frac{u''}{\sqrt{1 + (u')^2}} - \frac{(u')^2 \cdot u''}{\sqrt{1 + (u')^2}^3} \\ &= \frac{u''\, (1 + (u'(x)^2) - (u')^2 \cdot u''}{\sqrt{1 + (u')^2}^3} = \frac{u''}{\sqrt{1 + (u')^2}^3} \end{aligned}$$

This equation is satisfied if $u''(x) = 0$, thus we find the straight line again. ♢

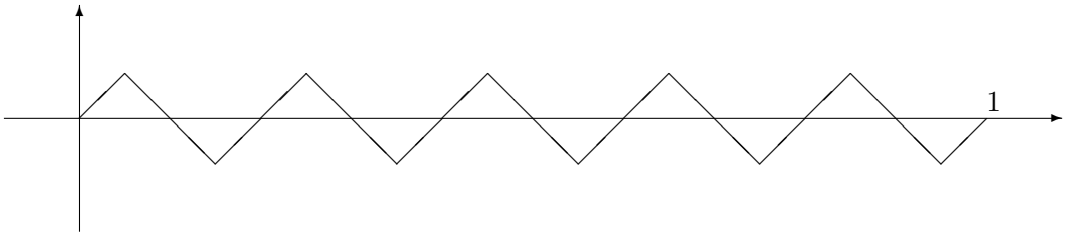
3–9 Example : The Euler Lagrange equations are a necessary condition, but not a sufficient condition. There are functionals where the Euler Lagrange equation can be solved, but the functional will not attain a minimal value. As an example consider the functional

$$F\,(u) = \int_0^1 \left((u'(x))^2 - 1\right)^2 + u(x)^2\, dx \quad \text{with} \quad u(0) = u(1) = 0$$

which leads to the Euler Lagrange equation

$$\begin{aligned} \frac{d}{dx}\, 2\, \left((u'(x))^2 - 1\right)\, 2\, u'(x) &= 2\, u(x) \\ 4\, (u'(x))^2\, u''(x) + \left((u'(x))^2 - 1\right)\, 2\, u''(x) &= u(x) \\ 2\, \left(3\, (u'(x))^2 - 1\right)\, u''(x) &= u(x) \end{aligned}$$

This is a second order differential equation. A solution of this equation, including boundary conditions, is given by $u(x) = 0$. For this solution we have $F(u) = 1$. But this is not the minimal value of the functional. To verify this fact consider the triangular function in the figure below.



As the slope of the function is ± 1 the first part of the functional vanishes and we only have to compute

$$F(u) = \int_0^1 u(x)^2 \, dx$$

As the number of triangles is increased their height will decrease and this integral will obviously converge to zero. ♢

3.1.5 Quadratics functionals and second order linear boundary value problems

If for given functions $a(x)$, $b(x)$ and $g(x)$ the functional

$$F(u) = \int_a^b \frac{1}{2} a(x) \, (u'(x))^2 + \frac{1}{2} b(x) \, u(x)^2 + g(x) \cdot u(x) \, dx \tag{3.1}$$

has to be minimised, then we obtain the Euler Lagrange equation

$$\begin{aligned} \frac{d}{dx} f_{u'} &= f_u \\ \frac{d}{dx} \left(a(x) \frac{d \, u(x)}{dx} \right) &= b(x) \, u(x) + g(x) \end{aligned}$$

This is a linear, second order differential equation which has to be supplemented with appropriate boundary conditions. If the value at one of the endpoints is given then this is called a **Dirichlet boundary condition**. If we are free to choose the value at the boundary then this is called a **Neumann boundary condition**. Theorem 3–7 implies that the second situation leads to a natural boundary condition

$$a(x) \frac{d \, u}{dx} = 0 \quad \text{for} \quad x = a \quad \text{or} \quad x = b$$

If we wish to consider a non-homogeneous boundary conditions

$$a(x) \frac{d \, u}{dx} = r(x) \quad \text{for} \quad x = a \quad \text{or} \quad x = b$$

then the functional has to be supplemented by (see exercise 3–6)

$$F(u) + r(a) \, u(a) - r(b) \, u(b)$$

Thus the above approach shows that many second order differential equation corresponds to an extremal point for a properly chosen functional.

Many physical, mechanical and electrical problems lead to this type of equation as can be seen in Table 3.1 (Source: [OttoPete92, p. 63]). Examples of this type are considered in section 4.3.

differential equation	problem description			constitutive law
$\frac{d}{dx}\left(A\,k\frac{dT}{dx}\right)+Q=0$	one–dimensional heat flow	$T=$ $A=$ $k=$ $Q=$	temperature area thermal conductivity heat supply	Fourier's law $q=-k\,\frac{dT}{dx}$
$\frac{d}{dx}\left(A\,E\frac{du}{dx}\right)+b=0$	axially loaded elastic bar	$u=$ $A=$ $E=$ $b=$	displacement area Young's modulus axial loading	Hooke's law $\sigma=E\,\frac{du}{dx}$ σ = stress
$\frac{d}{dx}\left(S\frac{dw}{dx}\right)+p=0$	transversely loaded flexible string	$w=$ $S=$ $p=$	deflection string force lateral loading	
$\frac{d}{dx}\left(A\,D\frac{dc}{dx}\right)+Q=0$	one dimensional diffusion	$c=$ $A=$ $D=$ $Q=$	concentration area Diffusion coefficient external supply	Fick's law $q=-D\,\frac{dc}{dx}$ q = flux
$\frac{d}{dx}\left(A\,\gamma\,\frac{dV}{dx}\right)+Q=0$	one dimensional electric current	$V=$ $A=$ $\gamma=$ $Q=$	voltage area electric conductivity charge supply	Ohm's law $q=-\gamma\,\frac{dV}{dx}$ q = charge flux
$\frac{d}{dx}\left(A\,\frac{D^2}{32\nu}\,\frac{dp}{dx}\right)+Q=0$	laminar flow in a pipe (Poisseuille flow)	$p=$ $A=$ $D=$ $\mu=$ $Q=$	pressure area diameter viscosity fluid supply	$q=\frac{D^2}{32\nu}\,\frac{dp}{dx}$ $q=$ volume flux

Table 3.1: Examples of second order differential equations

3.1.6 First integrals

If the function f in integral of an variational problem does not depend on x or u, then the Euler Lagrange equation can be simplified and we can conclude that certain expressions have to be constant, i.e. we have a **first integral**.

3–10 Result : *If the function f in the functional*

$$F(u) = \int_a^b f\left(x, u'(x)\right)\, dx$$

does not explicitly depend on u then the expression

$$f_{u'}(x, u'(x)) = C_1$$

is a first integral. ◇

Proof : Since f does not depend on u we have $f_u = 0$ and the Euler Lagrange equation simplifies to

$$\frac{d}{dx}\, f_{u'} = 0$$

Thus the expression in the result can not depend on x. □

3–11 Result : *If the function f in the functional*

$$F(u) = \int_a^b f\left(u(x), u'(x)\right)\, dx$$

does not explicitly depend on x then the expression

$$u'\, f_{u'} - f = C_1$$

is a first integral. ◇

Proof : Since f does not depend on x the chain rule implies

$$\frac{d}{dx}\, f = f_u\, u' + f_{u'}\, u''$$

We use the Euler Lagrange equation

$$\frac{d}{dx}\, f_{u'} = f_u$$

to show that the total derivative of the expression in question vanishes.

$$\begin{aligned} \frac{d}{dx}\, \left(u'\, f_{u'} - f\right) &= u'' f_{u'} + u'\, \frac{d\, f_{u'}}{dx} - \frac{d\, f}{dx} \\ &= u'' f_{u'} + u'\, f_u - \left(f_u\, u' + f_{u'}\, u''\right) = 0 \end{aligned}$$

□

3–12 Example : The problem of the shortest connection between two points leads to the functional

$$F(u) = \int_a^b \sqrt{1 + (u'(x))^2}\, dx$$

Since the integrand does not explicitly depend on u we have the first integral

$$C = f_{u'}(x, (u'(x)) = \frac{u'(x)}{\sqrt{1 + (u'(x))^2}}$$

Thus we conclude

$$\begin{aligned} C\ \sqrt{1 + (u'(x))^2} &= u'(x) \\ C^2\ \left(1 + (u'(x))^2\right) &= (u'(x))^2 \\ (u'(x))^2 &= \frac{C^2}{1 - C^2} \end{aligned}$$

and we obtain again the result that the slope has to be constant. ♢

3.1.7 Functionals depending on several functions

If a functional

$$F(u_1, u_2) = \int_a^b f(x, u_1(x), u_2(x), u_1'(x), u_2'(x))\, dx$$

depends on two functions u_1 and u_2 and their first derivatives, the situation changes slightly. If the functional is minimised then the new function

$$g(\varepsilon) = F(u_1 + \varepsilon \cdot \eta, u_2) = \int_a^b f(x, u_1(x) + \varepsilon \cdot \eta(x), u_2(x), u_1'(x) + \varepsilon \cdot \eta'(x), u_2'(x))\, dx$$

needs to have a minimum for $\varepsilon = 0$ and thus

$$\frac{d}{d\varepsilon}\, g(\varepsilon)\Big|_{\varepsilon=0} = 0$$

This leads to the necessary condition

$$0 = \int_a^b f_{u_1}(x, u_1(x), u_2(x), u_1'(x), u_2'(x)) \cdot \eta(x) f_{u_1'}(x, u_1(x), u_2(x), u_1'(x), u_2'(x)) \cdot \eta'(x)\, dx$$

for all admissible functions η. Integration by parts and using the fundamental lemma we have one Euler Lagrange equation

$$\frac{d}{dx}\, f_{u_1'}(x, \vec{u}(x), \vec{u}'(x)) = f_{u_1}(x, \vec{u}(x), \vec{u}'(x))$$

Identical arguments apply to the dependence on the second function u_2 and we have **one Euler Lagrange equation for each function** the functional depends on.

This situation occurs often for mechanical problems when using Hamilton's principle of least action.

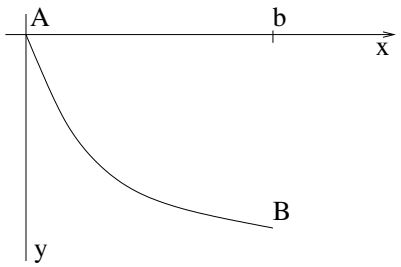


Figure 3.2: The brachistochrone by Johann Bernoulli

3.2 Examples

3.2.1 Brachistochrone problem

In 1696 Johann Bernoulli asked the following question: *Given two points A and B in a vertical plane. What is the path to be used by a moving particle to get from point A to point B in the shortest time possible?* It is assume that gravity is the only force available and friction is ignored. The situation is shown in figure 3.2. The techniques used here to solve the problem were not available to Johann Bernoulli, he used Fermat's principle from optics to find a solution.

The velocity at a given height y is known to be $v = \sqrt{2\,g}\,\sqrt{y}$. Now we assume that the curve is given by a function $y = u(x)$. The length element along the path is

$$ds = \sqrt{1 + u'(x)^2}\,dx$$

and thus the total travel time is

$$T(u) = \int_0^b \frac{ds}{v} = \frac{1}{\sqrt{2\,g}} \int_0^b \frac{\sqrt{1 + u'(x)^2}}{\sqrt{u(x)}}\,dx$$

we have to find a function u to render this integral minimal. As the integrand does not explicitly depend on x we have a first integral

$$u'\,f_{u'} - f = u'\,\frac{u'}{\sqrt{u}\,\sqrt{1 + (u')^2}} - \frac{\sqrt{1 + (u')^2}}{\sqrt{u}} = C$$

This equation can be solved for u'. The calculations are shown below.

$$\begin{aligned}
\frac{(u')^2}{\sqrt{1 + (u')^2}} - \sqrt{1 + (u')^2} &= C\,\sqrt{u} \\
(u')^2 - (1 + (u')^2) &= C\,\sqrt{u}\,\sqrt{1 + (u')^2} \\
\frac{-1}{C\,\sqrt{u}} &= \sqrt{1 + (u')^2} \\
1 + (u')^2 &= \frac{1}{C^2\,u} \\
\frac{d\,u}{dx} &= \sqrt{\frac{1}{C^2\,u} - 1} = \sqrt{\frac{1 - C^2\,u}{C^2\,u}}
\end{aligned}$$

Separation of variables leads to the integral. To compute the integral on the right we use the substitution $C^2\,u = \sin^2\theta$ and $du = \frac{2}{C^2}\,\sin\theta\,\cos\theta\,d\theta$

$$\int 1\,dx = \int \sqrt{\frac{C^2\,u}{1 - C^2\,u}}\,du$$

$$\begin{aligned} x &= \int \sqrt{\frac{\sin^2\theta}{1-\sin^2\theta}}\ \frac{2}{C^2}\ \sin\theta\ \cos\theta\ d\theta = \int \sqrt{\frac{\sin^2\theta}{\cos^2\theta}}\ \frac{2}{C^2}\ \sin\theta\ \cos\theta\ d\theta \\ \frac{C^2}{2}\ x &= \int \sin^2\theta\ d\theta = \frac{1}{4}\ (2\,\theta - \sin(2\,\theta)) + k \end{aligned}$$

Using

$$y = u(x) = \frac{1}{C^2}\ \sin^2\theta = \frac{1}{2\,C^2}\,(1-\cos(2\theta))$$

and the new parameter $t = 2\,\theta$ we find the parametrisation of a cycloid.

$$\begin{aligned} x(t) &= \tfrac{1}{2\,C^2}\ (t-\sin t) + x_0 \\ y(t) &= \tfrac{1}{2\,C^2}\ (1-\cos t) \end{aligned} \qquad \text{where} \quad t \in \mathbb{R}$$

This curve can be generated by attaching a pen at the perimeter of a wheel with diameter $1/C$ and then roll it in direction x. Find the graph of a cycloid of a wheel with radius 1 in figure 3.3. Observe that orientation of the vertical axis is different from figure 3.2.

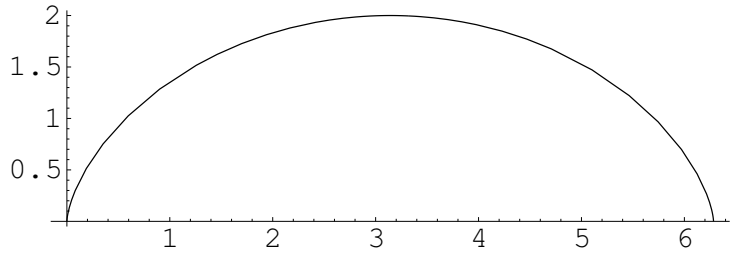


Figure 3.3: Graph of a cycloid

3–13 Example : The *Mathematica* code below will find and plot the curve connecting the origin with the point $(1,1)$ such that the time of flight will be minimal.

Mathematica

```
Clear[x,y,c]
x[t_]:= 1/(2 c) (t-Sin[t])
y[t_]:= 1/(2 c) (1-Cos[t])
sol=FindRoot[{x[t]==1,y[t]==1},{t,1.0},{c,0.5}]
ParametricPlot[ {x[t],y[t]}/. sol[[2]],{t,0,t/.sol[[1]]},
AspectRatio ->Automatic];
```

♢

3–14 Example : If the particle is to move from the origin to the vertical line $x = b$ as first as possible but the height at $x = b$ is not prescribed then we have to use the natural boundary condition

$$\begin{aligned} f_{u'}(x, u(b), u'(b)) &= 0 \\ \frac{u'}{\sqrt{u}\,\sqrt{1+(u')^2}} &= 0 \\ u'(b) &= 0 \end{aligned}$$

Thus we know that the optimal solution will hit the line $x = b$ horizontally. For the parametrisation of the cycloid we have thus the necessary condition

$$\frac{d}{dt}\,y(t) = \frac{d}{dt}\,\frac{1}{2\,C^2}(1-\cos t) = \frac{\sin t}{2\,C^2} = 0$$

This implies $t = \pi$ and from

$$b = x(\pi) = \frac{1}{2\,C^2}\;(\pi - \sin\pi) + x_0 = \frac{\pi}{2\,C}$$

we conclude $C = \frac{\pi}{2\,b}$ Thus we have the exact parametrisation of the cycloid and can now compute the level at which the particle will hit the line $x = b$ by

$$y(\pi) = \frac{1}{2\,C^2}\;(1 - \cos\pi) = \frac{2\,b}{\pi}$$

$\diamondsuit$

3.2.2 Transverse deflection of a string

A perfectly flexible string (such as a violin string) is stretched tightly between two fixed points $x = 0$ and $x = L$. The tension is known to be S (units N). The string is pulled out of its horizontal position by a known vertical force F (units $\frac{\mathrm{N}}{\mathrm{m}}$). Its new position can be described by a function $y(x)$. Thus the string will be stretched and will finally have a total length of $L = \Delta L$. We assume that $\Delta L \ll L$. Due to this stretching the elastic energy in the string will increase by $S\,\Delta L$. To compute the increase in energy we have to find the length as a functional of the vertical displacement function $y(x)$. From calculus we (should) know

$$L(y) = \int_0^L \sqrt{1 + (y'(x))^2}\;dx \approx \int_0^L 1 + \frac{1}{2}\;(y'(x))^2\;dx$$

where we used the approximation $\sqrt{1+x} \approx 1 + x/2$ for $x \ll 1$. Thus the increase in elastic energy is given by

$$U_{elast}(y) = \frac{S}{2}\;\int_0^L (y'(x))^2\;dx$$

The external vertical force can be described to a potential energy of the form

$$U_{pot}(y) = -\int_0^L F(x)\cdot y(x)\;dx$$

Thus we arrive at at total energy

$$U(y) = \int_0^L \frac{S}{2}\,(y'(x))^2 - F(x)\cdot y(x)\;dx$$

The string will be at rest if this energy is minimal and we find a standard problem for the calculus of variations. The Euler Lagrange equation for this problem is

$$\begin{aligned}
\frac{d}{dx}\,f_{y'} &= f_y(x) \\
\frac{d}{dx}\;\big(S\cdot y'(x)\big) &= -F(x) \\
y''(x) &= \frac{-F(x)}{S}
\end{aligned}$$

This ordinary differential equation of order 2 can be solved with the boundary conditions $y(0) = y(L) = 0$.

As a simple example we consider the deflection of the string due to its weight. If the mass per meter is given by ρ then we have $F = -g\rho$ and the equation to be solved is

$$y''(x) = \frac{+g\rho}{S} \quad \text{with} \quad y(0) = y(L) = 0$$

The unique solution is

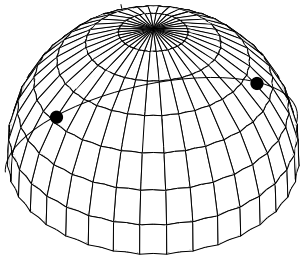
$$y(x) = \frac{g\rho}{2\,S}\;(x - L)\,x$$

thus the string is shaped like a parabola.

3.2.3 Geodesics on a sphere

On the right find the northern hemisphere (radius $R = 6300$ km) with the two cities Zürich and Salt Lake City shown. The geographic data of the towns is given below

	latitude $\frac{\pi}{2} - \theta$	longitude ϕ
Zürich	$47°$	$+8°$
Salt Lake City	$41°$	$-112°$



Airlines prefer to use the shortest connection possible and will fly north first, only to turn back south. We use calculus of variations to show that the shortest connection follows a great circle.

Find the Euler Lagrange equation

To achieve this we use spherical coordinates. The longitude equals the angle ϕ and the latitude is given by $\pi/2 - \theta$. We represent the connection from Salt Lake City to Zürich by writing the latitude as a function of the longitude, i.e. $\phi = v(\theta)$. To find the total length L we need to express the length element dl in terms of spherical coordinates. A few calculations lead to

$$\begin{aligned} d\phi &= v'(\theta)\, d\theta \\ dl^2 &= R^2 \left(\sin^2\theta\, d\phi^2 + d\theta^2\right) \\ &= R^2 \left(\sin^2\theta\, v'(\theta) + 1\right) d\theta^2 \\ dl &= R\sqrt{\sin^2\theta\, (v'(\theta))^2 + 1}\, d\theta \\ L &= R\int_{\theta_0}^{\theta_1} \sqrt{\sin^2\theta\, (v'(\theta))^2 + 1}\, d\theta \end{aligned}$$

Thus we have a variational problem with

$$\begin{aligned} f(\theta, v, v') &= R\sqrt{\sin^2\theta\, (v')^2 + 1} \\ \frac{\partial}{\partial v} f(\theta, v, v') &= 0 \\ \frac{\partial}{\partial v'} f(\theta, v, v') &= R\,\frac{\sin^2\theta\, v'}{\sqrt{\sin^2\theta\, (v')^2 + 1}} \end{aligned}$$

As the function f does not explicitly depend on the function v we have a first integral

$$\begin{aligned} f_{v'} &= C_1 \\ \frac{\sin^2\theta\, v'(\theta)}{\sqrt{\sin^2\theta\, (v'(\theta))^2 + 1}} &= \frac{C_1}{R} = C_2 \\ \sin^4\theta (v'(\theta))^2 &= C_2^2 \left(\sin^2\theta\, (v'(\theta))^2 + 1\right) \\ v'(\theta) &= \frac{\pm C_2}{\sqrt{\sin^4\theta - C_2^2\, \sin^2\theta}} \end{aligned}$$

Proof that all solutions are great circles

The above differential equation could be solved, but the computations are rather involved. Thus we resort to a slightly different argument to conclude that the optimal solution is along a great circle.

If we consider the special case of $v'(\theta_0) = 0$ for some $v(\theta_0) \neq 0$ then we can conclude that $C_2 = 0$ thus $v'(\theta) = 0$ for all θ. This solution is along a great circle with constant longitude. In general this will not be the case. Knowing the locations of the two towns we can introduce a new system of spherical coordinates, such that the two cities are on a great circle with the same 'longitude' in the new system. The connecting path is then a great circle. This does not depend on our choice of coordinate system. Thus the two cities are to be connected along a great circle.

The connection from Zürich to Salt Lake City

With the given data we have the locations of the two cities given by

$$\vec{Z} = R \begin{pmatrix} \cos 47^\circ \ \cos 8^\circ \\ \cos 47^\circ \ \sin 8^\circ \\ \sin 47^\circ \end{pmatrix} \approx \begin{pmatrix} 4254.78 \\ 597.97 \\ 4607.53 \end{pmatrix} km$$

and

$$\vec{S} = R \begin{pmatrix} \cos 41^\circ \ \cos 112^\circ \\ -\cos 41^\circ \ \sin 112^\circ \\ \sin 41^\circ \end{pmatrix} \approx \begin{pmatrix} -1781.13 \\ -4408.45 \\ 4133.17 \end{pmatrix} km$$

The distance along a straight line, cutting the interior of the earth is given by

$$d_1 = \|\vec{Z} - \vec{S}\| \approx 7856.3 \text{ km}$$

To find the distance between the cities along the surface we have to compute the angle between the two vectors

$$\cos\alpha = \frac{\langle \vec{Z}, \vec{S} \rangle}{\|\vec{Z}\| \ \|\vec{S}\|} \approx 0.222 \Longrightarrow \alpha \approx 1.346 \approx 77.15^\circ$$

and then conclude

$$d_2 = R\,\alpha \approx 8482.72 \text{ km}$$

The distance along the surface is slightly larger, as it should be. The vector $\vec{n}$ normal to $\vec{Z}$ and $\vec{S}$ is a normal vector to the plane supporting the great circle. It can be normalised.

$$\vec{n}_1 = \vec{S} \times \vec{Z} \approx \begin{pmatrix} -2.35126 \\ 2.66176 \\ 1.8258 \end{pmatrix} 10^7 \quad \text{normalises to} \quad \vec{n}_2 = \begin{pmatrix} -0.588792 \\ 0.666546 \\ 0.457209 \end{pmatrix}$$

The angle between the z axis and $\vec{n}_2$ indicates the maximal latitude along the connection and is given by

$$\arccos 0.457 \approx 62.8^\circ$$

This shows why flights from Zürich to Salt Lake City first go north and then turn south again.

3.3 Hamilton's principle of least action

Some of the notes in this section are adapted from [Wein74, p. 72]

We consider a system of particles subject to given geometric constraints and otherwise influenced by forces which are functions only of the positions of the particles. In addition we require the system to be conservative, i.e. the forces can be written as the gradient of a **potential energy** V of the system. We denote the n **degrees of freedom** of the system with $\vec{q} = (q_1, q_2, \ldots, q_n)^T$. The **kinetic energy** T of the system is

the extension of the basic formula $E = \frac{1}{2}\, m\, v^2$. With those we form the **Lagrange function** L of the system by

$$L(\vec{q}, \dot{\vec{q}}) = T(\vec{q}, \dot{\vec{q}}) - V(\vec{q})$$

The fundamental principle of Hamilton now reads:

> The actual motion of a system with the above Lagrangian L is such as to render the (Hamilton's) integral
>
> $$I = \int_{t_1}^{t_2} (T - V)\; dt = \int_{t_1}^{t_2} L(\vec{q}, \dot{\vec{q}})\; dt$$
>
> an extremum with respect to all twice differentiable functions $\vec{q}(t)$. Here t_1 and t_2 are arbitrary times.

This is a situation where we (usually) have multiple dependent variables and thus the Euler Lagrange equations imply

$$\frac{d}{dt}\frac{\partial\, L}{\partial \dot{q}_i} = \frac{\partial\, L}{\partial q_i} \quad \text{for} \quad i = 1, 2, \ldots, n$$

These differential equations apply to many mechanical setups, as the subsequent examples show.

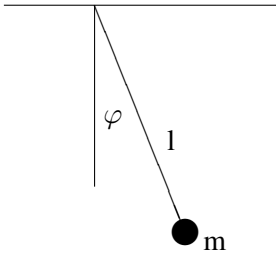
3.3.1 A simple pendulum

For a simple pendulum of length l we have

$$T(\varphi, \dot{\varphi}) = \frac{1}{2}\, m\, l^2\, (\dot{\varphi})^2 \quad \text{and} \quad V(\varphi) = -m\, l\, g\, \cos\varphi$$

and thus the Lagrange function

$$L = T - V = \frac{1}{2}\, m\, l^2\, (\dot{\varphi})^2 + m\, l\, g\, \cos\varphi$$



The only degree of freedom is $q_1 = \varphi$ and the functional to be minimised is

$$\int_a^b \frac{1}{2}\, m\, l^2\, (\dot{\varphi})^2 + m\, l\, g\, \cos\varphi\; d\varphi$$

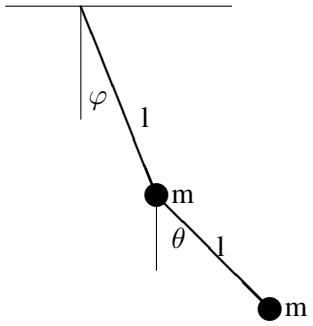
The Euler Lagrange equation leads to

$$\begin{aligned} \frac{d}{dt}\frac{\partial\, L}{\partial \dot{\varphi}} &= \frac{\partial\, L}{\partial \varphi} \\ \frac{d}{dt} m\, l^2\, \dot{\varphi} &= -m\, l\, g\, \sin\varphi \\ \ddot{\varphi} &= -\frac{g}{l}\, \sin\varphi \end{aligned}$$

This is the well known differential equation describing a pendulum. One can certainly derive the same equation using Newton's law.

3.3.2 A double pendulum

The calculations for this problem given in many books on classical mechanics, e.g. [Gree77]. A double pendulum consists of two particles with mass m suspended by massless rods of length l. Assuming that all takes place in a vertical plane we have two degrees of freedom: the two angles φ and θ. The potential energy is not too hard to find as



$$V(\varphi,\theta) = -m\,l\,g\,(2\,\cos\varphi + \cos\theta)$$

The velocity of the upper particle is

$$v_1 = l\,\dot\varphi$$

To find the kinetic energy we need the velocity of the lower particle. The velocity vector is equal to the vector sum of the velocity of the upper particle and the velocity of the lower particle relative to the upper particle. Since the two vectors differ in direction by an angle of $\varphi - \theta$ we can use the law of cosine to find the absolute velocity as[1]

$$v_2 = l\,\sqrt{\dot\varphi^2 + \dot\theta^2 + 2\,\dot\varphi\,\dot\theta\,\cos(\varphi-\theta)}$$

Thus the total kinetic energy is

$$T(\dot\varphi,\dot\theta) = \frac{m\,l^2}{2}\,\left(2\,\dot\varphi^2 + \dot\theta^2 + 2\,\dot\varphi\,\dot\theta\,\cos(\varphi-\theta)\right)$$

and the Lagrange function is

$$L = T - V = \frac{m\,l^2}{2}\,\left(2\,\dot\varphi^2 + \dot\theta^2 + 2\,\dot\varphi\,\dot\theta\,\cos(\varphi-\theta)\right) + m\,l\,g\,(2\,\cos\varphi + \cos\theta)$$

The Lagrange equation for the free variable φ is obtained from

$$\begin{aligned}
\frac{\partial\,L}{\partial\dot\varphi} &= m\,l^2\,\left(2\,\dot\varphi + \dot\theta\,\cos(\varphi-\theta)\right)\\
\frac{d}{dt}\frac{\partial\,L}{\partial\dot\varphi} &= m\,l^2\,\left(2\,\ddot\varphi + \ddot\theta\,\cos(\varphi-\theta) - \dot\theta\,(\dot\varphi-\dot\theta)\,\sin(\varphi-\theta)\right)\\
\frac{\partial\,L}{\partial\varphi} &= -m\,l^2\,\dot\varphi\,\dot\theta\,\sin(\varphi-\theta) - m\,l\,g\,2\,\sin\varphi
\end{aligned}$$

which, upon substitution into the Euler Lagrange equation, yields

$$m\,l^2\,\left(2\,\ddot\varphi + \ddot\theta\,\cos(\varphi-\theta) + \dot\theta^2\,\sin(\varphi-\theta)\right) = -m\,l\,g\,2\,\sin\varphi$$

In a similar fashion the θ equation is obtained from

$$\begin{aligned}
\frac{\partial\,L}{\partial\dot\theta} &= m\,l^2\,\left(\dot\theta + \dot\varphi\,\cos(\varphi-\theta)\right)\\
\frac{d}{dt}\frac{\partial\,L}{\partial\dot\theta} &= m\,l^2\,\left(\ddot\theta + \ddot\varphi\,\cos(\varphi-\theta) - \dot\varphi\,(\dot\varphi-\dot\theta)\,\sin(\varphi-\theta)\right)\\
\frac{\partial\,L}{\partial\theta} &= -m\,l^2\,\dot\varphi\,\dot\theta\,\sin(\varphi-\theta) - m\,l\,g\,\sin\theta
\end{aligned}$$

yielding

$$m\,l^2\,\left(\ddot\theta + \ddot\varphi\,\cos(\varphi-\theta) - \dot\varphi^2\,\sin(\varphi-\theta)\right) = -m\,l\,g\,\sin\theta$$

[1] Another approach is to use cartesian coordinates $x(\varphi,\theta) = l\,(\sin\varphi + \sin\theta)$, $y(\varphi,\theta) = -l\,(\cos\varphi + \cos\theta)$ and a few calculations

Those two equations can be divided by $m\,l^2$ and then lead to a system of ordinary differential equations of order 2.

$$\begin{aligned} 2\,\ddot{\varphi} + \ddot{\theta}\,\cos(\varphi-\theta) + \dot{\theta}^2\,\sin(\varphi-\theta) &= -\frac{g}{l}\,2\,\sin\varphi \\ \ddot{\theta} + \ddot{\varphi}\,\cos(\varphi-\theta) - \dot{\varphi}^2\,\sin(\varphi-\theta) &= -\frac{g}{l}\,\sin\theta \end{aligned}$$

By isolating the second order terms on the left we arrive at

$$\begin{bmatrix} 2 & \cos(\varphi-\theta) \\ \cos(\varphi-\theta) & 1 \end{bmatrix} \begin{pmatrix} \ddot{\varphi} \\ \ddot{\theta} \end{pmatrix} = \sin(\varphi-\theta) \begin{pmatrix} -\dot{\theta}^2 \\ \dot{\varphi}^2 \end{pmatrix} - \frac{g}{l} \begin{pmatrix} 2\,\sin\varphi \\ \sin\theta \end{pmatrix}$$

The matrix on the left hand side is always invertible and thus this differential equation can reliably be solved by numerical procedures.

If we assume that all angles and velocities are small we may use the approximations $\cos(\varphi-\theta) \approx 1$ and $\sin x \approx x$ to obtain the **linearised** differential equation

$$\begin{bmatrix} 2 & 1 \\ 1 & 1 \end{bmatrix} \begin{pmatrix} \ddot{\varphi} \\ \ddot{\theta} \end{pmatrix} = -\frac{g}{l} \begin{pmatrix} 2\,\varphi \\ \theta \end{pmatrix}$$

Solving for the highest order derivative we obtain

$$\begin{pmatrix} \ddot{\varphi} \\ \ddot{\theta} \end{pmatrix} = -\frac{g}{l} \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix} \begin{pmatrix} 2\,\varphi \\ \theta \end{pmatrix} = -\frac{g}{l} \begin{bmatrix} 2 & -1 \\ -2 & 2 \end{bmatrix} \begin{pmatrix} \varphi \\ \theta \end{pmatrix}$$

This linear system of equations could be solved explicitly using eigenvalues and eigenvectors. The solution will be valid for small angles and velocities.

3.3.3 A pendulum with moving support

A chariot of mass m_1 with an attached pendulum of length l and mass m_2 is moving freely. The situation is shown in figure 3.4. In this example the independent variable is time t and the two general coordinates are x and θ, i.e.

$$\vec{u} = \begin{pmatrix} x \\ \theta \end{pmatrix}$$

and potential and kinetic energy are given by

$$\begin{aligned} V(x,\theta) &= -m_2\,l\,g\,\cos\theta \\ T(x,\theta,\dot{x},\dot{\theta}) &= \frac{m_1}{2}\,\dot{x}^2 + \frac{m_2}{2}\,\left((\dot{x}+l\,\cos\theta\,\dot{\theta})^2 + (l\,\sin\theta\,\dot{\theta})^2\right) \\ &= \frac{m_1}{2}\,\dot{x}^2 + \frac{m_2}{2}\,\left(\dot{x}^2 + 2\,l\,\dot{x}\,\cos\theta\,\dot{\theta} + l^2\,\dot{\theta}^2\right) \end{aligned}$$

Thus for the Lagrange function $L = T - V$ we have two Euler Lagrange equations. The first equation deals with the dependence on the function $x(t)$.

$$\begin{aligned} \frac{d}{dt}\,L_{\dot{x}}(x,\theta,\dot{x},\dot{\theta}) &= L_x(x,\theta,\dot{x},\dot{\theta}) \\ \frac{d}{dt}\,\left((m_1+m_2)\,\dot{x} + m_2\,l\,\cos\theta\,\dot{\theta}\right) &= 0 \end{aligned}$$

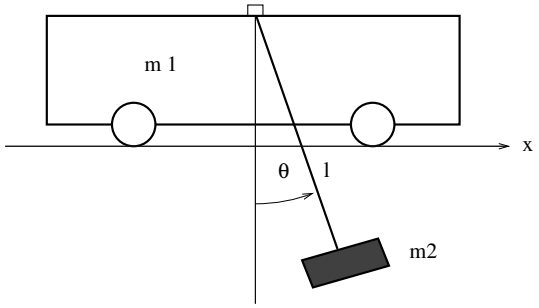


Figure 3.4: Pendulum with moving support

From this we can conclude that the momentum in x direction is conserved. The first equation deals with the dependence on the function $\theta(t)$.

$$\begin{aligned}
\frac{d}{dt}\, L_{\dot\theta}(x,\theta,\dot x,\dot\theta) &= L_\theta(x,\theta,\dot x,\dot\theta)\\
m_2\,\frac{d}{dt}\,\Big(l\,\dot x\,\cos\theta + l^2\,\dot\theta\Big) &= -m_2\,l\,\dot x\,\dot\theta\,\sin\theta - m_2\,l\,g\,\sin\theta\\
\frac{d}{dt}\,\Big(\dot x\,\cos\theta + l\,\dot\theta\Big) &= -\dot x\,\dot\theta\,\sin\theta - g\,\sin\theta\\
\ddot x\,\cos\theta - \dot x\,\dot\theta\,\sin\theta + l\,\ddot\theta &= -\dot x\,\dot\theta\,\sin\theta - g\,\sin\theta\\
\ddot x\,\cos\theta + l\,\ddot\theta &= -g\,\sin\theta
\end{aligned}$$

This is a second order differential equation for the functions $x(t)$ and $\theta(t)$. The two equations can be combined and we arrive at the system

$$\begin{aligned}
(m_1+m_2)\,\ddot x + m_2\,l\,\cos\theta\,\ddot\theta &= m_2\,l\,\sin\theta\,(\dot\theta)^2\\
\ddot x\,\cos\theta + l\,\ddot\theta &= -g\,\sin\theta
\end{aligned}$$

With the help of a matrix the system can be solved for the highest occurring derivatives. A simple computation shows that the determinant of the matrix does not vanish and thus we can always find an inverse matrix.

$$\frac{d^2}{dt^2}\begin{pmatrix} x\\ \theta\end{pmatrix} = \begin{bmatrix} m_1+m_2 & m_2\,l\,\cos\theta\\ \cos\theta & l\end{bmatrix}^{-1}\begin{pmatrix} m_2\,l\,\sin\theta\,(\dot\theta)^2\\ -g\,\sin\theta\end{pmatrix}$$

This is a very convenient form to produce numerical solutions for the problem at hand.

The above model does not consider friction. Now we want to include some friction on the moving chariot. This is not elementary, as the potential V can not depend on the velocity $\dot x$, but there is a trick to be used.

1. Introduce a constant force F applied to the chariot to the system. This is done by modifying the potential V accordingly.
2. Find the corresponding differential equations.
3. Set the force $F = -\alpha\dot x$

To take the additional force F into account we have to modify the potential energy

$$V(x,\theta) = -m_2\,l\,g\,\cos\theta - x\cdot F$$

The Euler Lagrange equation for the variable θ will not be affected by this change, but the equation for x turns out to be

$$\begin{aligned} \frac{d}{dt}\left((m_1+m_2)\,\dot{x}+m_2\,l\,\cos\theta\,\dot{\theta}\right) &= F \\ (m_1+m_2)\,\ddot{x}+m_2\,l\,\cos\theta\,\ddot{\theta} &= m_2\,l\,\sin\theta\,(\dot{\theta})^2+F \end{aligned}$$

and the full system is now given by

$$\frac{d^2}{dt^2}\begin{pmatrix} x \\ \theta \end{pmatrix} = \begin{bmatrix} m_1+m_2 & m_2\,l\,\cos\theta \\ \cos\theta & l \end{bmatrix}^{-1} \begin{pmatrix} m_2\,l\,\sin\theta\,(\dot{\theta})^2+F \\ -g\,\sin\theta \end{pmatrix}$$

Now it is easy to replace F by $-\alpha\dot{x}$ and one obtains

$$\frac{d^2}{dt^2}\begin{pmatrix} x \\ \theta \end{pmatrix} = \begin{bmatrix} m_1+m_2 & m_2\,l\,\cos\theta \\ \cos\theta & l \end{bmatrix}^{-1} \begin{pmatrix} m_2\,l\,\sin\theta\,(\dot{\theta})^2-\alpha\,\dot{x} \\ -g\,\sin\theta \end{pmatrix}$$

This equation can be solved with *Mathematica*. First define the equations using the above matrix notation.

Mathematica

```
rhs:= Inverse[{{m1+m2,m2 l Cos[w[t]]},
              {Cos[w[t]],l}}].{m2 l Sin[w[t]](w'[t])^2- al x'[t],-g Sin[w[t]]};
eq:={x''[t]==rhs[[1]],w''[t]==rhs[[2]]}
```

Then choose the constants and initial conditions for the specific setup and compute the numerical approximation to the solution.

Mathematica

```
g=9.81;m1=1;m2=8; l =1;al=0.5;
initval={x[0]==0 ,x'[0]==0,w[0]==Pi/6,w'[0]==0};
nsol=NDSolve[Flatten[{eq,initval}],{x[t],w[t]},{t,0,10},MaxSteps -> 2000]
```

Define the functions to be plotted and generate an array of plots for the solution (see figure 3.5).

Mathematica

```
xn[t_] = x[t]/.nsol;
wn[t_] = w[t]/.nsol;
g1 = Plot[xn[t],{t,0,10},AxesLabel->{"t","x"},DisplayFunction->Identity];
g2 = Plot[wn[t],{t,0,10},AxesLabel->{"t","angle"},DisplayFunction->Identity];
Show[GraphicsArray[{g1,g2}],DisplayFunction ->$DisplayFunction];
```

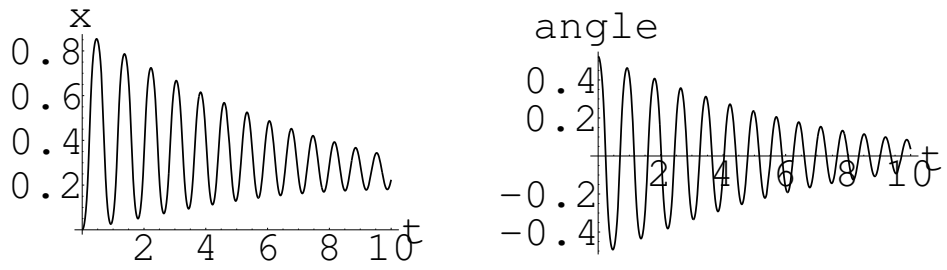


Figure 3.5: Numerical solution for a pendulum with moving support

3.4 An isoperimetric problem

If for function of multiple variable we want to find an extremum of the function $f(\vec{x})$ subject to to the constraint $g(\vec{x}) = c$ then we have to use a **Lagrange multiplier**, i.e. there is a number $\lambda \in \mathbb{R}$ such that

$$\begin{array}{l}\vec{x} \text{ critical point of } f(\vec{x}) \\ \text{subject to } g(\vec{x}) = c\end{array} \qquad \Longrightarrow \qquad \vec{\nabla} f(\vec{x}) + \lambda \, \vec{\nabla} g(x) = \vec{0}$$

The idea and technique of Lagrange multipliers can be applied to functionals too. We will not go into details here but consider the problem of maximum enclosed area by a curve of given length.

If a function $u(x)$ is a critical function for the functional

$$F(u) = \int_a^b f(x, u, u') \, dx \quad \text{subject to} \quad G(u) = \int_a^b g(x, u, u') \, dx = c$$

then there is a multiplier $\lambda \in \mathbb{R}$ such that $u(x)$ is a critical function for

$$H(u) = F(u) + \lambda \, G(u) = \int_a^b f(x, u, u') + \lambda \, g(x, u, u') \, dx$$

Thus the corresponding Euler Lagrange equation is

$$\begin{aligned} \frac{d}{dx} \, h_{u'} &= h_u \\ \frac{d}{dx} \, f_{u'} + \lambda \, \frac{d}{dx} \, g_{u'} &= f_u + \lambda \, g_u \end{aligned}$$

As an example we search the curve in $\mathbb{R}^2$ of given length L enclosing the largest area possible. We examine parametrised curves

$$\vec{u}(t) = \begin{pmatrix} x(t) \\ y(t) \end{pmatrix} \quad \text{for} \quad a \leq t \leq b$$

This problem has two dependent functions $x(t)$ and $y(t)$. The total length of the curve is given by

$$G(\vec{u}) = \int_a^b \sqrt{\dot{x}^2 + \dot{y}^2} \, dt = L$$

The total area can be computed by

$$F(\vec{u}) = \frac{1}{2} \int_a^b x \, \dot{y} - y \, \dot{x} \, dt$$

Using the method of Lagrange multipliers we have to examine the functional

$$\int_a^b \frac{1}{2} \, (x \, \dot{y} - y \, \dot{x}) + \lambda \sqrt{\dot{x}^2 + \dot{y}^2} \, dt$$

and are lead to the Lagrange equations

$$\frac{d}{dt} \left(\frac{-y}{2} + \lambda \, \frac{\dot{x}}{\sqrt{\dot{x}^2 + \dot{y}^2}} \right) = \frac{\dot{y}}{2} \quad \text{and} \quad \frac{d}{dt} \left(\frac{x}{2} + \lambda \, \frac{\dot{y}}{\sqrt{\dot{x}^2 + \dot{y}^2}} \right) = \frac{-\dot{x}}{2}$$

Those equations can directly be integrated with respect to t and we get

$$\lambda \, \frac{\dot{x}}{\sqrt{\dot{x}^2 + \dot{y}^2}} - y = -C_1 \quad \text{and} \quad \lambda \, \frac{\dot{y}}{\sqrt{\dot{x}^2 + \dot{y}^2}} + x = C_2$$

or also

$$\lambda \frac{\dot{x}}{\sqrt{\dot{x}^2+\dot{y}^2}} = y - C_1 \quad \text{and} \quad -\lambda \frac{\dot{y}}{\sqrt{\dot{x}^2+\dot{y}^2}} = x - C_2$$

By squaring and adding we get

$$(x - C_2)^2 + (y - C_1)^2 = \lambda^2$$

which is the equation of a circle of radius $|\lambda|$ and center at $x = C_2$ and $y = C_1$.

The formula used to compute the total area requires that the curve has to be closed. There are other situations and in exercise 3–4 a similar situation is considered.

3.5 Laser beam deflected by a heat source

A solid with a flat, horizontal surface is heated, thus there will be a temperature gradient above this surface. Then a laser beam will travel parallel to the surface. As the speed of light v depends on the temperature , which decreases as we move further away from the surface. we consider the speed of light to be given by a function of the form

$$v(x, u) = v_0 + k\, u$$

where u is the distance above the surface, x the horizontal coordinate, v_0 the speed of light on a given level and the gradient of the speed of light is k. This can also be reformulated using the index of refraction n related to the speed by

$$n(x, u) = \frac{c}{v(x, u)} = \frac{c}{v_0 + k\, u}$$

where c is the speed of light in empty space.

3.5.1 Dependence of the speed of light on the temperature

In [CRC95, §10–302] find data for the index of refraction as a function of pressure, wavelength and the temperature. The index can be computed by

$$(n-1)\cdot 10^8 = 8342.13 + 2406030\,(130-\sigma^2)^{-1} + 15997\,(38.9-\sigma^2)^{-1}$$

where

$\lambda_{vac} = \frac{1}{\sigma}$	wavelength	μm
T	temperature	$^\circ C$
p	pressure	$\frac{\mathrm{N}}{\mathrm{m}^2}$

If the temperature is different from $15^\circ C$ and the pressure p deviates of $101.325\ kP$, then the value of $(n-1)$ in the formula above has to be multiplied with

$$\frac{p\,(1 + p\,(61.3 - t)\cdot 10^{-10})}{96095.4\,(1 + 0.003661\,t)}$$

Those formulas can be coded in *Mathematica*

Mathematica

```
n1[sigma_]:=1+ (8342.13+ 2406030*(130-sigma^2)^(-1)+
                15997*(38.9-sigma^2)^(-1) )*10^(-8)
nfactor[p_,T_] := (p (1+p*(61.3-T)* 10^(-10)))/(96095.4*(1+0.003661*T))
n[lambda_] := n1[1/lambda]
n[lambda_,T_,p_] := (n[lambda]-1)*nfactor[p,T]+1
```

Then the entry in the table in [CRC95] for a wavelength of $\lambda = 0.4\ \mu m$ can be verified by

Mathematica

```
(n[0.4]-1)*10^8
(n[0.4,15,101.325*10^3]-1)*10^8
.
28274.8
28274.8
```

Since we are interested in the gradient with respect to the temperature T now compute

Mathematica

```
nGradt[lambda_,T_,p_] =D[n[lambda,T,p],T];
nGradt[0.4,15,101.325*10^3]
.
-9.84117*10^(-7)
```

Thus if the temperature increases by 1° C then the index of refraction decreases by 10^{-6}. This leads to

$$\begin{aligned} \Delta n &\approx -10^{-6}\,\Delta T \\ \Delta v &= \frac{c}{n+\Delta n} - \frac{c}{n} \approx -c\,\Delta n \approx 10^{-6}\,c\,\Delta T \end{aligned}$$

If the rate of change of the temperature is known this will alow to find the coefficient k in $v(v,u) = v_0 + k\,u$.

3.5.2 Find the time of travel

For a given function $u(x)$ the time of travel T can be found. **Fermat's principle** implies that the time T required to travel from $x = a$ to $x = b$ will be a critical value. The time needed to travel from x to $x + \Delta x$ is approximately given by

$$\Delta T = \frac{\text{distance}}{\text{speed}} \approx \frac{\sqrt{1 + (u'(x))^2}}{v\,(x,u)}$$

Thus for the total time we consider the functional

$$T\,(u) = \int_a^b \frac{\sqrt{1 + (u'(x))^2}}{v\,(x,u)}\,dx$$

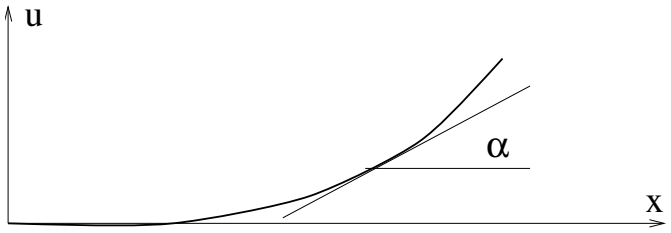


Figure 3.6: Laser beam in a medium with variable refraction index

3.5.3 Solution using a first integral

We have to minimise a functional based on the function

$$f\,(u,u') = \frac{\sqrt{1 + (u')^2}}{v_0 + k\,u}$$

Since the function f is explicitly independent on x we have a first integral (see Result 3–11).

$$\begin{aligned} u'\, f_{u'} - f &= -C_1 \\ u' \, \frac{u'}{\sqrt{1+(u')^2}\,(v_0+k\,u)} - \frac{\sqrt{1+(u')^2}}{v_0+k\,u} &= -C_1 \\ \frac{(u')^2-(1+(u')^2)}{\sqrt{1+(u')^2}\,(v_0+k\,u)} &= -C_1 \\ \frac{1}{\sqrt{1+(u')^2}\,(v_0+k\,u)} &= C_1 \end{aligned}$$

The above leads to a nonlinear differential equation of first order.

$$(u')^2 = -1 + \frac{1}{C_1^2\,(v_0+k\,u)^2} = \frac{1-C_1^2\,(v_0+k\,u)^2}{C_1^2\,(v_0+k\,u)^2}$$

By a separation of variables we obtain

$$\begin{aligned} \sqrt{\frac{C_1^2\,(v_0+k\,u)^2}{1-C_1^2\,(v_0+k\,u)^2}}\,du &= dx \\ \int \frac{\pm C_1\,(v_0+k\,u)}{\sqrt{1-C_1^2\,(v_0+k\,u)^2}}\,du &= \int dx \\ \pm\frac{1}{k\,C_1}\sqrt{1-C_1^2\,(v_0+k\,u)^2} &= x+C_2 \\ C_1^2\,(v_0+k\,u)^2 &= 1-k^2C_1^2\,(x+C_2)^2 \\ (v_0+k\,u)^2 &= \frac{1}{C_1^2}-k^2\,(x+C_2)^2 \\ k\,u &= -v_0+\sqrt{\frac{1}{C_1^2}-k^2\,(x+C_2)^2} \end{aligned}$$

This integration will fail if the denominator vanishes which is equivalent to $u'(x)=0$. This indicates that for horizontal beams this approach is not suitable.

Using the initial conditions $u(0)=0$ and $u'(0)=u_1$ we can determine the constants C_1 and C_2. First set $x=0$, then differentiate with respect to to x and set $x=0$. We conclude

$$\begin{aligned} 0 &= -v_0+\sqrt{\frac{1}{C_1^2}-k^2C_2^2} \\ k\,u_1 &= \frac{-k^2C_2}{\sqrt{1/C_1^2-k^2(0+C_2)^2}} = \frac{-k^2C_2}{v_0} \end{aligned}$$

From this we arrive at

$$C_2 = -\frac{v_0\,u_1}{k} \quad \text{and} \quad C_1^2 = \frac{1}{v_0^2+k^2C_2^2} = \frac{1}{v_0^2(1+u_1^2)}$$

Now we finally obtain

$$\begin{aligned} u(x) &= \frac{1}{k}\left(-v_0+\sqrt{v_0^2(1+u_1^2)-k^2\,(x-\frac{v_0\,u_1}{k})^2}\right) \\ &= \frac{1}{k}\left(-v_0+\sqrt{v_0^2+v_0^2u_1^2-k^2x^2+2\,x\,v_0\,k\,u_1-v_0^2u_1^2}\right) \\ &= \frac{1}{k}\left(-v_0+\sqrt{v_0^2+2\,x\,v_0\,k\,u_1-k^2x^2}\right) \end{aligned}$$

Situation with vanishing initial slope

In the particular situation $u_1 = u'(0) = 0$ we have

$$u(x) = \frac{1}{k}\left(-v_0 + \sqrt{v_0^2 - k^2 x^2}\right) = \frac{v_0}{k}\left(-1 + \sqrt{1 - \frac{k^2}{v_0^2}\,x^2}\right) \approx -\frac{v_0}{k}\,\frac{k^2}{2\,v_0^2}\,x^2 = -\frac{k}{2\,v_0}\,x^2$$

Thus the laser beam will take the shape of a parabola, at least for values close to the highest/lowest point of the beam. But this is the „dangerous region“ for the separation of variables above. But as soon as we are away from the extremal point the solution should be valid.

The exact solution

Considering the intermediate results in the calculation for the separation of variables we see that the solution passing through $u(0) = 0$ satisfies the following equations.

$$\begin{aligned}
(v_0 + k\,u)^2 &= \frac{1}{C_1^2} - k^2\,(x + C_2)^2 \\
(v_0 + k\,u)^2 &= v_0^2(1 + u_1^2) - k^2\,(x - \frac{v_0 u_1}{k})^2 \\
&= v_0^2(1 + u_1^2) - (k\,x - v_0 u_1)^2 \\
(v_0 + k\,u)^2 + (k\,x - v_0 u_1)^2 &= v_0^2(1 + u_1^2) \\
(u + \frac{v_0}{k})^2 + (x - \frac{v_0 u_1}{k})^2 &= \frac{v_0^2(1 + u_1^2)}{k^2}
\end{aligned}$$

This is the equation of a circle in the xu plane with center at $x = \frac{v_0 u_1}{k}$ and $u = -\frac{v_0}{k}$. The radius of the circle is $\frac{v_0}{k}\sqrt{1 + u_1^2}$. The points with vertical slope are at the level $u = -\frac{v_0}{k}$ and due to $v(x,u) = v_0^k\,u$ the speed would be zero at those points.

3.5.4 Solution using an approximation

Since the expressions will be rather lengthy we assume $v(x,u) = v_0 + k\,u$ and use the approximation (valid if $|u'| \ll 1$)

$$\sqrt{1 + (u')^2} \approx 1 + \frac{1}{2}\,(u')^2$$

Now we use the Euler Lagrange equation from Theorem 3–7. This leads to

$$\begin{aligned}
f\,(x,u,u') &= \frac{1 + \frac{1}{2}\,(u')^2}{v_0 + k\,u} \\
\frac{d}{dx}\,f_{u'}(x,u,u') &= f_u(x,u) \\
\frac{d}{dx}\,\frac{u'}{v_0 + k\,u} &= \frac{-k\,(1 + \frac{1}{2}\,(u')^2)}{(v_0 + k\,u)^2} \\
\frac{u''}{v_0 + k\,u} - \frac{k\,(u')^2}{(v_0 + k\,u)^2} &= \frac{-k\,(1 + \frac{1}{2}\,(u')^2)}{(v_0 + k\,u)^2} \\
u'' &= \frac{-k + \frac{k}{2}\,(u')^2)}{(v_0 + k\,u)} = -\frac{k}{(v_0 + k\,u)} + \frac{k}{2\,(v_0 + k\,u)}\,(u')^2 \\
u'' &\approx \frac{-k}{v_0 + k\,u} \quad \text{if} \quad (u')^2 \ll 1
\end{aligned}$$

Thus we obtain a very simple formula. If we use in addition $k\,u \ll v_0$, then we find the following for the height $u(x)$ of the laser beam.

$$u''(x) \approx \frac{-k}{v_0}$$

But this formula should yield sufficiently precise results in many situations.

If the gradient k of the speed of light is known as a function of x then an integration will lead to the total change of the angle as the ray of light crosses the area. We find with the obvious definition of $K(x)$

$$\begin{aligned} u'(x) &= u'(0) + \int_0^x u''(z)\,dz = u'(0) - \int_0^x \frac{k(z)}{v_0}\,dz = u(0) - \frac{1}{v_0}\,K(x) \\ u(x) &= u(0) + \int_0^x u'(z)\,dz = u(0) + x\,u'(0) + \frac{1}{v_0}\int_0^x K(z)\,dz \end{aligned}$$

For a given experimental setup the last two formulas may help to analyse the structure of the temperature gradient.

3.6 Bending of a circular plate

The goal of this section is to compute the bending of a circular plate under constant pressure. Knowing the deflection at the midpoint one can find the applied pressure. The integral to be minimised is different from the previous examples, it does contain second order derivatives of the unknown function. We have to derive the Euler Lagrange equation for this type of problem.

3.6.1 Energy of bending

The energy of bending bar with cross section A, thickness h and moment of inertia I is given by

$$U = \int_0^L \frac{E\,I}{2}\,(u''(x))^2\,dx$$

This is correct if the displacement depends on x only. If the displacement depends on x and y then further effects have to be taken into account. The energy of a plate described by the vertical displacement $z = u(x,y)$ for $(x,y) \in \Omega$ with thickness h is given by

$$U = \frac{h^3\,E}{24\,(1-\sigma^2)} \iint_\Omega (u_{xx}+u_{yy})^2 - 2\,(1-\sigma)\,(u_{xx}\,u_{yy} - u_{xy}^2)\,dx\,dy$$

where E is **Young's modulus** and σ is **Poisson's ratio**. More information can be found in [Wein74, §10] or [LandLifs75].

3.6.2 Using polar coordinates

Using the standard notation Δ for the **Laplace operator** we have in cartesian and polar coordinates Δu by

$$u_{xx} + u_{yy} = \Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \frac{1}{r}\,\frac{\partial}{\partial r}\left(r\,\frac{\partial u}{\partial r}\right) + \frac{1}{r^2}\,\frac{\partial^2 u}{\partial \phi^2}$$

For a circular plate $0 \le r \le R$ with the deflection depending on r only we find

$$F(u) \quad = \quad = \frac{E\,h^3}{24\,(1-\sigma^2)} \int_0^R \left(\frac{1}{r}\,(r\,u')'\right)^2\,2\,\pi\,r\,dr = \frac{\pi\,E\,h^3}{12\,(1-\sigma^2)} \int_0^R \frac{1}{r}\left((r\,u')'\right)^2\,dr$$

Set $k = \frac{\pi\, E\, h^3}{6\,(1-\sigma^2)}$ and choose a fixed test function $\phi(r)$ and a parameter ε then we find

$$\begin{aligned} F(u+\varepsilon\phi) &= \frac{1}{2}\int_0^R \frac{k}{r}\,\Big(\big(r\,u' + \varepsilon\, r\,\phi'\big)'\Big)^2\,\,dr \\ \frac{d}{d\varepsilon}\,F(u+\varepsilon\phi) &= \int_0^R \frac{k}{r}\,\Big(\big(r\,u' + \varepsilon\, r\,\phi'\big)'\Big)\cdot\big(r\,\phi'\big)'\,\,dr \\ \frac{d}{d\varepsilon}\,F(u+\varepsilon\phi)\Big|_{\varepsilon=0} &= \int_0^R \frac{k}{r}\,(r\,u')'\cdot\big(r\,\phi'\big)'\,\,dr \end{aligned}$$

Now we can perform two integrations by parts to move all derivatives from the test function $\phi(r)$ to the function $u(r)$

$$\begin{aligned} \frac{d}{d\varepsilon}\,F(u+\varepsilon\phi)\Big|_{\varepsilon=0} &= \int_0^R \frac{k}{r}\,(r\,u')'\cdot\big(r\,\phi'\big)'\,\,dr \\ &= \frac{k}{r}\,(r\,u')'\cdot\big(r\,\phi'\big)\Big|_{r=0}^{R} - \int_0^R \left(\frac{k}{r}\,(r\,u')'\right)'\cdot\big(r\,\phi'\big)\,\,dr \\ &= k\,(r\,u')'\cdot\phi' - r\left(\frac{k}{r}\,(r\,u')'\right)'\cdot\phi\,\Big|_{r=0}^{R} + \int_0^R \left(r\,\left(\frac{k}{r}\,(r\,u')'\right)'\right)'\cdot\phi\,\,dr \end{aligned}$$

Next we consider the second part of the elastic energy. The expression in question is (suppressing some constants)

$$P(u) = \iint_\Omega u_{xx}\,u_{yy} - u_{xy}^2\,\,dx\,dy$$

$$\begin{aligned} P(u) &= \iint_\Omega u_{xx}\,u_{yy} - u_{xy}^2\,\,dx\,dy \\ &= \frac{1}{2}\iint_\Omega \frac{\partial}{\partial x}\,\left(u_x\,u_{yy} - u_y\,u_{xy}\right) + \frac{\partial}{\partial y}\,\left(u_y\,u_{xx} - u_x\,u_{xy}\right)\,\,dx\,dy \\ &= \frac{1}{2}\oint_{\partial\Omega} \begin{pmatrix} u_x\,u_{yy} - u_y\,u_{xy} \\ u_y\,u_{xx} - u_x\,u_{xy} \end{pmatrix}\cdot\vec{n}\,\,ds \\ &= \frac{2\,\pi\,R}{2}\,u'(R)\,\frac{u'(R)}{R} = \pi\,u'(R)^2 \\ P(u+\phi) - P(u) &\approx 2\,\pi\,u'(R)\cdot\phi'(R) \\ -\frac{h^3\,E\,2\,(1-\sigma)}{24\,(1-\sigma^2)}\,\left(P(u+\phi) - P(u)\right) &\approx -k\,(1-\sigma)\,u'(R)\cdot\phi'(R) \end{aligned}$$

For small perturbations ϕ we find

$$\begin{aligned} P(u+\phi) - P(u) &\approx \iint_\Omega \phi_{xx}\,u_{yy} + u_{xx}\,\phi_{yy} - 2\,u_{xy}\,\phi_{xy}\,\,dx\,dy \\ &= \iint_\Omega \frac{\partial}{\partial x}\,\left(\phi_x\,u_{yy} - \phi_y\,u_{xy}\right) + \frac{\partial}{\partial y}\,\left(\phi_y\,u_{xx} - \phi_x\,u_{xy}\right)\,\,dx\,dy \\ &= \oint_{\partial\Omega} \begin{pmatrix} \phi_x\,u_{yy} - \phi_y\,u_{xy} \\ \phi_y\,u_{xx} - \phi_x\,u_{xy} \end{pmatrix}\cdot\vec{n}\,\,ds \end{aligned}$$

For a circular plate the integrand has to be constant along the boundary $r = R$ and the functions u and ϕ depend on r only. The derivatives with respect to x and y are

$$\begin{aligned}
\frac{d}{dx}\,u(r) &= u'(r)\,\frac{x}{\sqrt{x^2+y^2}} \\
\frac{d}{dy}\,u(r) &= u'(r)\,\frac{y}{\sqrt{x^2+y^2}} \\
\frac{d^2}{dx^2}\,u(r) &= u''(r)\,\frac{x^2}{x^2+y^2} + u'(r)\,\frac{1}{\sqrt{x^2+y^2}} - u'(r)\,\frac{x^2}{\sqrt{x^2+y^2}^3} \\
\frac{d^2}{dy^2}\,u(r) &= u''(r)\,\frac{y^2}{x^2+y^2} + u'(r)\,\frac{1}{\sqrt{x^2+y^2}} - u'(r)\,\frac{y^2}{\sqrt{x^2+y^2}^3} \\
\frac{d^2}{dx\,dy}\,u(r) &= u''(r)\,\frac{x\,y}{x^2+y^2} - u'(r)\,\frac{x\,y}{\sqrt{x^2+y^2}^3}
\end{aligned}$$

At the point $x = R$ and $y = 0$ we find

$$\begin{pmatrix} \phi_x\,u_{yy} - \phi_y\,u_{xy} \\ \phi_y\,u_{xx} - \phi_x\,u_{xy} \end{pmatrix} \cdot \vec{n} = \phi_x\,u_{yy} - \phi_y\,u_{xy} = \phi'(R)\,\frac{u'(R)}{R}$$

As the situation is radially symmetric the contribution can not depend on the angle. Thus in the variation of the total energy we find a contribution of

$$-\frac{E\,h^3}{24\,(1-\sigma^2)}\,2\,(1-\sigma)\,\phi'(R)\,\frac{u'(R)}{R}\,2\,\pi\,R = -k\,(1-\sigma)\,u'(R)\cdot\phi'(R)$$

where $k = \frac{\pi\,E\,h^3}{6\,(1-\sigma^2)}$ is the same constant as above. Observe that this term does not lead to any contribution in the integral, but only to boundary terms.

3.6.3 Energy due to external pressure and the Euler Lagrange equation

Sofar we considered the internal energy of bending only. The external pressure p applied to the upper side of the plate will lead to an additional energy term

$$F_e(u) = \int_0^R p\,2\,\pi\,r\cdot u\,dr$$

By computations much simpler than the above we obtain an additional term to that variation

$$\int_0^R p\,2\,\pi\,r\cdot\phi\,dr$$

The total variation can be described by

$$\begin{aligned}
\frac{d}{d\varepsilon}\,U(u+\varepsilon\phi)\Big|_{\varepsilon=0} &= \left(k\,(r\,u')'\cdot\phi' - r\left(\frac{k}{r}\,(r\,u')'\right)'\cdot\phi\right)\Big|_{r=0}^{R} - 2\,k\,(1-\sigma)\,u'(R)\cdot\phi'(R) \\
&\quad + \int_0^R \left(r\left(\frac{k}{r}\,(r\,u')'\right)'\right)'\cdot\phi + p\,2\,\pi\,r\cdot\phi\,dr
\end{aligned}$$

This expression has to vanish for "all" test function $\phi(r)$. Using the fundamental lemma 3–2 we conclude that the expression under the integral has to vanish. Since the expressions are not overly simple we compute

some of the derivatives.

$$\begin{aligned} r\left(\frac{k}{r}\,(r\,u')'\right)' &= r\left(\frac{k}{r}\,(u'+r\,u'')\right)' = r\left(\frac{k\,u'}{r}+k\,u''\right)' \\ &= r\left(\frac{k\,u''}{r}-\frac{k\,u'}{r^2}+k\,u'''\right) = k\,u''-\frac{k\,u'}{r}+k\,r\,u''' \\ \left(r\left(\frac{k}{r}\,(r\,u')'\right)'\right)' &= \frac{k\,u'}{r^2}-\frac{k\,u''}{r}+2\,k\,u'''+k\,r\,u^{(4)} \end{aligned}$$

Thus the differential equation to be solved is

$$\frac{\pi\,E\,h^3}{6\,(1-\sigma^2)}\left(\frac{u'}{r^2}-\frac{u''}{r}+2\,u'''+r\,u^{(4)}\right) = -2\,\pi\,r\,p$$

This can be rewritten as

$$r^3\,u^{(4)}+2\,r^2\,u'''-r\,u''+u' = -\frac{12\,(1-\sigma^2)}{E\,h^3}\,r^3\,p$$

This is a linear differential equation with non constant coefficients. Its general solution is given as sum of the general solution of the homogeneous problem and one particular solution. Standard methods will lead to the result

$$u(r) = c_1 + c_2\ \ln r + c_3\,r^2 + c_4\,r^2\,\ln r - \frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3}\,r^4$$

As the deflection (and its derivatives) can not have singularities at $r = 0$ we conclude $c_2 = c_4 = 0$. By choosing $u(0) = 0$ we find $c_1 = 0$ and thus the solution is

$$u(r) = c_3\,r^2 - \frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3}\,r^4$$

The constant c_3 has to be determined by another boundary condition. We carry out those computations for two special cases.

3.6.4 Clamped edge at $r = R$

If we require $u'(R) = 0$ then

$$u'(R) = c_3\,2\,R - \frac{4\cdot 12\,p\,(1-\sigma^2)}{64\,E\,h^3}\,R^3 = 0 \qquad \Longrightarrow \qquad c_3 = 2\,\frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3}\,R^2$$

The maximal deflection will be at $r = R$ and we have

$$u(R) = 2\,\frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3}\,R^2\,R^2 - \frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3}\,R^4 = \frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3}\,R^4$$

This result is confirmed in [Hart52].

3.6.5 Simply supported edge at $r = R$

If the angle at $r = R$ is free to move then we have to use the natural boundary condition, i.e. the terms in the variation of the energy containing $\phi'(R)$ have to vanish at $r = R$.

$$(r\,u')' - (1-\sigma)\,u'(r) = \left(c_3\,2\,r^2 - 4\,\frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3}\,r^4\right)' - (1-\sigma)\,\left(c_3\,2\,r - 4\,\frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3}\,r^3\right) = 0$$

This leads to

$$c_3 \ (4\,R - 2\,(1-\sigma)\,R) = \frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3} \ \left(16\,R^3 - 4\,(1-\sigma)\,R^3\right)$$

with the solution

$$c_3 = \frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3} \ \frac{16-4\,(1-\sigma)}{4-2\,(1-\sigma)} \ R^2 = \frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3} \ \frac{6+2\,\sigma}{1+\sigma} \ R^2$$

and thus

$$u(R) = \frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3} \ \left(\frac{6+2\,\sigma}{1+\sigma} - 1\right) \ R^4 = \frac{12\,p\,(1-\sigma^2)}{64\,E\,h^3} \ \left(\frac{5+\sigma}{1+\sigma}\right) \ R^4$$

We obtain the same result as for a clamped plate, except for the factor $(5+\sigma)/(1+\sigma)$. We conclude that the maximal deflection will increase by this factor if the plate is simply supported instead of clamped. For a typical value of $\sigma = 0.3$ of Poisson's ratio the factor is approximately 4.08. This is confirmed by the results in [Hart52].

Since the dependence of the maximal deflection on the parameters is explicitly given one can now quickly decide whether a given design of a pressure sensor might work.

3.6.6 Introduce new variable

If the plate is clamed we use $u(R) = u'(R) = 0$ we find the total energy of the plate as

$$F(u) = \frac{\pi\, E\, h^3}{12\,(1-\sigma^2)} \int_0^R \frac{1}{r} \left(\left(r\, u' \right)' \right)^2 \, dr + \int_0^R 2\,\pi\, r\, p \cdot u \, dr$$

$$F(u) = \frac{\pi\, E\, h^3}{12\,(1-\sigma^2)} \int_0^R \frac{1}{r} \left(\left(r\, u' \right)' \right)^2 \, dr - \int_0^R \pi\, r^2\, p \cdot u' \, dr$$

Let $v(r) = r\, u'(r)$

$$F(v) = \frac{\pi\, E\, h^3}{12\,(1-\sigma^2)} \int_0^R \frac{1}{r}\, v'^2 \, dr - \int_0^R \pi\, r\, p \cdot v \, dr$$

$$\begin{aligned} F(v+\phi) - F(v) &\approx k \int_0^R \frac{2}{r}\, v' \cdot \phi' \, dr - \int_0^R \pi\, r\, p \cdot \phi \, dr \\ &= \left(k\, \frac{2\, v'(r)}{r} \cdot \phi(r) \right) \Big|_{r=0}^{R} - \int_0^R \left(\left(k\, \frac{v'}{r} \right)' + \pi\, r\, p \right) \cdot \phi \, dr \end{aligned}$$

Thus

$$\left(k\, \frac{v'}{r} \right)' = -\pi\, r\, p$$

and thus

$$\begin{aligned} k\, \frac{v'}{r} &= -\frac{\pi\, p}{2}\, r^2 + c_1 \\ k\, v' &= -\frac{\pi\, p}{2}\, r^3 + c_1\, r \\ k\, r\, u' = k\, v &= -\frac{\pi\, p}{8}\, r^4 + \frac{c_1}{2}\, r^2 + c_2 \\ k\, u' &= -\frac{\pi\, p}{8}\, r^3 + \frac{c_1}{2}\, r + \frac{c_2}{r} \\ k\, u &= -\frac{\pi\, p}{32}\, r^4 + \frac{c_1}{4}\, r^2 + c_2\, \ln r + c_3 \\ k\, u &= -\frac{\pi\, p}{32}\, r^4 + \frac{c_1}{4}\, r^2 \qquad (\quad \text{use} \quad u(0) = u'(0) = 0 \qquad) \\ 0 &= -\frac{\pi\, p}{8}\, R^3 + \frac{c_1}{2}\, R \qquad (\quad \text{use} \quad u'(R) = 0 \qquad) \\ c_1 &= \frac{\pi\, p}{4}\, R^2 \\ k\, u(r) &= -\frac{\pi\, p}{32}\, r^4 + \frac{\pi\, p}{16}\, R^2\, r^2 = \frac{\pi\, p}{32}\, \left(2\, R^2 - r^2 \right)\, r^2 \\ k\, u(R) &= \frac{\pi\, p}{32}\, R^4 \\ U(R) &= \frac{6\,(1-\sigma^2)}{\pi\, E\, h^3}\, \frac{\pi\, p}{32}\, R^4 = \frac{3\,(1-\sigma^2)\, p}{16\, E\, h^3}\, R^4 \end{aligned}$$

3.6.7 Eigenfrequencies of a clamped plate

$$\begin{aligned} F(u) &= \frac{k}{2} \int_0^R \frac{1}{r} \left(\left(r\, u' \right)' \right)^2 \, dr + \int_0^R 2\,\pi\, r\, p \cdot u \, dr \\ F(u+\phi) - F(u) &\approx k \int_0^R \frac{1}{r} \left(r\, u' \right)' \cdot \left(r\, \phi' \right)' \, dr + \int_0^R 2\,\pi\, r\, p \cdot \phi \, dr \end{aligned}$$

$$
\begin{aligned}
&= \text{boundary} - k \int_0^R \left(\frac{1}{r} \left(r\, u' \right)' \right)' \cdot r\, \phi'\, dr + \int_0^R 2\,\pi\, r\, p \cdot \phi\, dr \\
&= \text{boundary} + k \int_0^R \left(r \left(\frac{1}{r} \left(r\, u' \right)' \right)' \right)' \cdot \phi\, dr + \int_0^R 2\,\pi\, r\, p \cdot \phi\, dr
\end{aligned}
$$

Euler–Lagrange equation

$$
k \left(r \left(\frac{1}{r} \left(r\, u' \right)' \right)' \right)' + 2\,\pi\, r\, p = 0
$$

Replace the pressure p by $h\,\rho\,\ddot{u} = -\omega^2\, h\,\rho\, u$ to examine harmonic vibration with angular velocity ω. This leads to

$$
\begin{aligned}
\frac{\pi\, E\, h^3}{6\,(1-\sigma^2)} \left(r \left(\frac{1}{r} \left(r\, u' \right)' \right)' \right)' &= 2\,\pi\, r\, \omega^2\, h\,\rho\, u \\
\left(r \left(\frac{1}{r} \left(r\, u' \right)' \right)' \right)' &= \lambda\, r\, u
\end{aligned}
$$

with

$$
\lambda = \frac{6\,(1-\sigma^2)}{\pi\, E\, h^3}\, 2\,\pi\, \omega^2\, h\,\rho = \frac{12\,(1-\sigma^2)\,\rho}{E\, h^2}\, \omega^2
$$

3.7 Exercises

• Exercise 3–1:
Consider a curve $y = u(x)$ for $a \le x \le b$ where $0 < a < b$, $u(a)$ and $u(b)$ are given. This curve is rotated about the x axis and a surface of revolution is generated.

(a) Show that the area of the surface of revolution is given by

$$A(u) = 2\,\pi \int_a^b u(x)\,\sqrt{1 + (u'(x))^2}\,dx$$

This is only correct if $u(x) \ge 0$ for all $a < x < b$.

(b) Find the Euler Lagrange equation for this area to be minimal.

(c) Use separation of variables to find a solution to this differential equation. Find the integral to be computed in a table. The result will be a catenary.

• Exercise 3–2:
Write the elastic energy of a bar ($a \le x \le b$) with variable cross section $A(x)$ as a function depending on the displacement function $u(x)$. The energy will be minimal.

(a) Then use the Euler Lagrange equation to find a differential equation for the displacement function.

(b) Find the exact solution of the bar in section 2.2.2. Assume that the displacement $u(b) = u(100) = B$ is known.

(c) The variational problem will have a first integral. What is the physical interpretation of this first integral?

(d) Compute the force F at the right end point as a function of B. Use the strain and Hooke's law.

• Exercise 3–3:
Use the Euler Lagrange equations to show that the connection between two points is given by a straight line. You may assume that y and z are functions of the free variable x.

• Exercise 3–4:
A curve of known length L is described by $y = u(x) \ge 0$ for $0 \le x \le b$. It is required that $u(0) = 0$ but the value of $u(b)$ is not prescribed. The area A under between the curve, the x axis and the vertical line $x = b$ is computed by an integral, as is the length L.

$$A = \int_0^b u(x)\,dx \quad \text{and} \quad L = \int_0^b \sqrt{1 + (u'(x))^2}\,dx$$

Show that the curve enclosing the maximal area is a section of a circle and will have a horizontal tangent line at $x = b$.

Hint : Use the Euler Lagrange equation and show that the curvature κ is constant.

$$\kappa = \frac{u''(x)}{\sqrt{1 + (u'(x))^2}^3}$$

• Exercise 3–5:
A cylinder with radius R is partially filled with a given volume V_0 of water and then rotated about the z axis with angular velocity ω. The surface of the water can be described by a function $z = u(r)$.

(a) Find the volume V, the potential energy U and the kinetic energy T as functionals of the function u by appropriate integration.

(b) Hamilton's principle of least action implies that the functional $L(u) = T(u) - U(u)$ will be minimal at the stable state. But the restriction of the given volume V has to be taken into account. The theory about isoperimetric problem indicates that there will be a Lagrange multiplier $\lambda \in \mathbb{R}$ such that the functional $L(u) + \lambda V(u)$ will admit a critical value.

$$\text{Find minimum of} \quad T(u) - U(u) \quad \text{subject to} \quad V(u) = V_0$$

Use the Euler Lagrange equation to find $u(r)$.

• Exercise 3–6:
For given functions $a(x)$, $b(x)$ and $g(x)$ consider the functional

$$F(u) = \int_a^b \frac{1}{2}\, a(x)\, (u'(x))^2 + \frac{1}{2}\, b(x)\, u(x)^2 + g(x) \cdot u(x)\; dx + r(a)\, u(a) - r(b)\, u(b)$$

Use Theorem 3–7 (or its proof) to show that a minimiser of this functional has to solve the Neumann boundary value problem

$$\begin{aligned} \frac{d}{dx}\left(a(x)\, \frac{d\, u(x)}{dx} \right) &= b(x)\, u(x) + g(x) \quad \text{for} \quad a < x < b \\ a(x)\, \frac{d\, u(x)}{dx}\bigg|_{x=a} &= r(a) \\ a(x)\, \frac{d\, u(x)}{dx}\bigg|_{x=b} &= r(b) \end{aligned}$$

• Exercise 3–7:
Consider the functional

$$F(u) = \frac{1}{2} \int_a^b A(x)\; \big(u''(x)\big)^2\; dx$$

This does **not** fit exactly into the framework of Theorem 3–7. Use the ideas and procedures of the calculus of variations to show, that a necessary condition for an extremum is given by

$$\frac{d^2}{dx^2}\; \big(A(x)\, u''(x)\big) = 0 \quad \text{for} \quad a < x < b$$

A possible application of this type of problem is the bending of a bar. Set $A(x) = E\, I(x)$ (E=modulus of elasticity, $I(x)$= moment of inertia of the cross section) and let $u(x)$ denote the vertical displacement. If a vertical force $f(x)$ (units $\frac{\mathrm{N}}{\mathrm{m}}$) is applied the functional

$$F(u) = \frac{1}{2} \int_a^b E\, I(x)\; \big(u''(x)\big)^2 + u(x) \cdot f(x)\; dx$$

has to be minimised. For constant cross section this should lead to the equation

$$E\, I\, u^{(4)}(x) = -f(x)$$

Chapter 4

Finite Element problems in one variable

In this chapter we will thoroughly discuss the problem of solving second order boundary value problems in one variable by means of the finite element method. To achieve this ambitious goal we proceed in separate steps:

- Use basic physical principles to derive and discuss the heat equation in one or multiple variables. Consider the special case of a thin circular plate.
- Introduce weak formulation of differential equations.
- Introduce the general problem to be solved in this section and list a few of the possible application areas.
- Construct the most simple finite element in one variable carefully: piecewise linear interpolation.
- Construct a vastly improved element using piecewise quadratic interpolation and Gauss integration.
- Solve a few sample problems.

4.1 The heat equation

As a real world example we first want to derive the heat equation for solids and the examine it in a few special situations.

4.1.1 Basic physics

The **heat capacity** c of a material gives the amount of energy needed to raise the temperature T one kilogramme of the material by one degree K (Kelvin). The **thermal conductivity** k of a material indicates the amount of energy transmitted trough a plate with thickness 1 m and 1 m^2 area if the temperatures at the two sides differ by 1 K. In table 4.1 find values for c and k for some typical materials. For homogeneous materials the values of c and k will not depend on the location $\vec{x}$. For some materials the values can depend on the temperature T, but we will not consider this case since the resulting equations would be nonlinear.

The **flux of thermal energy** is a vector indicates the direction of the flow and the amount of thermal energy flowing per second and square meter. **Fourier's law** of heat conduction can be stated as

$$\vec{q} = -k \nabla T \tag{4.1}$$

This basic law of physics indicates that the thermal energy will flow from hot spots to areas with lower temperature. For some simple situations we will examine the consequences of this equation. The only other basic physical principle to be used is **conservation of energy**. Some of the variables and symbols used in this section are shown in table 4.2.

	heat capacity at $20°C$	heat conductivity
symbol	c	k
unit	$\frac{\text{kJ}}{\text{kg K}}$	$\frac{\text{W}}{\text{m K}}$
iron	0.452	74
steel	0.42 - 0.51	45
copper	0.383	384
water	4.182	0.598

Table 4.1: Some values of heat related constants

	symbol	unit
density of energy	u	$\frac{\text{J}}{\text{m}^3}$
temperature	T	K
heat capacity	c	$\frac{\text{J}}{\text{K kg}}$
density	ρ	$\frac{\text{kg}}{\text{m}^3}$
heat conductivity	k	$\frac{\text{J}}{\text{s m K}}$
heat flux	$\vec{q}$	$\frac{\text{J}}{\text{s m}^2}$
external energy source density	f	$\frac{\text{J}}{\text{s m}^3}$

Table 4.2: Symbols and variables for heat conduction

4.1.2 One dimensional heat equation

If a temperature T over a solid (with constant cross section A) is known to depend on one coordinate x only, then the change of temperature ΔT measured over a distance Δx will lead to a flow of thermal energy ΔQ. If the time difference is Δt then (4.1) reads as

$$\frac{\Delta Q}{\Delta t} = -k\ A\ \frac{\Delta T}{\Delta x}$$

It we choose the temperature T as unknown variable we find on the interval $a \le x \le b$ the thermal energy

$$E(t) = \int_a^b u(t,x)\ dx = \int_a^b \rho\, c\, T(t,x)\ dx$$

Fourier's law leads to a total heat flux in direction x of

$$Q = -k\, A\ \frac{\partial T}{\partial x}$$

Now we compute the rate of change of energy in the same interval. The rate of change has to equal the total flux of energy into this interval plus the input from external sources

$$\begin{array}{ccccc} \text{total change} & = & \text{input through boundary} & + & \text{external sources} \\ \frac{d\,E(t)}{dt} & = & \left(-k\,A(a)\ \frac{\partial T(t,a)}{\partial x} + k\,A(b)\ \frac{\partial T(t,b)}{\partial x}\right) & + & \int_a^b f(t,x)\ dx \\ \int_a^b \rho\, c\, \frac{\partial\, T(t,x)}{\partial t}\ dx & = & \int_a^b \frac{\partial}{\partial x}\left(k\,A(x)\ \frac{\partial T(t,x)}{\partial x}\right)\ dx & + & \int_a^b f(t,x)\ dx \end{array}$$

At this point we used the conservation of energy. Since the above equation has to be correct for all possible values of a and b the expressions under the integrals have to be equal and we obtain the general equation for heat conduction in one variable

$$\rho\, c\, \frac{\partial\, T(t,x)}{\partial t}\ dx = \frac{\partial}{\partial x}\ \left(k\,A(x)\ \frac{\partial T(t,x)}{\partial x}\right) + f(t,x)$$

If we are only interested in steady state solution then the temperature T can not depend on t and thus we find

$$-\frac{\partial}{\partial x}\ \left(k\,A(x)\ \frac{\partial T(t,x)}{\partial x}\right) = f(t,x)$$

This second order differential equation has to be supplemented by boundary conditions, either prescribed temperature or prescribed energy flux. Then the equation can be solved by the finite element method.

4.1.3 Two dimensional heat equation, strong formulation

If the domain $G \subset \mathbb{R}^2$ with boundary curve C describes a thin plate with constant thickness h then we may assume that the temperature will depend on t, x and y only and not on z.

The total energy stored in that domain is given by

$$E\,(t) = \iint\limits_G h\,u\ dA = \iint\limits_G h\,c\,\rho\ T(t,x,y)\ dA$$

Again we compute the rate of change of energy $\frac{d}{dt}\,E$ and arrive at

$$\frac{d}{dt}\,E = \iint\limits_G h\,c\,\rho\,\frac{\partial\,T}{\partial t}\ dA = -\oint\limits_C h\,\vec{q}\cdot\vec{n}\ ds + \iint\limits_G h\,f\ dA$$

Using the divergence theorem on the second integral and Fourier's law we find

$$\begin{aligned}\iint\limits_G h\,c\,\rho\,\frac{\partial T}{\partial t}\,dA &= -\iint\limits_G \operatorname{div}(h\,\vec{q})\,dA + \iint\limits_G h\,f\,dA \\ &= \iint\limits_G \operatorname{div}\left(k\,h\nabla T\right)dA + \iint\limits_G h\,f\,dA \\ &= \iint\limits_G \operatorname{div}\left(k\,h\nabla T(t,x,y)\right)dA + \iint\limits_G h\,f\,dA\end{aligned}$$

This equation has to be correct for all possible domains G and not only for the actual physical domain. Thus the expressions under the integral have to be equal and we find

$$h\,c\,\rho\,\frac{\partial T}{\partial t} = \operatorname{div}\left(k\,h\nabla T(t,x,y)\right) + h\,f \tag{4.2}$$

If ρ, c, h and k are constant then we find in cartesian coordinates

$$c\,\rho\,\frac{\partial}{\partial t}\,T\,(t,x,y) = k\,\left(\frac{\partial^2}{\partial x^2}\,T\,(t,x,y) + \frac{\partial^2}{\partial y^2}\,T\,(t,x,y)\right) + f\,(t,x,y)$$

or shorter

$$c\,\rho\,\frac{\partial}{\partial t}\,T\,(t,x,y) = k\,\Delta\,T\,(t,x,y) + f\,(t,x,y)$$

where Δ is the well known **Laplace operator**. The heat equation is thus a second order partial differential equation with respect to the space variable x and y and of first order with respect to time t.

If the solution is know to be independent of time then we find

$$\operatorname{div}\left(k\,h\nabla T(t,x,y)\right) = -h\,f$$

If this equation is supplemented with boundary conditions then we can try to solve it numerically.

4.1.4 Steady state problem with radial symmetry

Suppose we have a radial plate of Radius R and the external heat source depends on the radius ρ only. Then the temperature T will also depend on ρ. Thus we rewrite the above differential equation in polar coordinates. If ρ, c, h and k are constant then we can use the Laplace operator in polar coordinates (see appendix A.2.4) and find

$$\Delta T(\rho,\phi,z) = \Delta T(\rho) = \frac{1}{\rho}\,\frac{\partial}{\partial\rho}(\rho\frac{\partial\,T}{\partial\rho}) + \frac{1}{\rho^2}\frac{\partial^2\,T}{\partial\phi^2} + \frac{\partial\,T}{\partial z^2} = \frac{1}{\rho}\,\frac{\partial}{\partial\rho}(\rho\frac{\partial\,T}{\partial\rho})$$

and thus the ordinary differential equation

$$\frac{1}{\rho}\,\frac{\partial}{\partial\rho}(\rho\frac{\partial\,T(\rho)}{\partial\rho}) = -\frac{1}{k}\,f(\rho)$$

or

$$\frac{\partial}{\partial\rho}(\rho\frac{\partial\,T(\rho)}{\partial\rho}) = -\frac{\rho}{k}\,f(\rho)$$

As an example we consider a plate of radius 3 which is heated uniformly on the ring $1 \le \rho \le 2$. The outer edge is kept on $T = 0$ and the heat conductivity is set to $k = 1$. The equation to be solved is

$$\frac{\partial}{\partial\rho}(\rho\frac{\partial\,T(\rho)}{\partial\rho}) = \rho\,T''(\rho) + T'(\rho) = -\rho\,f(\rho) = \begin{cases} 0 & \text{if} \quad 0 \le \rho < 1 \\ -\rho & \text{if} \quad 1 \le \rho < 2 \\ 0 & \text{if} \quad 0 \le \rho \le 3 \end{cases}$$

$$\begin{aligned} T(3) &= 0 \\ T'(0) &= 0 \end{aligned}$$

Solution with *Mathematica*

Figure 4.1 shows the heating function $f(\rho)$ and section of the resulting temperature as a function of the radius. The results were generated with the code below.

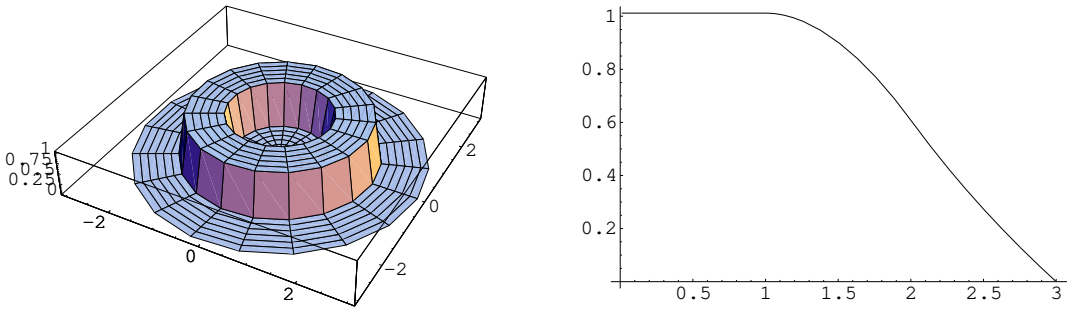


Figure 4.1: Heat source on a ring and the resulting temperature distribution

Mathematica

```
Clear[f,T,Tn]
f[r_] = Which[r<1,0,r<2,1,True,0];
g1=ParametricPlot3D[{r Cos[phi],r Sin[phi],f[r]},{phi,0,2 Pi},{r,0,3}];
nsol=NDSolve[{D[r D[T[r],r],r]== -r*f[r],T[3]==0,Derivative[1][T][0.01]==0},
             T[r],{r,0.01,3}, Method -> RungeKutta];
Tn[r_] = T[r]/.nsol[[1]];
g2=Plot[Tn[r],{r,0.01,3},PlotRange -> All];
```

Mathematica had to be tricked into solving this equation. The boundary condition $T'(0) = 0$ had to be specified at a small value of ρ and the numerical scheme yielding a solution seems to be Runge–Kutta. A sizeable number of attempts failed before the above created an answer. A good finite element code will avoid both those problems.

Exact solution

The problem above can be solved exactly. This is useful to verify the numerical schemes and the coding of the algorithms. The computations are elementary but lengthy.

- For $0 < \rho < 1$ we have the differential equation

$$\frac{\partial}{\partial \rho}\,\rho\,\frac{\partial\, T(\rho)}{\partial \rho} = 0 \quad \text{with} \quad T'(0) = 0$$

  and thus the solution $T(\rho) = c_1$.

- For $2 < \rho < 3$ we have again the differential equation

$$\frac{\partial}{\partial \rho}\,\rho\,\frac{\partial\, T(\rho)}{\partial \rho} = 0 \quad \text{with} \quad T(3) = 0$$

  This can be written as $T'(\rho) = \frac{c_2}{\rho}$ with the solution

$$T(\rho) = c_2\, \ln \frac{\rho}{3}$$

- For $1 < \rho < 2$ the differential equation is

$$\frac{\partial}{\partial \rho} \rho \frac{\partial T(\rho)}{\partial \rho} = -\rho$$

and by integration

$$\begin{aligned} \rho \frac{\partial T(\rho)}{\partial \rho} &= -\frac{\rho^2}{2} + c_3 \\ \frac{\partial T(\rho)}{\partial \rho} &= -\frac{\rho}{2} + \frac{c_3}{\rho} \\ T(\rho) &= -\frac{\rho^2}{4} + c_3 \ln \rho + c_4 \end{aligned}$$

- Now we have to use the two compatibility conditions at $\rho = 1$ and $\rho = 2$. Values and derivatives at those points have to coincide. This leads to

$$\begin{aligned} c_1 &= -\frac{1^2}{4} + c_3 \ln 1 + c_4 \\ 0 &= -\frac{2 \cdot 1}{4} + \frac{c_3}{1} \\ c_2 \ln \frac{2}{3} &= -\frac{2^2}{4} + c_3 \ln 2 + c_4 \\ c_2 \frac{1}{2} &= -\frac{2 \cdot 2}{4} + \frac{c_3}{2} \end{aligned}$$

This leads to a system of 4 equations for the 4 unknown constants c_i

$$\begin{bmatrix} 1 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \\ 0 & \ln \frac{2}{3} & -\ln 2 & -1 \\ 0 & \frac{1}{2} & \frac{-1}{2} & 0 \end{bmatrix} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{pmatrix} = \begin{pmatrix} \frac{-1}{4} \\ \frac{-1}{2} \\ -1 \\ -1 \end{pmatrix}$$

The solution is given by

$$\begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{pmatrix} = \begin{pmatrix} 1.01162 \\ -1.5 \\ 0.5 \\ 1.26162 \end{pmatrix}$$

The result can be seen in figure 4.1. This example shows that an exact solution can be available, but the computations necessary to obtain it can be prohibitive (which was not the case here). If the shape of the heating function $f(\rho)$ is modified the computations have to be redone and there are no guarantees that an exact solution can be found. A stable numerical scheme to find an approximate solution will be useful and easy to use, once the method is programmed, see section 4.7.2.

4.2 Weak solutions

4.2.1 Two dimensional heat equation, weak formulation

The general steady state heat equation in two variables is

$$\operatorname{div}(k\, h \nabla T(x, y)) + h\, f = 0$$

This equation has to be valid within the domain $G \subset \mathbb{R}^2$. The boundary ∂G of the domain is divided up into two disjoint parts Γ_1 and Γ_2. On Γ_1 the temperature is equals to a given function g_1 (Dirichlet condition) and on Γ_2 the heat flux is prescribed to be g_2 (Neumann condition). This leads to the problem

$$\begin{aligned} \nabla \cdot (k\, h \nabla T(x,y)) &= -h\, f \qquad \text{for} \quad (x,y) \in G \\ T(t,x,y) &= g_1(x,y) \qquad \text{for} \quad (x,y) \in \Gamma_1 \\ h\, k\, \nabla T(t,x,y) \cdot \vec{n} &= g_2(x,y) \qquad \text{for} \quad (x,y) \in \Gamma_2 \end{aligned} \tag{4.3}$$

Now we consider test functions $\phi(x,y)$ that vanish on Γ_1. The differential equations is integrated over the domain G. For sake of shorter notation the arguments (x,y) are suppressed.

$$\iint_G \phi\ \operatorname{div}(k\, h \nabla T)\ dA + \iint_G \phi\ h\ f\ dA = 0$$

Now we use a version of the divergence theorem (see appendix A.3)

$$\iint_G f\ (\operatorname{div} \vec{v})\ dA \quad = \quad \oint_{\partial G} f\, \vec{v} \cdot \vec{n}\ dA - \iint_G (\operatorname{grad} f) \cdot \vec{v}\ dA$$

and arrive at

$$\begin{aligned} 0 &= \oint_{\partial G} \phi\ (k\, h \nabla T) \cdot \vec{n}\ ds + \iint_G -\nabla\phi \cdot (k\, h \nabla T) + \phi\, h\, f\ dA \\ &= \oint_{\Gamma_1 \cup \Gamma_2} \phi\, k\, h\ \nabla T \cdot \vec{n}\ ds + \iint_G -k\, h\ \nabla\phi \cdot \nabla T + \phi\, h\, f\ dA \\ &= \oint_{\Gamma_2} \phi\, g_2\ ds + \iint_G -k\, h\ \nabla\phi \cdot \nabla T + \phi\, h\, f\ dA \end{aligned}$$

This leads to the definition of a **weak solution** of the equation (4.3). The function $T(x,y)$ is called a weak solution iff $T(x,y) = g_1$ on Γ_1 and

$$\iint_G k\, h\ \nabla\phi \cdot \nabla T\ dA = \oint_{\Gamma_2} \phi\, g_2\ ds + \iint_G h\, f\ dA \tag{4.4}$$

for 'all' functions ϕ vanishing on Γ_1.

4.2.2 Advantages of weak solutions

At first sight the definition of a weak solution semms to be rather artificial, but it has a few decisive advantages over the classical definition of a solution. The concept of weak solutions is the foundation of **Hilbert space methods** to solve partial differential equations. The following comparison of weak and strong formulations of differential equations is quoted from [OttoPete92, p. 59].

We have shown that the same differential equation can be formulated in a strong or weak form, where the weak formulation is more involved and requires more mathematical results. So one may ask why we have taken the trouble to obtain a weak form. The answer is fundamental for the establishment of the finite element method: it is the weak form that the finite element formulation is based on.

In order to understand this point consider the differential equation in the strong form, i.e. (4.3). It appears that the unknown function T is differentiated twice. The finite element approach is an approximate method, so in one way or another we have to replace the true T-function by an approximate one. If this

approximation is made directly in (4.3), we need to deal with an approximating function, which is at least twice differentiable. In the weak formulation, however, only the first derivative of the temperature function T enters, (cf. (4.4)). That is, if we choose the weak form as the basis of the approximation, we may deal with approximating functions which only need to be differentiable once. This aspect clearly favours the weak form compared with the strong form, and it also suggests the terminology of weak and strong forms.

Another point, closely related to the above is the fact that weak forms provide in fact a more general formulation than strong forms. For the weak form we only need the derivative of the temperature T to be integrable (i.e. piecewise continuous). The temperature T need not be twice differentiable in simple applications. Consider a plate consisting of two different materials with different heat conduction coefficients k_1 and k_2 and consider a point of discontinuity of k. The heat flux vector is given by $\vec{q} = -k \operatorname{grad} T$. This flux has to be continuous. On the two sides of the separation line of the materials we find

$$-k_1 \nabla T \mid_{\text{material 1}} = -k_2 \nabla T \mid_{\text{material 2}}$$

As k changes the value the derivatives of T have to jump. Thus the derivatives of T may not be continuous and there is no hope to find a second derivative. Weak formulations and finite element formulations are perfectly capable to incorporate this type of problem.

In the next section we show that weak formulation may also be obtained by means of minimising a well chosen functional. This will link the previous chapter to weak solutions.

4.2.3 Weak solution of heat equation on a circular plate

For cylindrical coordinates and functions depending on the radius only we use

$$\nabla T(\rho) = \operatorname{grad} T(\rho) = \frac{\partial T}{\partial \rho}\, \vec{e}_\rho$$

The domain to be considered is a circle of radius R and the temperature at the edge has to be zero. The boundary consists of Γ_1 only and h and k are assumed to be constant, thus equation (4.4) now reads as

$$\begin{aligned}
\iint_G k\, h\, \nabla\phi \cdot \nabla T \; dA &= \iint_G \phi\, h\, f \; dA \\
h \iint_G k\, (\frac{\partial \phi}{\partial \rho}\, \vec{e}_\rho) \cdot (\frac{\partial T}{\partial \rho}\, \vec{e}_\rho) \; dA &= h \iint_G \phi\, f \; dA \\
\int_0^R 2\,\pi\,\rho\, k\, \phi'(\rho) \cdot T'(\rho) \; d\rho &= \int_0^R 2\,\pi\,\rho\, \phi(\rho)\, f(\rho) \; d\rho \\
\int_0^R \rho\, k\, \phi'(\rho) \cdot T'(\rho) \; d\rho &= \int_0^R \rho\, \phi(\rho)\, f(\rho) \; d\rho
\end{aligned}$$

This is the weak formulation of the steady state heat problem on a disk, assuming that the solution will depend on the radius ρ only.

Now consider the functional

$$F(T) = \int_0^R \frac{\rho\, k}{2} \; \big(T'(\rho)\big)^2 - \rho\, T(\rho)\, f(\rho) \; dA$$

and find

$$\frac{d\, F(T + \varepsilon\phi)}{d\varepsilon}\bigg|_{\varepsilon=0} = \int_0^R \rho\, k\, \phi'(\rho) \cdot T'(\rho) - \rho\, \phi(\rho)\, f(\rho) \; d\rho$$

Observe that the function $T(\rho)$ being a weak solution of the heat equation is equivalent to $T(\rho)$ being a critical point of the functional F, or in fact a minimiser of F. Thus we are lead to a minimisation problem, to be solved by the finite element method.

4.3 The general one dimensional problem

In section 3.1.5, equation (3.1) we considered functionals of the form

$$F(u) = \int_a^b \frac{1}{2}\, a(x)\, (u'(x))^2 + \frac{1}{2}\, b(x)\, u(x)^2 + g(x) \cdot u(x)\; dx$$

for given functions $a(x)$, $b(x)$ and $g(x)$. The minimising function has to solve the corresponding Euler Lagrange equation

$$\frac{d}{dx}\left(a(x)\,\frac{d\,u(x)}{dx}\right) = b(x)\,u(x) + g(x)$$

This second order, linear differential equation has to be supplemented with appropriate boundary conditions. We may use **Dirichlet boundary conditions** or **Neumann boundary conditions**.

Aiming for a numerical approximation to the true solution we divide the interval $[a, b]$ in n subintervals with a partition $a = x_0 < x_1 < x_2 < \ldots < x_n = b$. The length of the i–th interval is $\Delta x_i = x_i - x_{i-1}$. Such a subinterval is also called **element**. Instead of all (smooth) functions on $[a, b]$ we consider only special types of functions. The goal is to minimise the functional $F(u)$ amongst this type of function. To acchieve this we first outline the steps to be performed:

1. Find the energy on each element, using matrices.
2. Find a formula for the total energy, using a matrix.
3. Take boundary conditions into account.
4. Convert to a system of linear equations and solve.
5. Extract information about the solution, e.g. a plot.

We will carry out the above steps for two different type of elements:

1. First we use elements with linear functions and the simplest integration procedure possible. The individual steps should be clearly visible and easy to understand. We could solve general problems with this procedure, but it would not be very efficient.
2. As a second case we examine quadratic elements and use Gauss integration. Due to those improvements we find a few more calculations to be done, but the resulting final code is rather efficient in solving general problems.

4.4 First order elements

To simplify the calculation we first consider only the situation $b(x) = 0$ in equation (3.1). At the end this contribution will be taken into account too.

On each subinterval $[x_{i-1}, x_i]$ we consider a linear function. The function u is required to be continuous: the value at the right endpoint of one subinterval has to coincide with the value at the left end point of the next subinterval. This is a **compatibility condition**. We obtain a continuous, piecewise linear function $u(x)$ by a linear interpolation using the points of support $(x_i, u(x_i)) = (x_i, u_i)$. Thus a complete description of the function is given by the two vectors

$$\vec{x} = (x_0, x_1, x_2, \ldots, x_n)^T \quad \text{and} \quad \vec{u} = (u_0, u_1, u_2, \ldots, u_n)^T$$

This discretisation is visualized in Figure 4.2.

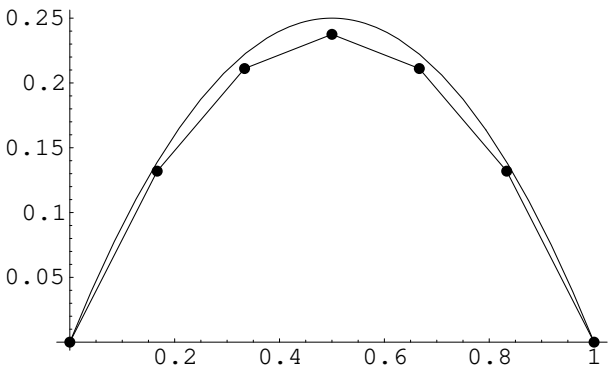


Figure 4.2: Approximation of a function by linear elements

4.4.1 Description of one element with a linear function

For the i–th element we have to consider $x_{i-1} \le x \le x_i$ with the values u_{i-1} and u_i at the endpoints. For the values of x in the interval we have

$$u(x) = u_{i-1} + \frac{u_i - u_{i-1}}{\Delta x_i}\,(x - x_{i-1})$$

and the derivative is a constant given by

$$u'(x) = \frac{u_i - u_{i-1}}{\Delta x_i}$$

Thus the contribution to the functional $F(\vec{u})$ is given by

$$F_i = \int_{x_{i-1}}^{x_i} \frac{1}{2}\,a(x)\,(u'(x))^2 + g(x)\cdot u(x)\;dx$$

Since we want to deal with general function $a(x)$ and $g(x)$ we have to approximate this integral numerically. A simple approach is to use the trapezoidal rule

$$\int_{x_{i-1}}^{x_i} f(x)\;dx \approx \frac{f(x_{i-1}) + f(x_i)}{2}\;\Delta x_i$$

This corresponds to replacing a linear function by the value at the midpoint of the interval. With this we arrive at

$$\begin{aligned}
F_i &= \int_{x_{i-1}}^{x_i} \frac{1}{2}\,a(x)\,(u'(x))^2 + g(x)\cdot u(x)\;dx \\
&\approx \frac{1}{2}\left(\frac{a(x_{i-1}) + a(x_i)}{2}\;\left(\frac{u_i - u_{i-1}}{\Delta x_i}\right)^2 + (g(x_{i-1})\,u_{i-1} + g(x_i)\,u_i)\right)\Delta x_i \\
&= \frac{1}{2}\left(\frac{a(x_{i-1}) + a(x_i)}{2\,\Delta x_i}\;(u_i - u_{i-1})^2\right) + (g(x_{i-1})\,u_{i-1} + g(x_i)\,u_i)\;\frac{\Delta x_i}{2} \\
&= \frac{1}{2}\,\frac{a(x_{i-1}) + a(x_i)}{2\,\Delta x_i}\,\langle\begin{pmatrix} u_{i-1} \\ u_i \end{pmatrix}, \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\begin{pmatrix} u_{i-1} \\ u_i \end{pmatrix}\rangle + \frac{\Delta x_i}{2}\,\langle\begin{pmatrix} u_{i-1} \\ u_i \end{pmatrix}, \begin{pmatrix} g(x_{i-1}) \\ g(x_i) \end{pmatrix}\rangle \\
&= \frac{1}{2}\,\langle\begin{pmatrix} u_{i-1} \\ u_i \end{pmatrix}, \mathbf{K}_i\begin{pmatrix} u_{i-1} \\ u_i \end{pmatrix}\rangle + \langle\begin{pmatrix} u_{i-1} \\ u_i \end{pmatrix}, \vec{b}_i\rangle
\end{aligned}$$

where the **element stiffness matrix $\mathbf{K}_i$** is given by

$$\mathbf{K}_i = \frac{a(x_{i-1}) + a(x_i)}{2\,\Delta x_i} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} k_i & -k_i \\ -k_i & k_i \end{bmatrix}$$

The coefficient k_i is equals to the average value of the function $a(x)$ at the two endpoints of the element, divided by the length of the element. The vector $\vec{b}_i$ by

$$\vec{b}_i = -\frac{\Delta x_i}{2} \begin{pmatrix} g(x_{i-1}) \\ g(x_i) \end{pmatrix} = \begin{pmatrix} b_{i,0} \\ b_{i,1} \end{pmatrix}$$

4.4.2 Add up the contributions from the elements

Now we have to compute the total functional by

$$F(\vec{u}) = \sum_{i=1}^{n} F_i$$

To examine the effects we consider first three terms only

$$\begin{aligned} F_1 + F_2 + F_3 &= +\frac{1}{2}\left\langle \begin{pmatrix} u_0 \\ u_1 \end{pmatrix}, \begin{bmatrix} k_1 & -k_1 \\ -k_1 & k_1 \end{bmatrix} \begin{pmatrix} u_0 \\ u_1 \end{pmatrix} \right\rangle + \left\langle \begin{pmatrix} u_0 \\ u_1 \end{pmatrix}, \begin{pmatrix} b_{1,0} \\ b_{1,1} \end{pmatrix} \right\rangle \\ &\quad +\frac{1}{2}\left\langle \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, \begin{bmatrix} k_2 & -k_2 \\ -k_2 & k_2 \end{bmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \right\rangle + \left\langle \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, \begin{pmatrix} b_{2,0} \\ b_{2,1} \end{pmatrix} \right\rangle \\ &\quad +\frac{1}{2}\left\langle \begin{pmatrix} u_2 \\ u_3 \end{pmatrix}, \begin{bmatrix} k_3 & -k_3 \\ -k_3 & k_3 \end{bmatrix} \begin{pmatrix} u_2 \\ u_3 \end{pmatrix} \right\rangle + \left\langle \begin{pmatrix} u_2 \\ u_3 \end{pmatrix}, \begin{pmatrix} b_{3,0} \\ b_{3,1} \end{pmatrix} \right\rangle \\ &= +\frac{1}{2}\left\langle \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{pmatrix}, \begin{bmatrix} k_1 & -k_1 & 0 & 0 \\ -k_1 & k_1+k_2 & -k_2 & 0 \\ 0 & -k_2 & k_2+k_3 & -k_3 \\ 0 & 0 & -k_3 & k_3 \end{bmatrix} \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{pmatrix} \right\rangle \\ &\quad +\left\langle \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{pmatrix}, \begin{pmatrix} b_{1,0} \\ b_{1,1}+b_{2,0} \\ b_{2,1}+b_{3,0} \\ b_{3,1} \end{pmatrix} \right\rangle \end{aligned}$$

With the above in mind it should not be too difficult to realise that the **total stiffness matrix K** and the vector $\vec{b}$ in

$$F(\vec{u}) = \sum_{i=1}^{n} F_i = \frac{1}{2}\,\langle \vec{u}, \mathbf{K}\vec{u} \rangle + \langle \vec{u}, \vec{b} \rangle$$

are given by

$$\begin{aligned} &\mathbf{K} \text{ is a } (n+1)\times(n+1), \text{ tridiagonal, symmetric matrix} \\ &\text{first upper diagonal} \quad = (-k_1, -k_2, -k_3, \ldots, -k_{n-1}, -k_n) \\ &\text{main diagonal} \quad = (k_1, k_1+k_2, k_2+k_3, \ldots, k_{n-1}+k_n, k_n) \\ &\text{first lower diagonal} \quad = (-k_1, -k_2, -k_3, \ldots, -k_{n-1}, -k_n) \\ &(n+1) \text{ vector } \vec{b} \quad = (b_{1,0}, b_{1,1}+b_{2,0}, b_{2,1}+b_{3,0}, \ldots, b_{n-1,1}+b_{n,0}, b_{n,1}) \end{aligned}$$

Now we have the functional F expressed as function of the unknown vector $\vec{u}$, containing the values of the function $u(x)$ at the points of support x_i. The main goal is to minimise F amongst all admissible functions.

4.4.3 Solve the system of linear equations and use boundary conditions

Minimising the F of the above form is equivalent to solving the system of linear equations

$$\mathbf{K} \cdot \vec{u} = -\vec{b} \tag{4.5}$$

(see Result 1–4). If one of the boundary values at $x = a$ or $x = b$ is fixed, the corresponding component u_0 or u_n of the vector $\vec{u}$ has to be equal to that given value and we are not free to choose it any more. As an example we consider $u(a) = u_0 = A$. In the derivation of the system (4.5) we used derivatives of F with respect to the degrees of freedom u_i. Since u_0 is not free any more we can not set the corresponding derivative to 0 and thus have to remove the first equation in (4.5). Since $u_0 = A$ is known its contribution to the other equations is known and we can move it to the right hand side in (4.5). This leads to a modified equation with one less variable and one less unknown. The ideas are used in the example below. Using algorithms from numerical linear algebra the set of equations can often be solved. Then we can do a piecewise linear interpolation to find an approximate solution to the original minimisation problem.

4.4.4 Examples

4–1 Example : We try to solve the boundary value problem[1]

$$u''(x) = 1 \quad \text{for} \quad 0 < x < 1 \quad \text{and} \quad u(0) = 3 \quad , \quad u'(1) = 0$$

This corresponds to minimising (Euler Lagrange equation)

$$F(u) = \int_0^1 \frac{1}{2} \left((u''(x) \right)^2 + u(x)\, dx$$

amongst all smooth functions with $u(0) = 3$. The condition $u'(1) = 0$ corresponds to a natural boundary condition.

We will use five elements of equal length $h = 0.2$. To adapt the notation of equation (3.1) to this example we have to set $a(x) = 1$ and $g(x) = -1$. Thus the element stiffness matrices $\mathbf{K}_i$ are given by

$$\mathbf{K}_i = \begin{bmatrix} \frac{1}{h} & -\frac{1}{h} \\ -\frac{1}{h} & \frac{1}{h} \end{bmatrix}$$

and the vectors $\vec{b}_i$ by

$$\vec{b}_i = \begin{pmatrix} \frac{-h}{2} \\ \frac{-h}{2} \end{pmatrix}$$

The functional to be minimised is given by

$$F(\vec{u}) = \left\langle \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{pmatrix}, \begin{bmatrix} \frac{1}{h} & -\frac{1}{h} & 0 & 0 & 0 & 0 \\ -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & 0 & 0 & 0 \\ 0 & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & 0 & 0 \\ 0 & 0 & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & 0 \\ 0 & 0 & 0 & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} \\ 0 & 0 & 0 & 0 & -\frac{1}{h} & \frac{1}{h} \end{bmatrix} \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{pmatrix} \right\rangle + \left\langle \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{pmatrix}, \begin{pmatrix} \frac{-h}{2} \\ -h \\ -h \\ -h \\ -h \\ \frac{-h}{2} \end{pmatrix} \right\rangle$$

[1] The exact solution is $u(x) = 3 - x + \frac{1}{2}\, x^2$.

This set of 6 linear equations for 6 unknowns does not take the boundary condition $u(0) = u_0 = 3$ into account. There are at least two different methods to incorporate the condition.

- **Remove one equation using the boundary condition**
  This has to be minimised with respect to $u_1, u_2, \ldots u_5$. The derivative of the functional F with respect to u_0 has to be ignored, i.e. we have to drop the first equation. Using the condition $u(0) = u_0 = 3$ we arrive at the system

$$\begin{bmatrix} -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & 0 & 0 & 0 \\ 0 & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & 0 & 0 \\ 0 & 0 & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & 0 \\ 0 & 0 & 0 & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} \\ 0 & 0 & 0 & 0 & -\frac{1}{h} & \frac{1}{h} \end{bmatrix} \begin{pmatrix} 3 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{pmatrix} = - \begin{pmatrix} h \\ h \\ h \\ h \\ \frac{h}{2} \end{pmatrix}$$

  The contribution of the first component $u_0 = 3$ can be brought to the other side and we arrive at

$$\begin{bmatrix} \frac{2}{h} & -\frac{1}{h} & 0 & 0 & 0 \\ -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & 0 & 0 \\ 0 & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & 0 \\ 0 & 0 & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} \\ 0 & 0 & 0 & -\frac{1}{h} & \frac{1}{h} \end{bmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{pmatrix} = \begin{pmatrix} -h \\ -h \\ -h \\ -h \\ \frac{-h}{2} \end{pmatrix} + \begin{pmatrix} \frac{3}{h} \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

  This system of five equations has to be solved.

- **Modify one equation to take the boundary condition into account**
  Instead of moving the contribution of $u_0 = 3$ to the right hand side we can also add this linear equation to the five others and „reintroduce“ the unknown u_0 by

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & 0 & 0 & 0 \\ 0 & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & 0 & 0 \\ 0 & 0 & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} & 0 \\ 0 & 0 & 0 & -\frac{1}{h} & \frac{2}{h} & -\frac{1}{h} \\ 0 & 0 & 0 & 0 & -\frac{1}{h} & \frac{1}{h} \end{bmatrix} \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{pmatrix} = \begin{pmatrix} 3 \\ -h \\ -h \\ -h \\ -h \\ \frac{-h}{2} \end{pmatrix}$$

  Now we can solve this system of 6 equations. The result is the values of the approximate solution at the points of support.

The numerical calculations are best done with some software. First we find the exact solution with *Mathematica*.

Mathematica

```
Clear[x,sol,uexact,u]
sol=DSolve[{u''[x]==1,u[0]==3,u'[1]==0}, u[x],x];
uexact[x_] = u[x]/.sol[[1]]
```

Now generate the matrix and vector for the linear system to be solved, then solve.

Mathematica

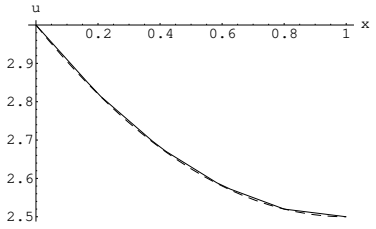


Figure 4.3: Exact and approximate solution to an elementary boundary value problem

```
h=0.2;
x=Table[k*h,{k,0,5}];
(* construct the matrix m *)
m=Table[0,{6},{6}];
m[[1,1]]=1;
For[k=2,k<=5,k++,m[[k,k]]=2/h;m[[k,k-1]]=-1/h;m[[k,k+1]]=-1/h;]
m[[6,5]]=-1/h;m[[6,6]]=1/h;
(* construct the vector b *)
b=Table[-h,{6}];
b[[1]]=3; b[[6]]=-h/2;
(* Solve the equation *)
u=LinearSolve[m,b]
```

With the help of a piecewise linear interpolation we can compare the exact solution and the approximate solution.

Mathematica

```
data=Transpose[{x,u}]
unum=Interpolation[data,InterpolationOrder ->1];
Plot[{uexact[x],unum[x]},{x,0,1},
                AxesLabel->{"x","u"},
PlotStyle->{Dashing[{0.03,0.02}],Dashing[{}]}];
```

As can be seen in figure 4.3 the differences are very small. At the points of support the two solutions actually coincide, but this is due to the fact that the exact solution is a polynomial of degree two. This will usually not occur. ♢

4–2 Example : In chapter 2.2 we considered a horizontal bar with variable cross section $A(x)$, stretched by a force F. The energy of the bar is given by

$$E(u) = \frac{E}{2} \int_0^{100} A(x)\, u'(x)^2 \, dx - F \cdot u(100)$$

This problem can be solved by the method we used for the previous example, except for the force term $F\, u(100)$ at the right end point. This term can easily be incorporated. we divide the bar of length $L = 100$ again in three elements of equal length $L_i = \frac{100}{3}$ and use

$$\vec{x} = \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0 \\ \frac{100}{3} \\ \frac{200}{3} \\ 100 \end{pmatrix} \quad , \quad \vec{u} = \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} u(0) \\ u(\frac{100}{3}) \\ u(\frac{200}{3}) \\ u(100) \end{pmatrix} \quad , \quad \begin{pmatrix} A_0 \\ A_1 \\ A_2 \\ A_3 \end{pmatrix} = \begin{pmatrix} A(0) \\ A(\frac{100}{3}) \\ A(\frac{200}{3}) \\ A(100) \end{pmatrix}$$

After some calculation we obtain the global stiffness matrix

$$\mathbf{K} = \begin{bmatrix} \frac{A_0+A_1}{2\,L_1} & -\frac{A_0+A_1}{2\,L_1} & 0 & 0 \\ -\frac{A_0+A_1}{2\,L_1} & \frac{A_0+A_1}{2\,L_1} + \frac{A_1+A_2}{2\,L_2} & -\frac{A_1+A_2}{2\,L_2} & 0 \\ 0 & -\frac{A_1+A_2}{2\,L_2} & \frac{A_1+A_2}{2\,L_2} + \frac{A_2+A_3}{2\,L_3} & -\frac{A_2+A_3}{2\,L_3} \\ 0 & 0 & -\frac{A_2+A_3}{2\,L_3} & \frac{A_2+A_3}{2\,L_3} \end{bmatrix}$$

and the vector

$$\vec{b} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ F \end{pmatrix}$$

The energy E can now be written as

$$E(\vec{u}) = \frac{1}{2}\,\langle \vec{u}\,,\, \mathbf{K}\vec{u}\rangle - \langle \vec{u}\,,\, \vec{b}\rangle$$

The minimiser (subject to the condition $u(0) = u_0 = 0$) is identical to the solution found in chapter 2.2. $\diamondsuit$

4.4.5 General situation

Equation (3.1) is

$$F(u) = \int_a^b \frac{1}{2}\,a(x)\,(u'(x))^2 + \frac{1}{2}\,b(x)\,u(x)^2 + g(x)\cdot u(x)\;dx$$

and the corresponding ordinary differential equation is

$$\frac{d}{dx}\left(a(x)\,\frac{d\,u(x)}{dx}\right) - b(x)\,u(x) = g(x) \tag{4.6}$$

Sofar we ignored the contribution of the term $b\,u^2$ to the functional. On each element now consider the additional term

$$\begin{aligned} \int_{x_{i-1}}^{x_i} \frac{1}{2}\,b(x)\,u(x)^2\;dx &\approx \frac{1}{2}\left(\frac{b(x_{i-1})\,u_{i-1}^2 + b(x_i)\,u_i^2}{2}\right)\Delta x_i \\ &= \frac{1}{2}\,\langle \begin{pmatrix} u_{i-1} \\ u_i \end{pmatrix}, \begin{bmatrix} \frac{b(x_{i-1})\,\Delta x_i}{2} & 0 \\ 0 & \frac{b(x_i)\,\Delta x_i}{2} \end{bmatrix} \begin{pmatrix} u_{i-1} \\ u_i \end{pmatrix}\rangle \end{aligned}$$

This contribution has to be subtracted in the matrices in the previous sections. With the notations

$$a_i = a(x_i) \quad , \quad b_i = b(x_i) \quad \text{and} \quad g_i = g(x_i)$$

we find the contribution F_i by the i–th element to be

$$F_i = \frac{1}{2}\,\langle \begin{pmatrix} u_{i-1} \\ u_i \end{pmatrix}, \mathbf{K}_i \begin{pmatrix} u_{i-1} \\ u_i \end{pmatrix}\rangle + \langle \begin{pmatrix} u_{i-1} \\ u_i \end{pmatrix}, \begin{pmatrix} \frac{g_{i-1}\,\Delta x_i}{2} \\ \frac{g_i\,\Delta x_i}{2} \end{pmatrix}\rangle$$

where

$$\mathbf{K}_i = \begin{bmatrix} \frac{a_{i-1}+a_i}{2\,\Delta x_i} + \frac{b_{i-1}\,\Delta x_i}{2} & -\frac{a_{i-1}+a_i}{2\,\Delta x_i} \\ -\frac{a_{i-1}+a_i}{2\,\Delta x_i} & \frac{a_{i-1}+a_i}{2\,\Delta x_i} + \frac{b_i\,\Delta x_i}{2} \end{bmatrix}$$

These contributions from the individual elements have to be added up properly to find the global stiffness matrix $\mathbf{K}$ with the entries $k_{i,j}$ for $0 \le i, j \le n$. We find

$$\begin{aligned} k_{0,0} &= \tfrac{a_0+a_1}{2\,\Delta x_1} + \tfrac{b_0\,\Delta x_1}{2} \\ k_{i,i} &= \tfrac{a_{i-1}+a_i}{2\,\Delta x_i} + \tfrac{b_{i-1}\,\Delta x_i}{2} + \tfrac{a_i+a_{i+1}}{2\,\Delta x_{i+1}} + \tfrac{b_i\,\Delta x_{i+1}}{2} \qquad \text{for} \quad 1 \le i \le n-1 \\ k_{i,i-1} = k_{i,i+1} &= -\tfrac{a_{i-1}+a_i}{2\,\Delta x_i} \qquad \text{for} \quad 0 \le i \le n \\ k_{n,n} &= \tfrac{a_{n-1}+a_n}{2\,\Delta x_n} + \tfrac{b_n\,\Delta x_n}{2} \end{aligned}$$

The right hand side vector $\vec{b}$ is given by

$$b_0 = \frac{g_0\,\Delta x_1}{2} \quad , \quad b_i = \frac{g_i\,(\Delta x_{i+1} + \Delta x_i)}{2} \quad \text{for} \quad 1 \le i \le n-1 \quad \text{and} \quad b_n = \frac{g_n\,\Delta x_n}{2}$$

To find the approximate solution to equation (4.6) we have to solve $(n+1)$ linear equations for $(n+1)$ unknowns.

$$\mathbf{K}\,\vec{u} = \vec{b}$$

But before solving the equations the boundary conditions have to be implemented.

4–3 Example : If the functions a, b and g are constants then the differential equation to be solved is

$$a\,u''(x) - b\,u(x) = g$$

Now we choose a uniform step size $\Delta x_i = h$ and the system of linear equations simplifies drastically.

$$\begin{bmatrix} \frac{a}{h}+\frac{bh}{2} & -\frac{a}{h} & & & & & \\ -\frac{a}{h} & \frac{2a}{h}+bh & -\frac{a}{h} & & & & \\ & -\frac{a}{h} & \frac{2a}{h}+bh & -\frac{a}{h} & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & -\frac{a}{h} & \frac{2a}{h}+bh & -\frac{a}{h} & \\ & & & & -\frac{a}{h} & \frac{a}{h}+\frac{bh}{2} \end{bmatrix} \cdot \begin{pmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_{n-1} \\ u_n \end{pmatrix} = \begin{pmatrix} \frac{gh}{2} \\ g\,h \\ g\,h \\ \vdots \\ g\,h \\ \frac{gh}{2} \end{pmatrix}$$

The numbers not shown are all equal to 0. As an example consider the third equation

$$a\,\left(\frac{-u_1 + 2\,u_2 - u_3}{h}\right) + b\,h\,u_2 = g\,h$$

Generally we find for $1 \le i \le n-1$

$$a\,\left(\frac{-u_{i-1} + 2\,u_i - u_{i+1}}{h^2}\right) + b\,u_i = g \tag{4.7}$$

In this special case the method of finite elements leads to a **finite difference approximation** to the differential equation (4.6). ♢

4.5 Second order element with Gauss integration

The goal is to find a minimiser of the functional

$$F(u) = \int_a^b \frac{1}{2}\,a(x)\,(u'(x))^2 + \frac{1}{2}\,b(x)\,u(x)^2 + g(x)\cdot u(x)\;dx$$

The solution found in the previous section is an approximate solution to the original problem. There are two sources of possible approximation errors:

1. Instead of searching among all smooth functions we considered only piecewise continuous functions. To reduce this error more functions have to be considered. This can be achieved by choosing more, smaller elements. Another option is to use the same size and number of elements but use more general functions on each element, e.g. parabola instead of lines.

2. The integration on each element is not exact, but performed with the trapezoidal rule. To reduce this error one can user better integration routines, e.g. Simpson's rule or Gaussian quadrature.

In this section we will implement improvements in both areas. To simplify the calculations we first examine a standard element for the functional of the above type with $-h \leq x \leq h$. As degrees of freedom we chose the values at the two endpoints and the midpoint. The situation is shown in figure 4.4. We will keep track of the size of the errors as the length h of the element gets closer to 0.

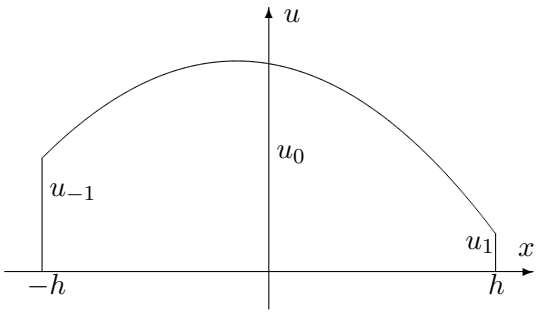


Figure 4.4: Second order element

4.5.1 Linear and quadratic interpolation

A given function $U(x)$ on an interval $a \leq x \leq b$ is discretised by evaluation u at a finite set of nodes x_i. Using the values u_i at those points we try to reconstruct the function by interpolation. This leads to a function $u(x)$. There are different methods of interpolation but the goal is to keep the distance of $U(x)$ and $u(x)$ as small as possible.

Linear interpolation

We want to find a polynomial $u(x)$ of degree 1 such that

$$u(-h/2) = u_{-1} \quad \text{and} \quad u(/2) = u_1$$

It is easy to see that the solution is

$$u(x) = \frac{u_1 + u_{-1}}{2} + \frac{u_1 - u_{-1}}{h}\, x$$

One can verify that for smooth functions the difference between the linear interpolation and the true function is of the order h^2 within the interval $-h/2 \leq x \leq h/2$. The derivative of the linear function is constant

$$u'(x) = \frac{u_1 - u_{-1}}{h}$$

The error for the derivative is of the order h.

Quadratic interpolation

We want to find a polynomial $u(x)$ of degree 2 such that

$$u(-h) = u_{-1} \quad , \quad u(0) = u_0 \quad \text{and} \quad u(h) = u_1$$

There is a variety of methods to find the unique solution

$$u(x) = u_0 + \frac{u_1 - u_{-1}}{2\,h}\,x + \frac{u_1 - 2\,u_0 + u_{-1}}{2\,h^2}\,x^2$$

One can verify that for smooth functions the difference between the quadratic interpolation and the true function is of the order h^3 within the interval $-h \le x \le h$. The derivative is a linear function

$$u'(x) = \frac{u_1 - u_{-1}}{2\,h} + \frac{u_1 - 2\,u_0 + u_{-1}}{h^2}\,x$$

The error for the derivative is of the order h^2.

	linear interpolation	quadratic interpolation
original function	$U(x)$ smooth	$U(x)$ smooth
discretisation	$\downarrow$	$\downarrow$
vector	$\vec{u} \in \mathbb{R}^n$	$\vec{u} \in \mathbb{R}^n$
interpolation	$\downarrow$	$\downarrow$
function	$u(x)$ piecewise linear	$u(x)$ piecewise quadratic
error of function	$u(x) - U(x) = O(h^2)$	$u(x) - U(x) = O(h^3)$
error of derivative	$u'(x) - U'(x) = O(h)$	$u'(x) - U'(x) = O(h^2)$

Table 4.3: Comparison of linear and quadratic interpolation

Linear interpolation leads to errors of the order h^2 for the function and order h for the derivative. Thus quadratic interpolation is a clear improvement if h is sufficiently small. The result are shown in table 4.3.

4.5.2 Gauss integration

Gauss integration with two nodes

If we integrate a know function $f(x)$ over a given interval of length h (e.g. $-\frac{h}{2} \le x \le \frac{h}{2}$) we can also evaluate the function at two nodes in the interval and replace the function by a straight line passing through those points. For the classical trapezoidal rule the nodes are the two endpoints. The main idea of Gauss integration is to choose the nodes such that as many polynomials as possible are integrated correctly. Due to symmetry it is reasonable to choose the two nodes at $\pm x_1$ with equal weight w_1. The situation is shown in figure 4.5. Then the integral is approximated by

$$w_1\,f(-x_1) + w_1\,f(x_1) \approx \int_{-h/2}^{h/2} f(x)\,dx$$

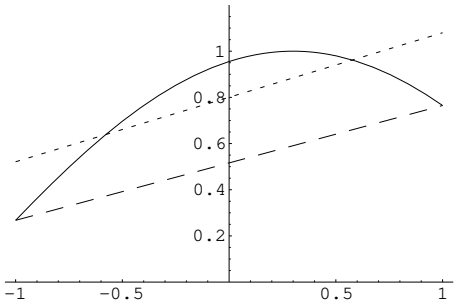


Figure 4.5: Gauss integration and trapezoidal rule

The table below is looking at monomials of increasing degree and verifying that the integral is computed exactly

$f(x)$	$w_1\, f(-x_1) + w_1\, f(x_1) = \int_{-h/2}^{h/2} f(x)\, dx$	resulting equation
1	$w_1 + w_1 = h$	$2\, w_1 = h$
x	$-w_1\, x_1 + w_1\, x_1 = 0$	
x^2	$w_1\, x_1^2 + w_1\, x_1^2 = \frac{1}{12}\, h^3$	$2\, w_1\, x_1^2 = \frac{1}{12}\, h^3$
x^3	$-w_1\, x_1^3 + w_1\, x_1^3 = 0$	

The two conditions in the above table lead to $w_1 = \frac{h}{2}$ and $x_1 = \frac{1}{2\sqrt{3}}\, h$. Thus we find the approximation

$$\int_{-h/2}^{h/2} f(x)\, dx \approx \frac{h}{2}\ \left(f(-\frac{1}{2\sqrt{3}}\, h) + f(\frac{1}{2\sqrt{3}}\, h) \right)$$

The monomial x^4 will not be integrated exactly. We find

$$w_1\, x_1^4 + w_1\, x_1^4 = \frac{1}{16 \cdot 9}\, h^5 \quad \text{but} \quad \int_{-h/2}^{h/2} x^4\, dx = \frac{1}{5 \cdot 16}\, h^5$$

Now we integrate a general function by replacing it by its Taylor approximation

$$f(x) \approx f(0) = f'(0)\, x + \frac{f''(0)}{2}\, x^2 + \frac{f^{(3)}(0)}{6}\, x^3 + \frac{f^{(4)}(0)}{24}\, x^4 + O(x^5)$$

$$\begin{aligned} f(x) &= f(0) + f'(0)\, x + \frac{f''(0)}{2}\, x^2 + \frac{f^{(3)}(0)}{6}\, x^3 + \frac{f^{(4)}(0)}{24}\, x^4 + O(x^5) \\ \int_{-h/2}^{h/2} f(x)\, dx &= \int_{-h/2}^{h/2} f(0) + f'(0)\, x + \frac{f''(0)}{2}\, x^2 + \frac{f^{(3)}(0)}{6}\, x^3 + \frac{f^{(4)}(0)}{24}\, x^4\, dx + O(h^6) \end{aligned}$$

Combining the above we find

$$w_1\, f(-x_1) + w_1\, f(x_1) - \int_{-h/2}^{h/2} f(x)\, dx = \frac{f^{(4)}(0)}{24 \cdot 16}\ \left(\frac{1}{9} - \frac{1}{5} \right)\ h^5 + O(h^6)$$

Thus the local integration error is of the order h^5. The standard trapezoidal rule (nodes at endpoints) will generate an error integrating x^2 and the integration error is of the order h^3. For small values of h integration by Gauss is clearly superior.

Gauss integration with three nodes

Instead of two nodes we can consider three nodes at $x_0 = 0$ and $\pm x_1$ with corresponding weights and approximate the integral by

$$w_1\, f(-x_1) + w_0\, f(0) + w_1\, f(x_1) \approx \int_{-h/2}^{h/2} f(x)\, dx$$

As before we generate a table of integrals for monomials of increasing order to find equations for w_0, w_1 and x_1.

$f(x)$	$w_1\, f(-x_1) + w_0\, f(0) + w_1\, f(x_1) = \int_{-h/2}^{h/2} f(x)\, dx$	resulting equation
1	$w_1 + w_0 + w_1 = h$	$w_0 + 2\, w_1 = h$
x	$-w_1\, x_1 + w_1\, x_1 = 0$	
x^2	$w_1\, x_1^2 + w_1\, x_1^2 = \frac{1}{12}\, h^3$	$2\, w_1\, x_1^2 = \frac{1}{12}\, h^3$
x^3	$-w_1\, x_1^3 + w_1\, x_1^3 = 0$	
x^4	$w_1\, x_1^4 + w_1\, x_1^4 = \frac{1}{16\cdot 5}\, h^5$	$2\, w_1\, x_1^4 = \frac{1}{80}\, h^5$
x^5	$-w_1\, x_1^5 + w_1\, x_1^5 = 0$	

Dividing the last two equations yields $x_1^2 = \frac{12}{80}\, h^2$ or $x_1 = \frac{\sqrt{3}}{2\sqrt{5}}\, h$. Using this we find $w_1 = \frac{h^3}{24\, x_1^2} = \frac{5}{18}\, h$. and the $w_0 = \frac{8}{18}\, h$. We have the approximation

$$\int_{-h/2}^{h/2} f(x)\, dx \approx \frac{h}{18}\, \left(5\, f(-\frac{\sqrt{3}}{2\sqrt{5}}\, h) + 8\, f(0) + 5\, f(\frac{\sqrt{3}}{2\sqrt{5}}\, h) \right)$$

The monomial x^6 will not be integrated exactly and thus the local integration error is of the order h^7. *Mathematica* has a package to generate all the information computed by hand above.

Mathematica

```
Needs["NumericalMath`GaussianQuadrature`"]
GaussianQuadratureWeights[3, -h/2, h/2]
GaussianQuadratureError[3,f, -h/2, h/2]
.
{{-0.387298 h,0.277778 h},{0,0.444444 h},{0.387298 h,0.277778 h}}

            -7  7  (6)
-4.96032 10   h  f
```

If the integration is performed over an integral of length $2h$ then the coefficients have to be adapted. The result is

$$\int_{-h}^{h} f(x)\, dx \approx \frac{h}{9}\, \left(5\, f(-\frac{\sqrt{3}}{\sqrt{5}}\, h) + 8\, f(0) + 5\, f(\frac{\sqrt{3}}{\sqrt{5}}\, h) \right)$$

Gauss integration is a very powerful tool. As an example verify that the integral of $\sin x$ from 0 to $\pi/2$ with three nodes is computed with an error smaller than 10^{-5}. As a consequence Gauss integration is used almost exclusively for finite element problems. Gauss will not perform well if there are edges **within** one element as the expression will typically not be often differentiable. This situation can be avoided when choosing the location of the elements. If Gauss integration is used properly the errors due to approximate integration can be kept very small.

We will from now on assume that the integral are computed exactly, knowing that the error should be small.

4.5.3 Construction of an improved element

The goal is to combine the results of the two previous sections and construct an elements with

- three internal degrees of freedom and a quadratic interpolation within the element.
- Gaussian integration with three nodes to evaluate the functional.

If for given functions $a(x)$, $b(x)$ and $g(x)$ the expression

$$F(u) = \int_{-h}^{h} \frac{1}{2}\, a(x)\, (u'(x))^2 + \frac{1}{2}\, b(x)\, u(x)^2 + g(x) \cdot u(x)\; dx$$

has to be written in the form

$$\frac{1}{2}\, \langle \vec{u}\,,\, \mathbf{K}\, \vec{u} \rangle - \langle \vec{u}\,,\, \vec{b} \rangle$$

Now we consider the three contributions in the above integral separately.

Integrating the expression $\int b(x)\, u(x)^2\; dx$

Since it is the easiest term we first want to apply the above ideas to the expression

$$2\, F_2(u) = \int_{-h}^{h} b(x)\, u(x)^2\; dx$$

only. The path of computations for one element is as follows

1. discretise and construct an interpolating function
2. evaluate the interpolated function at the nodes needed for Gauss integration
3. evaluate the integral

The obvious discretisation is $u_{-1} = u(-h)$, $u_0 = u(0)$ and $u_1 = u(h)$. The interpolated function is given by

$$u(x) = u_0 + \frac{u_1 - u_{-1}}{2\, h}\; x + \frac{u_1 - 2\, u_0 + u_{-1}}{2\, h^2}\; x^2$$

The Gaussian integration points are at $x_{-1} = -\frac{\sqrt{3}}{\sqrt{5}}\, h$, $x_0 = 0$ and $x_1 = \frac{\sqrt{3}}{\sqrt{5}}\, h$. The function to be integrated has to be evaluated at the three nodes. As an intermediate result we compute u at the nodes and obtain the values p_{-1}, p_0 and p_1.

$$\begin{aligned}
\begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix} &= \begin{pmatrix} u(x_{-1}) \\ u(x_0) \\ u(x_1) \end{pmatrix} = \begin{pmatrix} u_0 + \frac{u_1 - u_{-1}}{2\, h}\; x_{-1} + \frac{u_1 - 2\, u_0 + u_{-1}}{2\, h^2}\; x_{-1}^2 \\ u_0 \\ u_0 + \frac{u_1 - u_{-1}}{2\, h}\; x_1 + \frac{u_1 - 2\, u_0 + u_{-1}}{2\, h^2}\; x_1^2 \end{pmatrix} \\
&= \begin{bmatrix} -\frac{x_{-1}}{2\, h} + \frac{x_{-1}^2}{2\, h^2} & 1 - \frac{x_{-1}^2}{h^2} & \frac{x_{-1}}{2\, h} + \frac{x_{-1}^2}{2\, h^2} \\ 0 & 1 & 0 \\ -\frac{x_1}{2\, h} + \frac{x_1^2}{2\, h^2} & 1 - \frac{x_1^2}{h^2} & \frac{x_1}{2\, h} + \frac{x_1^2}{2\, h^2} \end{bmatrix} \cdot \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix} \\
&= \begin{bmatrix} \frac{\sqrt{3}}{2\sqrt{5}} + \frac{3}{10} & \frac{2}{5} & -\frac{\sqrt{3}}{2\sqrt{5}} + \frac{3}{10} \\ 0 & 1 & 0 \\ -\frac{\sqrt{3}}{2\sqrt{5}} + \frac{3}{10} & \frac{2}{5} & \frac{\sqrt{3}}{2\sqrt{5}} + \frac{3}{10} \end{bmatrix} \cdot \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix} = \mathbf{M}_0 \cdot \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix}
\end{aligned}$$

The matrix $\mathbf{M}_0$ allows to find the values of the interpolated function at the Gauss nodes, when the values of the function at the end and mid points are known. It is worth observing that the matrix $\mathbf{M}_0$ contains constants only. This matrix is one part of the description of the element. Using these values p_i we can find the values of the function to be integrated

$$\begin{aligned} b(x_{-1})\,u(x_{-1})^2 &= b(x_{-1})\,p_{-1}^2 \\ b(x_0)\,u(x_0)^2 &= b(0)\,p_0^2 \\ b(x_1)\,u(x_1)^2 &= b(x_1)\,p_1^2 \end{aligned}$$

The integral is (approximately) given by

$$\begin{aligned} \int_{-h}^{h} b(x)\,u(x)^2\,dx &\approx \frac{h}{9}\,\left(5\,b(x_{-1})\,p_{-1}^2 + 8\,b(x_0)\,p_0^2 + 5\,b(x_1)\,p_1^2\right) \\ &= \langle \begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix}, \begin{bmatrix} \frac{5\,h}{9}\,b(x_{-1}) & 0 & 0 \\ 0 & \frac{8\,h}{9}\,b(x_0) & 0 \\ 0 & 0 & \frac{5\,h}{9}\,b(x_1) \end{bmatrix} \begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix} \rangle \\ &= \langle \mathbf{M}_0 \cdot \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix}, \begin{bmatrix} \frac{5\,h}{9}\,b(x_{-1}) & 0 & 0 \\ 0 & \frac{8\,h}{9}\,b(x_0) & 0 \\ 0 & 0 & \frac{5\,h}{9}\,b(x_1) \end{bmatrix} \cdot \mathbf{M}_0 \cdot \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix} \rangle \\ &= \langle \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix}, \mathbf{M}_0^T \cdot \begin{bmatrix} \frac{5\,h}{9}\,b(x_{-1}) & 0 & 0 \\ 0 & \frac{8\,h}{9}\,b(x_0) & 0 \\ 0 & 0 & \frac{5\,h}{9}\,b(x_1) \end{bmatrix} \cdot \mathbf{M}_0 \cdot \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix} \rangle \end{aligned}$$

This notation incorporates the steps discretisation, interpolation and Gauss integration in one formula.

Integrating the expression $\int a(x)\,(u'(x))^2\,dx$

Now we compute

$$2\,F_1(u) = \int_{-h}^{h} a(x)\,(u'(x))^2\,dx$$

by a very similar method. The only additional problem stems from using the derivative u' instead of u at the Gauss nodes. The derivative of the interpolated function is

$$u'(x) = \frac{u_1 - u_{-1}}{2\,h} + \frac{u_1 - 2\,u_0 + u_{-1}}{h^2}\,x$$

We find

$$\begin{aligned} \begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix} &= \begin{pmatrix} u'(x_{-1}) \\ u'(x_0) \\ u'(x_1) \end{pmatrix} = \begin{pmatrix} \frac{u_1-u_{-1}}{2\,h} + \frac{u_1-2\,u_0+u_{-1}}{h^2}\,x_{-1} \\ \frac{u_1-u_{-1}}{2\,h} \\ \frac{u_1-u_{-1}}{2\,h} + \frac{u_1-2\,u_0+u_{-1}}{h^2}\,x_1 \end{pmatrix} \\ &= \begin{bmatrix} -\frac{1}{2\,h} - \frac{\sqrt{3}}{\sqrt{5}\,h} & \frac{2\sqrt{3}}{\sqrt{5}\,h} & \frac{1}{2\,h} - \frac{\sqrt{3}}{\sqrt{5}\,h} \\ -\frac{1}{2\,h} & 0 & \frac{1}{2\,h} \\ -\frac{1}{2\,h} + \frac{\sqrt{3}}{\sqrt{5}\,h} & -\frac{2\sqrt{3}}{\sqrt{5}\,h} & \frac{1}{2\,h} + \frac{\sqrt{3}}{\sqrt{5}\,h} \end{bmatrix} \cdot \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix} = \mathbf{M}_1 \cdot \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix} \end{aligned}$$

Except for the division by h the matrix $\mathbf{M}_1$ is constant. As above we find

$$\int_{-h}^{h} a(x)\,(u'(x))^2\,dx \approx \langle \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix}, \mathbf{M}_1^T \begin{bmatrix} \frac{5\,h}{9}\,a(x_{-1}) & 0 & 0 \\ 0 & \frac{8\,h}{9}\,a(x_0) & 0 \\ 0 & 0 & \frac{5\,h}{9}\,a(x_1) \end{bmatrix} \mathbf{M}_1 \cdot \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix} \rangle$$

Integrating the expression $\int g(x)\,u(x)\,dx$

Now we consider

$$F_3(u) = \int_{-h}^{h} g(x)\,u(x)\,dx$$

This is comparable to the first integral to be computed. Again we have to compute the values of the function u at the Gauss nodes by

$$\begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix} = \begin{pmatrix} u(x_{-1}) \\ u(x_0) \\ u(x_1) \end{pmatrix} = \begin{bmatrix} \frac{\sqrt{3}}{2\sqrt{5}} + \frac{3}{10} & \frac{2}{5} & -\frac{\sqrt{3}}{2\sqrt{5}} + \frac{3}{10} \\ 0 & 1 & 0 \\ -\frac{\sqrt{3}}{2\sqrt{5}} + \frac{3}{10} & \frac{2}{5} & \frac{\sqrt{3}}{2\sqrt{5}} + \frac{3}{10} \end{bmatrix} \cdot \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix} = \mathbf{M}_0 \cdot \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix}$$

Using these values p_i and the values of g at the Gauss nodes to find

$$\begin{aligned} g(x_{-1})\,u(x_{-1}) &= b(x_{-1})\,p_{-1} \\ g(x_0)\,u(x_0) &= b(0)\,p_0 \\ g(x_1)\,u(x_1) &= b(x_1)\,p_1 \end{aligned}$$

The integral is (approximately) given by

$$\begin{aligned} \int_{-h}^{h} g(x)\,u(x)\,dx &\approx \frac{h}{9}\,\left(5\,g(x_{-1})\,p_{-1} + 8\,g(x_0)\,p_0 + 5\,g(x_1)\,p_1\right) \\ &= \left\langle \begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix}, \begin{pmatrix} \frac{5h}{9}\,g(x_{-1}) \\ \frac{8h}{9}\,g(x_0) \\ \frac{5h}{9}\,g(x_1) \end{pmatrix} \right\rangle = \left\langle \begin{pmatrix} u_{-1} \\ u_0 \\ u_1 \end{pmatrix}, \mathbf{M}_0^T \cdot \begin{pmatrix} \frac{5h}{9}\,g(x_{-1}) \\ \frac{8h}{9}\,g(x_0) \\ \frac{5h}{9}\,g(x_1) \end{pmatrix} \right\rangle \end{aligned}$$

Combining all contributions

Finally we can combine the results above and arrive at

$$F(u) = \int_{-h}^{h} \frac{1}{2}\,a(x)\,(u'(x))^2 + \frac{1}{2}\,b(x)\,u(x)^2 + g(x)\cdot u(x)\,dx = \frac{1}{2}\,\langle \vec{u}\,,\,\mathbf{K}\,\vec{u}\rangle - \langle \vec{u}\,,\,\vec{b}\rangle$$

with

$$\mathbf{K} = \mathbf{M}_1^T \begin{bmatrix} \frac{5h}{9}\,a(x_{-1}) & 0 & 0 \\ 0 & \frac{8h}{9}\,a(x_0) & 0 \\ 0 & 0 & \frac{5h}{9}\,a(x_1) \end{bmatrix} \mathbf{M}_1 + \mathbf{M}_0^T \cdot \begin{bmatrix} \frac{5h}{9}\,b(x_{-1}) & 0 & 0 \\ 0 & \frac{8h}{9}\,b(x_0) & 0 \\ 0 & 0 & \frac{5h}{9}\,b(x_1) \end{bmatrix} \cdot \mathbf{M}_0$$

and

$$\vec{b} = -\mathbf{M}_0^T \cdot \begin{pmatrix} \frac{5h}{9}\,g(x_{-1}) \\ \frac{8h}{9}\,g(x_0) \\ \frac{5h}{9}\,g(x_1) \end{pmatrix}$$

where

$$\mathbf{M}_1 = \begin{bmatrix} -\frac{1}{2\,h} - \frac{\sqrt{3}}{\sqrt{5}\,h} & \frac{2\sqrt{3}}{\sqrt{5}\,h} & \frac{1}{2\,h} - \frac{\sqrt{3}}{\sqrt{5}\,h} \\ -\frac{1}{2\,h} & 0 & \frac{1}{2\,h} \\ -\frac{1}{2\,h} + \frac{\sqrt{3}}{\sqrt{5}\,h} & -\frac{2\sqrt{3}}{\sqrt{5}\,h} & \frac{1}{2\,h} + \frac{\sqrt{3}}{\sqrt{5}\,h} \end{bmatrix} \quad \text{and} \quad \mathbf{M}_0 = \begin{bmatrix} \frac{\sqrt{3}}{2\sqrt{5}} + \frac{3}{10} & \frac{2}{5} & -\frac{\sqrt{3}}{2\sqrt{5}} + \frac{3}{10} \\ 0 & 1 & 0 \\ -\frac{\sqrt{3}}{2\sqrt{5}} + \frac{3}{10} & \frac{2}{5} & \frac{\sqrt{3}}{2\sqrt{5}} + \frac{3}{10} \end{bmatrix}$$

Now we have a complete description of an element with second order interpolation and Gauss integration with three nodes.

Results from linear algebra show that the matrix **K** is positive definite if the two function $a(x)$ and $b(x)$ are strictly positive. Thus the matrix **K** is invertible. If we only know that a is strictly positive and b might be zero, then **K** is known to be positive semidefinite.

4.5.4 Comparison of interpolation and integration methods

In table 4.4 find a simple comparison of first and second order elements with different integration methods. The table indicates that Gauss integration leads to very small integration errors if h is small. Thus Gauss integration is used almost exclusively in finite element constructions. The discretisation error is usually a much larger contribution to the total error and we will concentrate on its influence in the next section.

	first order element	second order element
Degrees of freedom	2	3
Length of one element	h	$2\,h$
Interpolation of $u(x)$ by	straight line	parabola
Order of error of $u(x)$	h^2	h^3
Derivative $u'(x)$ is	constant	linear
Order of error of $u'(x)$	h	h^2
Integration method	trapezoidal	Simpson's rule
Local integration error	h^3	h^4
Integration method		Gauss, 3 points
Local integration error		h^7

Table 4.4: Comparison of interpolation errors of first and second order elements

Sofar we considered elements of first and second order only, but the methods do apply to higher order elements too. Some applications (e.g. bending of beams or plates) **require** higher order elements. An example is examined in section 4.8.

4.6 Code in *Mathematica* for second order boundary value problems

In figure 4.6 find *Mathematica* code to solve a second order boundary value problem on an interval. This package has to be loaded by `<<BVP.m` . The differential equation

$$\frac{d}{dx}\left(a(x)\,\frac{d\,u(x)}{dx}\right) - b(x)\,u(x) = f(x)$$

can be solved on an interval with either Neumann or Dirichlet boundary conditions. As an example consider the equation

$$\begin{aligned} -u''(x) - 2\,u(x) &= -x^3 \qquad \text{for} \quad 0 < x < 1 \\ u(0) &= 0 \\ u(1) &= 0.1 \end{aligned}$$

to be solved with 3 elements (4 points) by the code below.

Mathematica

```
<<BVP.m;
Clear[a,b,c,f,x,n]
a=Function[x,-1];
b=Function[x,2];
f=Function[x,-x^3];
n=3;
x=Table[t,{t,0,1,1/n}];
data=BVP[a,b,f,x,{"D","D"},{0,0.1}];
```

The result `data` contains a list of values of independent and dependent variables. With this information the solution can be plotted by `ListPlot[]` and combined with the exact solution.

Mathematica

```
g1=ListPlot[data,PlotStyle -> PointSize[0.02],DisplayFunction ->Identity] ;
yn[t_]=y[t]/.DSolve[{D[a[t]*y'[t],t]-b[t]*y[t]==f[t],
                              y[0]==0,y[1]==0.1},y[t],t];
g2=Plot[yn[t],{t,0,1},DisplayFunction -> Identity];
Show[{g1,g2},PlotRange -> All,DisplayFunction ->$DisplayFunction];
```

Since the second order finite element solution is based on piecewise quadratic interpolation we define a *Mathematica* function to compute this interpolation using the result of the `BVP[]` command. Find the content of `Interpol2.m` in figure 4.7. With the command in this package we can then plot the solution and its derivative. In figure 4.8 find plots of the FEM solution and the exact solution generated by

Mathematica

```
<<Interpol2.m
Clear[x]
yfem[x_]:=Interpol2[x,data];
Plot[{yfem[x],yn[x]},{x,0,1}];
dyfem[x_]:=DInterpol2[x,data];  dyn[x_]=D[yn[x],x];
Plot[{dyfem[x],dyn[x]},{x,0,1}, PlotRange ->All];
```

The right half in figure 4.8 clearly shows that the derivative of the finite element solution is a piecewise linear, **non**continuous function, as the derivative 'jumps'.

Mathematica

```
BeginPackage["BVP`"];
BVP::usage="BVP[a,b,f,xvalues,BCType,BCValues]
solves a second order boundary value problem";

Begin["`Private`"];
M0={{0.6872983346207417, 0.4,  -0.08729833462074164},  {0, 1, 0},
  {-0.08729833462074164,  0.4, 0.6872983346207417}};
M1={{-1.274596669241483, 1.549193338482966, -0.2745966692414834},
    {-(1/2), 0, 1/2},
    {0.2745966692414834, -1.549193338482966, 1.274596669241483}};

ElementStiffnessMatrix[a_,b_,{xl_,xr_}] :=
  Module[{pl,pc,pr,h,avalues,bvalues},
     pc = (xl+xr)/2;  h=(xr-xl)/2;
     {pl,pr}={pc-Sqrt[3/5]*h,pc+Sqrt[3/5]*h};
     avalues=Map[a,{pl,pc,pr}];
     bvalues=Map[b,{pl,pc,pr}];
     Return[Transpose[M1].DiagonalMatrix[avalues*{5,8,5}/9].M1/h+
            Transpose[M0].DiagonalMatrix[bvalues*{5,8,5}/9*h].M0];]
ElementRHS[f_,{xl_,xr_}] :=
   Module[{pl,pc,pr,w,h,fvalues},
     pc = (xl+xr)/2;   h=(xr-xl)/2;
     {pl,pr}={pc-Sqrt[3/5]*h,pc+Sqrt[3/5]*h};
     fvalues=Map[f,{pl,pc,pr}];
     Return[-Transpose[M0].(fvalues*{5,8,5})/9*h]]
BVP[a_,b_,f_,xvalues_,BCType_,BCValues_]:=
    Module[{n,x,u,M,mElement,RHS,mRHS},
      n=Length[xvalues];  x=Table[0,{2*n-1}];
      For[e=1,e<=n,e++,x[[2*e-1]]=xvalues[[e]]];
         For[e=1,e<n,e++, x[[2*e]]=(xvalues[[e]]+xvalues[[e+1]])/2];
      M=Table[0,{2*n-1},{2*n-1}];
      RHS=Table[0,{2*n-1}];
      For[e=1,e<n,e++,
        mElement=N[ElementStiffnessMatrix[a,b,
                              {xvalues[[e]],xvalues[[e+1]]}]];
        For[i=1,i<=3,i++,
          For[j=1,j<=3,j++, M[[(e-1)*2+i,(e-1)*2+j]]+=mElement[[i,j]]]];
        mRHS=N[ElementRHS[f,{xvalues[[e]],xvalues[[e+1]]}]];
        For[i=1,i<=3,i++, RHS[[(e-1)*2+i]]+=mRHS[[i]]];
      ];
      Which[BCType[[1]]=="D"||BCType[[1]]=="d",
        M[[1]]=Table[0,{2*n-1}];M[[1,1]]=1;RHS[[1]]=BCValues[[1]],
        BCType[[1]]=="N"||BCType[[1]]=="n", RHS[[1]]-=BCValues[[1]]];
      Which[BCType[[2]]=="D"||BCType[[2]]=="d",
        M[[2*n-1]]=Table[0,{2*n-1}];M[[2*n-1,2*n-1]]=1;
                                    RHS[[2*n-1]]=BCValues[[2]],
        BCType[[2]]=="N"||BCType[[2]]=="n", RHS[[2*n-1]]-=BCValues[[2]]];
      u=LinearSolve[M,RHS];
      Return[Transpose[{x,u}]];];
End[];
EndPackage[];
```

Figure 4.6: `BVP.m`, *Mathematica* code to solve a boundary value problem

Mathematica

```
BeginPackage["Interpol2`"];
Interpol2::usage="Interpol2[x,data]
returns the value of the piecewise quadratic interpolation function,
determined by the given data";
DInterpol2::usage="DInterpol2[x,data]
returns the value of the derivative of the piecewise quadratic interpolation
function, determined by the given data";

Begin["`Private`"];
Interpol2[x_,data_]:= Module[{xp,yp,pos,f},
                {xp,yp}=Transpose[data];
        pos=Floor[(Length[Select[xp,(#<=x)&]]+1)/2];
        f=Interpolation[data[[{2*pos-1,2*pos,2*pos+1}]],InterpolationOrder -> 2];
        f[x]]
DInterpol2[x_,data_]:= Module[{xp,yp,pos,f},
                {xp,yp}=Transpose[data];
        pos=Floor[(Length[Select[xp,(#<=x)&]]+1)/2];
        f=Interpolation[data[[{2*pos-1,2*pos,2*pos+1}]],InterpolationOrder -> 2];
        f'[x]]
End[];
EndPackage[];
```

Figure 4.7: `Interpol2.m`, *Mathematica* code to compute piecewise quadratic interpolation

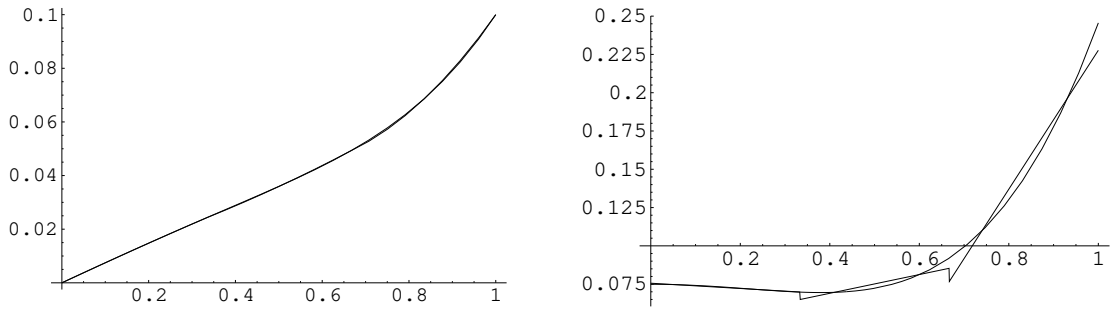


Figure 4.8: Test problem for the second order finite element, plot of solutions and its derivatives

4.7 Examples

4.7.1 The FEM solution to the standard truss problem

To test the produces in the previous section we use the example in section 2.2.2. The solution with 5 elements is now easily computed by

Mathematica

```
<<BVP.m;
<<Interpol2.m;
Clear[A,b,f,x]
A=Function[x,-(10-0.09*x)*3*10^6];
b=Function[x,0];
f=Function[x,0];
n=5;
x=Table[s,{s,0,100,100/n}];
data=BVP[A,b,f,x,{"D","N"},{0,2*10^4}];
```

and leads to the solution shown in figure 4.9. This has to be compared with figure 2.7 on page 25. There you find the solution with 10 elements of order 1. Both approaches require to solve a system of 11 linear equations and thus the computational effort is comparable. But the second order elements lead to a better approximation¿ To vervy this, compare the two graphs for the strain.

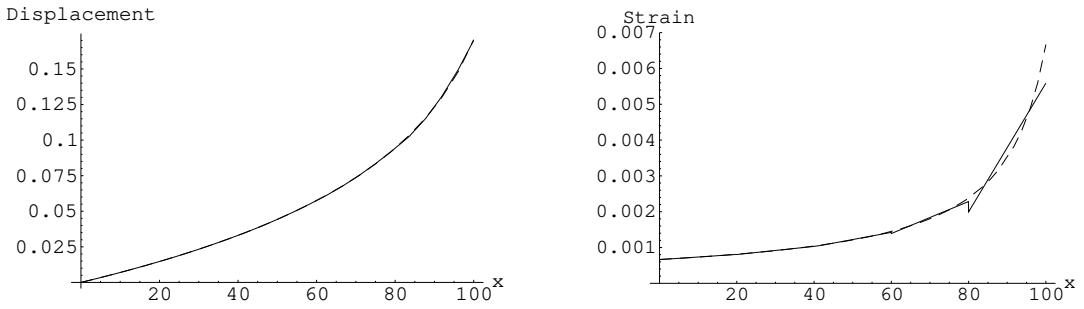


Figure 4.9: Displacement and strain in a truss with 5 elements of order 2

In figure 4.10 find the relative error of the displacement function. It can be seen that the error are larger towards the right end point and thus more elements should be placed there. The accuracy in the left half might be sufficient already.

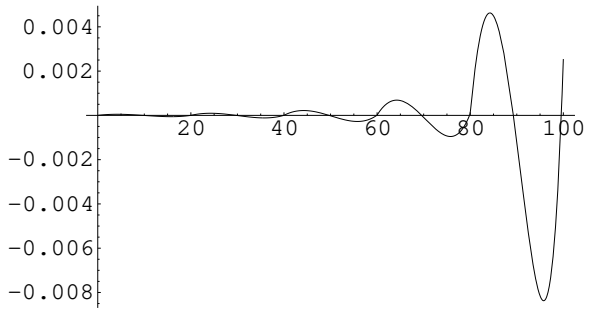


Figure 4.10: Relative error of the displacement function

4.7.2 Radial heat problem

In section 4.1.4 (page 65) the boundary value problem

$$\frac{\partial}{\partial\rho}(\rho\frac{\partial\, T(\rho)}{\partial\rho}) = \rho\, T''(\rho) + T'(\rho) \;=\; -\rho\, f(\rho) \;=\; \begin{cases} 0 & \text{if} \quad 0 \le \rho < 1 \\ -\rho & \text{if} \quad 1 \le \rho < 2 \\ 0 & \text{if} \quad 0 \le \rho \le 3 \end{cases}$$

$$\begin{aligned} T(3) &= 0 \\ T'(0) &= 0 \end{aligned}$$

had to be solved for the unknown temperature T as function of the radius ρ. The solution can now be computed by the code below. Figure 4.11 shows the approximate and the exact solution. The exact solution is shown as a solid line and the FEM solution is indicated by the points. A simple computation (with *Mathematica*) show that the relative error of the approximation is smaller than 0.3%.

Mathematica

```
<<BVP.m;
Clear[a,b,c,f,x,t,n];
a=Function[x,x];
b=Function[x,0];
f=Function[x,Which[x<1,0,x<2,-x,True,0]];
n=3;
x=Table[t,{t,0,3,3/n}];
data=BVP[a,b,f,x,{"N","D"},{0,0}];
ListPlot[data,PlotStyle -> PointSize[0.01],DisplayFunction -> Identity];
```

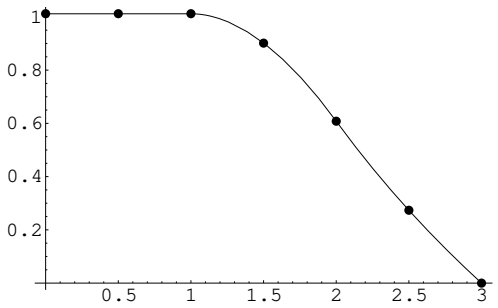


Figure 4.11: Solution of radial heat equation

In this example it is important to choose the elements such that the two points $x = 1$ and $x = 2$ are separating different elements. The reason are the discontinuities of the function $f(x)$ at those points. If they would be inside an element, then the solution can not be often differentiable on the individual elements and thus the high accuracy of Gauss integration would not apply. This can be experimentally verified by the above code, e.g. set $n = 10$ and observe that the error is considerably larger than for $n = 3$.

4.8 Vibrations of a beam

We want to examine vibration of a beam which is clamped at one end and free at the other end. In particular we want to find the resonance frequencies of such a bar. This information is often useful when designing a mechanical system.

4.8.1 Description of the static situation

We examine the transversal deformation of an elastic bar in horizontal position. Its displacement is given by a function $u(x)$ where $0 \leq x \leq L$.

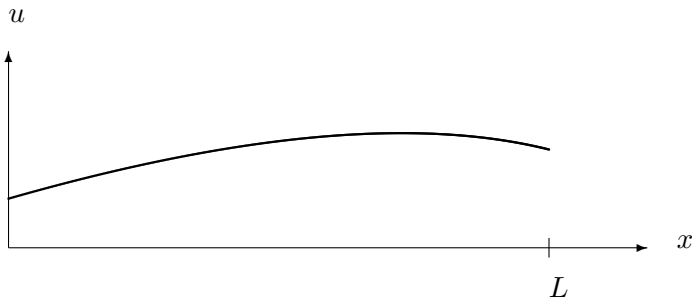


Figure 4.12: Bending of a beam

The basic equations from physics are

$$\begin{aligned} EI\, u''(x) &= M(x) \\ EI\, u^{(4)}(x) &= \frac{d^2}{dx^2} M(x) = f(x) \end{aligned}$$

where the meaning of the symbols is given in table 4.5.

symbol	description	units
x	horizontal coordinate $0 \leq x \leq L$	m
y	vertical displacement	m
L	length of the beam	m
I	moment of inertia of the cross section	m^4
E	modulus of elasticity	$\frac{N}{m^2}$
f	vertical force per length	$\frac{N}{m}$
M	bending moment	$N\,m$
ρ	mass per length	$\frac{kg}{m}$

Table 4.5: Description of the physical quantities

The static situation can be considered as a minimization problem with the functional

$$F(u) = \int_0^L \frac{E\,I(x)}{2} \left((u''(x)\right)^2 - f(x) \cdot u(x)\, dx$$

Calculus of variations and the notation $a(x) = E\,I(x)$ imply

$$\begin{aligned} 0 &= \int_0^L a(x)\,u''(x)\cdot\varphi''(x) - f(x)\cdot\varphi(x)\,dx \\ &= a(x)\,u''(x)\cdot\varphi'(x)\,\big|_{x=0}^{L} - \int_0^L \big(a(x)\,u''(x)\big)'\cdot\varphi'(x) - f(x)\cdot\varphi(x)\,dx \\ &= \big(a(x)\,u''(x)\cdot\varphi'(x) - (a(x)\,u''(x))'\cdot\varphi(x)\big)\,\big|_{x=0}^{L} + \int_0^L \big(a(x)\,u''(x)\big)''\cdot\varphi(x) - f(x)\cdot\varphi(x)\,dx \end{aligned}$$

The above expression has to vanish for all smooth test-functions $\varphi(x)$ with $\varphi(0) = \varphi'(0) = 0$. Thus we obtain

$$\begin{aligned} (a(x)\,u''(x))'' &= f(x) \quad \text{for all} \quad 0 < x < L \\ u(0) = u'(0) &= 0 \\ a(L)\,u''(L) &= 0 \\ (a(L)\,u''(L))' &= 0 \end{aligned}$$

Using $a(L) > 0$ the boundary condition simplifies to

$$\begin{aligned} (a(x)\,u''(x))'' &= f(x) \quad \text{for all} \quad 0 < x < L \\ u(0) = u'(0) &= 0 \\ u''(L) = u'''(L) &= 0 \end{aligned}$$

For a given force density f we have to solve an ordinary differential equation of order 4 with boundary conditions.

4.8.2 Dynamic situation, separation of variables

Now we assume that the vertical displacement $u(t,x)$ may depend on the space variable x and the time t. For the static situation we considered the force $f\,\Delta x$ applied to a section of length Δx of the beam. For the beam to remain static the beam has to generate an internal elastic force of the same size, but opposite sign. If the are no external forces then this internal force will lead to an acceleration of this section of the bar. Using Newton's law we then arrive at a partial differential equation of order 4.

$$\rho\,\ddot{u}(t,x) = -\big(EI\,u''\big)''\,(t,x) \tag{4.8}$$

To find solutions of this equation we use the idea of **separation of variables**. We look for a solution in the form of a product of a function of t and a function of x only, i.e.

$$u(t,x) = T(t)\cdot y(x)$$

This ansatz will be used in the equation (4.8). We obtain

$$\begin{aligned} \rho\,\frac{\partial^2}{\partial t^2}\,(T(t)\cdot y(x)) &= -\frac{\partial^2}{\partial x^2}\left(EI\,\cdot\frac{\partial^2}{\partial x^2}\,(T(t)\cdot y(x))\right) \\ \rho\,y(x)\cdot\frac{\partial^2}{\partial t^2}\,T(t) &= -T(t)\,\frac{\partial^2}{\partial x^2}\left(EI\,\cdot\frac{\partial^2}{\partial x^2}\,y(x)\right) \\ \frac{\frac{\partial^2}{\partial t^2}\,T(t)}{T(t)} &= -\frac{1}{\rho(x)\,y(x)}\,\frac{\partial^2}{\partial x^2}\left(EI\,\cdot\frac{\partial^2}{\partial x^2}\,y(x)\right) \end{aligned}$$

Observe that the variables t and x are separated.

- The LHS (left hand side) depends on t only and thus the RHS (right hand side) depends on t only too.
- The RHS depends on x only and thus the LHS depends on x only too.
- The above too statements imply that RHS and LHS have to equal a constant which we call $-\omega^2$, ignoring a technical problem[2].

Thus we obtain a differential equation for time dependence

$$\frac{\partial^2}{\partial t^2}\, T(t) = -\omega^2\, T(t)$$

and an equation for space the space variables $y(x)$

$$\frac{1}{\rho}\,\frac{\partial^2}{\partial x^2}\left(EI\,\cdot\frac{\partial^2}{\partial x^2}\, y(x)\right) = \omega^2\, y(x)$$

or equivalently to the generalized eigenvalue problem

$$\frac{\partial^2}{\partial x^2}\left(EI\,\cdot\frac{\partial^2}{\partial x^2}\, y(x)\right) = \omega^2\, \rho(x)\, y(x) \tag{4.9}$$

If we manage to solve both of the above equations for a common value of ω then we have a solution of the original differential equation (4.8). This solution is given by $u(t,x) = T(t)\cdot y(x)$.

We replace one difficult problem by two simpler problems. The method can only be called successful if we manage to solve the two simpler problems.

4.8.3 From eigenvalues to frequencies

The general, complex solution of the above time equation is given by

$$T(t) = A_1\, e^{i\,\omega\, t} + A_2\, e^{-i\,\omega\, t}$$

As real solution we find the two linear independent functions $\cos(\omega t)$ and $\sin(\omega t)$. Thus we have periodic solutions

$$T(t) = A\, \cos(\omega t + \delta)$$

with the frequency $\nu = \frac{\omega}{2\,\pi}$ or the period $T = \frac{1}{\nu} = \frac{2\,\pi}{\omega}$.

The eigenvalue problem for the space dependence is given by

$$\frac{1}{\rho}\,\frac{\partial^2}{\partial x^2}\left(EI\,\cdot\frac{\partial^2}{\partial x^2}\, y(x)\right) = \lambda\, y(x) \tag{4.10}$$

If we find an eigenvalue λ the we can use $\omega^2 = \lambda$ to determine ω and thus we find ν and the period T.

$$\omega = \sqrt{\lambda} \quad \text{resp.} \quad \nu = \frac{\omega}{2\,\pi} = \frac{\sqrt{\lambda}}{2\,\pi}$$

This shows that the eigenvalues of the space problem determine the frequencies with which the bar can vibrate.

[2]The notation $-\omega^2$ quietly assumes that this constant is negative. This assumption can be justified.

4.8.4 A beam with constant cross section

If E, I and ρ are independent on x then we can proceed computing with exact expressions. Use the notation

$$k^4 = \frac{\rho}{EI} \quad \text{or} \quad k = \sqrt[4]{\frac{\rho}{EI}}$$

to arrive at

$$\frac{d^4}{dx^4}\, y(x) = \omega^2\, k^4\, y(x)$$

Thus we search a function whose fourth derivative equals the original function, up to a constant factor. The general form of those functions is

$$y(x) = A\, \cos(k\sqrt{\omega}\, x) + B\, \cosh(k\sqrt{\omega}\, x) + C\, \sin(k\sqrt{\omega}\, x) + D\, \sinh(k\sqrt{\omega}\, x)$$

Now we can use the 4 boundary conditions to determine the constants A, B, C and D. Using the condition at $x = 0$ we conclude

$$\begin{array}{rcl@{\qquad}rcl} y(0) & = & 0 & \Longrightarrow & A + B & = & 0 \\ y'(0) & = & 0 & \Longrightarrow & k\sqrt{\omega}\, C + k\sqrt{\omega}\, D & = & 0 \end{array}$$

This simplifies the general solution to

$$y(x) = A\, \big(\cos(k\sqrt{\omega}\, x) - \cosh(k\sqrt{\omega}\, x)\big) + C\, \big(\sin(k\sqrt{\omega}\, x) - \sinh(k\sqrt{\omega}\, x)\big)$$

Now use the constraints $y''(L) = 0$ and $y'''(L) = 0$ to find two equations for the unknowns A and C.

$$\begin{array}{rcl} A\, \big(-\cos(k\sqrt{\omega}\, L) - \cosh(k\sqrt{\omega}\, L)\big) + C\, \big(-\sin(k\sqrt{\omega}\, L) - \sinh(k\sqrt{\omega}\, L)\big) & = & 0 \\ A\, \big(\sin(k\sqrt{\omega}\, L) - \sinh(k\sqrt{\omega}\, L)\big) + C\, \big(-\cos(k\sqrt{\omega}\, L) - \cosh(k\sqrt{\omega}\, L)\big) & = & 0 \end{array}$$

This is a linear, homogeneous system of linear equation for A and C. Unless the determinant of the matrix vanishes we only have the trivial solution $A = C = 0$, leading to $y(x) = 0$. This is not a very interesting situation.

We have to determine the values of ω for which the determinate vanishes, i.e.

$$\big(\cos(k\sqrt{\omega}\, L) + \cosh(k\sqrt{\omega}\, L)\big)^2 + \sin^2(k\sqrt{\omega}\, L) - \sinh^2(k\sqrt{\omega}\, L) = 0$$

Using $\sin^2 z + \cos^2 z = 1$ and $\cosh^2 z - \sinh^2 z = 1$ rewrite this equations to

$$2 + 2\, \cos(k\sqrt{\omega}\, L) \cdot \cosh(k\sqrt{\omega}\, L) = 0$$

Thus the interesting values are determined by the zeros of the auxiliary function

$$g(z) = 1 + \cos(z) \cdot \cosh(z) = 0$$

Using a simple graphic and *Mathematica* we determine the first few zeros of this function. Exact solution are not possible and we have to settle for approximate values. Since the $\cosh z$ function grows rapidly as $z \gg 1$ the expression $\cos z$ has to be close to 0 for $g(z)$ to vanish. We conclude that the zeros of $g(z)$ should be close to the known zeros of $\cos z$. This is correct, except for the first zero. Using the information we can use the *Mathematica* command `FindRoot[]` to determine the zeros with the help of Newton's algorithm. This is implemented in the code below.

Mathematica

```
g[z_]:= 1+Cos[z]*Cosh[z];
Plot[g[z],{z,0,20},PlotRange->{-1,1}];
{z/.FindRoot[g[z]==0,{z,2}],
    Table[z/.FindRoot[g[z]==0,{z,Pi/2+k*Pi}],{k,1,6}]}//Flatten
.
{1.8751,4.69409,7.85476,10.9955,14.1372,17.2788,20.4204}
```

Using the above values for $z = k\sqrt{\omega}L$ we find the eigenvalues

$$\lambda = \omega^2 = \left(\frac{z}{k\,L}\right)^4 = \frac{EI}{\rho}\,\frac{z^4}{L^4}$$

and the frequencies

$$\nu = \frac{\omega}{2\,\pi} = \frac{1}{2\,\pi}\,\sqrt{\frac{EI}{\rho}}\,\frac{z^2}{L^2}$$

This information is useful to determine the frequencies for a beam with constant cross section and to obtain estimates for more complicated situations. To obtain better results for non constant cross section we have to use numerical methods. The next section presents one of the possible solutions by a finite element method.

4.8.5 FEM description of the static situation

To solve the static problem we have to minimize the functional

$$F(u) = \int_0^L \frac{E\,I(x)}{2}\;\;\big(u''(x)\big)^2 - f(x)\cdot u(x)\;dx$$

The methods to be used are very similar to section 4.5, but we now have to find integrals of second derivatives.

First we examine the integral on the standard interval $[-h/2\,,\,h/2]$ of length h . For beams the first derivative $u'(x)$ should be continuous, thus we choose $u(-h/2)$, $u'(-h/2)$, $u(h/2)$ and $u'(h/2)$ as degrees of freedom for the element construction. We can then cover the length of the bean by a piecewise cubic interpolation, assuring that the function and the slope are continuous along the beam.

Cubic interpolation in the interval $[-h/2\,,\,h/2]$

On the interval $[\frac{h}{2}\,,\,\frac{h}{2}]$ we consider polynomials of degree 3 as functions

$$u(x) = c_0 + c_1\,x + c_2\,x^2 + c_3\,x^3$$

The values and slopes at the endpoints are the degrees of freedom.

$$a_1 = u(\frac{-h}{2}) \quad , \quad a_2 = u'(\frac{-h}{2}) \quad , \quad a_3 = u(\frac{h}{2}) \quad , \quad a_4 = u'(\frac{h}{2})$$

We need the coefficients c_i of the polynomial as function of the degrees of freedom. Consider the system of linear equations

$$\begin{array}{rcccccc}
u(\frac{-h}{2}) & = & c_0 & -c_1\,\frac{h}{2} & +c_2\,\frac{h^2}{4} & -c_3\,\frac{h^3}{8} & = & a_1 \\
u'(\frac{-h}{2}) & = & & c_1 & -c_2\,h & +c_3\,\frac{3\,h^2}{4} & = & a_2 \\
u(\frac{h}{2}) & = & c_0 & +c_1\,\frac{h}{2} & +c_2\,\frac{h^2}{4} & +c_3\,\frac{h^3}{8} & = & a_3 \\
u'(\frac{h}{2}) & = & & c_1 & +c_2\,h & +c_3\,\frac{3\,h^2}{4} & = & a_4
\end{array}$$

or using a matrix

$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} = \begin{bmatrix} 1 & -\frac{h}{2} & \frac{h^2}{4} & -\frac{h^3}{8} \\ 0 & 1 & -h & \frac{3\,h^2}{4} \\ 1 & \frac{h}{2} & \frac{h^2}{4} & \frac{h^3}{8} \\ 0 & 1 & h & \frac{3\,h^2}{4} \end{bmatrix} \cdot \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix}$$

Using the coefficients c_i we can compute the function $u(x)$ at any chosen point in the interval. This will be used to compute the integral with Gauss integration.

Contribution of $f(x) \cdot u(x)$ to the functional

For a Gauss integration with three points of support we need values of the function

$$u(x) = c_0 + c_1\,x + c_2\,x^2 + c_3\,x^3$$

at the points (see section 4.5.2)

$$p_{-1} = u(-\frac{h}{2}\sqrt{\frac{3}{5}}) \quad , \quad p_0 = u(0) \quad , \quad p_1 = u(\frac{h}{2}\sqrt{\frac{3}{5}})$$

Using the above interpolation we obtain

$$\begin{array}{rcl} p_{-1} &=& c_0 \;-\sqrt{\frac{3}{5}}\,\frac{h}{2}\,c_1 \;+\frac{3}{5}\,\frac{h^2}{4}\,c_2 \;-\frac{3}{5}\sqrt{\frac{3}{5}}\,\frac{h^3}{8}\,c_3 \\ p_0 &=& c_0 \\ p_1 &=& c_0 \;+\sqrt{\frac{3}{5}}\,\frac{h}{2}\,c_1 \;+\frac{3}{5}\,\frac{h^2}{4}\,c_2 \;+\frac{3}{5}\sqrt{\frac{3}{5}}\,\frac{h^3}{8}\,c_3 \end{array}$$

or using matrices

$$\begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix} = \begin{bmatrix} 1 & -\frac{1}{2}\sqrt{\frac{3}{5}}\,h & \frac{3}{20}\,h^2 & -\frac{3}{40}\sqrt{\frac{3}{5}}\,h^3 \\ 1 & 0 & 0 & 0 \\ 1 & \frac{1}{2}\sqrt{\frac{3}{5}}\,h & \frac{3}{20}\,h^2 & \frac{3}{40}\sqrt{\frac{3}{5}}\,h^3 \end{bmatrix} \cdot \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix}$$

We can also write

$$\begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix} = \begin{bmatrix} 1 & -\frac{1}{2}\sqrt{\frac{3}{5}}\,h & \frac{3}{20}\,h^2 & -\frac{3}{40}\sqrt{\frac{3}{5}}\,h^3 \\ 1 & 0 & 0 & 0 \\ 1 & \frac{1}{2}\sqrt{\frac{3}{5}}\,h & \frac{3}{20}\,h^2 & \frac{3}{40}\sqrt{\frac{3}{5}}\,h^3 \end{bmatrix} \cdot \begin{bmatrix} 1 & -\frac{h}{2} & \frac{h^2}{4} & -\frac{h^3}{8} \\ 0 & 1 & -h & \frac{3\,h^2}{4} \\ 1 & \frac{h}{2} & \frac{h^2}{4} & \frac{h^3}{8} \\ 0 & 1 & h & \frac{3\,h^2}{4} \end{bmatrix}^{-1} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

A calculation (done with *Mathematica*) shows, using the abbreviation $s = \sqrt{\frac{3}{5}}$,

$$\begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix} = \begin{bmatrix} \frac{1}{2} + s^3 & \frac{5+\sqrt{15}}{100}\,h & \frac{1}{2} - s^3 & \frac{-5+\sqrt{15}}{100}\,h \\ \frac{1}{2} & \frac{h}{8} & \frac{1}{2} & -\frac{h}{8} \\ \frac{1}{2} - s^3 & \frac{5-\sqrt{15}}{100}\,h & \frac{1}{2} + s^3 & \frac{-5-\sqrt{15}}{100}\,h \end{bmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \mathbf{M}_0 \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

Now we can compute the function $u(x)$ at the Gauss integration points, using the degrees of freedom of the element. Using a Gauss integration we find

$$\begin{aligned}
\int_{-h/2}^{h/2} f(x)\, u(x)\, dx &\approx \frac{h}{18}\left(5\, f(-\sqrt{\frac{3}{5}}\frac{h}{2})\, p_{-1} + 8\, f(0)\, p_0 + 5\, f(\sqrt{\frac{3}{5}}\frac{h}{2})\, p_{-1}\right) \\
&= \frac{h}{18}\,(5\, f_{-1}\, p_{-1} + 8\, f_0\, p_0 + 5\, f_1\, p_{-1}) \\
&= \langle \begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix},\ \frac{h}{18}\begin{pmatrix} 5\, f_{-1} \\ 8\, f_0 \\ 5\, f_1 \end{pmatrix}\rangle = \langle \mathbf{M}_0 \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix},\ \frac{h}{18}\begin{pmatrix} 5\, f_{-1} \\ 8\, f_0 \\ 5\, f_1 \end{pmatrix}\rangle \\
&= \langle \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix},\ \frac{h}{18}\,\mathbf{M}_0^T \cdot \begin{pmatrix} 5\, f_{-1} \\ 8\, f_0 \\ 5\, f_1 \end{pmatrix}\rangle
\end{aligned}$$

Contribution of $(u'')^2$ to the functional

In this case we need

$$u''(x) = 2\, c_2 + 6\, c_3\, x$$

at the three Gauss points

$$p_{-1} = u''(-\frac{h}{2}\sqrt{\frac{3}{5}}) \qquad , \qquad p_0 = u''(0) \qquad , \qquad p_1 = u''(\frac{h}{2}\sqrt{\frac{3}{5}})$$

Thus

$$\begin{aligned}
p_{-1} &= 2\, c_2 \quad -3\,\sqrt{\tfrac{3}{5}}\, h\, c_3 \\
p_0 &= 2\, c_2 \\
p_1 &= 2\, c_2 \quad +3\,\sqrt{\tfrac{3}{5}}\, h\, c_3
\end{aligned}$$

or

$$\begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix} = \begin{bmatrix} 0 & 0 & 2 & -3\,\sqrt{\frac{3}{5}}\, h \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 2 & 3\sqrt{\frac{3}{5}}\, h \end{bmatrix} \cdot \begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{pmatrix}$$

We may also write

$$\begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix} = \begin{bmatrix} 0 & 0 & 2 & -3\,\sqrt{\frac{3}{5}}\, h \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 2 & 3\sqrt{\frac{3}{5}}\, h \end{bmatrix} \cdot \begin{bmatrix} 1 & -\frac{h}{2} & \frac{h^2}{4} & -\frac{h^3}{8} \\ 0 & 1 & -h & \frac{3\,h^2}{4} \\ 1 & \frac{h}{2} & \frac{h^2}{4} & \frac{h^3}{8} \\ 0 & 1 & h & \frac{3\,h^2}{4} \end{bmatrix}^{-1} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

A calculation (with *Mathematica*) shows

$$\begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix} = \frac{1}{h} \begin{bmatrix} -\frac{6}{h}\sqrt{\frac{3}{5}} & -1-3\sqrt{\frac{3}{5}} & +\frac{6}{h}\sqrt{\frac{3}{5}} & 1-3\sqrt{\frac{3}{5}} \\ 0 & -1 & 0 & 1 \\ \frac{6}{h}\sqrt{\frac{3}{5}} & -1+3\sqrt{\frac{3}{5}} & -\frac{6}{h}\sqrt{\frac{3}{5}} & 1+3\sqrt{\frac{3}{5}} \end{bmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} = \mathbf{M}_2 \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix}$$

Using a Gauss integration we now arrive at

$$\begin{aligned} \frac{1}{2}\int_{-h/2}^{h/2} EI(x)\,(u''(x))^2\,dx &\approx \frac{1}{2}\frac{h}{18}\left(5\,EI(-\sqrt{\frac{3}{5}}\frac{h}{2})\,p_{-1}^2 + 8\,EI(0)\,p_0^2 + 5\,EI(\sqrt{\frac{3}{5}}\frac{h}{2})\,p_{-1}^2\right) \\ &= \frac{1}{2}\,\frac{h}{18}\left(5\,EI_{-1}\,p_{-1}^2 + 8\,EI_0\,p_0^2 + 5\,EI_1\,p_{-1}^2\right) \\ &= \frac{1}{2}\,\langle \begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix},\, \frac{h}{18} \begin{bmatrix} 5\,EI_{-1} & 0 & 0 \\ 0 & 8\,EI_0 & 0 \\ 0 & 0 & 5\,EI_1 \end{bmatrix} \cdot \begin{pmatrix} p_{-1} \\ p_0 \\ p_1 \end{pmatrix} \rangle \\ &= \frac{1}{2}\langle \mathbf{M}_2 \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix},\, \frac{h}{18} \begin{bmatrix} 5\,EI_{-1} & 0 & 0 \\ 0 & 8\,EI_0 & 0 \\ 0 & 0 & 5\,EI_1 \end{bmatrix} \cdot \mathbf{M}_2 \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \rangle \\ &= \frac{1}{2}\langle \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix},\, \frac{h}{18}\,\mathbf{M}_2^T \cdot \begin{bmatrix} 5\,EI_{-1} & 0 & 0 \\ 0 & 8\,EI_0 & 0 \\ 0 & 0 & 5\,EI_1 \end{bmatrix} \cdot \mathbf{M}_2 \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \rangle \end{aligned}$$

Now we have the element stiffness matrix. For the simplest case $EI(x) = 1$ we find

$$\frac{1}{2}\int_{-h/2}^{h/2} (u''(x))^2\,dx \approx \frac{1}{2}\,\langle \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix},\, \begin{bmatrix} \frac{12}{h^3} & \frac{6}{h^2} & -\frac{12}{h^3} & \frac{6}{h^2} \\ \frac{6}{h^2} & \frac{4}{h} & -\frac{6}{h^2} & \frac{2}{h} \\ -\frac{12}{h^3} & -\frac{6}{h^2} & \frac{12}{h^3} & -\frac{6}{h^2} \\ \frac{6}{h^2} & \frac{2}{h} & -\frac{6}{h^2} & \frac{4}{h} \end{bmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \rangle$$

If u'' is a polynomial with degree ≤ 2 then the above integration is exact.

4.8.6 Assembling the system of equations, *Octave* code and a few tests

For our standard element on the interval $-h/2 \le x \le h/2$ we have

$$\begin{aligned}
&\int_{-h/2}^{h/2} \ \tfrac{1}{2}\, EI(x)\, (u''(x))^2 \ - f(x)\cdot u(x) dx \approx \\
&\approx \ \tfrac{1}{2}\, \langle \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \ , \ \tfrac{h}{18}\, \mathbf{M}_2^T \cdot \begin{bmatrix} 5\,EI_{-1} & 0 & 0 \\ 0 & 8\,EI_0 & 0 \\ 0 & 0 & 5\,EI_1 \end{bmatrix} \cdot \mathbf{M}_2 \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \rangle \\
&\quad - \langle \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \ , \ \tfrac{h}{18}\, \mathbf{M}_0^T \cdot \begin{pmatrix} 5\,f_{-1} \\ 8\,f_0 \\ 5\,f_1 \end{pmatrix} \rangle
\end{aligned}$$

Now we want to solve the static problem for given functions $EI(x)$, $\rho(x)$ and $f(x)$. We also need to give the length L of the bean and the desired number of elements. This is done by choosing all boundary points of the elements.

As a concrete example we consider a beam of length $L = 1$ with a force density $f(x) = 2 - x$.

Octave

```
n=10;
L=1;
x=linspace(0,L,n+1);

function r= EI(x)  r=ones(size(x)); endfunction
function r = f(x) r=2-x;  endfunction
function r = rho(x) r=ones(size(x)); endfunction
```

To solve the problem we need a function to determine the element stiffness matrix, using the formulas of the previous section.

Octave

```
function mat = elementContribution(x,aFunc)
  h=x(2)-x(1);    s=sqrt(3/5);
  xm=(x(1)+x(2))/2;  xr=xm+s*h/2;  xl=xm-s*h/2;
  M2=[-6/h*s, -1-3*s,6/h*s,1-3*s;...
      0,-1,0,1;...
     6/h*s, -1+3*s,-6/h*s,1+3*s]/h;
  mat=h/18*M2'*...
       diag([5*feval(aFunc,xl),8*feval(aFunc,xm),5*feval(aFunc,xr)])*M2;
endfunction
```

Octave

```
function vec = elementVector(x,fFunc)
  h=x(2)-x(1);   s=sqrt(3/5);
  xm=(x(1)+x(2))/2;  xr=xm+s*h/2;  xl=xm-s*h/2;
  M0=[0.5+s^3, (5+sqrt(15))/100*h, 0.5-s^3, (-5+sqrt(15))/100*h;...
      0.5,h/8,0.5,-h/8;...
      0.5-s^3, (5-sqrt(15))/100*h, 0.5+s^3, (-5-sqrt(15))/100*h];
  vec=h/18*M0'*[5*feval(fFunc,xl);8*feval(fFunc,xm);5*feval(fFunc,xr)];
endfunction
```

As degrees of freedom we have the values of u_k and the slopes u'_k at the points of support x_k. Since for our problem the beam is clamped at $x = 0$ we have $u_0 = u'_0 = 0$. Thus we have a total of $2\,n$ degrees of freedom. Now we want to rewrite the function in the form

$$\begin{aligned} F(u) &= \int_0^L \frac{E\,I(x)}{2}\ \left(u''(x)\right)^2 - f(x)\cdot u(x)\,dx \\ &= \sum_{k=1}^{n}\left(\int_{x_{k-1}}^{x_k} \frac{E\,I(x)}{2}\ \left(u''(x)\right)^2 - f(x)\cdot u(x)\,dx\right) \\ &\approx \frac{1}{2}\left\langle \begin{pmatrix} u_1 \\ u'_1 \\ u_2 \\ u'_2 \\ \vdots \\ u_n \\ u'_n \end{pmatrix},\ \mathbf{A}\cdot \begin{pmatrix} u_1 \\ u'_1 \\ u_2 \\ u'_2 \\ \vdots \\ u_n \\ u'_n \end{pmatrix} \right\rangle + \left\langle \begin{pmatrix} u_1 \\ u'_1 \\ u_2 \\ u'_2 \\ \vdots \\ u_n \\ u'_n \end{pmatrix},\ \vec{b} \right\rangle \end{aligned}$$

Given the $2n \times 2n$ matrix $\mathbf{A}$ and the vector $\vec{b}$ we have to solve the linear system $\mathbf{A}\,\vec{u} = \vec{b}$. The matrix $\mathbf{A}$ contains all contributions from the individual elements. As an example consider the third element, its contribution will end up in rows and columns 3 through 6.

Octave

```
A=zeros(2*(length(x)-1));
b=zeros(2*(length(x)-1),1);

% Dirichlet conditions at left end point:  u(0)=u'(0)=0
Atmp=elementContribution([x(1), x(2)],'EI');
A(1:2,1:2) = Atmp(3:4,3:4);
btmp=elementVector([x(1), x(2)],'f');
b(1:2)=btmp(3:4);

for k=2:(length(x)-1);
  A(2*k-3:2*k,2*k-3:2*k) = A(2*k-3:2*k,2*k-3:2*k) + ...
                   elementContribution([x(k), x(k+1)],'EI');
  b(2*k-3:2*k) = b(2*k-3:2*k) + elementVector([x(k), x(k+1)],'fF');
endfor

u=A\b;
```

From the resulting vector $\vec{u}$ we can extract the values at the endpoints of the elements (ignoring the derivatives), complement it with $u(0) = 0$ and then plot the solution.

Octave

```
u=reshape(u,2,length(x)-1);
y=[0 u(1,:)]
plot(x,y)
```

A first test

Consider the functional

$$F(u) = \int_0^1 \frac{1}{2}\,(u''(x))^2 - u(x)dx$$

with the boundary condition $u(0) = u'(0) = 0$. The minimum is attained by the solution of the differential equation

$$u^{(4)}(x) = 1 \quad , \quad u(0) = u'(0) = u''(1) = u'''(1) = 0$$

Using $u(0) = u'(0) = 0$ we obtain

$$u(x) = \frac{1}{24}\, x^4 + c_1\, x^2 + c_2\, x^3$$

The remaining boundary conditions are satisfied if

$$\begin{array}{lllll} \frac{1}{2} & +2\, c_1 & +6\, c_2 & = & 0 \\ 1 & & +6\, c_2 & = & 0 \end{array}$$

Since the solution of the above system is $c_2 = -\frac{1}{6}$, $c_1 = \frac{1}{4}$ we find the exact solution $u(x)$ to be

$$u(x) = \frac{1}{24}\, x^4 + \frac{1}{4}\, x^2 - \frac{1}{6}\, x^3$$

From this we find easily $u(1) = \frac{1}{8}$ and $u'(1) = \frac{1}{6}$. The *Octave* code of the previous section reproduces this result exactly. This is no surprise as the exact solution is a polynomial of degree 3 and the Gauss integration is exact for polynomials up to degree 5 . Thus the integrals of $(u'')^2$ and u are exact.

A second test

The function $f(x) = 2 - x$ describes a load getting smaller as we move along the beam. The functional to be considered is

$$F(u) = \int_0^1 \frac{1}{2}\, (u''(x))^2 - (2 - x)\, u(x) dx$$

Mathematica is capable of solving the resulting differential equation exactly.

Mathematica

```
s=DSolve[{D[u[x],{x,4}]==2-x,u[0]==0,u'[0]==0,u''[1]==0,u'''[1]==0},u[x],x];
us[x_]=u[x]/.s[[1]]
.
 2    3    4    5
x    x    x    x
-- - -- + -- - ---
3    4    12   120
```

i.e. the exact solution is

$$u(x) = \frac{x^2}{3} - \frac{x^3}{4} + \frac{x^4}{12} - \frac{x^5}{120}$$

and thus $u(1) = \frac{19}{120} \approx 0.15833$ and $u'(1) = \frac{5}{24} \approx 0.20833$. Again the *Octave* code of the previous section reproduces this result exactly.

A third test

If $f(x) = \sin(x)$ the *Mathematica* yields the exact result $u(1) = 0.0821057$ and $u'(1) = 0.111622$. The FEM code with 10 elements of equal length gives $u(1) = 0.082106$ and $u'(1) = 0.11162$ and thus a very small error. Even a computation with only 3 elements yields a good approximation.

4.8.7 Finding eigenvalues

To determine the eigenvalues λ in (4.10) the function $f(x)$ has to be replaced by $\frac{1}{2}\,\lambda\,\rho(x)\,u(x)$, where λ is to be determined. This leads to [3]

$$F(u) = \int_{-h/2}^{h/2} \frac{1}{2}\, EI(x)\, (u''(x))^2 \; - \lambda\, \frac{1}{2}\, \rho(x)\, u^2(x) dx$$

For the standard element we obtain

$$\begin{aligned}
&\textstyle\int_{-h/2}^{h/2} \quad \frac{1}{2}\, EI(x)\, (u''(x))^2 - \lambda\, \frac{1}{2}\, \rho(x)\; u^2(x) dx \approx \\
&\approx \quad \frac{1}{2}\, \left\langle \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \;,\; \frac{h}{18}\, \mathbf{M}_2^T \cdot \begin{bmatrix} 5\, EI_{-1} & 0 & 0 \\ 0 & 8\, EI_0 & 0 \\ 0 & 0 & 5\, EI_1 \end{bmatrix} \cdot \mathbf{M}_2 \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \right\rangle \\
&\quad - \lambda\, \frac{1}{2}\, \left\langle \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \;,\; \frac{h}{18}\, \mathbf{M}_0^T \cdot \begin{bmatrix} 5\, \rho_{-1} & 0 & 0 \\ 0 & 8\, \rho_0 & 0 \\ 0 & 0 & 5\, \rho_1 \end{bmatrix} \mathbf{M}_0 \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{pmatrix} \right\rangle
\end{aligned}$$

The element contribution have to be combined again to a global expression

$$F(u) \approx \frac{1}{2}\, \langle \vec{u}\,,\, \mathbf{A} \cdot \vec{u} \rangle - \frac{1}{2}\, \lambda\, \langle \vec{u}\,,\, \mathbf{B} \cdot \vec{u} \rangle$$

The vector $\vec{u}$ is to be chosen such that F is minimal. The corresponding system of linear equations has to form

$$\mathbf{A}\, \vec{u} = \lambda\, \mathbf{B}\, \vec{u} \quad \text{or} \quad \mathbf{B}^{-1}\, \mathbf{A}\, \vec{u} = \lambda\, \vec{u}$$

Thus the eigenvalues of the matrix $\mathbf{B}^{-1} \cdot \mathbf{A}$ correspond to the eigenvalues of the differential operator

$$\frac{1}{\rho(x)}\, \frac{\partial^2}{\partial x^2} \left(EI(x) \; \cdot \frac{\partial^2}{\partial x^2}\, y(x) \right) = \lambda\, y(x)$$

It is often advantageous[4] not to invert the matrix $\mathbf{B}$ but to consider directly the **generalized eigenvalue problem**

$$\begin{aligned}
\mathbf{A}\, \vec{u} \quad &= \quad \lambda\, \mathbf{B}\, \vec{u} \\
\frac{\partial^2}{\partial x^2} \left(EI(x) \; \cdot \frac{\partial^2}{\partial x^2}\, y(x) \right) \quad &= \quad \lambda\, \rho(x)\, y(x)
\end{aligned}$$

This corresponds to a discretization of equation (4.9). This type of problem is examined in section 11.6.

Considering the above we need the element contributions of the expression $\rho(x)\, u^2(x)$. The arguments are very similar to the previous sections and thus not repeated here. The code below finds the corresponding element contribution.

Octave

[3] The arguments are similar to section 4.8.1.

[4] The matrix $\mathbf{B}^{-1} \cdot \mathbf{A}$ is **not** symmetric, even if $\mathbf{A}$ and $\mathbf{B}$ are symmetric

```
function mat =evMatrix(x,rhoFunc)
  h=x(2)-x(1);
  s=sqrt(3/5);
  xm=(x(1)+x(2))/2;
  xr=xm+s*h/2;
  xl=xm-s*h/2;
  M0=[0.5+s^3, (5+sqrt(15))/100*h, 0.5-s^3, (-5+sqrt(15))/100*h;...
      0.5,h/8,0.5,-h/8;...
      0.5-s^3, (5-sqrt(15))/100*h, 0.5+s^3, (-5-sqrt(15))/100*h];
  mat=h/18*M0'*diag([5*feval(rhoFunc,xl);8*feval(rhoFunc,xm);
                                        5*feval(rhoFunc,xr)])*M0;
endfunction
```

Now all building blocks are at our disposition and we write code to determine the 10 lowest frequencies and plot the shape of the first 4 eigenmodes.

Octave

```
evMat=zeros(2*(length(x)-1));
% Dirichlet conditions at left endpoint: u(0)=u'(0)=0
Atmp=evMatrix([x(1), x(2)],'rho');
evMat(1:2,1:2) = Atmp(3:4,3:4);

for k=2:(length(x)-1);
  evMat(2*k-3:2*k,2*k-3:2*k) =   evMat(2*k-3:2*k,2*k-3:2*k) +...
                evMatrix([x(k), x(k+1)],'rho');
endfor

neig=10;  %%% first 10 eigenvalues
[vec,lam]=eig(inv(evMat)*A);
[lam, index]=sort(diag(lam));
freq=sqrt(lam(1:neig))/(2*pi)

nvec=4;  %%% first 4 eigenmodes
ploty=zeros(nvec,length(x));
for k=1:nvec
  tmp=reshape(vec(:,index(k)) ,2,length(x)-1);
  tmp=tmp(1,:);
  tmp=tmp/max(abs(tmp));
  ploty(k,:)=[0 tmp];
endfor

plot(x,ploty)
```

For the case of constant cross section we already know the exact results from section 4.8.3. Table 4.6 show the results for 10 and 50 elements of equal length. Verify that the computation with more elements leads to smaller errors results, in particular for higher frequencies. The shape of the first 4 eigenmodes is shown in figure 4.13.

Octave

```
n=10;
L=1;
x=linspace(0,L,n+1);

function r= EI(x)
r=ones(size(x));
endfunction
```

```
function r = rho(x)
r=ones(size(x));
endfunction
```

Elements Frequency	10	50	exakt
1	0.55959	0.55959	0.55959
2	3.50701	3.50690	3.5069
3	9.82193	9.81942	9.8194
4	19.26067	19.24217	19.2421
5	31.89016	31.80877	31.8086
6	47.77946	47.51706	47.5166
7	67.05149	66.36742	66.3661
8	89.87237	88.36030	88.3573
9	116.31737	113.49639	113.490
10	144.93389	141.77674	141.764

Table 4.6: Eigen frequencies of a vibrating beam

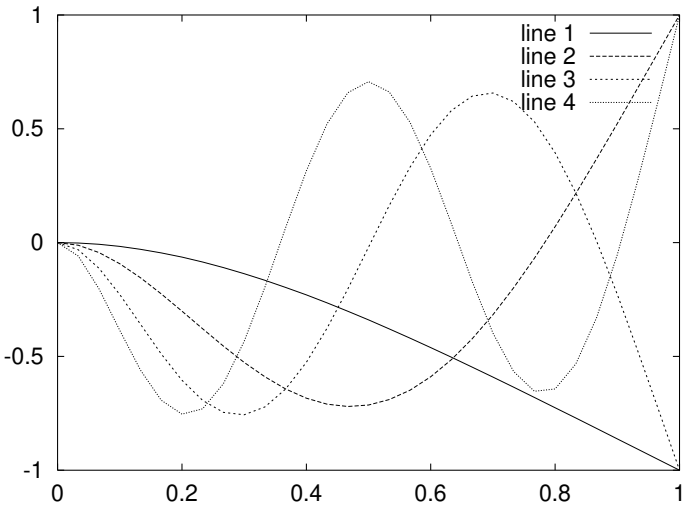


Figure 4.13: Eigenmodes for a beam with constant cross section

4.8.8 Design of a force sensor

Consider a block of metal with the cross section shown in figure 4.14. A vertical force F is applied at the right, leading to a deformation. The deformation can be measured we try to compute the applied force. This essentially corresponds to finding the spring constant (force/length) for this system. Since the vertical force applies only at the endpoint we have $f(x) = 0$ with the notation in the previous sections. Instead we have

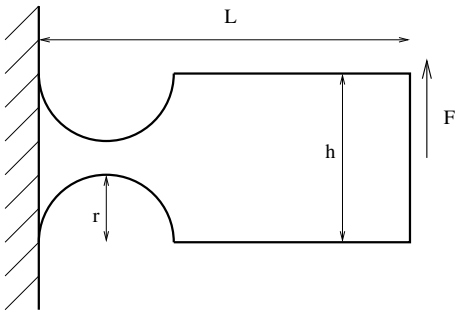


Figure 4.14: Setup for a force sensor

to add a potential energy $-F \cdot u(L)$ to the functional. Thus we have to minimize the modified functional

$$F(u) = \int_0^L \frac{1}{2}\, EI(x)\, (u''(x))^2 dx - F \cdot u(L)$$

The vector $\vec{b}$ in section 4.8.5 contains zeros only. Except for the second to last component which equals the value of F.

As a concrete example consider a block of aluminum of a sensor of width B with the following data.

h_0	=	6.0	mm	r	=	2.9	mm
L	=	20	mm	B	=	20	mm
ρ_0	=	$2.7 \cdot 10^{-6}$	$\frac{\text{kg}}{\text{mm}^3}$	E	=	$73 \cdot 10^3$	$\frac{\text{N}}{\text{mm}^2}$

The height $h(x)$ of the structure, as function of x, is given by

$$h(x) = \begin{cases} h_0 - 2\sqrt{r^2 - (x-r)^2} & \text{for} \quad 0 \le x \le 2\,r \\ h_0 & \text{for} \quad 2\,r \le x \le L \end{cases}$$

and the moment of inertia of the cross section is

$$I(x) = \frac{1}{12}\, B\, h(x)^3$$

The mass per length is

$$\rho(x) = \rho_0\, B\, h(x)$$

All essential data of the force sensor is known. Since most of the bending will happen in the narrow section we place more elements in that part. Initially we divide the section $0 \le x \le 2\,r$ into 4 elements of equal length and only one element on the right. Then all elements are twice split up. We end up with 20 elements.

Octave

```
global h0=6;
global r=2.9;
global width = 20
global L=20;
global E=73*10^3;
global rho0=2.7*10^(-6);

x=[0 r/2 r 3*r/2 2*r L];
```

```
%%% doubling the number of elemente %%%%
xt=sort([x x+[diff(x)/2 0]]);x=xt(1:length(xt)-1);
xt=sort([x x+[diff(x)/2 0]]);x=xt(1:length(xt)-1);

%%%% height as function of the horizontal position %%%%
function res=h(x)
  global h0 r;
  if(x<2*r)
    res=h0-2*sqrt(r^2-(x-r)^2);
  else
    res=h0;
  endif
endfunction

%%%% modulus of elasticity and moment of inertia of the surface %%%%
function r= EI(x)
global E width;
r=E/12*width*h(x)^3;
endfunction

%%%% mass per length %%%%
function r = rho(x)
global rho0,width;
r=rho0*h(x)*width;
endfunction
```

If we set $F = 1\,\mathrm{N}$ then the inverse of $u(L)$ will give us the spring constant. This is done by the code below. This example leads to a value of $u(L) = 0.267$ mm and thus a spring constant of $k = 3.75\ \frac{\mathrm{N}}{\mathrm{mm}}$. The code also generates a graphics with the shape of the deformation. Considering this graph in figure 4.15 it is obvious that the bending is concentrated to the section $2.5 < x < 3.5$. This should not be a surprise as the sensor has its thinnest section at $x \approx 3$.

Octave

```
function mat =elementContribution(x,aFunc)
  h=x(2)-x(1);
  s=sqrt(3/5);
  xm=(x(1)+x(2))/2;
  xr=xm+s*h/2;
  xl=xm-s*h/2;
  M2=[-6/h*s, -1-3*s,6/h*s,1-3*s; 0,-1,0,1; 6/h*s, -1+3*s,-6/h*s,1+3*s]/h;
  mat=h/18*M2'*diag([5*feval(aFunc,xl),8*feval(aFunc,xm),5*feval(aFunc,xr)])*M2;
endfunction

A=zeros(2*(length(x)-1));
b=zeros(2*(length(x)-1),1);
% Dirichlet condition at left end point:  u(0)=u'(0)=0
Atmp=elementContribution([x(1), x(2)],'EI');
A(1:2,1:2) = Atmp(3:4,3:4);
for k=2:(length(x)-1);
  A(2*k-3:2*k,2*k-3:2*k) = A(2*k-3:2*k,2*k-3:2*k)...
           +elementContribution([x(k), x(k+1)],'EI');
endfor
b(2*length(x)-3)=1e3;  # units in mm, not m

u=A\b;   %% solving the system

u=reshape(u,2,length(x)-1);
```

```
y=[0 u(1,:)];
plot(x,y)
displacement=y(length(y))
springConstant=1/displacement
```

To find the eigenmodes the number of nodes should be increased. Using 80 nodes we find the first four frequencies to be

$$\nu_1 \approx 7.2 \quad , \quad \nu_2 \approx 436 \quad , \quad \nu_3 \approx 4500 \quad , \quad \nu_4 \approx 11 \cdot 10^3$$

and the corresponding modes are shown in the right half of figure 4.15.

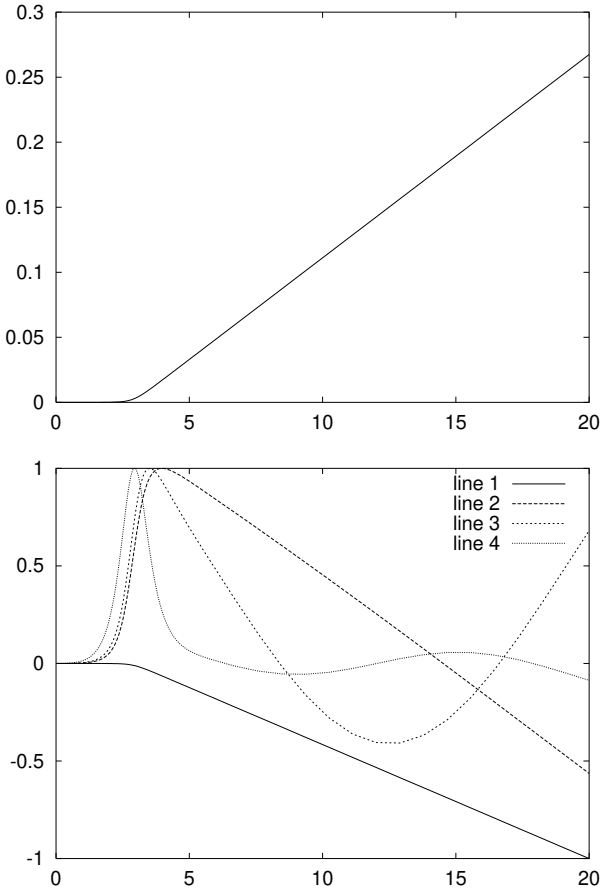


Figure 4.15: Displacment and first 4 eigenmodes of a force sensor

Once the above code is set up it is very easy to run the calculations again with modified parameters. Only very few entries have to be changed. This illustrates an advantage of **mathematical modeling**. It is very easy to modify parameters and 'see what happens'. This might save an engineer from having to built a few prototypes before finding a good solution.

4.9 Exercises

• **Exercise 4–1:**

Find an approximate solution to the boundary value problem

$$\begin{aligned} -x\,y''(x) - y'(x) + 7\,y(x) &= \sin x \qquad \text{for} \quad 0 < x < 3 \\ y(0) &= 3 \\ y'(3) &= 0 \end{aligned}$$

using the *Mathematica* command `BVP[]` with 10 elements.

• Exercise 4–2:
The simplest useful finite element in one variable is based on piecewise linear interpolation. It is advisable to use Gauss integration to improve accuracy. This will lead to better results than the construction in section 4.4.

- Construct the element matrices for a linear interpolation and Gauss integration with 2 points of support.
- Use the above result to write code in *Mathematica* similar to `BVP[]` to solve boundary values problems.
- Test your code with some typical examples.

• Exercise 4–3:
Recode the *Mathematica* command `BVP[]` in `MATLAB` and solve a few example problems to test the code. A good documentation should be provided.

• Exercise 4–4:
Find the element matrices for a 'new' element with the help of *Mathematica*.

- Use the values of the function at the two endpoints and at three interior points as the five degrees of freedom. An interpolating polynomial of degree four should be used.
- Base your integration on 5 Gauss points.
- Determine the order of convergence of this method.

One should be able to handle all terms in the functional.

$$F(u) = \int_a^b \frac{1}{2}\,a(x)\,(u'(x))^2 + \frac{1}{2}\,b(x)\,u(x)^2 + g(x)\cdot u(x)\;dx$$

The methods and calculations are similar to section 4.5.

• Exercise 4–5:
Use the results from the previous exercise to write code similar to `BVP[]`. Solve a few test problems.

• Exercise 4–6:
Find matrices for a 'new' element with *Mathematica*.

- Use the values of the function and its derivative at the endpoints as the four degrees of freedom. Cubic interpolation should be used.
- Base your integration on 3 Gauss points.
- Determine the order of convergence of this method.

One should be able to handle all terms in the functional.

$$F(u) = \int_a^b \frac{1}{2}\, a(x)\, (u'(x))^2 + \frac{1}{2}\, b(x)\, u(x)^2 + g(x) \cdot u(x)\; dx$$

The methods and calculations are similar to section 4.5, starting on page 77.

• Exercise 4–7:
Use the results from the previous exercise to write code similar to `BVP[]`. Since the unknown function and its derivative have to match at the connecting points of the intervals it is more difficult to assemble the matrices in this case.

• Exercise 4–8:
Find the element matrices for a 'new' element with the help of *Mathematica*.

- Use the values of the function and its derivative at the endpoints as the four degrees of freedom. Cubic interpolation should be used.
- Base your integration on 3 Gauss points.
- Determine the order of convergence of this method.

One should be able to handle all terms in the functional.

$$F(u) = \int_a^b \frac{1}{2}\, a(x)\, (u'(x))^2 + \frac{1}{2}\, b(x)\, u(x)^2 + g(x) \cdot u(x)\; dx$$

The methods and calculations are similar to section 4.5.

• Exercise 4–9:
Use the results from the previous exercise to write code similar to `BVP[]`. This approach has the advantage, that the approximate solution will be continuously differentiable, i.e. no „corners" in the solution.

Since the unknown function and its derivative have to match at the connecting points of the intervals it is more difficult to assemble the matrices in this case. Solve a few test problems.

• Exercise 4–10:
Find matrices for a 'new' element with *Mathematica*.

- Use the values of the function and its derivative at the endpoints as the four degrees of freedom. Cubic interpolation should be used.
- Base your integration on 3 Gauss points.

One should be able to handle all terms in the functional

$$F(u) = \int_a^b \frac{1}{2}\, a(x)\, (u''(x))^2 + \frac{1}{2}\, b(x)\, (u'(x))^2 + \frac{1}{2}\, c(x)\, u(x)^2 + g(x) \cdot u(x)\; dx$$

Thus boundary value problems of fourth order can be handled. The methods and calculations are comparable to section 4.5, starting on page 77.

• Exercise 4–11:
Use the results from the previous exercise to write code similar to `BVP[]`. This code will alow to solve problems of a bending beam or the bending of a circular plate.

Chapter 5

Convergence and finite difference schemes

5.1 Convergence of the approximate solutions for boundary value problems

The differential equations to be solved correspond to minimum of the functional

$$F(u) = \int_a^b \frac{1}{2}\, a(x)\, (u'(x))^2 + \frac{1}{2}\, b(x)\, u(x)^2 + g(x) \cdot u(x)\; dx$$

(see page 34). The corresponding differential equation is

$$\frac{d}{dx}\left(a(x)\, \frac{d\, u(x)}{dx} \right) - b(x)\, u(x) = g(x) \tag{5.1}$$

(see also equation (4.6) on page 76) and we also considered weak solution, i.e.

$$\int_a^b a(x)\, u'(x)\, \phi'(x) + b(x)\, u(x)\, \phi(x)\; dx = \int_a^b g(x)\, \phi(x)\; dx$$

for 'all' function $\phi(x)$. The boundary conditions have to be added. We can have Dirichlet or Neumann boundary conditions.

Dirichlet condition	$u(x) = h(x)$	for	$x = a$	or	$x = b$
Neumann condition	$a(x)\, \frac{d\, u}{dx} = r(x)$	for	$x = a$	or	$x = b$

From now on we assume that at either end point one boundary condition has to be satisfied, together with the differential equation (5.1). Thus we arrive at a **BVP** (**B**oundary **V**alue **P**roblem).

In general the exact u_e solution can not be found and we have to settle for an approximate solution u_h, where the parameter h corresponds to the typical length of the elements used for the approximation. Obviously we hope for the solution u_h to converge to the exact solution u_e as h approaches 0. It is the goal of this section to show under what circumstances this is in fact the case and also to determine the rate of convergence. The methods and ideas used can also be applied to partial differential equations of multiple variables.

5.1.1 Basic assumptions and regularity results

For the results of this section the be correct we need assumptions on the functions $a(x)$, $b(x)$ and $g(x)$, such that the solution of the boundary value problem is well behaved. Throughout the section we assume:

- $a(x)$, $b(x)$ and $g(x)$ continuous, bounded functions.

- There is a positive number α_0 such that $0 < \alpha_0 \leq a(x) \leq \alpha_1$ and $0 \leq b(x) \leq \beta_1$ for all x.
- The quadratic functional $F(u)$ is **strictly positive definite**. This condition is satisfied if
  - either at least one Dirichlet boundary condition is imposed.
  - or the function $b(x)$ is strictly positive on a subinterval.

  There are other combinations of conditions to arrive at a strictly positive functional, but the above two are easiest to verify.

With the above assumptions we have the following result, whose proof is left to mathematicians.

5–1 Theorem : *For each function $g(x)$ the BVP (5.1) has exactly one solution u. If the functions $a(x)$ and $b(x)$ are smooth then we have the estimate*

$$\int_a^b |u(x)|^2 + |u'(x)|^2 + |u''(x)|^2 \, dx \leq C \int_a^b |g(x)|^2 \, dx$$

for some constant C, independent on the function g.
As a rule of thumb we know that the solution u is $(k+2)$-times differentiable if g is k-times differentiable. ◇

This mathematical result tells us that there is a unique solution of the boundary value problem, but it does not give the solution. Now we use the finite element method to find numerical approximations $u_h(x)$ to this exact solution $u_e(x)$.

5.1.2 Function spaces, norms and continuous functionals

In view of the above definition of a weak solution we define for functions u and v

$$\begin{aligned} \langle u, v \rangle &:= \int_a^b u(x)\, v(x) \, dx \\ A(u, v) &:= \int_a^b a(x)\, u'(x)\, v'(x) + b(x)\, u(x)\, v(x) \, dx \end{aligned}$$

Basic properties of the integral imply that A is symmetric and linear with respect to each argument, i.e. for $\lambda_i \in \mathbb{R}$ we have

$$\begin{aligned} A(u, v) &= A(v, u) \\ A(\lambda_1 u_1 + \lambda_2 u_2, v) &= \lambda_1\, A(u_1, v) + \lambda_2\, A(u_2, v) \\ A(u, \lambda_1 v_1 + \lambda_2 v_2) &= \lambda_1\, A(u, v_1) + \lambda_2\, A(u, v_2) \end{aligned}$$

Now u is a weak solution if

$$A(u, \phi) = \langle g, \phi \rangle \quad \text{for all functions } \phi$$

We can also search for a minimum of

$$F(u) = \frac{1}{2}\, A(u, u) + \langle g, u \rangle$$

The only new aspect is a new notation. For the subsequent observations it is convenient to introduce two spaces of functions.

5–2 Definition : Let u be a piecewise differentiable function defined on the interval $[a, b]$. Then L_2 and V denote two sets of functions, both spaces equipped with a norm.[1]

$$\begin{aligned} L_2 &:= \{u : [a,b] \to \mathbb{R} \mid u \text{ is square integrable}\} \\ \|u\|_2^2 &:= \langle u, u\rangle = \int_a^b u^2(x)\, dx \end{aligned}$$

For the function $u(x)$ to be in the smaller subspace V we require the function u and its derivative u' to be square integrable and u has to satisfy the Dirichlet boundary condition (if there are any imposed). The norm in this space is given by

$$\|u\|_V^2 := \|u'\|_2^2 + \|u\|_2^2 = \int_a^b (u'(x))^2 + u^2(x)\, dx$$

L_2 and V are vector spaces and $\langle\, .\, ,\, .\,\rangle$ is a scalar product on L_2. Obviously we have

$$V \subset L_2 \quad \text{and} \quad \|u\|_2 \le \|u\|_V$$

Since the 'energy' to be minimised

$$F(u) = A(u,u) = \int_a^b a(x)\,(u'(x))^2 + b(x)\,u^2(x)\, dx$$

is closely related to $\|u\|_V^2$ this norm is often called an **energy norm**.

If $u\, v \in V$ the expression $A(u, v)$ can be computed and we find

$$\begin{aligned} |A(u,v)| &= \left| \int_a^b a(x)\,u'(x)\,v'(x) + b(x)\,u(x)\,v(x)\, dx \right| \\ &\le \int_a^b |a(x)|\,|u'(x)|\,|v'(x)| + |b(x)|\,|u(x)|\,|v(x)|\, dx \\ &\le \alpha_1 \int_a^b |u'(x)|\,|v'(x)|\, dx + \beta_1 \int_a^b |u(x)|\,|v(x)|\, dx \\ &\le \alpha_1 \|u'\|_2\, \|v'\|_2 + \beta_1 \|u\|_2\, \|v\|_2 \\ &\le (\alpha_1 + \beta_1)\, \|u\|_V\, \|v\|_V \end{aligned}$$

If we assume that $0 < \beta_0 \le b(x)$ for all x then

$$\begin{aligned} A(u,u) &= \int_a^b a(x)\,u'(x)\,u'(x) + b(x)\,u(x)\,u(x)\, dx \\ &\ge \int_a^b a(x)\,|u'(x)|^2 + b(x)\,|u(x)|^2\, dx \\ &\ge \int_a^b \alpha_0\,|u'(x)|^2 + \beta_0\,|u(x)|^2\, dx \\ &\ge \min\{\alpha_0, \beta_0\} \int_a^b |u'(x)|^2 + |u(x)|^2\, dx \\ &= \gamma_0 \|u\|_V^2 \end{aligned}$$

[1] A mathematically correct introduction of these function spaces is well beyond the scope of these notes. The tools of Lebesgue integration and completion of spaces is not available. As a consequence we ignore most of the mathematical problems.

It can be shown that the final inequality is correct as long as the assumptions in section 5.1.1 are satisfied. Thus we find

$$\gamma_0 \|u\|_V^2 \leq A(u,u) \leq (\alpha_1 + \beta_1) \|u\|_V^2 \quad \text{for all} \quad u \in V \tag{5.2}$$

This inequality is the starting point for most theoretical results on boundary value problems of the type we consider in these notes.

5.1.3 Convergence of the finite dimensional approximation

The space V contains all piecewise differentiable, continuous functions and thus V is not a finite dimensional vectors space. For a fixed parameter $h > 0$ we choose a discretisation of the interval $[a, b]$ in finite many intervals of typical length h. Then we consider only continuous functions that are polynomials of a given degree (e.g. 2) on each of the intervals, i.e. a piecewise quadratic function. This leads to a finite dimensional subspace V_h, i.e. finitely many degrees of freedom.

$$V_h \subset V \quad \text{finite dimensional subspace}$$

Instead of searching for a minimum on all of V we now only consider the functions in $V_h \subset V$ to find the minimiser of the functional. This is illustrated in table 5.1. We hope that the minimum $u_h \in V_h$ we close to the exact solution $u_e \in V$. The main goal of this section is to show that this is in fact the case. The ideas of proofs are adapted from [John87, p.54] and [Davi80, §7] and can also be used in more general situations, e.g. for differential equations with more independent variables. To simplify the proof of the abstract error estimate we use two lemmas.

	original problem	approximate problem
functional to minimise amongst functions	$F(u) = \frac{1}{2} A(u,u) + \langle g, u\rangle$ $u \in V$ (infinite dimensional)	$F(u_h) = \frac{1}{2} A(u_h,u_h) + \langle g, u_h\rangle$ $u_h \in V_h$ (finite dimensional)
necessary condition for minimum	$A(u,\phi) + \langle g, \phi\rangle = 0$ for all $\phi \in V$	$A(u_h,\phi_h) + \langle g, \phi_h\rangle = 0$ for all $\phi_h \in V_h$
main goal	$u_h \longrightarrow u$ as $h \to 0$	

Table 5.1: Minimisation of original and approximate problem

5–3 Lemma : *If u_h is a minimiser of the functional F on V_h, i.e.*

$$F(u_h) = \frac{1}{2} A(u_h,u_h) + \langle g, u_h\rangle \leq \frac{1}{2} A(v_h,v_h) + \langle g, v_h\rangle = F(v_h) \quad \textit{for all} \quad v_h \in V_h$$

then

$$A(u_h,\phi_h) + \langle g, \phi_h\rangle = 0 \quad \textit{for all} \quad \phi_h \in V_h$$

◇

Proof : The derivative of

$$\begin{aligned} g(t) &= F(u_h + t\phi) = \frac{1}{2} A(u_h + t\phi, u_h + t\phi) + \langle g, u_h + t\phi\rangle \\ &= \frac{1}{2} A(u_h,u_h) + \langle g, u_h\rangle + t\,(A(u_h,\phi) + \langle g,\phi\rangle) + t^2\,\frac{1}{2} A(\phi_h,\phi_h) \end{aligned}$$

has to vanish at $t = 0$ for all $\phi_h \in V_h$. This implies the result. □

5–4 Lemma : *Let $u_e \in V$ be the minimiser of the functional F on all of V and let $u_h \in V_h$ be the minimiser of*

$$F(\psi_h) = \frac{1}{2}\, A(\psi_h, \psi_h) + \langle g, \psi_h \rangle$$

amongst all $\psi_h \in V_h$. This implies that $u_h \in V_h$ is also the minimiser of

$$G(\psi_h) = A(u_e - \psi_h, u_e - \psi_h)$$

amongst all $\psi_h \in V_h$. ◇

Proof : If $u_h \in V_h$ minimises F in V_h and $u_e \in V$ minimises F in V then the previous lemma implies

$$\begin{aligned} A(u_e, \phi_h) &= -\langle g, \phi_h \rangle \quad \text{for all} \quad \phi_h \in V_h \\ A(u_h, \phi_h) &= -\langle g, \phi_h \rangle \quad \text{for all} \quad \phi_h \in V_h \end{aligned}$$

and thus $A(u_e - u_h, \phi_h) = 0$. This leads to

$$\begin{aligned} G(u_h + \phi_h) &= A(u_e - u_h - \phi_h, u_e - u_h - \phi_h) \\ &= A(u_e - u_h, u_e - u_h) - 2\, A(u_e - u_h, \phi_h) + A(\phi_h, \phi_h) \\ &= A(u_e - u_h, u_e - u_h) + A(\phi_h, \phi_h) \\ &\geq A(u_e - u_h, u_e - u_h) \end{aligned}$$

Equality occurs only if $\phi_h = 0$. Thus $\phi_h = 0 \in V_h$ is the unique minimiser of the above function and the result is established. □

5–5 Theorem : *(Abstract error estimate)*
If u_e is the minimiser of the functional

$$F(u) = \frac{1}{2}\, A(u, u) + \langle g, u \rangle$$

amongst all $u \in V$ and $u_h \in V_h$ is the minimiser of F amongst all u_h in the subspace $V_h \subset V$, then the distance of u_e and u_h (in the V–norm) can be estimated. There exists a positive constant k such that

$$\|u_e - u_h\|_V \leq k \min_{\psi_h \in V_h} \|u_e - \psi_h\|_V$$

The constant k is independent on h. ◇

The above result carries the name of **Lemma of Céa**.

As a consequence of this result we have to be able to approximate an the exact solution $u_e \in V$ by approximate function $\psi_h \in V_h$ and the error of the finite element solution $u_h \in V_h$ is smaller than the approximation error, except for the factor k. Thus the lemma reduces the question of estimating the error of the approximate solution to a question of estimationd the approximation error of a function in the appropriate norm. Standard interpolation result allow to estimate the error of the approximation, assuming some regularity on the exact solution u.

Proof : Use equation (5.2) and the above lemma to conclude that

$$\begin{aligned} \gamma_0\, \|u_e - u_h\|_V^2 &\leq A(u_e - u_h, u_e - u_h) \leq A(u_e - u_h - \phi_h, u_e - u_h - \phi_h) \\ &\leq (\alpha_1 + \beta_1)\, \|u_e - u_h - \phi_h\|_V^2 \quad \text{for all} \quad \phi_h \in V_h \end{aligned}$$

and thus

$$\|u_e - u_h\|_V \leq \sqrt{\frac{\alpha_1 + \beta_1}{\gamma_0}}\, \|u_e - u_h - \phi_h\|_V \quad \text{for all} \quad \phi_h \in V_h$$

As $\phi_h \in V_h$ is arbitrary we find the claimed result. □

Let u be a smooth function and divide the interval $[a,b]$ in subintervals of typical length h. On each subinterval the two endpoints (and may be interior points) are used to construct an interpolating function $\Pi_h u$. The operator Π_h can be considered a **projection operator** of V onto the finite dimensional subspace V_h. We have

$$\Pi_h \; : \; V \longrightarrow V_h \qquad , \qquad u \mapsto \Pi_h u$$

The following two results can be found in many books on calculus in one variable or numerical analysis. We omit the proof.

5–6 Result : *(Piecewise linear interpolation)*
*If u is at least twice differentiable we use the values at the endpoints of the subintervals to construct the function $\Pi_h u$, the **piecewise linear interpolation** function. Basic approximation theory implies that there is a constant M, such that*

$$\begin{aligned} |u(x) - \Pi_h u(x)| &\leq M\, h^2 \quad \text{for all} \quad a \leq x \leq b \\ |u'(x) - \Pi_h u'(x)| &\leq M\, h \quad \text{for all} \quad a \leq x \leq b \end{aligned}$$

Thus an integration implies that there is a constant c such that

$$\|u - \Pi_h u\|_V \leq c\, h$$

The constant depends on the maximal value of the second derivative of the original function u. ◇

5–7 Result : *(Piecewise quadratic interpolation)*
*If u is at least three times differentiable we use the values at the endpoints and the midpoint of the subintervals to construct the function $\Pi_h u$, the **piecewise quadratic interpolation** function. Basic approximation theory implies that there is a constant M, such that*

$$\begin{aligned} |u(x) - \Pi_h u(x)| &\leq M\, h^3 \quad \text{for all} \quad a \leq x \leq b \\ |u'(x) - \Pi_h u'(x)| &\leq M\, h^2 \quad \text{for all} \quad a \leq x \leq b \end{aligned}$$

Thus an integration implies that there is a constant c such that

$$\|u - \Pi_h u\|_V \leq c\, h^2$$

The constant depends on the maximal value of the third derivative of the original function u. ◇

To obtain an improved convergence result 5–10 we need an additional interpolation estimate. The proof of this result uses the lemma below.

5–8 Lemma : *Let $a = x_0 < x_1 < x_2 < \ldots < x_n = b$ a partition of the interval $[a,b]$ with $\max_i\{x_i - x_{i-1}\} \leq h$. If u is a continuous function on $[a,b]$ with $u(x_i) = 0$, twice differentiable on each subinterval $[x_{i-1}, x_i]$ then*

$$\|u\|_V \leq \sqrt{2}\, h\, \|u''\|_2 \quad \text{and} \quad \|u\|_2 \leq h^2\, \|u''\|_2$$

◇

Proof : Consider a function u with $u(0) = u(h) = 0$. Thus there is a $0 < \xi < h$ such that $u'(\xi) = 0$. Then we find for $0 \leq x \leq h$

$$\begin{aligned} u(x) &= u(0) + \int_0^x u'(s)\, ds \\ |u(x)| &\leq \int_0^x |u'(s)| \cdot 1\, ds \end{aligned}$$

$$\begin{aligned} &\leq \left(\int_0^x |u'(s)|^2\, ds\right)^{1/2} \cdot \left(\int_0^x 1\, ds\right)^{1/2} \\ &\leq \left(\int_0^h |u'(s)|^2\, ds\right)^{1/2} \cdot \sqrt{h} = \sqrt{h}\, \|u'\|_2 \\ \|u\|_2^2 &= \int_0^h |u(x)|^2\, dx \leq \int_0^h h\, \|u'\|_2^2\, dx = h^2\, \|u'\|_2^2 \end{aligned}$$

By similar arguments and the intermediate value theorem we arrive at

$$\begin{aligned} |u'(x)| &\leq \sqrt{h}\, \|u''\|_2 \\ \|u'\|_2^2 &= \int_0^h |u'(x)|^2\, dx \leq h^2\, \|u''\|_2^2 \end{aligned}$$

and thus conclude

$$\begin{aligned} \|u\|_2^2 &\leq h^2\, \|u'\|_2^2 \leq h^4\, \|u''\|_2^2 \\ \|u\|_V^2 &= \|u\|_2^2 + \|u'\|_2^2 \leq (h^4 + h^2)\, \|u''\|_2^2 \leq (1 + h^2)\, h^2\, \|u''\|_2^2 \leq 2\, h^2\, \|u''\|_2^2 \end{aligned}$$

if $h < 1$. Thus the required inequalities are verified on the interval $[0, h]$. Now consider the interval $[a, b]$ and its partition. The above computation is valid on each subinterval and a summation will lead to the claimed result. □

5–9 Result : *(Estimate of interpolation error)*
Let u be a twice differentiable function and $\Pi_h u$ the piecewise linear interpolation. Then $w = u - \Pi_h u$ satisfies the assumptions of the previous lemma and $(u - \Pi_h u)'' = u''$. Thus we conclude

$$\|u - \Pi_h u\|_V \leq c\, h\, \|u''\|_2$$

◇

Proof : This is an immediate consequence of the above lemma.

□

Now we have all the ingredients to state and proof the basic convergence results for finite element solutions to boundary value problems in one variable. The exact solution $u_0 \in V$ to be approximated is the minimum of the functional

$$F(u) = \int_a^b \frac{1}{2}\, a(x)\, (u'(x))^2 + \frac{1}{2}\, b(x)\, u(x)^2 + g(x) \cdot u(x)\, dx = \frac{1}{2}\, A(u, u) + \langle g, u \rangle$$

The exact solution u_0 is smooth (often differentiable) if g is (theorem 5–1). Instead of searching on the space V we restrict the search on the finite dimensional subspace V_h and arrive at the approximate minimiser u_h. Thus the error function $e = u_h - u_0$ has to be as small as possible for the approximation to be of a good quality. In fact we hope for a convergence

$$u_h \longrightarrow u_o \quad \text{as} \quad h \longrightarrow 0$$

in some sense to be specified.

5–10 Theorem : *If the subspace V_h is generated by the piecewise linear interpolation operator Π_h then we find*

$$\|u_h - u_0\|_V \leq C\,h \quad \textit{and} \quad \|u_h - u_0\|_2 \leq C_1\,h^2$$

for some constants C and C_1 independent on h.
We may say that

- *u_h converges to u_0 with an error proportional to h^2 as $h \to 0$.*
- *u_h' converges to u_0' with an error proportional to h as $h \to 0$.*

◇

Proof : The interpolation result 5–6 and the abstract error estimate 5–5 imply immediately

$$\|u_h - u_0\|_V \leq k \min_{\phi_h \in V_h} \|\psi_h - u_0\|_V \leq k\,\|\Pi_h u_0 - u_0\|_V \leq k\,c\,h$$

Which is already the first of the desired estimates. The second estimate requires considerably more work. The method of proof is known as **Nitsche trick** and is due to Nitsche and Aubin. A good presentation is given in [StraFix73, §3.4] or [KnabAnge00, Satz 3.37].

Let $w \in V$ be the minimiser of the functional

$$\frac{1}{2}\,A(u,u) + \langle e, u\rangle$$

The fundamental theorem 5–1 implies $\|w''\|_2^2 \leq k\,\|e\|_2^2$. The interpolation result 5–9 and theorem 5–5 imply

$$\begin{aligned} A(w,w) &\leq A(w - \Pi_h w, w - \Pi_h w) \leq (\alpha_1 + \beta_1)\,\|w - \Pi_h w\|_V^2 \\ &\leq c^2\,h^2\,(\alpha_1 + \beta_1)\,\|w''\|_2^2 \\ &\leq k\,c^2\,h^2\,(\alpha_1 + \beta_1)\,\|e\|_2^2 \end{aligned}$$

Since w is a minimiser of the functional we conclude

$$A(w,\psi) + \langle e, \psi\rangle = 0 \quad \text{for all} \quad \psi \in V$$

By choosing $\psi = e$ we arrive at

$$-A(w,e) = \langle e, e\rangle = \|e\|_2^2$$

Now use the Cauchy–Schwartz inequality to conclude that

$$\begin{aligned} \|e\|_2^2 &= |A(w,e)| \leq (A(w,w))^{1/2} \cdot (A(e,e))^{1/2} \\ &\leq c\,h\,\sqrt{k\,(\alpha_1 + \beta_1)}\,\|e\|_2 \cdot C\,h \end{aligned}$$

A division by $\|e\|_2$ leads to

$$\|e\|_2 \leq C_1\,h^2$$

□

5–11 Theorem : *If the subspace V_h is generated by the piecewise quadratic interpolation operator then we find*

$$\|u_h - u_0\|_V \leq C\,h^2 \quad \textit{and} \quad \|u_h - u_0\|_2 \leq C_1\,h^3$$

for some constants C and C_1 independent on h.
We may say that

- *u_h converges to u_0 with an error proportional to h^3 as $h \to 0$.*
- *u_h' converges to u_0' with an error proportional to h^2 as $h \to 0$.*

◇

Proof : The interpolation result 5–7 and the abstract error estimate 5–5 imply immediately

$$\|u_h - u_0\|_V \leq k \min_{\phi_h \in V_h} \|\psi_h - u_0\|_V \leq k \, \|\Pi_h u_0 - u_0\|_V \leq k \, c \, h^2$$

Which is already the first of the desired estimates. The second estimate would again ask for some more work, but the computations are identical to the proof of theorem 5–10. □

The method used in the *Mathematica* code `BVP[]` leads to the results in theorem 5–11. Thus we can illustrate the result with e few sample computations.

5–12 Example : Consider the boundary value problem

$$\begin{aligned} 2\,u'' - u &= \sin(\tfrac{\pi}{2} - x) \qquad \text{for} \quad 0 < x < 2 \\ u'(0) &= 0 \\ u(2) &= 0 \end{aligned}$$

Using the packages

Mathematica

```
<<BVP.m;
<<Interpol2.m;
```

we can find an finite element solution with the commands

Mathematica

```
Clear[a,b,c,f,x,t,n];
a=Function[x,2];
b=Function[x,-1];
f=Function[x,Sin[Pi/2-x]];
n=22;
x=Table[t,{t,0,2,2/n}];
data=BVP[a,b,f,x,{"N","D"},{0,0}];
ListPlot[data,PlotStyle -> PointSize[0.01]];

Clear[x]
uexact[x_]=y[x]/.DSolve[{D[a[x]*D[y[x],x],x]-b[x]*y[x]==f[x],y'[0]==0,y[2]==0},
                        y[x],x][[1]]
Duexact[x_]=D[uexact[x],x]
```

The last two lines compute the exact solution and its derivative.

The above code shows that it is possible to compute the solution for different numbers of elements, i.e. for smaller and smaller values of h. With the code below such a list is generated for values at $x = 1.01$.

Mathematica

```
testx=1.01;
res=Table[{n=3*k+1,x=Table[t,{t,0,2,2/n}];
            data=BVP[a,b,f,x,{"N","D"},{0,0}];
      Abs[Interpol2[testx,data]-uexact[testx]],
      Abs[DInterpol2[testx,data]-Duexact[testx]]},
   {k,1,30,3}]
```

Since $h \sim \frac{1}{n}$ we expect for the error function $e(x)$ the behaviour

$$e(x) \approx c_1 \, \frac{1}{n^3} \quad \text{and} \quad e'(x) \approx c_2 \, \frac{1}{n^2}$$

By applying the logarithm function we find

$$\log e(x) \approx \log c_1 - 3 \, \log n \quad \text{and} \quad \log e'(x) \approx \log c_2 - 2 \, \log n$$

Thus we should find straight lines with slopes of -3 (resp. -2). `BVP[]` shows that the rate of convergence is in fact h^3 for the error and h^2 for its derivative. The figure 5.1 shows the logarithmic plot of the error $e(1.01)$ as function of $\log n$.

Mathematica

```
resu=Map[Drop[#,-1]&,res];
ListPlot[Log[resu],PlotJoined ->True,PlotRange->All];

logresu=Log[Abs[resu]];
n=Length[resu];
(logresu[[1,2]]-logresu[[n,2]])/(logresu[[1,1]]-logresu[[n,1]])
.
-2.93686
```

Mathematica

```
logresu=Log[Abs[resu]];n=Length[logresu];
(logresu[[1,2]]-logresu[[n,2]])/(logresu[[1,1]]-logresu[[n,1]])
.
-2.17034
```

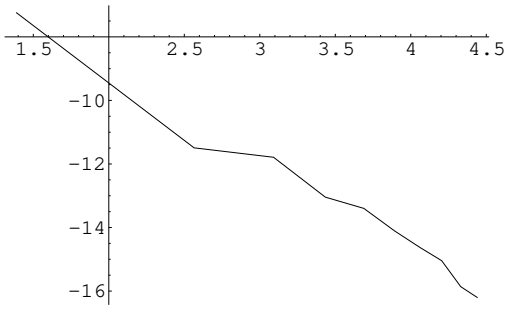


Figure 5.1: Logarithmic plot of the approximation error

Surprisingly the error $e(x)$ converges like h^4 at the grid points. This can be verified by choosing `testx=1.00` in the above calculations. The author has not found a good explanation for this yet. ♢

5.2 A finite difference approximation to an ordinary differential equation

As an first example we consider the ordinary differential equation

$$\dot{y}(t) = -\frac{1}{2}\, y(t) \quad \text{with} \quad y(0) = y_0$$

with the exact solution $y(t) = y_0\, e^{-t/2}$. Since

$$\dot{y}(t) = \lim_{h \to 0+} \frac{y(t+h) - y(t)}{h}$$

we use the approximate equation

$$\frac{y(t+h) - y(t)}{h} = -\frac{1}{2}\, y(t) \qquad \Longrightarrow \qquad y(t+h) = y(t) + h\, \frac{-1}{2}\, y(t) = \left(1 + h\, \frac{-1}{2}\right) y(t)$$

to construct the solution at discrete times $t_k = k \cdot h$. We obtain

$$y(k \cdot h) \approx y_k = (1 - \frac{h}{2})\, y_{k-1} = (1 - \frac{h}{2})^2\, y_{k-2} = \ldots = (1 - \frac{h}{2})^k\, y_0$$

If we divide the interval $[0\,,\,T]$ in n subintervals of equal length $h = \frac{T}{n}$ we obtain

$$y(T) \approx (1 - \frac{T}{2\,n})^n\, y_0 \underset{n\to\infty}{\longrightarrow} e^{-T/2}\, y_0 = y(T)$$

Thus the approximate solution converges to the exact solution as the stepsize h approaches 0 . Exact and approximate solution are shown in figure 5.2.

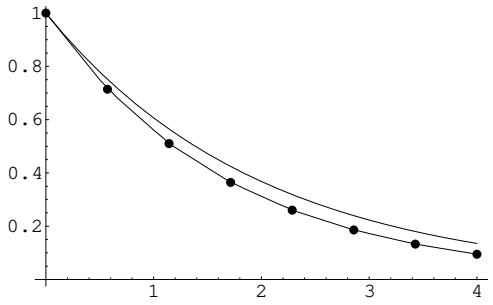


Figure 5.2: Exact and approximate solution to an ODE

5.2.1 Finite difference approximations

Using the Taylor approximation

$$y(t+x) = y(t) = y'(t) \cdot x + \frac{y''(t)}{2}\, x^2 + \frac{y'''(t)}{3!}\, x^3 + + \frac{y''(t)}{4!}\, x^4 + O(x^5)$$

with different values for x ($x = \pm h$) one can verify that

$$y'(t) = \frac{y(t+h) - y(t)}{h} - \frac{y''(t)}{2}\, h + O(h^2) = \frac{y(t+h) - y(t)}{h} + O(h)$$

This and other finite difference approximations of derivatives are given in table 5.2 .

forward difference	$y'(t) = \frac{y(t+h)-y(t)}{h} + O(h)$
backward difference	$y'(t) = \frac{y(t)-y(t-h)}{h} + O(h)$
centered difference	$y'(t) = \frac{y(t+h/2)-y(t-h/2)}{h} + O(h^2)$
	$y''(t) = \frac{y(t-h)-2\,y(t)+y(t+h)}{h^2} + O(h^2)$
	$y'''(t) = \frac{-y(t-h)+3\,y(t)-3\,y(t+h)+y(t+2\,h)}{h^3} + O(h)$
	$y'''(t) = \frac{-y(t-3\,h/2)+3\,y(t-h/2)-3\,y(t+h/2)+y(t+3\,h/2)}{h^3} + O(h^2)$
	$y^{(4)}(t) = \frac{y(t-2\,h)-4\,y(t-h)+6\,y(t)-4\,y(t+h)+y(t+2\,h)}{h^2} + O(h^2)$

Table 5.2: Finite difference approximations

As a sample problem we consider the initial values problem

$$y'(t) = -\lambda\, y(t) \quad \text{with} \quad y(0) = y_0 \tag{5.3}$$

with the exact solution $Y(t) = y_0\, e^{-\lambda t}$. We assume that $\lambda > 0$ and thus the solution is bounded and will converges to 0 as t tends to infinity. We expect approximate solution to exhibit the same behavior. Now we choose a step size $h > 0$ and discretize the time domain $0 \le t < \infty$ by $t_n = n\,h$. Then we try to find approximate values $y_n \approx Y(n\,h)$ be considering finite difference approximation to (5.3).

5–13 Definition : The finite difference approximation is said to be **consistent** if for a smooth solution $y(t)$ of equation (5.3) the difference approximation converges to the differential equation as h tends to 0.

The finite difference approximation is said to be **stable** if the absolut values of all solutions remain bounded, independent on h.

The finite difference approximation is said to be **convergent** if for a fixed value of T the approximate solution at T converges towards the exact solution $Y(T)$ as $h \to 0+$.

Now we examine stability and consistency of a few approximation schemes for equation (5.3).

5.2.2 Forward difference

Due to the first line in table 5.2 the forward difference approximation is consistent of order 1.

$$\begin{aligned} \frac{1}{h}\,(y(t+h) - y(t)) &= -\lambda\, y(t) \\ y(t+h) &= y(t) - h\,\lambda\, y(t) = (1 - \lambda\, h)\; y(t) \end{aligned}$$

With $y_i = y(i\,h)$ we find $y_{i+1} = (1 - \lambda\, h)\; y_i$. The initial condition now reads $y_0 = y(0)$. With an iteration we arrive at $y_n = (1 - \lambda\, h)^n\, y_0$.

$$\begin{aligned} y_n &= (1 - \lambda\, h)^n\; y_0 \longrightarrow 0 \quad \text{as} \quad n \to \infty \quad \text{if} \quad \lambda\, h < 2 \\ y_n &= (1 - \lambda\, h)^n\; y_0 \quad \text{diverges as} \quad n \to \infty \quad \text{if} \quad \lambda\, h > 2 \end{aligned}$$

Thus the solutions will only remain bounded if $0 < h < 2/\lambda$, thus we have **conditional stability**.

For a fixed value of the time T we consider $h = T/n$

$$y(T) \approx y_n = (1 - \lambda\, \frac{T}{n})^n\, y_0 \longrightarrow e^{-\lambda T}\, y_0 \quad \text{as} \quad n \to \infty$$

and thus the approximation scheme is convergent.

5.2.3 Backward difference

Again based on table 5.2 the backward difference approximation is consistent of order 1.

$$\begin{aligned} \frac{1}{h}\,(y(t+h) - y(t)) &= -\lambda\, y(t+h) \\ (1 + \lambda\, h)\; y(t+h) &= y(t) \\ y(t+h) &= \frac{1}{1 + \lambda\, h}\; y(t) \\ y_n &= \left(\frac{1}{1 + \lambda\, h}\right)^n\; y_0 \longrightarrow 0 \quad \text{as} \quad n \to \infty \end{aligned}$$

Since $\lambda\, h > 0$ we find that the system is **unconditionally stable**. **Convergence** is verified by

$$y(T) \approx y_n = \left(\frac{1}{1 + \lambda\, \frac{T}{n}}\right)^n\; y_0 = \left(1 + \frac{\lambda\, T}{n}\right)^{-n}\; y_0 \longrightarrow e^{-\lambda T}\, y_0 \quad \text{as} \quad n \to \infty$$

5.2.4 Centered difference

The centered difference scheme is consistent of order 2.

$$\begin{aligned}
\frac{1}{h}\,(y(t+h)-y(t)) &= -\lambda\,\frac{y(t)+y(t+h)}{2}\\
(1+\lambda\,h/2)\,y(t+h) &= (1-\lambda\,h/2)\,y(t)\\
y(t+h) &= \frac{1-\lambda\,h/2}{1+\lambda\,h/2}\,y(t)\\
y_n &= \left(\frac{1-\lambda\,h/2}{1+\lambda\,h/2}\right)^n y_0 \longrightarrow 0 \quad \text{as} \quad n\to\infty
\end{aligned}$$

With calculations similar to the backward difference scheme we verify that this approximation is also **unconditionally stable** and **convergent**.

$$\begin{aligned}
y(T) &\approx y_n = \left(\frac{1-\lambda\,\frac{T}{2\,n}}{1+\lambda\,\frac{T}{2\,n}}\right)^n y_0 = \left(1-\frac{2\cdot\lambda\,\frac{T}{2\,n}}{1+\lambda\,\frac{T}{2\,n}}\right)^n y_0\\
&= \left(1-\frac{\lambda\,T}{n+\lambda\,T/2}\right)^n y_0 \longrightarrow e^{-\lambda\,T}\,y_0 \quad \text{as} \quad n\to\infty
\end{aligned}$$

5.3 General difference approximations, consistency, stability and convergence

To explain the approximation behavior of finite difference schemes we use the example problem

$$-y''(x)=f(x) \quad \text{for} \quad 0<x<L \quad \text{with boundary conditions} \quad y(0)=y(L)=0 \tag{5.4}$$

We assume that for a given function f the exact solution is given by y. This differential equation is replaced by a difference equation. For $n\in\mathbb{N}$ discretize the interval by $x_k = k\cdot h = k\,\frac{L}{n+1}$ and then consider an approximate solution $u_k \approx y(k\cdot h)$ for $k=0,1,2,\ldots,n,n+1$. The finite difference approximation of the second derivative in list 5.2 leads for interior points to

$$-\frac{u_{k-1}-2\,u_k+u_{k+1}}{h^2} = f_k = f(k\cdot h) \quad \text{for} \quad k=1,2,3,\ldots,n \tag{5.5}$$

The boundary conditions lead to $u_0=u_{n+1}=0$. These equation can be written in the form

$$\frac{1}{h^2}\begin{bmatrix}
2 & -1 & & & & \\
-1 & 2 & -1 & & & \\
 & -1 & 2 & -1 & & \\
 & & \ddots & \ddots & \ddots & \\
 & & & -1 & 2 & -1\\
 & & & & -1 & 2
\end{bmatrix}\cdot\begin{pmatrix} u_1\\ u_2\\ u_3\\ \vdots\\ u_{n-1}\\ u_n\end{pmatrix} = \begin{pmatrix} f_1\\ f_2\\ f_3\\ \vdots\\ f_{n-1}\\ f_n\end{pmatrix}$$

The solution of this linear system will create the values of the approximate solution at the grid points. Exact and approximate solution are shown in figure 5.3. As $h\to 0$ we hope that u will converge to the exact solution y .

To examine the behavior of the approximate solution we use a general framework for finite difference approximations to boundary value problems.

Consider functions defined on a domain $\Omega\subset\mathbb{R}^N$ and for a fixed mesh size h cover the domain with a discrete set of points $x_k\in\Omega$. This leads to the following vector spaces:

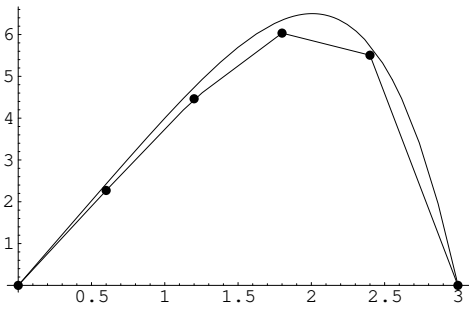


Figure 5.3: Exact and approximate solution to a boundary value problem

- E_1 is a space of functions defined on Ω. In the above example consider $u \in C^2([0,L]\,,\mathbb{R})$ with $u(0) = u(L) = 0$. On this space use the norm $\|u\|_{E_1} = \max\{|u(x)| \ : \ 0 \leq x \leq L\}$.
- E_2 is a space of functions defined on Ω. In the above example consider $f \in C^0([0,L]\,,\mathbb{R})$ with the norm $\|f\|_{E_2} = \max\{|f(x)| \ : \ 0 \leq x \leq L\}$.
- E_1^h is a space of discretized functions. In the above example consider $\vec{u} \in \mathbb{R}^n = E_1^h$, where $u_k = u(k \cdot h)$. The vector space E_1^h is equipped with the norm $\|\vec{u}\|_{E_1^h} = \max\{|u_k| \ : \ 1 \leq k \leq n\}$.
- E_2^h is also a space of discretized functions. In the above example consider $\vec{f} \in \mathbb{R}^n = E_1^h$, where $f_k = f(k \cdot h)$. The vector space E_2^h is equipped with the norm $\|\vec{f}\|_{E_2^h} = \max\{|f_k| \ : \ 1 \leq k \leq n\}$.

On these space we examine the following linear operations:

- For $u \in E_1$ let $F : E_1 \to E_2$ be the linear differential operator. In the above example $F(u) = u''$.
- For $\vec{u} \in E_1^h$ let $F_h : E_1^h \to E_2^h$ be the linear difference operator. In the above example

$$F_h(\vec{u})_k = \frac{u_{k-1} - 2\,u_k + u_{k+1}}{h^2}$$

- For $u \in E_1$ let $\vec{u} = P_1^h(u) \in E_1^h$ be the projection of the function $u \in E_1$ onto E_1^h. It is determined by evaluation the function at the points x_k.
- For $f \in E_2$ let $\vec{f} = P_2^h(f) \in E_2^h$ be the projection of the function $f \in E_2$ onto E_2^h. It is determined by evaluation the function at the points x_k.

The above operations are illustrated in figure 5.4

$$\begin{array}{ccc} u \in E_1 & \xrightarrow{\quad F \quad} & f \in E_2 \\ \Big\downarrow P_1^h & & \Big\downarrow P_2^h \\ u_h \in E_1^h & \xrightarrow{\quad F_h \quad} & f_h \in E_2^h \end{array}$$

$$\begin{array}{rcl} h & \longrightarrow & 0 \\ \|P_1^h u\| & \longrightarrow & \|u\| \\ \|P_2^h v\| & \longrightarrow & \|v\| \\ P_2^h(F(u)) & \approx & F_h(P_1^h(u)) \end{array}$$

Figure 5.4: A general approximation scheme for boundary value problems

5–14 Definition : For a given $f \in E_2$ let $u \in E_1$ be the solution of $F(u) = f$ and $\vec{u}$ the solution of $F_h(\vec{u}) = P_2^h(f)$.

- The above approximation scheme is said to be **convergent** of order p if

$$\|P_1^h(u) - \vec{u}\|_{E_1^h} \leq c_1\, h^p$$

  where the constant c_1 is independent on h, but it may depend on u.

- The above approximation scheme is said to be **consistent** of order p if

$$\|F_h(P_1^h(u)) - P_2^h(F(u))\|_{E_2^h} \leq c_2\, h^p$$

  where the constant c_2 is independent on h, but it may depend on u. This implies that the diagram in figure 5.4 is almost commutative as h approaches 0 .

- The above approximation scheme is said to be **stable** if the linear operator $F_h \in \mathcal{L}(E_1^h\,,\, E_2^h)$ is invertible and the exists a constant M, independent on h, such that

$$\|u_h\|_{E_1^h} \leq M\, \|F_h(u_h)\|_{E_2^h} \quad \text{for all} \quad u_h \in E_1^h$$

  This is equivalent to $\|F_h^{-1}\| \leq M$, i.e. the inverse linear operators of the approximate problems are uniformly bounded.

Now we can state a fundamental result for finite difference approximations to differential equations. The theorem is also known as **Lax equivalence theorem**. The result applies to a large variety of problems. We will examine a only a few of them.

5–15 Theorem : *If a finite difference scheme is consistent of order p and stable, then it is convergent of order p. A short formulation is:*

consistency and stability imply convergence

◇

Proof : Let u be the solution of $F(u) = f$ and $\vec{u}$ the solution of $F_h(\vec{u}) = P_2^h(f)$. Since the scheme is stable and consistent of order p we find

$$\begin{aligned}
\|P_1^h(u) - \vec{u}\|_{E_1^h} &= \|F_h^{-1}\left(F_h(P_1^h(u) - \vec{u})\right)\|_{E_1^h} \\
&\leq \|F_h^{-1}\|\, \|F_h(P_1^h(u)) - F_h(\vec{u})\|_{E_2^h} \\
&= M\, \|F_h(P_1^h(u)) - P_2^h(f)\|_{E_2^h} \\
&= M\, \|F_h(P_1^h(u)) - P_2^h(F(u))\|_{E_2^h} \\
&\leq M\, c\, h^p
\end{aligned}$$

Thus the finite difference approximation is convergent. □

The table 5.3 illustrates the abstract concept using the example equation (5.4).

	general problem	sample problem (5.4)
exact equation	$F(u) = f$	$u''(x) = f(x)$
approximate right hand side	$P_2^h(f) \in E_2^h$	$f_k = f(k \cdot h)$
differential/difference expression	$F(u) = u''$	$F_h(\vec{u}) = \frac{u_{k-1} - 2\,u_k + u_{k+1}}{h^2}$
approximate equation	$F_h(\vec{u}) = P_h^2(f)$	$\frac{u_{k-1} - 2\,u_k + u_{k+1}}{h^2} = f(k \cdot h)$
stability	$\|u_h\|_{E_1^h} \leq M \; \|F_h(u_h)\|_{E_2^h}$	$\max\{\|u_k\|\} \leq M \; \max\{\|f_k\|\}$
convergence, as $h \to 0$	$\|P_1^h(u) - \vec{u}\|_{E_1^h} \to 0$	$\max\{\|u(k \cdot h) - u_k\|\} \to 0$

Table 5.3: Exact and approximate boundary value problem

5–16 Result : *To verify convergence of the solution of the finite difference of approximation of equation (5.4) to the exact solution we have to assure that the scheme is consistent and stable.*

- ***Consistency**: According to table 5.2 (page 120) the scheme is consistent of order 2.*
- ***Stability**: Let $\vec{u}$ be the solution of the equation (5.5) with right hand side $\vec{f}$. Then*

$$\max_{1 \leq k \leq n} \{|u_k|\} \leq \frac{L^2}{2} \max_{1 \leq k \leq n} \{|f_k|\} \tag{5.6}$$

◇

Proof : The proof of stability of this finite difference scheme is based on a discrete maximum principle[2]. We proceed in two stages.

- The first claim a discrete maximum principle. If $f_k \geq 0$ for $k = 0, 1, 2, \ldots, n, (n+1)$ and

$$\frac{u_{k-1} - 2\,u_k + u_{k+1}}{h^2} = f_k = f(k \cdot h) \quad \text{for} \quad k = 1, 2, 3, \ldots, n$$

then

$$\max_{0 \leq k \leq n+1} \{u_k\} = \max\{u_0, u_{n+1}\}$$

To prove the statement we assume that $\max_{1 \leq k \leq n}\{|u_k|\} = u_i$ for some index $1 \leq i \leq n$. Then

$$\begin{aligned} u_{i-1} - 2\,u_i + u_{i+1} &= h^2\, f_i \\ u_i &= \frac{1}{2}\,(u_{i-1} + u_{i+1}) - h^2\, f_i \leq u_i - 0 \end{aligned}$$

Thus we find $u_{i-1} = u_i = u_{i+1}$ and $f_i = 0$. The process can be repeated with indices $i-1$ and $i+1$ to finally obtain the desired estimate. The computations also imply that $\vec{u} = \vec{0}$ is the only solution of the homogeneous problem, thus the matrix representing F_h is invertible.

- Use the vector $\vec{v} \in R^n$ defined by $v_k = \left(\frac{k}{h}\right)^2 = \left(\frac{k\,L}{n+1}\right)^2$. The vector corresponds to the discretization of the function $v(x) = x^2$. Verify that

$$\frac{v_{k-1} - 2\,v_k + v_{k+1}}{h^2} = 2 \quad \text{for} \quad k = 1, 2, 3, \ldots, n$$

[2]Readers familiar with partial differential equations will recognize the maximum principle and the construction of sub- and super–solutions to obtain à priori bounds.

Let $C = \max\{|f_k| \ : \ 1 \le k \le n\}$ and $f_k^+ = f_k + C \ge 0$. Then $\vec{w}^+ = \vec{u} + \frac{C}{2}\,\vec{v}$ is the solution of $F_h(\vec{w}^+) = \vec{f}^+$ and based on the first part of the proof and $u_0 = u_{n+1} = 0$ we find

$$\max_{1\le k\le n}\{u_k + \frac{C}{2}\,v_k\} \le \frac{C}{2}\ \max\{v_0, v_{k+1}\} = \frac{C}{2}\ L^2$$

A similar argument with $f_k^+ = f_k - C \le 0$ and $\vec{w}^- = \vec{u} - \frac{C}{2}\,\vec{v}$ implies

$$\min_{1\le k\le n}\{u_k - \frac{C}{2}\,v_k\} \ge -\frac{C}{2}\ L^2$$

These two inequalities imply

$$-\frac{C\ L^2}{2} \le u_k \le \frac{C\ L^2}{2} \quad \text{for} \quad k = 1, 2, 3, \ldots, n$$

and thus the stability estimate (5.6).

□

In this section we only introduced the basic keywords and illustrated them with one sample application. The above proof for stability of finite difference approximations to elliptic boundary value problems can be applied to two dimensional problems, e.g. [Smit84, p. 255]. Further information can be found in many books on numerical methods to solve PDE's are also in [IsaaKell66, §9.5], [Wlok82].

5.4 Parabolic problems, heat equation

A one dimensional heat equation is given by the partial differential equation

$$\begin{aligned} \tfrac{\partial}{\partial t}\,u(x,t) &= \kappa\ \tfrac{\partial^2}{\partial x^2}\,u(x,t) && \text{for} \quad 0 < x < L \quad \text{and} \quad t > 0 \\ u(0,t) = u(L,t) &= 0 && \text{for} \quad t > 0 \\ u(x,0) &= f(x) && \text{for} \quad 0 < x < L \end{aligned} \tag{5.7}$$

The maximum principle implies that for all $t \ge 0$ we find

$$\max\{|u(x,t)| \ : \ 0 \le x \le L\} \le \max\{|f(x)| \ : \ 0 \le x \le L\}$$

The finite difference scheme should satisfy this property too, leading to the stability condition.

The two dimensional domain $(x,t) \in [0,L] \times \mathbb{R}_+$ is discretized as illustrated in figure 5.5. For step sizes $\Delta x = \frac{L}{n+1}$ and Δt we set

$$u_{i,j} = u(i\cdot\Delta x\,,\ j\cdot\Delta t) \quad \text{for} \quad i = 0, 1, 2, \ldots, n, n+1 \quad \text{and} \quad j \ge 0$$

The boundary condition $u(0,t) = u(L,t) = 0$ implies $u_{0,j} = u_{n+1,j} = 0$ and the initial condition $u(x,0) = f(x)$ leads to $u_{i,0} = f(i\cdot\Delta x)$. The PDE (5.7) is replaced by a finite difference approximation on the grid shown in figure 5.5 and the result is examined.

5.4.1 A special matrix

The solution of the finite difference equation we be computed with the help of time steps, i.e. we use the values at one time level $t = j\cdot\Delta t$ and then compute the values at the next level $t + \Delta t = (j+1)\,\Delta t$. Thus we put all values at one time level $t = j\,\Delta t$ into a vector $\vec{u}_j$

$$\vec{u}_j = (u_{1,j}\ ,\ u_{2,j}\ ,\ u_{3,j}\ ,\ldots\ u_{n-1,j}\ ,\ u_{n,j})^T$$

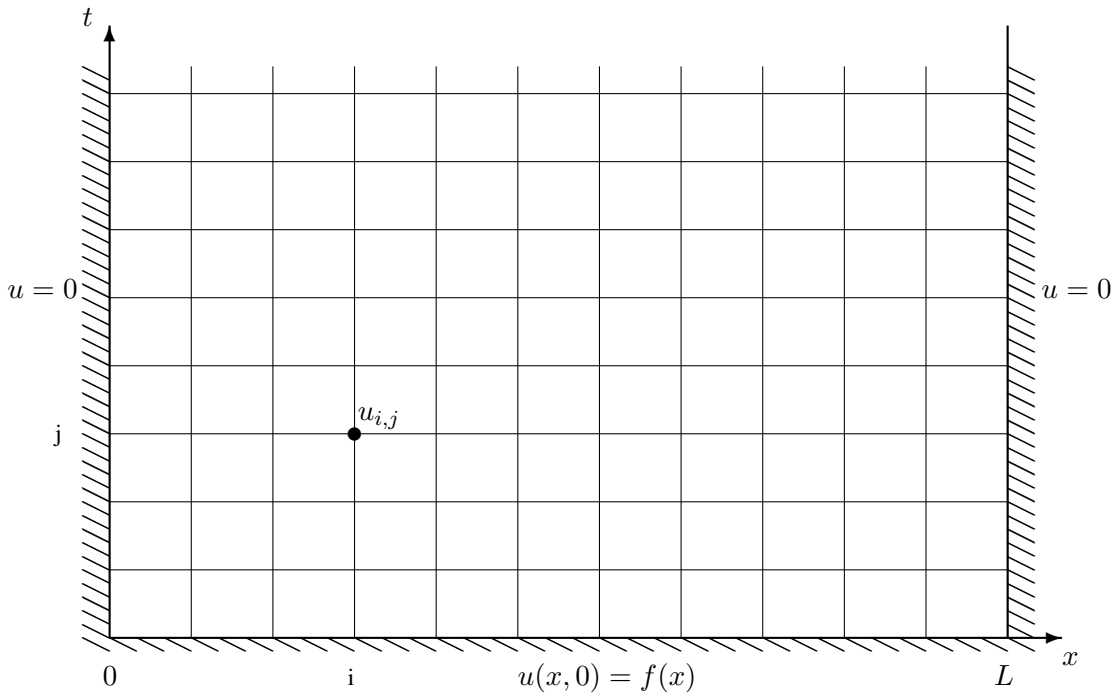


Figure 5.5: A finite difference grid for a heat equation

A good finite difference approximation to the second order space derivative is given by (see table 5.2)

$$\kappa \frac{\partial^2}{\partial x^2} u(x,t) = \kappa \frac{u(x-\Delta x) - 2\,u(x,t) + u(x+\Delta x,t)}{(\Delta x)^2} + O((\Delta x)^2) \tag{5.8}$$

Thus the values of the space derivatives at one time level can be written as

$$-\frac{\kappa}{(\Delta x)^2}\, \mathbf{A}_n \cdot \vec{u}_j$$

where the symmetric $n \times n$ matrix $\mathbf{A}_n$ is given by

$$\mathbf{A}_n = \begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}$$

To examine the different possible approximations to the equation (5.7) we need the exact eigenvalues and eigenvectors of $\mathbf{A}_n$ given by

$$\lambda_k = 2 + 2\,\cos\frac{k\,\pi}{n+1} = 4\,\sin^2\frac{k\,\pi}{2\,(n+1)}$$

and

$$\vec{v}_k = \left(\sin\frac{1\,k\,\pi}{n+1}, \sin\frac{2\,k\,\pi}{n+1}, \sin\frac{3\,k\,\pi}{n+1}, \ldots, \sin\frac{(n-1)\,k\,\pi}{n+1}, \sin\frac{n\,k\,\pi}{n+1}\right)^T$$

where $k = 1, 2, 3, \ldots, n$. Thus the eigenvectors are discretizations of the functions $\sin(k\,x)$ on the interval $[0\,,\,\pi]$. These functions have exactly k local extremas in the interval. The higher the value of k the more the eigenfunction will oscillate. For a proof of the above statements see [Smit84, p. 154].

Since the matrix $\mathbf{A}_n$ is symmetric the eigenvectors are orthogonal and form a basis. If the eigenvectors are normalized , then any vector $\vec{f}$ can be written as linear combination of normalized eigenvectors $\vec{v}_k$ of the matrix $\mathbf{A}_n$, i.e.

$$\vec{u} = \sum_{k=1}^{n} \alpha_k\ \vec{v}_k \quad \text{with} \quad \alpha_k = \langle \vec{u}, \vec{v}_k \rangle$$

For arbitrary $t \geq 0$ we may consider the vector $\vec{u}(t)$ of the discretized (in space) solution. The differential equation (5.8) reads as

$$\frac{d}{d\,t}\,\vec{u}(t) = -\frac{\kappa}{(\Delta x)^2}\ \mathbf{A}_n \cdot \vec{u}(t)$$

If the solution $\vec{u}(t)$ is written as linear combination of eigenvectors

$$\vec{u}(t) = \sum_{k=1}^{n} \alpha_k(t)\ \vec{v}_k$$

the above system of n linear equations is converted to n linear, first order differential equations

$$\frac{d}{d\,t}\,\alpha_k(t) = -\frac{\kappa}{(\Delta x)^2}\ \lambda_k\,\alpha_k(t) \quad \text{for} \quad k = 1, 2, 3, \ldots, n$$

The initial values for the coefficient functions are given by $\alpha_k(0) = \langle \vec{f}, \vec{v}_k \rangle$. For these equations we use the methods and results in section 5.2.

5.4.2 Explicit finite difference approximation to the heat equation

The time derivative in the PDE (5.7) can be approximated by a forward difference

$$\frac{\partial}{\partial t}\,u(x,t) = \frac{u(x, t+\Delta t) - u(t,x)}{\Delta t} + O(\Delta t)$$

This can be combined with the space derivatives in equation (5.8) to obtain the scheme illustrated in figure 5.6. The results in table 5.2 imply that the scheme is consistent with the error of the order $O(\Delta t) + O((\Delta x)^2)$.

$$\frac{u_{i,j+1} - u_{i,j}}{\Delta t} = \kappa\ \frac{u_{i-1,j} - 2\,u_{i,j} + u_{i+1,j}}{(\Delta x)^2}$$

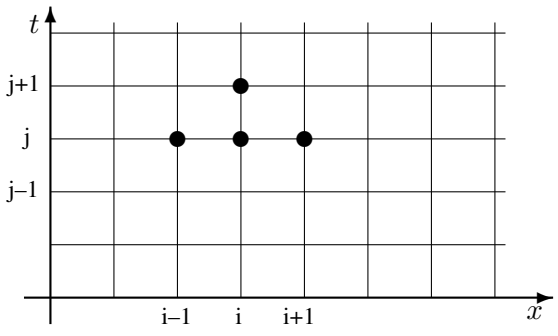


Figure 5.6: Explicit finite difference approximation

Using a matrix notation the finite difference equation can be written as

$$\vec{u}_{j+1} = \vec{u}_j - \frac{\kappa\,\Delta t}{(\Delta x)^2}\ \mathbf{A}_n \cdot \vec{u}_j = (\mathbb{I}_n - r\,\mathbf{A}_n) \cdot \vec{u}_j$$

with $r = \frac{\kappa\,\Delta t}{(\Delta x)^2}$. If the vector $\vec{u}_j$ is known the values at the next time level $\vec{u}_{j+1}$ can be computed without solving a system of linear equations, thus this is called an **explicit** method. Starting with the discretization of the initial values $\vec{u}_0 = \vec{f}$ and applying the above formula repeatedly we find the solution

$$\vec{u}_j = (\mathbb{I}_n - r\,\mathbf{A}_n)^j \cdot \vec{f}$$

The goal is to examine the stability of this finite difference scheme. Since for eigenvalues λ_k and eigenvectors $\vec{v}_k$ we have

$$(\mathbb{I}_n - r\,\mathbf{A}_n)^j \cdot \vec{v}_k = (1 - r\,\lambda_k)^j \cdot \vec{v}_k$$

and the solution will remain bounded as $j \to \infty$ only if $r\,\lambda_k < 2$ for all $k = 1, 2, 3, \ldots, n$. This corresponds to the stability condition.

Since we want to use the results of section 5.2.2 on solutions of the ordinary differential equation we translate to the coefficient functions $\alpha_k(t)$ and find

$$\frac{d}{d\,t}\,\alpha_k(t) = -\frac{\kappa}{(\Delta x)^2}\,\lambda_k\,\alpha_k(t) \quad \text{approximated by} \quad \frac{\alpha_k(t+h) - \alpha_k(t)}{h} = -\frac{\kappa\,\lambda_k}{(\Delta x)^2}\,\alpha_k(t)$$

and thus

$$\begin{aligned} \alpha_k(t+h) &= \left(1 - \frac{h\,\kappa}{(\Delta x)^2}\,\lambda_k\right)\,\alpha_k(t) \\ \alpha_k(j\cdot h) &= \left(1 - \frac{h\,\kappa}{(\Delta x)^2}\,\lambda_k\right)^j\,\alpha_k(0) \end{aligned}$$

The scheme is stable if the absolute value of the bracketed expression is smaller than 1, i.e.

$$\frac{h\,\kappa}{(\Delta x)^2}\,\lambda_k < 2$$

Since the largest eigenvalue of $\mathbf{A}_n$ is $\lambda_n = 4\,\sin^2\frac{n\,\pi}{2\,(n+1)} \approx 4\,\sin^2\frac{\pi}{2} = 4$ we find the stability condition

$$r = \frac{\kappa\,\Delta t}{(\Delta x)^2} < \frac{1}{2} \qquad \Longleftrightarrow \qquad \Delta t < \frac{1}{2\,\kappa}\,(\Delta x)^2$$

Thus we have **conditional stability**. The restriction on the size of the timestep Δt is severe, since for small Δx the Δt will need to be much smaller.

In figure 5.7 a solution of

$$\begin{aligned} \tfrac{\partial}{\partial t}\,u(x,t) &= \kappa\,\tfrac{\partial^2}{\partial x^2}\,u(x,t) && \text{for} \quad 0 < x < 1 \quad \text{and} \quad t > 0 \\ u(0,t) = u(1,t) &= 0 && \text{for} \quad t > 0 \\ u(x,0) &= \begin{cases} 2\,x & \text{for} \quad 0 < x \le 0.5 \\ 2 - 2\,x & \text{for} \quad 0.5 \le x < 1 \end{cases} \end{aligned}$$

is shown for values of r slightly smaller or larger than the critical value of $r = 0.5$. Since the largest eigenvalue of $\mathbf{A}_n$ will be the first to exhibit instability we examine the corresponding eigenvector

$$\vec{v}_n = \left(\sin\frac{1\,n\,\pi}{n+1}\,,\,\sin\frac{2\,n\,\pi}{n+1}\,,\,\sin\frac{3\,n\,\pi}{n+1}\,,\,\ldots\,,\,\sin\frac{(n-1)\,n\,\pi}{n+1}\,,\,\sin\frac{n\,k\,\pi}{n+1}\right)^T$$

the corresponding eigenfunction has n extrema in the interval. Thus the instability should exhibit n extrema, this is confirmed by figure 5.7 where the calculation is done with $n = 9$, as shown in the *Octave*–code below. The deviation from the correct solution exhibits 9 local extrema in the interval.

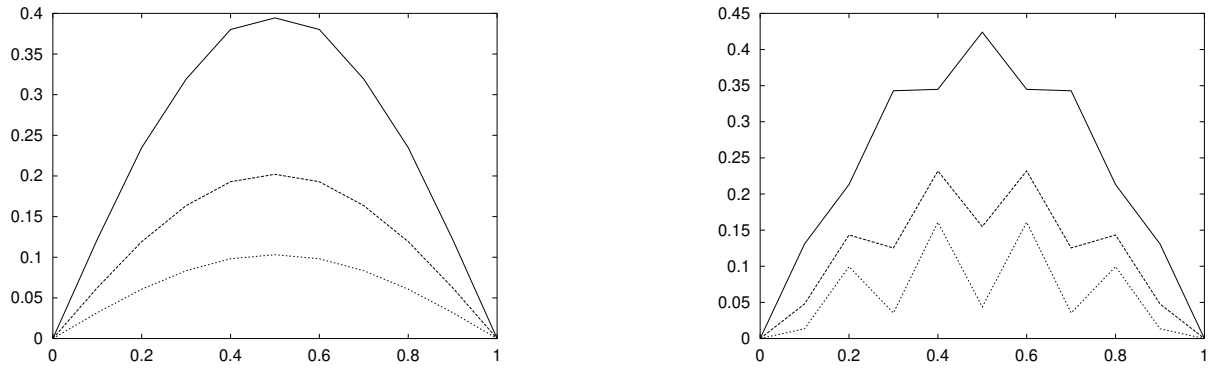


Figure 5.7: Solution of 1-d heat equation with explicit scheme with $r = 0.48$ and $r = 0.52$

Octave

```
global L=1;     # length of the space interval
n=9;            # number of interior grid points
r=0.48;         # ratio to compute time step
T=0.5;          # final time

function r=iv(x);
  global L;
  r=min([ 2*x/L,2-2*x/L]')';
endfunction

dx=L/(n+1);
dt= r*dx^2;
x=linspace(0,L,n+2)';

y=iv(x);
ynew=y;
gset nokey
gset yrange [0:1]
for t=0:dt:T+dt;
  for k=2:n+1
    ynew(k)=(1-2*r)*y(k)+r*(y(k-1)+y(k+1));
  endfor
  y=ynew;
  plot(x,y)
  sleep(0.5);
endfor
```

In the above code we verify that for each time step approximately $2 \cdot n$ multiplications/additions are necessary. Thus the computational cost of one time step is $2\,n$.

5.4.3 Implicit finite difference approximation to the heat equation

The time derivative in the PDE (5.7) can be approximated by a backward difference

$$\frac{\partial}{\partial t}\, u(x,t) = \frac{u(x,t) - u(t-\Delta t, x)}{\Delta t} + O(\Delta t)$$

This will lead to the finite difference scheme shown in figure 5.8. The results in table 5.2 again imply that the scheme is consistent with the error of the order $O(\Delta t) + O((\Delta x)^2)$.

$$\frac{u_{i,j+1} - u_{i,j}}{\Delta t} = \kappa \, \frac{u_{i-1,j+1} - 2\,u_{i,j+1} + u_{i+1,j+1}}{(\Delta x)^2}$$

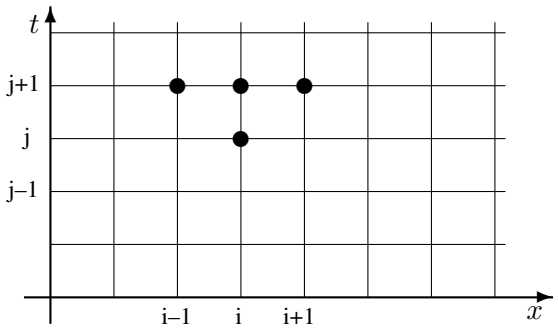


Figure 5.8: Implicit finite difference approximation

Using the matrix notation again we find

$$\vec{u}_{j+1} - \vec{u}_j = -\frac{\kappa\,\Delta t}{(\Delta x)^2}\,\mathbf{A}_n \cdot \vec{u}_j$$

or

$$(\mathbb{I}_n + r\,\mathbf{A}_n) \cdot \vec{u}_{j+1} = \vec{u}_j$$

with $r = \frac{\kappa\,\Delta t}{(\Delta x)^2}$. If the values $\vec{u}_j$ at a given time are known we have to solve a system of linear equations to determine the values $\vec{u}_{j+1}$ at the next time level. We have an **implicit** method. As in the previous section we can use the eigenvalues and vectors of $\mathbf{A}_n$ to examine stability of the scheme. We are lead to the iteration scheme

$$\vec{u}_j = (\mathbb{I}_n + r\,\mathbf{A}_n)^{-j} \cdot \vec{f}$$

and thus

$$(\mathbb{I}_n + r\,\mathbf{A}_n)^{-j} \cdot \vec{v}_k = \left(\frac{1}{1 + r\,\lambda_k}\right)^j \cdot \vec{v}_k$$

Since $\lambda_k > 0$ we find that this scheme is **unconditionally stable**, i.e. there are no restrictions on the step sizes Δx and Δt. This is confirmed by the results in figure 5.9. It was generated by code similar to the one below.

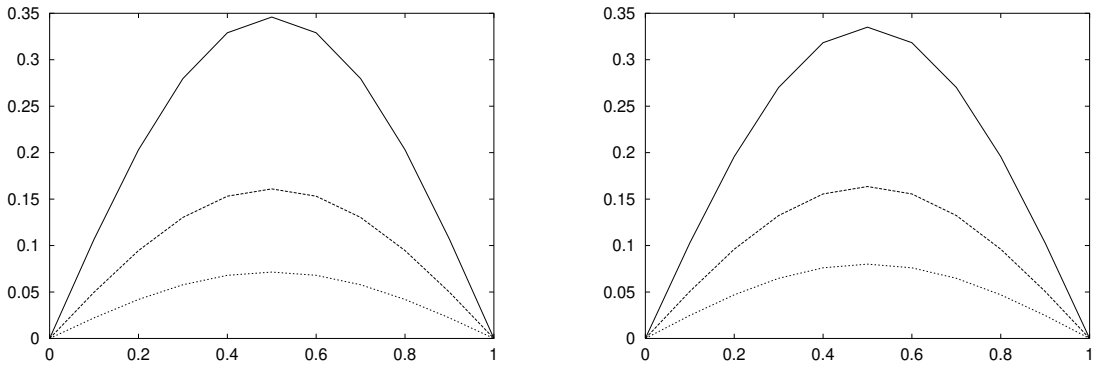


Figure 5.9: Solution of 1-d heat equation with implicit scheme with $r = 0.5$ and $r = 2.0$

Octave

```
global L=1;    # length of the space interval
n=9;           # number of interior grid points
```

```
r=2.0;        # ratio to compute time step
T=0.5;        # final time
plots=3;      # number of plots to be saved

function r=iv(x);
  global L;
  r=min([ 2*x/L,2-2*x/L]')';
endfunction

dx=L/(n+1);
dt=2*r*dx^2;
x=linspace(0,L,n+2)';
initval=iv(x(2:n+1));

yplot=zeros(plots,n+2);
plotc=1;

Adiag=ones(1,n)*(1+2*r);
Aoffdiag =-ones(1,n-1)*r;

y=initval;
for t=0:dt:T+dt;
  y=trisolve(Adiag,Aoffdiag,y);
  if min(abs(tplot-t))<dt/2
    yplot(plotc,2:n+1)=y';
    plotc++;
  endif
endfor
plot(x,yplot)
```

To perform one time-step one has to solve a system of n linear equations where the matrix is symmetric, tridiagonal and positive definite. There are excellent algorithms for this type of problem (e.g. [GoluVanLoan96]), requiring only $5\,n$ multiplications. If the matrix decomposition and the back-substitution are separately coded this can even be reduce to an operation count for one time-step of only $2\,n$ multiplication.

5.4.4 Crank–Nicolson approximation to the heat equation

When using a centered difference approximation

$$\frac{\partial}{\partial t}\,u(x,t) = \frac{u(x,t+\Delta t/2) - u(t-\Delta t/2,x)}{\Delta t} + O((\Delta t)^2)$$

at the midpoint between time levels we are lead to the scheme in figure 5.10. The results in table 5.2 again imply that the scheme is consistent with the error of the order $O((\Delta t)^2) + O((\Delta x)^2)$. Thus we gained one order of convergence in time.

The matrix notation leads to

$$\vec{u}_{j+1} - \vec{u}_j = -\frac{\kappa\ \Delta t}{2\ (\Delta x)^2}\ (\mathbf{A}_n \cdot \vec{u}_{j+1} + \mathbf{A}_n \cdot \vec{u}_j)$$

or

$$\left(\mathbb{I}_n + \frac{r}{2}\,\mathbf{A}_n\right) \cdot \vec{u}_{j+1} = \left(\mathbb{I}_n - \frac{r}{2}\,\mathbf{A}_n\right) \cdot \vec{u}_j$$

again with $r = \frac{\kappa\,\Delta t}{(\Delta x)^2}$. If the values $\vec{u}_j$ at a given time are known we have to multiply the vector with a matrix and then solve a system of linear equations to determine the values $\vec{u}_{j+1}$ at the next time level. We have

$$\frac{u_{i,j+1} - u_{i,j}}{\kappa\, \Delta t} = \frac{u_{i-1,j} - 2\, u_{i,j} + u_{i+1,j}}{2\, (\Delta x)^2} + \frac{u_{i-1,j+1} - 2\, u_{i,j+1} + u_{i+1,j+1}}{2\, (\Delta x)^2}$$

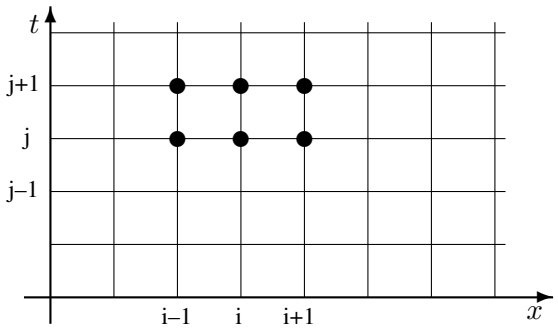


Figure 5.10: Crank–Nicolson finite difference approximation

an **implicit** method. As in the previous section we can use the eigenvalues and vectors of $\mathbf{A}_n$ to examine stability of the scheme. We are lead to examine the inequality

$$\left(\frac{2 - r\,\lambda_k}{2 + r\,\lambda_k}\right)^j < 1$$

Since $\lambda_k > 0$ we find that this scheme is also **unconditionally stable**.

5.4.5 General parabolic problems

In table 5.4 find a comparison of the three different finite difference approximations to equation (5.7). As a consequence one should use either an implicit method or Crank–Nicolson for this type of problem.

method	order of consistency	stability condition	operation count
explicit	$O(\Delta t) + O((\Delta x)^2)$	$\Delta t < \frac{1}{2\,\kappa}\,(\Delta x)^2$	$2\,n$
implicit	$O(\Delta t) + O((\Delta x)^2)$	unconditional	$2\,n$
Crank–Nicolson	$O((\Delta t)^2) + O((\Delta x)^2)$	unconditional	$4\,n$
advantage	Crank–Nicolson	implicit and C–N	implicit and explicit

Table 5.4: Comparison of finite difference schemes for the heat equation

In the previous section we considered only a special case of the space discretization operator $\mathbf{A} = \frac{\kappa}{(\Delta x)^2}\,\mathbf{A}_n$. A more general situation may be described by the equation

$$\frac{d}{dt}\,\vec{u}(t) = -\mathbf{A}\cdot\vec{u}(t)$$

where the symmetric, positive definite matrix $\mathbf{A}$ has eigenvalues $0 \le \lambda_1 \le \lambda_2 \le \lambda_3 \le \ldots \le \lambda_n$. When using either Crank–Nicolson or the fully implicit method the resulting finite difference scheme will be unconditionally stable. The explicit method leads to

$$\vec{u}(t + \Delta t) = \vec{u}(t) - \Delta t\ \mathbf{A}\cdot u(t) = (\mathbb{I} - \Delta t\ \mathbf{A})\cdot u(t)$$

and thus to the stability condition

$$\Delta t\cdot\lambda_n < 2 \qquad \Longleftrightarrow \qquad \Delta t < \frac{2}{\lambda_n}$$

This condition remains valid, also for problems with more than one space dimension. To use the explicit method for these type of problems one needs to estimate the largest eigenvalue of the space discretization. Estimates of this type can be given, based on the condition number of the discretization matrix, e.g. [KnabAnge00, Satz 3.45]. For higher space dimensions the effort to solve one linear system equations for the implicit methods will increase drastically, as the resulting matrices will not be tridiagonal, but only have a band structure. Nonetheless this structure can be used in efficient implementations. An introduction is given in chapter 11.

For many dynamic problems the mass matrix $\mathbf{M}$ has to be taken into account too. Consider a discretized systems of the form

$$\frac{d}{dt}\,\mathbf{M}\cdot\vec{u}(t) = -\mathbf{A}\cdot\vec{u}(t)$$

Often linear systems of equations with the matrix $\mathbf{M}$ are easily solved, e.g. $\mathbf{M}$ might be a diagonal matrix. The generalized eigenvalues and eigenvectors are nontrivial solutions of

$$\mathbf{A}\cdot\vec{v} = \lambda\,\mathbf{M}\cdot\vec{v}$$

The explicit discretization scheme leads to

$$\begin{aligned}\frac{1}{\Delta t}\,\mathbf{M}\;\left(\vec{u}(t+\Delta t)-\vec{u}(t)\right) &= -\mathbf{A}\cdot\vec{u}(t)\\ \vec{u}(t+\Delta t) &= \vec{u}(t)-\Delta t\,\mathbf{M}^{-1}\,\mathbf{A}\cdot\vec{u}(t) = \left(\mathbb{I}-\Delta t\,\mathbf{M}^{-1}\,\mathbf{A}\right)\cdot\vec{u}(t)\end{aligned}$$

Thus the stability condition is again $\Delta t < 2/\lambda_n$.

The fully implicit scheme will lead to

$$\begin{aligned}\frac{1}{\Delta t}\,\mathbf{M}\;\left(\vec{u}(t+\Delta t)-\vec{u}(t)\right) &= -\mathbf{A}\cdot\vec{u}(t+\Delta t)\\ \left(\mathbf{M}+\Delta t\;\mathbf{A}\right)\cdot\vec{u}(t+\Delta t) &= \mathbf{M}\cdot\vec{u}(t)\end{aligned}$$

and is unconditionally stable.

5.5 Hyperbolic problems, wave equation

The simplest form of a wave equation is

$$\begin{aligned}\tfrac{\partial^2}{\partial t^2}\,u(x,t) &= \kappa^2\,\tfrac{\partial^2}{\partial x^2}\,u(x,t) && \text{for}\quad 0<x<L \quad\text{and}\quad t>0\\ u(0,t)=u(L,t) &= 0 && \text{for}\quad t>0\\ u(x,0) &= u_0(x) && \text{for}\quad 0<x<L\\ \dot{u}(x,0) &= u_1(x) && \text{for}\quad 0<x<L\end{aligned} \tag{5.9}$$

Again we examine an explicit and an implicit approximation.

5.5.1 Explicit approximation

A finite difference approximation we be examined on a grid given by figure 5.5 . This scheme is consistent of order $(\Delta x)^2+(\Delta t)^2$.

$$\vec{u}_{j+1}-2\,\vec{u}_j+\vec{u}_{j-1} = -\kappa^2\,\frac{(\Delta t)^2}{(\Delta x)^2}\,\mathbf{A}_n\cdot\vec{u}_j$$

Since we have time derivatives of order 2 we first have to examine the ordinary differential equation

$$\ddot{y}(t) = -\lambda\,y(t) \quad \text{approximated by} \quad \frac{y(t+h)-2\,y(t)+y(t-h)}{h^2} = -\lambda\,y(h)$$

$$\frac{u_{i,j+1} - 2\,u_{i,j} + u_{i,j-1}}{(\Delta t)^2} = \kappa^2\,\frac{u_{i-1,j} - 2\,u_{i,j} + u_{i+1,j}}{(\Delta x)^2}$$

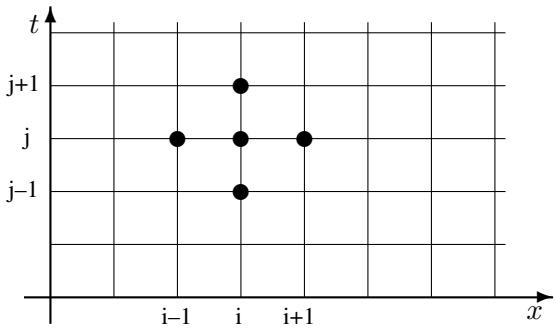


Figure 5.11: Explicit finite difference approximation for the wave equation

and thus

$$y(t+h) = 2\,y(t) - y(t-h) - h^2\,\lambda\,y(t)$$

Using a matrix we find

$$\begin{pmatrix} y(t) \\ y(t+h) \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 2-\lambda\,h^2 \end{bmatrix} \cdot \begin{pmatrix} y(t-h) \\ y(t) \end{pmatrix}$$

With an iteration we find

$$\begin{pmatrix} y(j\,h) \\ y((j+1)\,h) \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 2-\lambda\,h^2 \end{bmatrix}^j \cdot \begin{pmatrix} y(0) \\ y(h) \end{pmatrix}$$

These solutions remain bounded as $j \to \infty$ if the eigenvalues μ of the matrix have absolute values smaller than 1 . Thus we examine the solutions of the characteristic equation

$$\mu^2 - \mu\,(2 - \lambda\,h^2) + 1 = 0$$

Thus $\mu_1 \cdot \mu_2 = 1$ and for $|\mu_{1,2}| \leq 1$ to be correct we need conjugate complex values. The solutions are given by

$$\begin{aligned} \mu_{1,2} &= \frac{1}{2}\,\left(2 - \lambda\,h^2 \pm \sqrt{(2-\lambda\,h^2)^2 - 4}\right) \\ &= \frac{1}{2}\,\left(2 - \lambda\,h^2 \pm \sqrt{\lambda^2\,h^4 - 4\lambda\,h^2}\right) \\ &= \frac{2-\lambda\,h^2}{2} \pm \sqrt{\lambda\,h^2}\,\sqrt{\lambda\,h^2 - 4} \end{aligned}$$

Thus a necessary and sufficient condition for stability is

$$\lambda\,h^2 \leq 4 \qquad \Longleftrightarrow \qquad h^2 \leq \frac{4}{\lambda}$$

Now we return to the wave equation. With the notation from the previous section we can write the discretization scheme in figure 5.11 in the form

$$\vec{u}_{j+1} - 2\,\vec{u}_j + \vec{u}_{j-1} = -\kappa^2\,\frac{(\Delta t)^2}{(\Delta x)^2}\,\mathbf{A}_n \cdot \vec{u}_j$$

or when solved for $\vec{u}_{j+1}$

$$\vec{u}_{j+1} = (2\,\mathbb{I}_n - r\,\mathbf{A}_n)\cdot\vec{u}_j - \vec{u}_{j-1}$$

with $r = \kappa^2\,\frac{(\Delta t)^2}{(\Delta x)^2}$. With a block matrix notation this can be transformed in a form similar to the ODE situation above.

$$\begin{pmatrix} \vec{u}_j \\ \vec{u}_{j+1} \end{pmatrix} = \begin{bmatrix} 0 & \mathbb{I}_n \\ -\mathbb{I}_n & 2\,\mathbb{I}_n - r\,\mathbf{A}_n \end{bmatrix} \cdot \begin{pmatrix} \vec{u}_{j-1} \\ \vec{u}_j \end{pmatrix}$$

The stability condition for the ODE leads to $r\,\lambda_k \le 4$ for $k = 1, 2, 3, \ldots, n$. Since the largest eigenvalue is given by $\lambda_n = 4\,\sin^2\frac{n\,\pi}{2\,(n+1)} \approx 4\,\sin^2\pi/2 = 4$ we find the stability condition

$$r = \kappa^2\,\frac{(\Delta t)^2}{(\Delta x)^2} \le 1 \qquad \Longleftrightarrow \qquad \kappa^2\,(\Delta t)^2 \le (\Delta x)^2$$

Since the solution at two time levels has to be known to get the finite difference scheme started we have to use the initial conditions to construct the vectors u_0 and u_1. The first initial condition in equation (5.9) obviously implies that $\vec{u}_0$ should be the discretization of $u(x,0) = u_0(x)$. As $\vec{u}_1$ one can use the discretization of $u_0(x) + h\,u_1(x)$. The *Octave*–code below is an elementary implementation of the presented finite difference scheme.

Octave

```
global L=3;   # length of the space interval
n=150;        # number of interior grid points
r=0.8;        # ratio to compute time step
T=5;          # final time

function r=iv(x);  # initial displacement
  global L;
  r=min([2*x,2-2*x]')';  r=max([r,0*x]')';
endfunction

dx=L/(n+1);
dt=sqrt(r)*dx;
x=linspace(0,L,n+2)';

y0=iv(x);  y0(1)=0; y0(n+2)=0;
y1=y0;   # use zero initial speed
y2=y0;
for t=0:dt:T+dt;
  plot(x,y0)
  for k=2:n+1
    y2(k)=(2-2*r)*y1(k)+r*(y1(k-1)+y1(k+1))-y0(k);
  endfor
  y0=y1; y1=y2;
endfor
```

5.5.2 Implicit approximation

Since the explicit method is again conditionally stable we consider also an implicit method, which turns out to be unconditionally stable. The space discretization at time level j in the previous section is replaced by a weighted average of discretizations at levels $j-1$, j and $j+1$.

$$\frac{\vec{u}_{j+1} - 2\,\vec{u}_j + \vec{u}_{j-1}}{(\Delta t)^2} = -\frac{\kappa^2}{4\,(\Delta x)^2}\;(\mathbf{A}_n\cdot\vec{u}_{j+1} + 2\,\mathbf{A}_n\cdot\vec{u}_j + \mathbf{A}_n\cdot\vec{u}_{j-1})$$

The difference scheme is consistent of order $(\Delta x)^2 + (\Delta t)^2$. This leads to a linear system of equations for $\vec{u}_{j+1}$.

$$\left(1 + \kappa^2 \, \frac{(\Delta t)^2}{4\,(\Delta x)^2} \, \mathbf{A}_n\right) \, \vec{u}_{j+1} = \left(2 - 2\,\kappa^2 \, \frac{(\Delta t)^2}{4\,(\Delta x)^2} \, \mathbf{A}_n\right) \, \vec{u}_j - \left(1 + \kappa^2 \, \frac{(\Delta t)^2}{4\,(\Delta x)^2} \, \mathbf{A}_n\right) \, \vec{u}_{j-1}$$

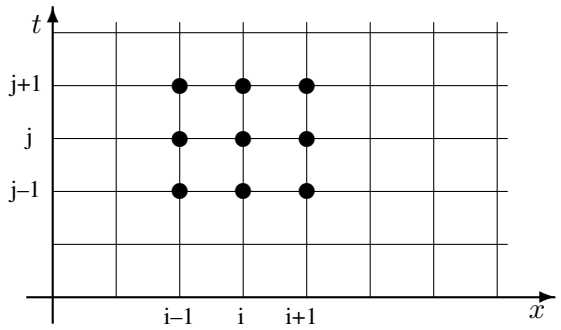


Figure 5.12: Implicit finite difference approximation for the wave equation

By considering eigenvalues and vectors we use

$$c = \kappa^2 \, \lambda \, \frac{(\Delta t)^2}{4\,(\Delta x)^2} \geq 0$$

and are lead to examine the time discretization

$$\begin{aligned} y(t+\Delta t) - 2\,y(t) + y(t-\Delta t) &= -\kappa^2 \, \lambda \, \frac{(\Delta t)^2}{4\,(\Delta x)^2} \, (y(t+\Delta t) + 2\,y(t) + y(t-\Delta t)) \\ (1+c) \;\; y(t+\Delta t) &= (2-2\,c) \;\; y(t) - (1+c) \;\; y(t-\Delta t) \\ \begin{pmatrix} y(t) \\ y(t+\Delta t) \end{pmatrix} &= \begin{bmatrix} 0 & 1 \\ -1 & \frac{2-2\,c}{1+c} \end{bmatrix} \begin{pmatrix} y(t-\Delta t) \\ y(t) \end{pmatrix} \end{aligned}$$

Examine the eigenvalues of this matrix

$$\det \begin{bmatrix} -\mu & 1 \\ -1 & 2\,\frac{1-c}{1+c} - \mu \end{bmatrix} = \mu^2 - \frac{2-2\,c}{1+c}\,\mu + 1 = 0$$

and observe that $\mu_1 \cdot \mu_2 = 1$ and

$$4 \, \left(\frac{1-c}{1+c}\right)^2 - 4 \leq 0$$

Thus the two values are complex conjugate. This imples $|\mu_1| = |\mu_2| = 1$ and the scheme is unconditionally stable.

5.5.3 General wave type problems

A more general form of a wave type, dynamic boundary value problem may be given in the form

$$\frac{d}{dt^2} \, \mathbf{M} \cdot \vec{u}(t) = -\mathbf{A} \cdot \vec{u}(t)$$

and the corresponding explicit scheme is

$$\mathbf{M} \cdot (\vec{u}_{j+1} - 2\,\vec{u}_j + \vec{u}_{j-1}) = -(\Delta t)^2 \, \mathbf{A} \cdot \vec{u}_j$$

or

$$\mathbf{M} \cdot \vec{u}_{j+1} = \left(2\,\mathbf{M} - (\Delta t)^2\,\mathbf{A}\right) \cdot \vec{u}_j - \mathbf{M} \cdot \vec{u}_{j-1}$$

The scheme will be stable if

$$(\Delta t)^2 < 4/\lambda_n$$

where λ_n is the largest of the generalized eigenvalues, i.e. nonzero solutions of

$$\mathbf{A} \cdot \vec{v} = \lambda\,\mathbf{M} \cdot \vec{v}$$

5.6 Comments and bibliography

- In [Ciar02] a very precise presentation of convergence of finite element methods for elliptic problems is given.
- The text book [KnabAnge00, §3] gives a brief recapitulation of the results in the book by Ph. Ciarlet.
- A general presentation of the concepts **stability** and **consistency** was given by H. B. Keller in [Kell92, Appendix C.2] in a reprint of an other article [Kell75]. It is also applicable to nonlinear problems.

Chapter 6

Calculus of variations for functions for multiple variables

6.1 An electrostatic example

The energy density ρ in an electric field $\vec{E}$ is given by

$$\rho = \|\vec{E}\|^2 = E_1^2 + E_2^2 + E_3^2$$

The relation between the electric field $\vec{E}$ and the potential Φ is

$$\vec{E} = -\operatorname{grad} V = -\vec{\nabla}\Phi$$

Thus the total energy in a domain Ω is given by

$$\text{Energy } = \iiint_\Omega \|\vec{E}\|^2 \, dV = \iiint_\Omega \left(\frac{\partial\,\Phi}{\partial x}\right)^2 + \left(\frac{\partial\,\Phi}{\partial y}\right)^2 + \left(\frac{\partial\,\Phi}{\partial z}\right)^2 \, dV$$

Now consider the situation where $\Omega \subset \mathbb{R}^3$ is very long in the z direction with a constant cross section parallel to the xy–plane. Then we can restrict the calculations to a domain in $\mathbb{R}^2$, again called Ω. Use the notation

$$\Phi_x = \frac{\partial \Phi}{\partial x} \quad \text{and} \quad \Phi_y = \frac{\partial \Phi}{\partial y}$$

then the above is given by a two dimensional integral.

$$I\left(\Phi_x, \Phi_y\right) = \iint_\Omega \Phi_x^2 + \Phi_y^2 \, dx\, dy$$

It is a fundamental physical law that the electric fields will be such that this energy will be minimized, if there are no electric charges. Thus we are lead to a problem of the type

$$\text{Minimise} \quad F\left(u_x, u_y\right) = \iint_\Omega f\left(u_x, u_y\right) \, dx\, dy \quad \text{where} \quad f(u_x, u_y) = u_x^2 + u_y^2$$

where f is a given function and the function $u = u(x, y)$ might be the unknown minimizer of the functional. In the next section we will find that the Euler Lagrange equation for this type of problem is

$$\frac{\partial}{\partial x}\frac{\partial\, f}{\partial u_x} + \frac{\partial}{\partial x}\frac{\partial\, f}{\partial u_x} = 0$$

In the example of the electric field above we find the Euler Lagrange equation

$$\begin{aligned} f(\Phi_x, \Phi_y) &= \Phi_x^2 + \Phi_y^2 \\ 2\,\frac{\partial\,\Phi_x}{\partial x} + 2\,\frac{\partial\,\Phi_y}{\partial y} &= 0 \\ \Phi_{xx} + \Phi_{yy} = \Delta\Phi &= 0 \end{aligned}$$

Thus the electric potential Φ has to satisfy the second order differential equation $\Delta\Phi = 0$ and appropriate boundary conditions.

In this slightly simpler example we can also argue directly by using calculus for multiple variables. We choose an arbitrary perturbation function η and a real parameter ε and define

$$F(\varepsilon) = \iint\limits_{\Omega} (\Phi_x + \varepsilon\,\eta_x)^2 + (\Phi_y + \varepsilon\,\eta_y)^2\; dx\,dy$$

Since Φ is the function minimizing the energy the function F has a minimum at $\varepsilon = 0$ and thus the derivative should be equal to zero. This leads to the necessary condition

$$\begin{aligned} \frac{\partial}{\partial\varepsilon}\,F(\varepsilon) &= 2 \iint\limits_{\Omega} (\Phi_x + \varepsilon\,\eta_x)\;\eta_x + (\Phi_y + \varepsilon\,\eta_y)\,\eta_y\; dx\,dy \\ \frac{\partial}{\partial\varepsilon}\,F(\varepsilon)\Big|_{\varepsilon=0} &= 2 \iint\limits_{\Omega} \Phi_x\,\eta_x + \Phi_y\,\eta_y\; dx\,dy \\ &= 2 \iint\limits_{\Omega} \begin{pmatrix} \Phi_x \\ \Phi_y \end{pmatrix} \cdot \begin{pmatrix} \eta_x \\ \eta_y \end{pmatrix}\; dx\,dy \;=\; 0 \end{aligned}$$

If Φ is to be the optimal solution (with minimal energy), then the above double integral has to vanish for 'all' test functions η. Thus Green's theorem (see appendix A.3) and the product rule

$$\begin{aligned} \operatorname{div}\left(\eta \begin{pmatrix} \Phi_x \\ \Phi_y \end{pmatrix}\right) &= \frac{d\,(\eta\,\Phi_x)}{dx} + \frac{d\,(\eta\,\Phi_y)}{dy} = \eta\,\frac{d\,\Phi_x}{dx} + \Phi_x\,\frac{d\,\eta}{dx} + \eta\,\frac{d\,\Phi_y}{dy} + \Phi_y\,\frac{d\,\eta}{dy} \\ &= \eta\;(\Phi_{xx} + \Phi_{yy}) + \begin{pmatrix} \eta_x \\ \eta_y \end{pmatrix} \cdot \begin{pmatrix} \Phi_x \\ \Phi_y \end{pmatrix} \end{aligned}$$

imply

$$\iint\limits_{\Omega} \begin{pmatrix} \Phi_x \\ \Phi_y \end{pmatrix} \cdot \begin{pmatrix} \eta_x \\ \eta_y \end{pmatrix}\; dx\,dy = \iint\limits_{\Omega} \eta\;(\Phi_{xx} + \Phi_{yy})\; dx\,dy + \oint\limits_{\partial\Omega} \eta\,\vec{n} \cdot \begin{pmatrix} \Phi_x \\ \Phi_y \end{pmatrix}\; ds = 0$$

Since this expression has to vanish for arbitrary functions η we conclude that the **Laplace equation** below has to be satisfied.

$$\begin{aligned} \Phi_{xx} + \Phi_{yy} &= 0 \quad \text{in} \quad \Omega \\ \vec{n} \cdot \begin{pmatrix} \Phi_x \\ \Phi_y \end{pmatrix} &= 0 \quad \text{on} \quad \Gamma \end{aligned}$$

This has to be supplemented with the condition for prescribed values of the electric potential on parts of the boundary.

6.2 Minimization of a functional of two variables

For a 'nice' domain $\Omega \subset \mathbb{R}^2$ and a known function f we try to find a function $u(x,y)$, such that the functional

$$F(u) = \iint\limits_{\Omega} f(x,y,u,u_x,u_y)\, dA$$

is minimized. We use the notation

$$\nabla u = \begin{pmatrix} u_x \\ u_y \end{pmatrix} = \begin{pmatrix} \frac{d\,u}{dx} \\ \frac{d\,u}{dy} \end{pmatrix}$$

We assume that the boundary $\partial\Omega = \Gamma_1 \cup \Gamma_2$ of the domain consists of two disjoint parts. On Γ_1 the value of the function $u(x,y)$ are given by a known function $g(x,y)$, while on Γ_2 we are free to choose u. The outer unit normal vector is denoted by $\vec{n}$. We reuse ideas and techniques from chapter 3.

We assume that the function $u = u(x,y)$ is the optimal solution and consider perturbation $w(x,y) = u(x,y) = \varepsilon \cdot \eta(x,y)$ for 'arbitrary' function η and a small parameter $\varepsilon \in \mathbb{R}$. Since u is a mini-miser of the functional F we have the necessary condition

$$\frac{\partial}{\partial \varepsilon} F(u + \varepsilon \cdot \eta)\Big|_{\varepsilon=0} = 0 \quad \text{for 'all' functions } \eta$$

Using a linear approximation[1] we find

$$f(x,y,u+\varepsilon\eta,u_x+\varepsilon\eta_x,u_y+\varepsilon\eta_y) \approx f(x,y,u,u_x,u_y) + \varepsilon\,\frac{\partial f}{\partial u}\,\eta + \varepsilon\,\frac{\partial f}{\partial u_x}\,\eta_x + \varepsilon\,\frac{\partial f}{\partial u_y}\,\eta_y$$

and thus

$$F(u+\varepsilon\cdot\eta) \approx F(u) + \varepsilon \iint\limits_{\Omega} \frac{\partial f}{\partial u}\,\eta + \frac{\partial f}{\partial u_x}\,\eta_x + \frac{\partial f}{\partial u_y}\,\eta_y\, dA$$

If F is minimal for $\varepsilon = 0$ we find

$$0 = \iint\limits_{\Omega} \frac{\partial f}{\partial u}\,\eta + \begin{pmatrix} \frac{\partial f}{\partial u_x} \\ \frac{\partial f}{\partial u_y} \end{pmatrix} \cdot \begin{pmatrix} \eta_x \\ \eta_y \end{pmatrix} dA$$

The divergence theorem in Appendix A.3 reads as

$$\iint\limits_{G} \nabla f \cdot \nabla g\, dA = \oint\limits_{\partial G} f\,\nabla g \cdot \vec{n}\, ds - \iint\limits_{G} f\,\Delta g\, dA$$

$$\iint\limits_{G} f\,(\operatorname{div}\vec{v})\, dA = \oint\limits_{\partial G} f\,\vec{v}\cdot\vec{n}\, ds - \iint\limits_{G} \operatorname{grad} f \cdot \vec{v}\, dA$$

and thus leads to

$$\begin{aligned} 0 &= \iint\limits_{\Omega} \eta\,\frac{\partial f}{\partial u} + \begin{pmatrix} \eta_x \\ \eta_y \end{pmatrix} \cdot \begin{pmatrix} \frac{\partial f}{\partial u_x} \\ \frac{\partial f}{\partial u_y} \end{pmatrix} dA \\ &= \iint\limits_{\Omega} \eta\,\frac{\partial f}{\partial u} + \eta\left(\frac{d}{dx}\frac{\partial f}{\partial u_x} + \frac{d}{dy}\frac{\partial f}{\partial u_y}\right) dA + \oint\limits_{\partial\Omega} \eta \begin{pmatrix} \frac{\partial f}{\partial u_x} \\ \frac{\partial f}{\partial u_y} \end{pmatrix} \cdot \vec{n}\, ds \\ &= \iint\limits_{\Omega} \eta\left(\frac{\partial f}{\partial u} + \frac{d}{dx}\frac{\partial f}{\partial u_x} + \frac{d}{dy}\frac{\partial f}{\partial u_y}\right) dA + \oint\limits_{\partial\Omega} \eta \begin{pmatrix} \frac{\partial f}{\partial u_x} \\ \frac{\partial f}{\partial u_y} \end{pmatrix} \cdot \vec{n}\, ds \end{aligned}$$

The fundamental lemma of the calculus of variations for two variables now implies

[1] Instead of using a linear approximation we may also differentiate the resulting integral with respect to the parameter ε and then set $\varepsilon = 0$

$$\begin{array}{rcll} \frac{\partial f}{\partial u} + \frac{d}{dx}\frac{\partial f}{\partial u_x} + \frac{d}{dy}\frac{\partial f}{\partial u_y} & = & 0 & \text{for} \quad (x,y) \in \Omega \\ u(x,y) & = & g(x,y) & \text{for} \quad (x,y) \in \Gamma_1 \\ \begin{pmatrix} \frac{\partial f}{\partial u_x} \\ \frac{\partial f}{\partial u_y} \end{pmatrix} \cdot \vec{n} & = & 0 & \text{for} \quad (x,y) \in \Gamma_2 \end{array} \tag{6.1}$$

These are the **Euler Lagrange equations** for problems with two independent variables x and y. For general functions f this partial differential equations may be nonlinear and thus usually difficult to solve. The most common case are linear differential equations, which are generated by quadratic functionals F .

6.3 The general quadratic functional

Consider a domain $\Omega \subset \mathbb{R}^2$ with a boundary $\partial\Omega = \Gamma_1 \cup \Gamma_2$ consisting of two disjoint parts. For given functions a, f, g_1 and g (all depending on x and y) we search a yet unknown function u, such that the functional

$$F(u) = \iint\limits_\Omega \frac{1}{2}\, a\, (\nabla u)^2 + \frac{1}{2}\, b\, u^2 + f \cdot u \; dA - \int_{\Gamma_2} g_2\, u \; ds \tag{6.2}$$

is minimal amongst all functions u which satisfy

$$u(x,y) = g_1(x,y) \quad \text{for} \quad (x,y) \in \Gamma_1$$

To find the necessary equations we assume that ϕ and $\nabla\phi$ are small and use the approximations

$$\begin{array}{rcl} (u+\phi)^2 & = & u^2 + 2\,u\,\phi + \phi^2 \approx u^2 + 2\,u\,\phi \\ (\nabla(u+\phi))^2 & = & \nabla u \cdot \nabla u + 2\,\nabla u \cdot \nabla\phi + \nabla\phi \cdot \nabla\phi \approx \nabla u \cdot \nabla u + 2\,\nabla u \cdot \nabla\phi \end{array}$$

to conclude

$$\begin{array}{rcl} F(u+\phi) - F(u) & \approx & \displaystyle\iint\limits_\Omega a\,\nabla u \cdot \nabla\phi + b\,u\,\phi + f \cdot \phi \; dA - \int_{\Gamma_2} g_2\,\phi \; ds \\ & = & \displaystyle\iint\limits_\Omega (-\nabla(\,a\,\nabla u) + b\,u + f) \cdot \phi \; dA + \int_\Gamma a\,\vec{n} \cdot \nabla u\;\phi \; ds - \int_{\Gamma_2} g_2\,\phi \; ds \\ & = & \displaystyle\iint\limits_\Omega (-\nabla(\,a\,\nabla u) + b\,u + f) \cdot \phi \; dA + \int_{\Gamma_2} (a\,\vec{n} \cdot \nabla u - g_2)\,\phi \; ds \end{array}$$

The test-function ϕ is arbitrary, but has to vanish on Γ_1. If the functional F is minimal for the function u then the above integral has to vanish for all test-functions ϕ. First consider only test-functions that vanish on Γ_2 and use the fundamental lemma A–2 to conclude that the expression in the bracket in the integral over the domain Ω has to be zero. Then use arbitrary test functions ϕ to conclude that the expression in the integral over Γ_2 has to vanish too. Thus the resulting linear partial differential equation with boundary conditions is given by

$$\begin{array}{rcll} \nabla \cdot (a\,\nabla u) - b\,u & = & f & \text{for} \quad (x,y) \in \Omega \\ u & = & g_1 & \text{for} \quad (x,y) \in \Gamma_1 \\ a\,\vec{n} \cdot \nabla u & = & g_2 & \text{for} \quad (x,y) \in \Gamma_2 \end{array} \tag{6.3}$$

The functions a, b f and g_i are known functions and we have to determine the solution u, all depending on the independent variables $(x, y) \in \Omega$. The vector $\vec{n}$ is the **outer unit normal vector**. The expression

$$\vec{n} \cdot \nabla u = n_1 \frac{\partial u}{\partial x} + n_2 \frac{\partial u}{\partial y} = \frac{\partial u}{\partial \vec{n}}$$

equals the directional derivative of the function u in the direction of the outer normal $\vec{n}$.

A list of typical applications of elliptic equations of second order is shown in table 6.1, copied from [Redd84, p. 32]. The static heat conduction problem in section 4.1.3 (page 64) is a further example. A description of the ground water flow problem is given in [OttoPete92, p. 87]. This table clearly illustrates the importance of the above type of problem.

6.4 A minimal surface problem

On the boundary of a domain $\Omega \subset \mathbb{R}^2$ the values of a function u are given by g. We search for the function u, such that the total area of the surface is minimal. A physical setup of this is given by soap bubbles. The boundary can be given by a closed piece of wire in space. Then the soap film will form a surface between the wire with minimal area. Further description can be found in [HildTrom86] and [Isen78]. The roof of the Olympic Stadium in Munich is a well known example of a minimal surface.

From calculus we know that the area of a surface given by $z = u(x, y)$ and $(x, y) \in \Omega \subset \mathbb{R}^2$ is given by

$$F(u) = \iint_\Omega \sqrt{1 + u_x^2 + u_y^2}\, dA$$

Thus we have an example of the type in section 6.2 with

$$\begin{aligned} f(u_x, u_y) &= \sqrt{1 + u_x^2 + u_y^2} \\ \frac{\partial}{\partial u_x} f(u_x, u_y) &= \frac{u_x}{\sqrt{1 + u_x^2 + u_y^2}} \\ \frac{\partial}{\partial u_y} f(u_x, u_y) &= \frac{u_y}{\sqrt{1 + u_x^2 + u_y^2}} \end{aligned}$$

Thus the Euler Lagrange equation 6.1 for this problem reads as

$$\begin{aligned} \frac{\partial f}{\partial u} + \frac{\partial}{\partial x}\frac{\partial f}{\partial u_x} + \frac{\partial}{\partial y}\frac{\partial f}{\partial u_y} &= 0 \\ \frac{\partial}{\partial x} \frac{u_x}{\sqrt{1 + u_x^2 + u_y^2}} + \frac{\partial}{\partial y} \frac{u_y}{\sqrt{1 + u_x^2 + u_y^2}} &= 0 \end{aligned}$$

This is a nonlinear partial differential equation.

If the slopes are known to be small ($|u_x| \ll 1$ and $|u_y| \ll 1$) we may use the Taylor approximation $\sqrt{1+s} \approx 1 + \frac{1}{2}\, s$ and obtain an approximate functional

$$F(u) \approx \iint_\Omega 1 + \frac{1}{2}\, \left(u_x^2 + u_y^2\right)\, dA$$

Minimizing this new functional is equivalent to finding a minimum of the quadratic functional

$$F_l(u) = \iint_\Omega \frac{1}{2}\, (\nabla u)^2\, dA$$

Field of application	Primary variable	Material constant	Source variable	Secondary variables
General situation	u	a	f	q, $\frac{\partial u}{\partial x}$, $\frac{\partial u}{\partial y}$
Heat transfer	Temperature T	Conductivity k	Heat source Q	Heat flow density $\vec{q}$ $\vec{q} = -k\nabla T$
Electrostatics	Scalar potential Φ	Dielectric constant ε	Charge density ρ	Electric flux density D
Magnetostatics	Magnetic potential Φ	Permeability ν	Charge density ρ	Magnetic flux density B
Transverse deflection of elastic membrane	Transverse deflection u	Tension of membrane T	Transversely distributed load	Normal force q
Torsion of a bar	Warping function ϕ	Constant 1	Constant 0	Stress τ $\tau_{xz} = \frac{E\,\alpha}{2\,(1+\nu)}\,(-y + \frac{\partial\,\phi}{\partial x})$ $\tau_{yz} = \frac{E\,\alpha}{2\,(1+\nu)}\,(x + \frac{\partial\,\phi}{\partial y})$
Irrotational flow of an ideal fluid	Stream function Ψ Velocity potential Φ	Density ρ	Mass production σ (usually zero)	Velocity $(u, v)^T$ $\frac{d\Psi}{\partial x} = -u$ $\frac{d\Psi}{\partial y} = v$ $\frac{d\Phi}{\partial x} = u$ $\frac{d\Phi}{\partial y} = v$
Ground-water flow	Piezometric head Φ	Permeability K	Recharge Q (or pumping $-Q$)	seepage $q = K\,\frac{\partial\Phi}{\partial n}$ velocities $u = -K\,\frac{d\Phi}{\partial x}$ $v = -K\,\frac{d\Psi}{\partial y}$

Table 6.1: Some examples of Poisson's equation $-\nabla\,(a\,\nabla u) = f$

and thus we have an example of section 6.3 with $a = 1$ and $b = f = 0$. The corresponding linear partial differential equation is thus given by equation (6.3) and simplifies to

$$\begin{aligned} \Delta u = u_{xx} + u_{yy} &= 0 \quad \text{for} \quad (x, y) \in \Omega \\ u &= g \quad \text{for} \quad (x, y) \in \Gamma \end{aligned}$$

This is a standard problem for the Finite Element Method.

Chapter 7

Finite element problems in two variables

We want to find solutions of the linear partial differential equation (6.3)

$$\begin{aligned} \nabla \cdot (a\,\nabla u) - b\,u &= f \quad \text{for} \quad (x,y) \in \Omega \\ u &= g_1 \quad \text{for} \quad (x,y) \in \Gamma_1 \\ a\,\frac{\partial u}{\partial \vec{n}} u &= g_2 \quad \text{for} \quad (x,y) \in \Gamma_2 \end{aligned} \tag{7.1}$$

This corresponds to a minimizer of the functional given in equation (6.2)

$$F(u) = \iint\limits_{\Omega} \left(\frac{1}{2}\,a\,(\nabla u)^2 + \frac{1}{2}\,b\,u^2 + f \cdot u \right)\, dA - \int_{\Gamma_2} g_2\,u\,ds$$

with respect to all function u that coincide with g_1 on Γ_1.

Under rather general assumption, similar to section 5.1.1 on page 110 the quadratic functional $F(u)$ is strictly positive definite and the partial differential equation (7.1) will have a unique solution. The Finite Element Method will lead to an approximation u_h of the exact solution u. The convergence results in section 5.1 (see page 110ff) can be applied to this problem too, thus we use a similar notation. Let V be the function space[1] of once differentiable function, where all derivative are square integrable.

$$V = \{ u \in C^2(\Omega) \mid \iint\limits_{\Omega} u^2 + u_x^2 + u_y^2\, dA < \infty \}$$

The space V is infinite dimensional and will be approximated by a finite dimensional subspace $V_h \subset V$. On this subspace we will minimize the above functional F (see also table 5.1 on page 113).

7.1 Description of the general procedure

In this section we describe the algorithm for a Finite Element code, based on **piecewise linear interpolation**. This will be done in a few separate steps:

- Approximate the domain $\Omega \subset \mathbb{R}^2$ by a collection of triangles and consider the values of the functions u at the nodes as degrees of freedom. On each triangle the functions u is replaced by a linear function. The set of all those piecewise linear functions will form the finite dimensional subspace V_h, where the parameter h is a measure of the typical size of the triangles.
- Compute the contribution of each triangle to the functional $F(u)$.

[1] The mathematically correct spaces are Sobolev spaces, but we omit the technicalities

- Compute the contribution of the boundary sections to the functional $F(u)$.
- Add up all the above contributions and take the boundary conditions into account. Then minimize

$$F(u) \approx F_h(\vec{u}) = \frac{1}{2}\,\langle \vec{u}\,,\, \mathbf{A}\,\vec{u}\rangle + \langle \vec{u}\,,\, \vec{b}\rangle$$

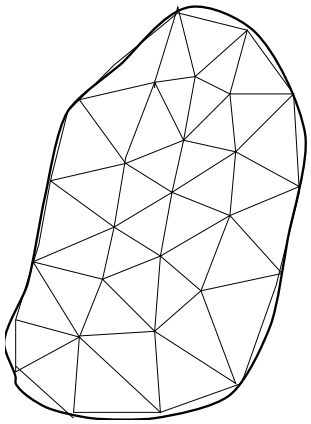
  which corresponds to solving the linear system of equations

$$\mathbf{A}\,\vec{u} + \vec{b} = \vec{0}$$

- The solution is then the minimum of the functional F amongst all functions u in the subspace V_h.

7.1.1 Approximation of the domain Ω, triangularization

We approximate the domain Ω by a finite set of triangles. If the domain has only straight line segments as boundary then we can generate the domain exactly. This leads to a finite number of nodes $(x_i, y_i) \in \Omega$, $1 \le i \le n$. On each triangle the function u is to be replaced by a plane, determined by the values $u_i = u\,(x_i, y_i)$ at the nodes. Thus we have a finite number of degrees of freedom. Integral over the domain Ω are now split up into a sum of integrals over each triangle Δ_k



$$\iint\limits_{\Omega} \ldots dA = \sum_k \iint\limits_{\Delta_k} \ldots dA$$

On each triangle Δ_k the integration will be done with the linearly interpolated function u. The above integral will be replaced by a summation over all triangles and we have to find the values u_i such that the value of the sum is minimal.

7.1.2 Integration over one triangle

If a triangle is given by its three corners $\vec{x}_1$, $\vec{x}_2$ and $\vec{x}_3$, then its area A is given by a cross product calculation

$$A = \frac{1}{2}\,\|(\vec{x}_2 - \vec{x}_1) \times (\vec{x}_3 - \vec{x}_1)\| = \frac{1}{2}\,\left|(x_2 - x_1)\cdot(y_3 - y_1) - (y_2 - y_1)\cdot(x_3 - x_1)\right|$$

If the values of a general function f are given at the tree corners of the triangle by f_1, f_2 and f_3 we can replace the exact function by a linearly interpolated function and find an approximate integral by

$$\int_{\Delta} f\,dA \approx A \cdot \frac{f_1 + f_2 + f_3}{3}$$

Observe that there is a systematic error due to replacing the true function by an approximate, linear function.

This leads to

$$\begin{aligned}
\int_{\Delta} f \cdot u\; dA &\approx \frac{A}{3}\,(f_1\,u_1 + f_2\,u_2 + f_3\,u_3) \\
\int_{\Delta} \frac{1}{2}\,b\,u^2\; dA &\approx \frac{1}{2}\,\frac{A}{3}\,(b_1\,u_1^2 + b_2\,u_2^2 + b_3\,u_3^2)
\end{aligned}$$

Using a vector notation we find

$$\int_{\Delta} \frac{1}{2}\,b\,u^2 + f \cdot u\; dA \approx \frac{1}{2}\,\frac{A}{3}\,\langle \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}\,,\, \begin{bmatrix} b_1 & 0 & 0 \\ 0 & b_2 & 0 \\ 0 & 0 & b_3 \end{bmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \rangle + \frac{A}{3}\,\langle \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix}\,,\, \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \rangle$$

This is one of the contributions in equation (6.2). To examine the other contribution we first need to compute the gradient of the function u. If the true function is replaced by a linear interpolation on the triangle, then the gradient is constant on this triangle and can be determined with the help of a normal vector of the plane passing through the three points

$$\begin{pmatrix} x_1 \\ y_1 \\ u_1 \end{pmatrix} \quad , \quad \begin{pmatrix} x_2 \\ y_2 \\ u_2 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} x_3 \\ y_3 \\ u_3 \end{pmatrix}$$

A normal vector is given by the vector product

$$\vec{n} = \begin{pmatrix} x_2 - x_1 \\ y_2 - y_1 \\ u_2 - u_1 \end{pmatrix} \times \begin{pmatrix} x_3 - x_1 \\ y_3 - y_1 \\ u_3 - u_1 \end{pmatrix} = \begin{pmatrix} +(y_2 - y_1) \cdot (u_3 - u_1) - (u_2 - u_1) \cdot (y_3 - y_1) \\ -(x_2 - x_1) \cdot (u_3 - u_1) + (u_2 - u_1) \cdot (x_3 - x_1) \\ +(x_2 - x_1) \cdot (y_3 - y_1) - (y_2 - y_1) \cdot (x_3 - x_1) \end{pmatrix}$$

The third component of this vector equals twice the oriented[2] area A of the triangle. To obtain the gradient in the first two components the vector has to be normalized, such that the third component equals -1. We find

$$\nabla u = \begin{pmatrix} \frac{d\,u}{\partial x} \\ \frac{d\,u}{\partial y} \end{pmatrix} = \frac{-1}{2\,A} \begin{pmatrix} +(y_2 - y_1) \cdot (u_3 - u_1) - (u_2 - u_1) \cdot (y_3 - y_1) \\ -(x_2 - x_1) \cdot (u_3 - u_1) + (u_2 - u_1) \cdot (x_3 - x_1) \end{pmatrix}$$

This formula can be written as

$$\nabla u = \frac{-1}{2\,A} \begin{bmatrix} (y_3 - y_2) & (y_1 - y_3) & (y_2 - y_1) \\ (x_2 - x_3) & (x_3 - x_1) & (x_1 - x_2) \end{bmatrix} \cdot \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \frac{-1}{2\,A}\, \mathbf{M} \cdot \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \tag{7.2}$$

This leads to

$$\langle \nabla u\,,\, \nabla u \rangle = \frac{1}{4\,A^2}\, \langle \mathbf{M} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}\,,\, \mathbf{M} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \rangle = \frac{1}{4\,A^2}\, \langle \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}\,,\, \mathbf{M}^T \cdot \mathbf{M} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \rangle$$

and thus

$$\int_\Delta \frac{1}{2}\, a\, (\nabla u)^2\, dA \approx \frac{a_1 + a_2 + a_3}{2 \cdot 3 \cdot 4\,A}\, \langle \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}\,,\, \mathbf{M}^T \cdot \mathbf{M} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \rangle$$

Exercise 7–1 verifies that the matrix $\mathbf{M}^T \cdot \mathbf{M}$ is symmetric and positive semidefinite. The expression vanishes if and only if $u_1 = u_2 = u_3$.

Collecting the above results we find

$$\int_\Delta \frac{1}{2}\, a\, (\nabla u)^2 + \frac{1}{2}\, b\, u^2 + f \cdot u\, dA \approx \frac{1}{2}\, \langle \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}\,,\, \mathbf{A}_\Delta \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} \rangle + \langle \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}\,,\, \vec{b}_\Delta \rangle$$

[2] We quietly assumed that the third component of $\vec{n}$ is positive. As we use only the square of the gradient the influence of this ignorance will disappear.

where

$$\mathbf{A}_\Delta = \frac{a_1+a_2+a_3}{12\,A}\,\mathbf{M}^T\cdot\mathbf{M}+\frac{A}{3}\begin{bmatrix} b_1 & 0 & 0\\ 0 & b_2 & 0\\ 0 & 0 & b_3\end{bmatrix} \tag{7.3}$$

$$\vec{b}_\Delta = \frac{A}{3}\begin{pmatrix} f_1\\ f_2\\ f_3\end{pmatrix} \tag{7.4}$$

If $b_i \geq 0$ then the element stiffness matrix $\mathbf{A}_\Delta$ is positive semidefinite.

7.1.3 Integration the contribution on the boundary

For the section Γ_2 of the boundary with Neumann boundary condition we have to approximate the integral.

$$\int_{\Gamma_2} g_2\,u\;ds$$

As each section consists of a straight line we choose the simple approximation

$$\frac{\text{length}}{2}\;(\text{value at left endpoint} + \text{value at right endpoint})$$

If we denote the two endpoints by $\vec{x}_1$ and $\vec{x}_2$ and the values of the function u by u_1, u_2, then we find the approximation

$$\int_{\text{edge}} g_2\,u\;ds \approx \frac{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}}{2}\,\langle\begin{pmatrix} u_1\\ u_2\end{pmatrix},\begin{pmatrix} g_2(\vec{x}_1)\\ g_2(\vec{x}_2)\end{pmatrix}\rangle$$

This approximate integration leads to the exact result if the function $u(\vec{x})\cdot g(\vec{x})$ is a polynomial of degree 1, or a constant.

An improved approach can be based on Gauss integration (see section 4.5.2). Instead of the two endpoints $\vec{x}_1$ and $\vec{x}_2$ we used the values at the two Gauss integration points

$$\begin{aligned}\vec{p}_1 &= \tfrac{1}{2}\,(\vec{x}_1+\vec{x}_2)+\tfrac{1}{2\sqrt{3}}\,(\vec{x}_1-\vec{x}_2)\\ \vec{p}_2 &= \tfrac{1}{2}\,(\vec{x}_1+\vec{x}_2)-\tfrac{1}{2\sqrt{3}}\,(\vec{x}_1-\vec{x}_2)\end{aligned}$$

By linear interpolation between the points $\vec{x}_1$ and $\vec{x}_2$ we find the values of the function u at the Gauss points to be

$$\begin{aligned}u(\vec{p}_1) &= (1-w)\,u_1+w\,u_2\\ u(\vec{p}_2) &= w\,u_1+(1-w)\,u_2\end{aligned}$$

where $w=\frac{1-1/\sqrt{3}}{2}\approx 0.211325$. This leads to the approximation

$$\int_{\text{edge}} g_2\,u\;ds \approx \frac{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}}{2}\,\langle\begin{pmatrix} u_1\\ u_2\end{pmatrix},\begin{pmatrix} (1-w)\,g_2(\vec{p}_1)+w\,g_2(\vec{p}_2)\\ w\,g_2(\vec{p}_1)+(1-w)\,g_2(\vec{p}_2)\end{pmatrix}\rangle$$

This approximate integration leads to the exact result if the function $u(\vec{x})\cdot g(\vec{x})$ is a polynomial of degree 3, or less. In addition the function g_2 need not be evaluated at the endpoints but at interior points of the segment. This is helpful at corners with two different types of boundary conditions. Thus this approach is clearly preferable.

7.1.4 Assembling the system of equations

Using the results of the previous sections we write the contribution of each triangle Δ_k to the functional F as

$$\frac{1}{2}\,\langle \vec{u}_\Delta\,,\,\mathbf{A}_\Delta\,\vec{u}_\Delta\rangle + \langle \vec{u}_\Delta\,,\,\vec{b}_\Delta\rangle$$

where the vector $\vec{u}_\Delta$ contains the values of the function u at the three corners of the triangle and $\mathbf{A}_\Delta$ is the **element stiffness matrix**. For each $u \in V_h$ we have

$$F(u) = \sum_\Delta \frac{1}{2}\,\langle \vec{u}_\Delta\,,\,\mathbf{A}_\Delta\,\vec{u}_\Delta\rangle + \langle \vec{u}_\Delta\,,\,\vec{b}_\Delta\rangle$$

where the summation is over all triangles. Now we want to find the **global stiffness matrix A** and the vector $\vec{b}$ such that

$$F(u) = \frac{1}{2}\,\langle \vec{u}\,,\,\mathbf{A}\,\vec{u}\rangle + \langle \vec{u}\,,\,\vec{b}\rangle \quad \text{for all} \quad u \in V_h$$

the vector $\vec{u}$ consists of the values of the unknown function u at all the nodes of the triangularization. Thus we will use both symbols u and $\vec{u}$ to denote the function.

for each element (resp. edge)	
$\mathbf{A}_\Delta$	3×3 matrix, symmetric
$\vec{b}_\Delta$	vector with 3 components
$\vec{u}_\Delta$	vector of unknowns with 3 components
for the complete structure	
$\mathbf{A}$	$n \times n$ matrix, symmetric
$\vec{b}$	vector with n components
$\vec{u}$	vector of unknowns with n components

The problem to solve now is to combine the contributions from all triangles and edges.

To illustrate this procedure we use an example. If the triangle Δ has the corners (x_1, y_1), (x_4, y_4) and (x_5, y_5) and the matrix is given by

$$A_\Delta = \begin{bmatrix} 11 & -2 & -3 \\ -2 & 22 & -1 \\ -3 & -1 & 99 \end{bmatrix} \quad \text{and} \quad \vec{b}_\Delta = \begin{pmatrix} 4 \\ 7 \\ 11 \end{pmatrix}$$

and the complete structure has 7 nodes. Then we extend the above matrix artificially to a 7×7 matrix by adding rows and columns of zeros The nonzero numbers of the element matrix $\mathbf{A}_\Delta$ have to go into rows/columns 1, 4 and 5 of the global matrix $\mathbf{A}$. Thus we construct

$$\overline{A_\Delta} = \begin{bmatrix} 11 & 0 & 0 & -2 & -3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -2 & 0 & 0 & 22 & -1 & 0 & 0 \\ -3 & 0 & 0 & -1 & 99 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad \overline{\vec{b}_\Delta} = \begin{pmatrix} 4 \\ 0 \\ 0 \\ 7 \\ 11 \\ 0 \\ 0 \end{pmatrix}$$

Observe that the new matrix $\overline{\mathbf{A}_\Delta}$ is still symmetric and

$$\frac{1}{2}\,\langle \vec{u}_\Delta\,,\,\mathbf{A}_\Delta\,\vec{u}_\Delta\rangle + \langle \vec{u}_\Delta\,,\,\vec{b}_\Delta\rangle = \frac{1}{2}\,\langle \vec{u}\,,\,\overline{\mathbf{A}_\Delta}\,\vec{u}\rangle + \langle \vec{u}\,,\,\overline{\vec{b}_\Delta}\rangle$$

where the scalar product is computed with vectors with 3 components on the left, respectively 7 components on the right. The size of the new matrices and vectors are independent of the element and the numbering of the nodes decides on where to put the nonzero numbers. This process has to be done for each element. Verify that

$$F(u) = \frac{1}{2}\,\langle \vec{u}\,,\,\mathbf{A}\,\vec{u}\rangle + \langle \vec{u}\,,\,\vec{b}\rangle = \sum_\Delta \frac{1}{2}\,\langle \vec{u}\,,\,\overline{\mathbf{A}_\Delta}\,\vec{u}\rangle + \langle \vec{u}\,,\,\overline{\vec{b}_\Delta}\rangle$$

and thus

$$\mathbf{A} = \sum_\Delta \overline{\mathbf{A}_\Delta} \quad \text{and} \quad \vec{b} = \sum_\Delta \overline{\vec{b}_\Delta}$$

i.e. a matrices and vectors can be added up. If the Finite Element Method is to be implemented on a computer then the matrices $\overline{\mathbf{A}_\Delta}$ should not be created explicitely but rather the algorithm below can be used.

- create a $n \times n$ matrix $\mathbf{A}$ and the n vector $\vec{b}$, filled with zeros.
- for each element Δ:
  - compute $\mathbf{A}_\Delta$ and $\vec{b}_\Delta$
  - add the values in $\mathbf{A}_\Delta$ and $\vec{b}_\Delta$ to the correct rows/columns in $\mathbf{A}$ and $\vec{b}$.
- the final matrix $\mathbf{A}$ and vector $\vec{b}$ can then be used to solve for the unknown values $\vec{u}$ by minimizing

$$F(\vec{u}) = \frac{1}{2}\,\langle \vec{u}\,,\,\mathbf{A}\,\vec{u}\rangle + \langle \vec{u}\,,\,\vec{b}\rangle$$

  Observe that the Dirichlet boundary condition $u = g_1$ on Γ_1 has not been taken into account yet.

7.1.5 Taking the Dirichlet boundary condition into account

If we were free to choose all components of the vector $\vec{u}$ then the minimum of $F(\vec{u})$ is attained at the solution of $\mathbf{A}\,\vec{u} + \vec{b} = \vec{0}$. This is a consequence of the result 1–4 (page 4). But if the point $\vec{x}_i$ is on the Dirichlet boundary Γ_1 the value of u_i is given by $g_1(\vec{x}_i)$ and we obtain no corresponding equation. If we are free to choose the value of u_i then the derivative of $F(\vec{u})$ with respect to u_i has to vanish. This leads to the linear equation

$$\sum_j a_{i,j}\,u_j = -b_i$$

where $a_{a,j}$ are the components of the matrix $\mathbf{A}$. If $\vec{x}_k \in \Gamma_1$ then the value of the solution is given by $u_k = g_1(\vec{x}_k)$. Then the above equation can be written as

$$\sum_{\vec{x}_j \notin \Gamma_1} a_{i,j}\,u_j = -b_i - \sum_{\vec{x}_k \in \Gamma_1} a_{i,g}\,g_1(\vec{x}_k)$$

If $\vec{x}_i \in \Gamma_1$ then the derivative of $F(\vec{u})$ with respect to u_i is not necessarily zero and thus we do not obtain a corresponding equation. This leads to a reduced matrix $\mathbf{A}_r$ and vector $\vec{b}_r$ and a smaller system of linear equations

$$\mathbf{A}_r\,\vec{u}_r = -\vec{b}_r$$

to be solved. Since we have exactly as many unknowns u_i as equations the matrix $\mathbf{A}_r$ will be square.

7.1.6 Applying periodic boundary conditions

The total energy of the system is given by

$$F(\vec{u}) = \frac{1}{2}\,\langle \vec{u}\,,\, \mathbf{A}\,\vec{u}\rangle + \langle \vec{u}\,,\,\vec{b}\rangle$$

If we are to consider a problem with periodic boundary conditions, then some of the degrees of freedom have to be eliminated, i.e. we have the additional condition $u_p = u_q$ for some values of p and q. Since those two degrees of freedom are coupled now we define a new function $f(t) = F(\vec{u})$, where $u_p = u_q = t$. Then the total derivative of $f(t)$, with respect to t equals the sum of the partial derivatives of $F(\vec{u})$ with respect to u_p and u_q. In exercise 1–3 we found

$$\frac{\partial}{\partial u_p}\,F(\vec{u}) = \sum_{j=1,}^{n} a_{p,j}\,u_j + b_p$$

Thus we arrive at a modified system of linear equations to be solved.

- replace all coefficients in row p in the matrix $\mathbf{A}$ by $a_{p,j} + a_{q,j}$, i.e. add row q to row p
- replace b_p by $b_p + b_q$ in the vector $\vec{b}$
- replace row q in the matrix $\mathbf{A}$ by the condition $u_p - u_q = 0$

This leads to a modified system of equations to be solved. Unfortunately the symmetry of the matrix $\mathbf{A}$ is lost. If the symmetry is important we may modify the above procedure. We arrive at a system of fewer equations, but have the extra condition $u_p = u_q$.

- replace all coefficients in row p in the matrix $\mathbf{A}$ by $a_{p,j} + a_{q,j}$, i.e. add row q to row p
- replace all coefficients in column p in the matrix $\mathbf{A}$ by $a_{j,p} + a_{j,q}$, i.e. add column q to column p
- replace b_p by $b_p + b_q$ in the vector $\vec{b}$
- the resulting new matrix $\mathbf{A}$ has one fewer row and column.
- take the condition $u_p - u_q = 0$ into account later on.

7.1.7 Solving the set of linear equations, visualization and interpretation

After solving the above equation for $\vec{u}_r$ all components of the original vector $\vec{u}$ can be constructed by adding the components corresponding to nodes on Γ_1. Now the data of an approximate solution of problem (7.1) is generated and can be analyzed or visualized by appropriate tools.

7.2 The eigenvalue problem

Instead of equation (7.1) we examine the general eigenvalues problem

$$\begin{aligned} -\nabla\cdot(a\,\nabla u) + b\,u &= \lambda\,f\,u \qquad \text{for} \quad (x,y)\in\Omega \\ u &= 0 \qquad\quad\;\; \text{for} \quad (x,y)\in\Gamma_1 \\ a\,\tfrac{\partial u}{\partial \vec{n}}u &= 0 \qquad\quad\;\; \text{for} \quad (x,y)\in\Gamma_2 \end{aligned} \tag{7.5}$$

For known functions a, b and f the equation has always the trivial solution $u = 0$. The number $\lambda \in \mathbb{R}$ is called an **eigenvalue** if the problem has a nontrivial solution. The function u is called an **eigenfunction**. This is equivalent to finding a function u and a value of λ such that

$$\iint\limits_{\Omega} (-\nabla(\,a\,\nabla u) + b\,u)\cdot\phi\;dA = \lambda \iint\limits_{\Omega} f\,u\,\phi\;dA$$

for all test functions ϕ vanishing on Γ_1 .

Now we first examine the contribution to the integral on the RHS when integrating over one triangle Δ of the mesh. In section 7.1.2 we found

$$\int_{\Delta} f\;dA \approx A\cdot\frac{f_1+f_2+f_3}{3}$$

This leads to

$$\begin{aligned}\int_{\Delta} f\cdot u\cdot\phi\;dA &\approx \frac{A}{3}\,(f_1\,u_1\,\phi_1 + f_2\,u_2\,\phi_2 + f_3\,u_3\,\phi_3)\\ &= \frac{A}{3}\,\langle\begin{bmatrix} f_1 & 0 & 0\\ 0 & f_2 & 0\\ 0 & 0 & f_3\end{bmatrix}\begin{pmatrix} u_1\\ u_2\\ u_3\end{pmatrix},\begin{pmatrix}\phi_1\\ \phi_2\\ \phi_3\end{pmatrix}\rangle\\ &= \langle\mathbf{M}_{\Delta}\,\vec{u}_{\Delta}\,,\,\vec{\phi}_{\Delta}\rangle\end{aligned}$$

By comparing with computations in the previous section we find

$$\mathbf{B}_{\Delta} = \mathrm{diag}(\vec{b}_{\Delta})$$

i.e. the coefficients of the matrix $\mathbf{B}_{\Delta}$ may be computed by the same algorithm as the components of $\vec{b}_{\Delta}$ in equation (7.4). By adding the contributions of all triangles we find

$$\iint\limits_{\Omega} f\cdot u\cdot\phi\;dA \approx \langle\mathbf{B}\,\vec{u}\,,\,\vec{\phi}\rangle$$

where $\mathbf{B} = \mathrm{diag}(\vec{b})$ is a diagonal matrix whose entries are given by the contribution of the RHS f in equation (7.1) to the vector $\vec{b}$.

The previous section implies

$$\iint\limits_{\Omega} (-\nabla(\,a\,\nabla u) + b\,u)\cdot\phi\;dA \approx \langle\mathbf{A}\,\vec{u}\,,\,\vec{\phi}\rangle$$

where the global stiffness matrix $\mathbf{A}$ is identical to the one in the previous section. Thus the FEM approximation of the above integral is given by

$$\langle\mathbf{A}\,\vec{u}\,,\,\vec{\phi}\rangle = \lambda\,\langle\mathbf{B}\,\vec{u}\,,\,\vec{\phi}\rangle$$

for all vectors $\vec{\phi}\in\mathbb{R}^n$. This is equivalent to the **generalized eigenvalue problem**

$$\mathbf{A}\,\vec{u} = \lambda\,\mathbf{B}\,\vec{u}$$

There in no need to write new code to set up this problem since the matrix $\mathbf{A}$ and $\mathbf{B} = \mathrm{diag}(\vec{b})$ are also given in the linear system $\mathbf{A}\,\vec{u} = -\vec{b}$ in the previous section. In section 11.6 an algorithm to compute the eigenvalues is presented.

7.3 From the finite element method to a finite difference method

We examine a very simple, but important special case of the general situation (7.1) on page 146.

$$\begin{aligned} u_{xx} + u_{yy} &= f(x,y) \qquad \text{for} \quad (x,y) \in \Omega = (0,a) \times (0,b) \\ u(x,y) &= 0 \qquad \text{for} \quad (x,y) \text{ on boundary } \partial\Omega \end{aligned} \tag{7.6}$$

This corresponds to the problem of minimizing the functional

$$F(u) = \int_\Omega \frac{1}{2} \left(\left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 \right) + u \cdot f \; dA$$

The domain Ω is shown in figure 7.1 with a uniform rectangular mesh. The mesh has nx interior nodes in x direction and ny interior nodes in y direction. The nodes are uniformly spaced. In the shown example we have $nx = 18$ and $ny = 5$. The step sizes are then given by

$$hx = \frac{a}{nx+1} \quad \text{and} \quad hy = \frac{b}{ny+1}$$

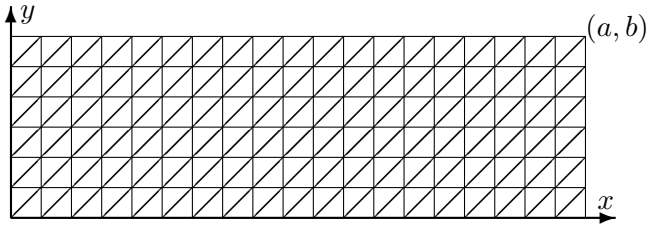


Figure 7.1: A simple rectangular mesh

7.3.1 Element contributions

In the mesh in figure 7.1 we find only two types of triangles, shown in figure 7.2 and thus we can compute all element contributions with the help of those two standard triangles.

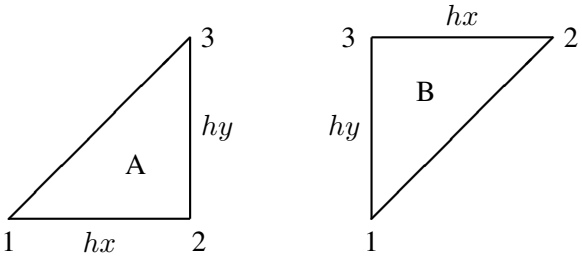


Figure 7.2: The two types of triangles in a rectangular mesh

Contributions from type A triangles

If the values u_i of the function at the three corners of a type A triangle are known then we have the gradient of the linearly interpolated function given by

$$\nabla u = \frac{-1}{2\,A}\begin{bmatrix} (y_3-y_2) & (y_1-y_3) & (y_2-y_1) \\ (x_2-x_3) & (x_3-x_1) & (x_1-x_2) \end{bmatrix} \cdot \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \frac{-1}{2\,A}\,\mathbf{M}\cdot\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}$$

Using equation (7.2) we find

$$\nabla u = \frac{-1}{2\,A}\,\mathbf{M}\cdot\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \frac{-1}{hx\cdot hy}\begin{bmatrix} hy & -hy & 0 \\ 0 & hx & -hx \end{bmatrix}\cdot\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}$$

and

$$\mathbf{M}^T\cdot\mathbf{M} = \begin{bmatrix} hy & 0 \\ -hy & hx \\ 0 & -hx \end{bmatrix}\cdot\begin{bmatrix} hy & -hy & 0 \\ 0 & hx & -hx \end{bmatrix} = \begin{bmatrix} hy^2 & -hy^2 & 0 \\ -hy^2 & hy^2+hx^2 & -hx^2 \\ 0 & -hx^2 & hx^2 \end{bmatrix}$$

Observe that the zeros in the off-diagonal corners are based on the facts $y_1 = y_2$ and $x_2 = x_3$. Equation (7.3) now leads to the element stiffness matrix

$$\begin{aligned} \mathbf{A}_{\Delta_A} &= \frac{a_1+a_2+a_3}{12\,A}\,\mathbf{M}^T\cdot\mathbf{M} \\ &= \frac{1}{2\,hx\,hy}\begin{bmatrix} hy^2 & -hy^2 & 0 \\ -hy^2 & hy^2+hx^2 & -hx^2 \\ 0 & -hx^2 & hx^2 \end{bmatrix} \end{aligned}$$

and

$$\vec{b}_{\Delta_A} = \frac{A}{3}\begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} = \frac{hx\,hy}{6}\begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix}$$

Contributions from type B triangles

Similar calculations lead to the element stiffness matrix

$$\mathbf{A}_{\Delta_B} = \frac{1}{2\,hx\,hy}\begin{bmatrix} hx^2 & 0 & -hx^2 \\ 0 & hy^2 & -hy^2 \\ -hx^2 & -hy^2 & hy^2+hx^2 \end{bmatrix}$$

and $\vec{b}_{\Delta_B} = \vec{b}_{\Delta_A}$. The details of the computations are left as an exercise.

7.3.2 The linear equation associated with an interior node

Now we construct the system of linear equations for the boundary value problem (7.6) on the mesh in figure 7.1. For this we consider the mesh point in column i and row j (starting at the bottom) in the mesh and denote the value of the solution at this point by $u_{i,j}$. In figure 7.1 we see that $u_{i,j}$ is directly connected

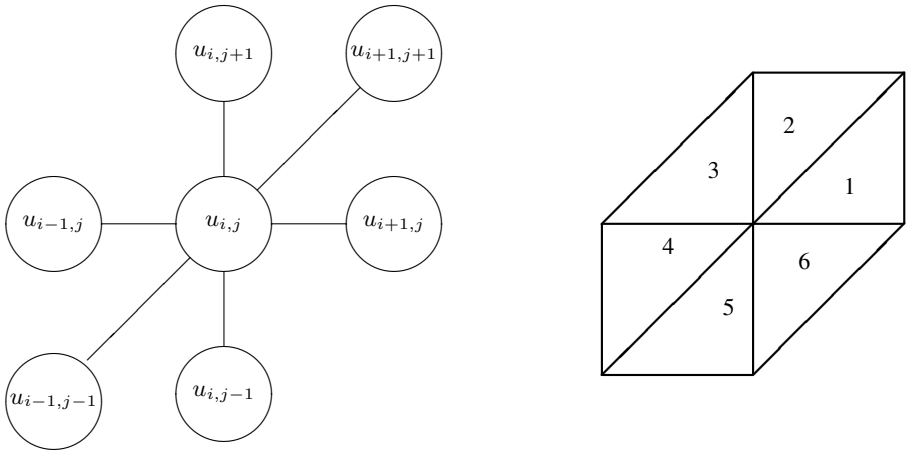


Figure 7.3: FEM stencil and neighboring triangles of a mesh point

to 6 neighboring points and 6 triangles are used to built the connections. This is visualized in figure 7.3, the left part of that figure represents the **stencil** of the point in the mesh.

We first examine the contributions to $\int_\Omega u \cdot f \; dA$ involving the coefficients $u_{i,j}$. As the coefficients in $\vec{b}_{\Delta_A} = \vec{b}_{\Delta_B}$ are all constant we obtain six contributions of the size $\frac{hx\,hy}{6}\; f_{i,j}$ leading to a total of $hx\;hy\;f_{i,j}$

$$f_{i,j} \quad \rightarrow \quad 6 \left(\frac{hx\,hy}{6} \right) = hx\;hy$$

With similar arguments we have to examine the contributions to $\frac{1}{2} \int_\Omega \nabla u \cdot \nabla u \; dA$. The expressions below, multiplied by $\frac{1}{2}\, u_{i,j}$ will contribute. Use the basic element matrices $\mathbf{A}_{\Delta_A}$, $\mathbf{A}_{\Delta_B}$ and figure 7.3 to verify the terms below.

$$\begin{aligned}
u_{i,j} \quad &\rightarrow \quad \frac{1}{2\;hx\;hy} \left(hy^2 + hx^2 + (hx^2 + hy^2) + hy^2 + hx^2 + (hx^2 + hy^2) \right) = \frac{2\;(hx^2 + hy^2)}{hx\;hy} \\
u_{i+1,j} \quad &\rightarrow \quad \frac{1}{2\;hx\;hy} \left(-hx^2 + 0 + 0 + 0 + 0 - hx^2 \right) = \frac{-hx^2}{hx\;hy} \\
u_{i-1,j} \quad &\rightarrow \quad \frac{1}{2\;hx\;hy} \left(0 + 0 - hx^2 - hx^2 + 0 + 0 \right) = \frac{-hx^2}{hx\;hy} \\
u_{i,j+1} \quad &\rightarrow \quad \frac{1}{2\;hx\;hy} \left(0 - hy^2 - hy^2 + 0 + 0 + 0 \right) = \frac{-hy^2}{hx\;hy} \\
u_{i,j-1} \quad &\rightarrow \quad \frac{1}{2\;hx\;hy} \left(0 + 0 + 0 + 0 - hy^2 - hy^2 \right) = \frac{-hy^2}{hx\;hy} \\
u_{i+1,j+1} \quad &\rightarrow \quad \frac{1}{2\;hx\;hy} \left(0 + 0 + 0 + 0 + 0 + 0 \right) = 0 \\
u_{i-1,j-1} \quad &\rightarrow \quad \frac{1}{2\;hx\;hy} \left(0 + 0 + 0 + 0 + 0 + 0 \right) = 0
\end{aligned}$$

Observe that the two diagonal connections in the stencil lead to zero contributions. This is correct for the rectangular mesh. Thus the resulting equation for the degree of freedom $u_{i,j}$ is given by

$$u_{i,j}\; \frac{2\;(hx^2 + hy^2)}{hx\;hy} - u_{i+1,j}\; \frac{hx^2}{hx\;hy} - u_{i-1,j}\; \frac{hx^2}{hx\;hy} - u_{i,j+1}\; \frac{hy^2}{hx\;hy} - u_{i,j-1}\; \frac{hy^2}{hx\;hy} + f_{i,j}\; hx\;hy = 0$$

or by rearranging

$$\frac{-u_{i+1,j} + 2\;u_{i,j} - u_{i-1,j}}{hx^2} + \frac{-u_{i,j+1} + 2\;u_{i,j} - u_{i,j-1}}{hy^2} = -f_{i,j} \tag{7.7}$$

For the special case $hx = hy$ we obtain

$$\frac{1}{hx^2}\ (4\ u_{i,j} - u_{i+1,j} - u_{i-1,j} - u_{i,j+1} - u_{i,j-1}) = -f_{i,j}$$

This is the **finite difference** approximation to the differential equation (7.6). It can be obtained directly[3]. This leads to the well know finite difference stencil in figure 7.4.

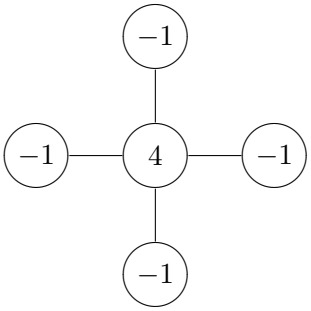


Figure 7.4: Finite difference stencil for $-u_{xx} - u_{yy}$ if $hx = hy$

7.3.3 Assembling the system of linear equations

Now we want to examine the system of linear equations to be solved. For this we have to number the degrees of freedom (interior nodes) in the mesh 7.1 sequentially. Starting in the lower left corner we number row by row. Thus the first row of interior nodes obtains numbers 1 through nx, the second row numbers $nx + 1$ through $2\,nx$. The top right corner will have the largest number $nx \cdot ny$. Thus if we are to move one row up we have to increase the number by nx. Steps to the right increase the number by 1. Now we use equation (7.7) to generate the square matrix **A** of size $(nx \cdot ny) \times (nx \cdot ny)$. We also have to use the Dirichlet boundary conditions in (7.6). As a stencil reaches the boundary the corresponding contribution will be set to 0. We generate the matrix by the following rules:

- Use the coefficient of $u_{i,j}$ to determine that along the diagonal all entries are $\frac{2}{hx^2} + \frac{2}{hy^2}$.
- The coefficient of $u_{i\pm1,j}$ in (7.7) is $\frac{-1}{hx^2}$ and thus the first upper and lower diagonal are filled with this number. This corresponds to the stencil reaching one step to the right and left.
- If the point is on the last interior column then there is no contribution by the point on its right, due to the boundary condition. Thus if the number k of the point is a multiple of nx we have to set $a_{k,k+1}$ to 0. If the point is on the first interior column then there is no contribution by the point on its left, due to the boundary condition. Thus if the number k of the point is a multiple of nx plus 1 then we have to set $a_{k,k-1}$ to 0. The matrix remains symmetric by these operations.
- The coefficient of $u_{i,j\pm1}$ in (7.7) is $\frac{-1}{hy^2}$. This number has to be filled into the diagonals of by nx units above and below the main diagonal. This corresponds to the stencil reaching one step up and down.
- The first row of interior points should not 'see' contributions from below, due to the boundary conditions. This is automatically the case since for the first nx rows of the matrix the nx-subdiagonal diagonal reaches out of the matrix on the left. A similar effect occurs for the top row.

[3] Use $u'_{i-1/2} \approx \frac{1}{h}\ (u_i - u_{i-1})$, $u'_{i+1/2} \approx \frac{1}{h}\ (u_{i+1} - u_{i1})$ and $u''_i \approx \frac{1}{h}\ (u'_{i+1/2} - u'_{i-1/2})$ to arrive at

$$-u''_i \approx \frac{-u_{i+1} + 2\ u_i - u_{i-1}}{h^2}$$

The above procedure may be implemented as in the C++ code segment below. Since the matrix **A** has to be symmetric some of the above steps can be skipped. The data type `TMatSymBand` assures that the lower part of **A** is correct if we supply the upper part.

```
TMatSymBand A(nx*ny,nx+1);                          // create the matrix
A.fill(0.0);                                        // initialize all values to 0
A.fillDiag(0,2.0*(1/(hx*hx)+1/(hy*hy)));            // fill the main diagonal
A.fillDiag(1,-1/(hx*hx));                           // fill the first upper diagonal
A.fillDiag(nx,-1/(hy*hy));                          // fill the upper diagonal nx
for(int j=1;j<ny;j++){ A(j*nx,j*nx+1)=0.0;}         // apply the boundary conditions
```

One has to observe that the resulting matrix has a band structure: all entries further than nx away from the main diagonal will remain zero. As an example we can consider the case $a = b = 1$ and $nx = 3$, $ny = 4$. This leads to $\frac{1}{hx^2} = 16$ and $\frac{1}{hy^2} = 25$. The mesh, the numbering of the interior nodes and an approximate solution of the differential equation are shown in figure 7.5.

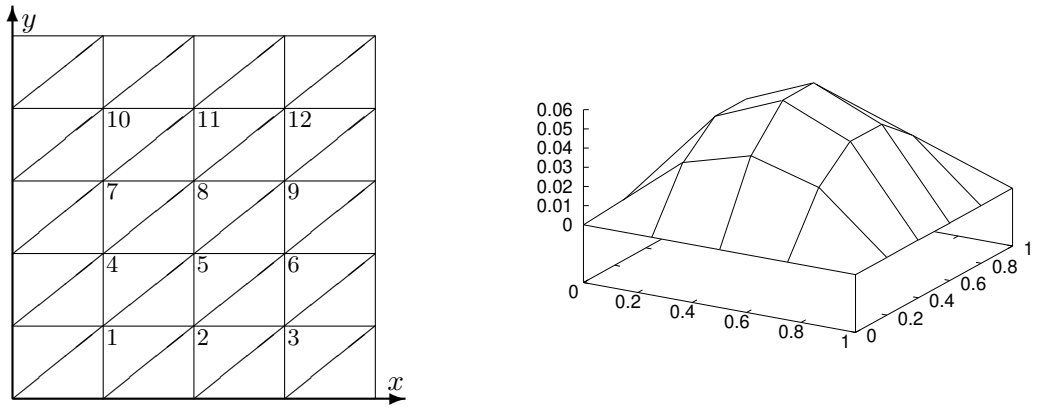


Figure 7.5: Solution on a small rectangular grid

The resulting symmetric matrix **A** with semi-bandwidth 4 is

$$\mathbf{A} = \begin{bmatrix}
82 & -16 & 0 & -25 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
-16 & 82 & -16 & 0 & -25 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
0 & -16 & 82 & 0 & 0 & -25 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
-25 & 0 & 0 & 82 & -16 & 0 & -25 & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & -25 & 0 & -16 & 82 & -16 & 0 & -25 & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & -25 & 0 & -16 & 82 & 0 & 0 & -25 & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & -25 & 0 & 0 & 82 & -16 & 0 & -25 & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & -25 & 0 & -16 & 82 & -16 & 0 & -25 & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & -25 & 0 & -16 & 82 & 0 & 0 & -25 \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -25 & 0 & 0 & 82 & -16 & 0 \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -25 & 0 & -16 & 82 & -16 \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & -25 & 0 & -16 & 82
\end{bmatrix}$$

If we choose the function $f(x, y) = 2\,(x-1)\,x + 2\,(y-1)\,y$ then the vector $\vec{b}$ is given by

$$\vec{b} = -\,(0.695\,,\,0.820\,,\,0.695\,,\,0.855\,,\,0.980\,,\,0.855\,,\,0.855\,,\,0.980\,,\,0.855\,,\,0.695\,,\,0.820\,,\,0.695)^T$$

and we can solve the system of 12 linear equations $\mathbf{A} \cdot \vec{x} = -\vec{b}$ and obtain the solution.

$$\vec{x} = (0.03\,,\ 0.04\,,\ 0.03\,,\ 0.045\,,\ 0.06\,,\ 0.045\,,\ 0.045\,,\ 0.06\,,\ 0.045\,,\ 0.03\,,\ 0.04\,,\ 0.03)^T$$

This leads to the solution in figure 7.5.

If we set $nx = 30$ and $ny = 40$ then we find a system of 1200 linear equations, but the matrix has semi-bandwidth 31. We can still setup and solve the system and arrive at the solution in figure 7.6.

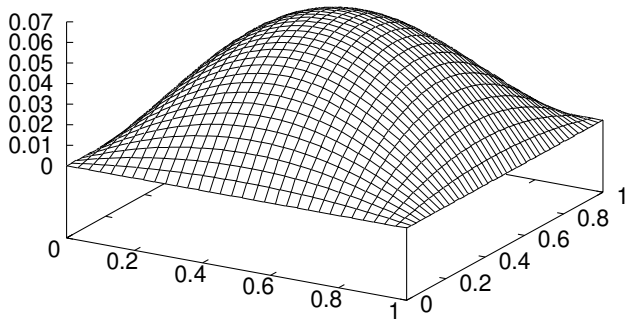


Figure 7.6: Solution on a larger rectangular grid

7.4 FEM code in *Mathematica*

The computations of the previous section can be implemented in *Mathematica* and then the code will be used to solve a problems. We present the code and illustrate it usage by simple examples.

7.4.1 Description of the sample problem

The sample problem to be considered is

$$\begin{aligned} u_{xx} + u_{yy} &= -1 && \text{for} \quad (x,y) \in (0,5) \times (0,4) \\ u(x,y) &= \tfrac{y}{10} && \text{for} \quad x \in (0,5) \text{ and } y \in \{0,4\} \\ \tfrac{\partial u}{\partial \vec{n}} u(x,y) &= -1 && \text{for} \quad x \in \{0,5\} \text{ and } y \in (0,4) \end{aligned}$$

i.e. in equation (7.1) we set

$$a(x,y) = 1 \quad , \quad b(x,y) = 0 \quad , \quad f(x,y) = -1 \quad , \quad g_1(x,y) = \frac{y}{10} \quad \text{and} \quad g_2(x,y) = 1$$

The graph of the solution is shown in figure 7.7. We use the left part to illustrate the code in *Mathematica*.

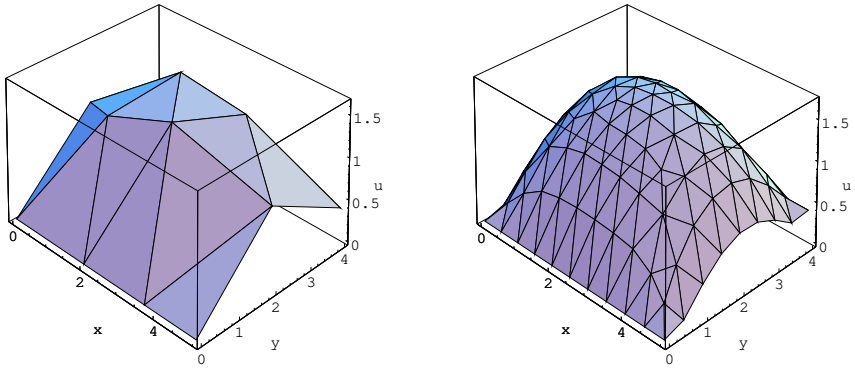


Figure 7.7: Solution of the test problem with few and many triangles

7.4.2 Mesh generation by *EasyMesh*

The program *EasyMesh* can be used to generate meshes for rather general domains. Documentation and source for this program can be found at the WWW site of the author[4]. The program can be compiled on many platforms and the additional program *ShowMesh* will show the resulting mesh, but on X Window systems only.

As a simple example we consider a rectangle $0 < x < 5$ and $0 < y < 4$. We aim for a mesh with very few nodes. We consider a problem with Dirichlet condition at the lower and upper boundary (type 1) and Neumann conditions at the left and right boundary (type 2). The input file `test1.d` does contain all necessary information (see the documentation of *EasyMesh*) and the command `EasyMesh test1` will generate three output files:

`test1.d`	input file with all information on the domain
`test1.n`	output file with data of the nodes
`test1.e`	output file with data of the elements
`test1.s`	output file with data of the line segments

[4] `http://www-dinma.univ.trieste.it/~nirftc/research/easymesh/`

The three output files of *EasyMesh* have to be read by *Mathematica*. The resulting mesh is shown in figure 7.8, including the numbering of nodes and elements. Observe that the numbering of nodes and elements in the output files of *EasyMesh* and in the right part of figure 7.8 differ by one. This is due to that fact that *EasyMesh* starts numbering with 0 while we use 1 as the first number.

```
# this is the file test1.d #
#=========
| POINTS |
=========#
5 # number of points #

# Nodes which define the boundary #
0:   0   0   2   1
1:   5   0   2   1
2:   5   4   2   1
3:   0   4   2   1

# material marker #
4: 2   2    0   1 # material 1 #

#===========
| SEGMENTS |
===========#
4 # Number of segments #

# Boundary segments #
0:   0   1   1
1:   1   2   2
2:   2   3   1
3:   3   0   2
```

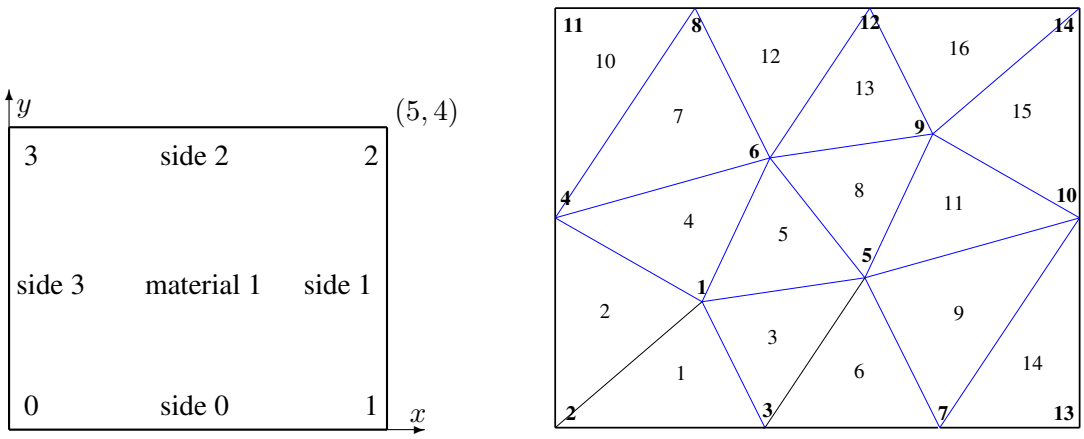


Figure 7.8: Input information for *EasyMesh* and the resulting mesh

7.4.3 Reading the mesh information

We have to read the three output files of *EasyMesh* and store the information about nodes, elements and segments in appropriate *Mathematica* variables. This will be put in one function `ReadMesh[]` to be found

in table 7.1. To be able to understand the code in this table one has to look at the precise format of the output files of *EasyMesh*.

Now read information from files `EasyMesh/test1.*` by

Mathematica

```
{nodes,elements,segments}=ReadMesh["EasyMesh/test1"];
```

and the command `MatrixForm[nodes]` will lead (almost) to the result below. The first line is added to document the format of the variable `nodes`. The marker indicates the type of node. In the input file the Dirichlet boundary was marked as type 1, thus the points 2, 7, 8 11, 12, 13 and 14 have markers 1 . The two points 4 and 10 on the Neumann boundary have markers 2 . The result has to be compared with the nodes in the right part of figure 7.8.

$$\text{nodes} = \begin{bmatrix} \text{number} & x \text{ coordinate} & y \text{ coordinate} & \text{marker} \\ 1 & 1.40 & 1.20 & 0 \\ 2 & 0.00 & 0.00 & 1 \\ 3 & 2.00 & 0.00 & 1 \\ 4 & 0.00 & 2.00 & 2 \\ 5 & 2.95 & 1.43 & 0 \\ 6 & 2.05 & 2.57 & 0 \\ 7 & 3.67 & 0.00 & 1 \\ 8 & 1.33 & 4.00 & 1 \\ 9 & 3.60 & 2.80 & 0 \\ 10 & 5.00 & 2.00 & 2 \\ 11 & 0.00 & 4.00 & 1 \\ 12 & 3.00 & 4.00 & 1 \\ 13 & 5.00 & 0.00 & 1 \\ 14 & 5.00 & 4.00 & 1 \end{bmatrix}$$

The format of the variable `elements` below needs no explanation. Since the rectangle consists of one type

Mathematica

```
ReadMesh[filename_String]:=Module[{instream,n,nodes,elements,segements},

(* read all the nodes *)
instream=OpenRead[StringJoin[filename,".n"]];
n=Read[instream,Number];
nodes=Table[{},{n}];
For[k=1,k<=n,k+=1,
      tmp=Read[instream,{Number,Character,Real,Real,Number}];
      nodes[[k]]=Drop[tmp,{2}];
      nodes[[k,1]]+=1; (* numbering starts at 1 *)
      ];
Close[instream];

(* read the elements *)
instream=OpenRead[StringJoin[filename,".e"]];
n=Read[instream,Number];
elements=Table[{},{n}];
For[k=1,k<=n,k+=1,
      tmp=Read[instream,{Number,Character,Number,Number,Number,
                         Number,Number,Number,Number,Number,Number,
                         Real,Real,Number}];
      elements[[k]]={tmp[[1]]+1,tmp[[3]]+1,tmp[[4]]+1,tmp[[5]]+1,Last[tmp]}
                            (* numbering starts at 1 *)
];
Close[instream];

(* read all segments *)
instream=OpenRead[StringJoin[filename,".s"]];
n=Read[instream,Number];
segments=Table[{},{n}];
For[k=1,k<=n,k+=1,
      tmp=Read[instream,{Number,Character,
                         Number,Number,Number,Number,Number}];
      segments[[k]]=Drop[Drop[tmp,2],{3,4}];
      segments[[k,1]]+=1;
      segments[[k,2]]+=1; (* numbering starts at 1 *)
];
Close[instream];
segments=Select[segments,Last[#]>0&];
(* return all results *)
{nodes,elements,segments}]
```

Table 7.1: *Mathematica* code for `ReadMesh[]`

only the last entry in each line equals 1 .

$$\text{elements} = \begin{bmatrix} \text{number} & \text{corner 1} & \text{corner 2} & \text{corner 3} & \text{material} \\ 1 & 1 & 2 & 3 & 1 \\ 2 & 1 & 4 & 2 & 1 \\ 3 & 1 & 3 & 5 & 1 \\ 4 & 1 & 6 & 4 & 1 \\ 5 & 1 & 5 & 6 & 1 \\ 6 & 5 & 3 & 7 & 1 \\ 7 & 4 & 6 & 8 & 1 \\ 8 & 5 & 9 & 6 & 1 \\ 9 & 5 & 7 & 10 & 1 \\ 10 & 4 & 8 & 11 & 1 \\ 11 & 5 & 10 & 9 & 1 \\ 12 & 6 & 12 & 8 & 1 \\ 13 & 6 & 9 & 12 & 1 \\ 14 & 10 & 7 & 13 & 1 \\ 15 & 9 & 10 & 14 & 1 \\ 16 & 9 & 14 & 12 & 1 \end{bmatrix}$$

The variable `segments` contains information about the boundary segments. The marker indicates Dirichlet (1) or Neumann (2) boundary conditions along the straight line segment.

$$\text{segments} = \begin{bmatrix} \text{node 1} & \text{node 2} & \text{marker} \\ 2 & 3 & 1 \\ 2 & 4 & 2 \\ 3 & 7 & 1 \\ 11 & 4 & 2 \\ 8 & 11 & 1 \\ 7 & 13 & 1 \\ 12 & 8 & 1 \\ 13 & 10 & 2 \\ 10 & 14 & 2 \\ 14 & 12 & 1 \end{bmatrix}$$

Now we have all information needed for the Finite Element Method.

7.4.4 Element and edge contributions

We have to compute contribution due to the integral

$$\iint_{\Delta} \frac{1}{2}\, a\, (\nabla u)^2 + \frac{1}{2}\, b\, u^2 + f \cdot u \; dA$$

for each triangle (see section 7.1.2). This cane be done with the function `ElementContribution[]`, it will generate the element stiffness matrix and also the vector.

Mathematica

```
(* Integration over one triangle: ElementContribution[] *)
ElementContribution[corners_List,aCoeff_Function,bCoeff_Function,
                    fCoeff_Function]:=
Module[{area,a,b,c,M},
a=Flatten[{corners[[2]]-corners[[1]],0}];
b=Flatten[{corners[[3]]-corners[[1]],0}];
c=Cross[a,b];
area=Abs[c[[3]]]/2;
M={
{corners[[3,2]]-corners[[2,2]],corners[[1,2]]-corners[[3,2]],
                               corners[[2,2]]-corners[[1,2]]},
{corners[[2,1]]-corners[[3,1]],corners[[3,1]]-corners[[1,1]],
                               corners[[1,1]]-corners[[2,1]]}};
{(aCoeff[corners[[1,1]],corners[[1,2]]]+aCoeff[corners[[2,1]],corners[[2,2]]] +
  aCoeff[corners[[3,1]],corners[[3,2]]])/(12*area)*
   Transpose[M].M +
 area/3* {{bCoeff[corners[[1,1]],corners[[1,2]]],0,0},
          {0,bCoeff[corners[[2,1]],corners[[2,2]]],0},
          {0,0,bCoeff[corners[[3,1]],corners[[3,2]]]}},
 area/3* {fCoeff[corners[[1,1]],corners[[1,2]]],
          fCoeff[corners[[2,1]],corners[[2,2]]],
          fCoeff[corners[[3,1]],corners[[3,2]]]}
}]
```

As an example we consider a triangle with corners at $(0,0)$, $(1,0)$ and $(1,0)$ with functions $a(x,y)=1$, $b(x,y)=0$ and $f(x,y)=7$.

Mathematica

```
a=Function[{x,y},1];
b=Function[{x,y},0];
f=Function[{x,y},7];
{elemMat,elemVec}=ElementContribution[{{0,0},{1,0},{0,1}},a,b,f]
.

        1     1       1   1          1      1
{{{1, -(-), -(-)}, {-(-), -, 0},  {-(-), 0, -}},
        2     2       2   2          2      2

  7  7  7
 {-, -, -}}
  6  6  6
```

The matrix and vector are given by

$$\begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & \frac{1}{2} & 0 \\ -\frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix} \quad \text{and} \quad \begin{pmatrix} \frac{7}{6} \\ \frac{7}{6} \\ \frac{7}{6} \end{pmatrix}$$

The integration over a boundary segment is done with similar code. The Gauss intgeration in section 7.1.3 leads to the code below.

Mathematica

```
ElementContributionEdge[corners_List,g_Function]:=
```

```
Module[{p1,p2,m,w},
  m=(corners[[1]]+corners[[2]])/2;
  p1=m+(corners[[1]]-corners[[2]])/Sqrt[3]/2;
  p2=m-(corners[[1]]-corners[[2]])/Sqrt[3]/2;
  w=N[(1-1/Sqrt[3])/2];
Sqrt[(corners[[1,1]]-corners[[2,1]])^2+(corners[[1,2]]-corners[[2,2]])^2]/2*
    {g[p1[[1]],p1[[2]]]*(1-w)+g[p2[[1]],p2[[2]]]*w,
     g[p1[[1]],p1[[2]]]*w    +g[p2[[1]],p2[[2]] ]*(1-w)}
]
```

7.4.5 Assembling the equations

Now that the contributions of isolated triangles and edges can be computed we have to generate the system of linear equations to be solved. At first we determine the actual degrees of freedom and number them.

Mathematica

```
(* Numbering of the degrees of freedom: FindDOF[] *)
FindDOF[nodes_]:=Module[{counter,Node2Degree},
counter=0;
Node2Degree=Table[0,{Length[nodes]}];
For[k=1,k<=Length[nodes],k+=1,
        If[nodes[[k,4]]==0,counter+=1;Node2Degree[[k]]=counter;,];
        If[nodes[[k,4]]==2,counter+=1;Node2Degree[[k]]=counter;,]
];
{counter,Node2Degree}];
```

For our example problem we obtain

Mathematica

```
{dof,Node2Degree} = FindDOF[nodes]
.
{6,{1,0,0,2,3,4,0,0,5,6,0,0,0,0}}
```

Thus in our sample problem we have only 6 nodes where the value of the function has to be computed, namely the nodes 1, 4, 5, 6, 9 and 10. This information gives the correct size of the global stiffness matrix, a 6×6 matrix for our example.

For each element the element stiffness matrix and vector have to be added to the global stiffness matrix and vector at the correct location, given in the vector `Node2Degree`.

Mathematica

```
insertElement[element_List,aCoeff_Function,bCoeff_Function,fCoeff_Function,
    gCoeff_Function,nodes_List,Node2Degree_List] :=
Block[{elMat,elVec,corners,dofs,tMat},
corners={{nodes[[element[[1]],2]],nodes[[element[[1]],3]]},
         {nodes[[element[[2]],2]],nodes[[element[[2]],3]]},
         {nodes[[element[[3]],2]],nodes[[element[[3]],3]]}};
{elMat,elVec}=ElementContribution[corners,aCoeff,bCoeff,fCoeff];
dofs={Node2Degree[[element[[1]]]],Node2Degree[[element[[2]]]],
                                  Node2Degree[[element[[3]]]]};
For[k1=1,k1<=3,k1+=1,
   If[dofs[[k1]]>0,
     For[k2=1,k2<=3,k2+=1,
        If[dofs[[k2]]>0,
           Aglobal[[dofs[[k1]],dofs[[k2]]]]+=elMat[[k1,k2]];,
        (*else*)
           bglobal[[dofs[[k1]]]]+=
           (elMat[[k1,k2]]*gCoeff[corners[[k2,1]],corners[[k2,2]] ]);
```

```
        ](*if k2*)
        ];(*for k2*)
      bglobal[[dofs[[k1]]]]+= elVec[[k1]];
      ](*if k1*);
 ](*for k1*);
 ](*block*)
```

The same has to be done for each segment on the boundary.

Mathematica

```
insertElementEdge[edge_List,g_Function,nodes_List,Node2Degree_List] :=
Block[{elVec,corners,dofs,k},
corners={{nodes[[edge[[1]],2]],nodes[[edge[[1]],3]]},
         {nodes[[edge[[2]],2]],nodes[[edge[[2]],3]]}};
elVec=ElementContributionEdge[corners,g];
dofs={Node2Degree[[edge[[1]]]], Node2Degree[[edge[[2]]]]};
For[k=1,k<=2,k+=1,
   If[dofs[[k]]>0,bglobal[[dofs[[k]]]] -= elVec[[k]] ];
](*for k*);
](*block*)
```

Now all necessary tools are available and we can attempt to solve the problem.

7.4.6 Solving the equations

All of the above routines are now combined a single command to solve a given boundary value problem.

Mathematica

```
FEMSolve[a_Function,b_Function,f_Function,g1_Function,g2_Function,
         nodes_List,elements_List,segments_List]:=
Block[{uglobal,dof,Node2Degree},
{dof , Node2Degree} = FindDOF[nodes];
Aglobal=Table[0,{dof},{dof}];
bglobal=Table[0,{dof}];
For[k=1,k<=Length[elements],k+=1,
   insertElement[Take[elements[[k]],{2,4}],a,b,f,g1,nodes,Node2Degree]] ;
For[k=1,k<=Length[segments],k+=1,
   If[segments[[k,3]]==2,
         insertElementEdge[Take[segments[[k]],2],g2,nodes,Node2Degree]]];
(* solve the system of linesr equations *)
uglobal=LinearSolve[Aglobal,-bglobal];
Table[{nodes[[k,2]],nodes[[k,3]],
      Which[Node2Degree[[k]]>0,uglobal[[Node2Degree[[k]]]],
           True,g1[nodes[[k,2]],nodes[[k,3]]]]},
     {k,Length[nodes]}]
]
```

Now the result can be generated with very little code, thanks to the above functions.

Mathematica

```
a=Function[{x,y},1];
b=Function[{x,y},0];
f=Function[{x,y},-1];
gDirichlet=Function[{x,y},y/10];
gNeumann=Function[{x,y},-1];

points=FEMSolve[a,b,f,gDirichlet,gNeumann,nodes,elements,segments]
;
```

```
{{1.4,1.200,1.290},{0.,0.,0.},{2.,0.,0.},{0.,2.,0.948}, {2.952,1.429,1.539},
{2.048,2.571,1.654},{3.667,0.,0.},{1.333,4.,0.4},{3.6,2.800,1.459},
{5.,2.,0.948},{0.,4.,0.4},{3.,4.,0.4},{5.,0.,0.},{5.,4.,0.4}}
```

The variable `points` contains the three components of all nodes of the mesh.

All of the above code can be put in one file `BVP2.m` and will form a *Mathematica* package. Only the two commands `FEMSolve[]` and `ReadMesh[]` have to be exported.

7.4.7 Visualization

With the information in `points` and `elements` the graph of the solution can be generated by the code below and leads to figure 7.7 on page 160.

Mathematica

```
g=FEMSurface[points,elements];
solution=Show[g,
        PlotRange ->All,
        Axes->True,
        AxesLabel->{"x","y","u"},
 AspectRatio ->1,
 ViewPoint->{4,-4,2}];
```

The package `BVP2.m` also contains command to visualizs the mesh and create level curves on the surface describing the solution. The example in figure 7.9 and the code below speaks for itself.

Mathematica

```
Show[MeshGraphics[nodes,elements]];

gsurf= FEMSurfaceNoMesh[points,elements];
glevel=LevelGraphics3D[points,elements,Table[le,{le,0,2,0.1}]];
gg4=Show[{gsurf,glevel},
        PlotRange ->All,
        Axes->True,
        AxesLabel->{"x","y","u"},
        AspectRatio ->1,
        ViewPoint->{4,-4,4}]
```

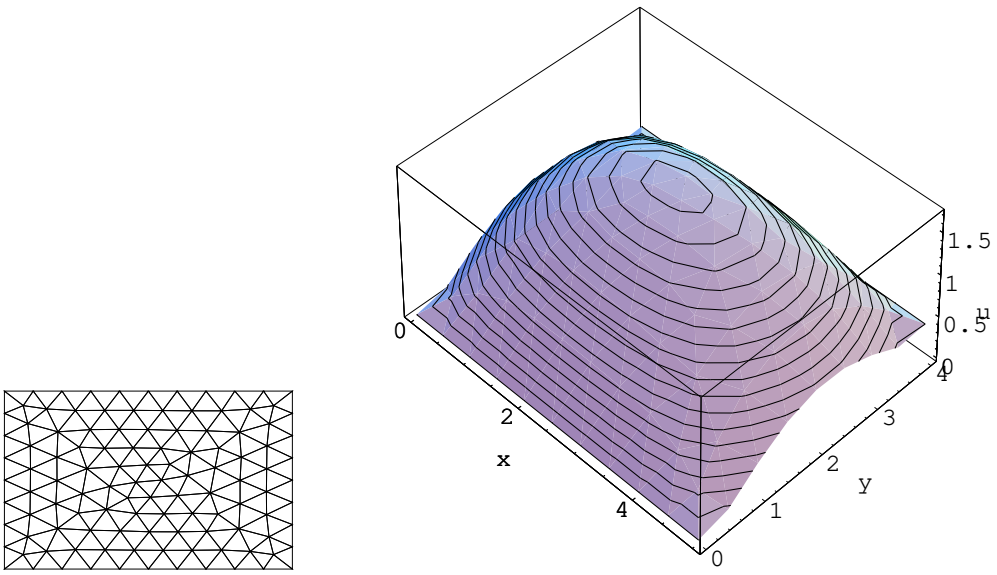


Figure 7.9: Visualization of the mesh and the solution of the test problem

7.5 Exercises

• Exercise 7–1:
According to equation (7.2) on page 148 the element stiffness matrix $\mathbf{A}_\Delta$ is given by

$$\mathbf{A}_\Delta = \frac{1}{4\,A^2}\,\mathbf{M}^T \cdot \mathbf{M} = \frac{1}{4\,A^2} \begin{bmatrix} (y_3 - y_2) & (x_2 - x_3) \\ (y_1 - y_3) & (x_3 - x_1) \\ (y_2 - y_1) & (x_1 - x_2) \end{bmatrix} \cdot \begin{bmatrix} (y_3 - y_2) & (y_1 - y_3) & (y_2 - y_1) \\ (x_2 - x_3) & (x_3 - x_1) & (x_1 - x_2) \end{bmatrix}$$

Consider the special case $x_1 = y_2 = 0$ and compute $\mathbf{A}_\Delta$. Then verify the following:

(a) The matrix is symmetric and all diagonal elements in $\mathbf{A}_\Delta$ are strictly positive.

(b) The sum of the entries along each row is zero. Thus the matrix is diagonally dominant (see section 11.2).

(c) Use the identity

$$\langle \mathbf{M} \cdot \vec{u}\,,\, \mathbf{M} \cdot \vec{u} \rangle = \|\mathbf{M} \cdot \vec{u}\|^2$$

to verify that $\mathbf{A}_\Delta$ is positive semidefinite. The above expression is zero if and only if $u_1 = u_2 = u_3$.

• Exercise 7–2:
Use the problem in section 7.3 with increasing values for $nx = ny$ to examine the error as a function of h. Use the exact solution $u(x, y) = -5\,\pi^2\,\sin(\pi\,x) \cdot \sin(2\,\pi\,y)$ of the equation $\Delta u = \sin(\pi\,x) \cdot \sin(2\,\pi\,y)$. Observe the average of the squared error and the computation time. Also keep track of the time needed to set up the system of equations and to solve the system. The basic code should be provided by the instructor.

Chapter 8

Some Applications

Most of the applications in this section are solve with help of the *Octave* package `FEMoctave`, available at [www:sha]. All problems can be solved with the help of the `MATLAB` toolbox.

8.1 Computing a capacitance

8.1.1 State the problem

We examine a circular plate capacitance as shown in figure 8.1. Based on the radial symmetry one should be able to consider a two dimensional section only for the computations.

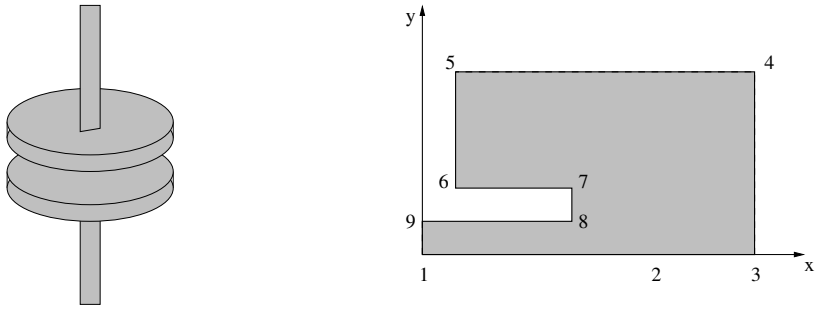


Figure 8.1: The capacitance and the section used for the modeling

Consider the voltage u as unknown. On the upper conductor we assume $u = 1$ and on the lower conductor $u = -1$. Based on the symmetry we consider a section only and use $u = 0$ in the plane centered between the conductors. We use the Laplace operator in cylindrical coordinates (see Appendix A.2.4). Based on the results listed in Appendix A.6 the following boundary value problem has to be solved.

$$\begin{aligned} \operatorname{div}(x\ \operatorname{grad} u(x,y)) &= 0 && \text{in domain} \\ u(x,0) &= 0 && \text{along edge } y = 0 \\ u(x,y) &= 1 && \text{along edges of upper conductor} \\ \tfrac{\partial\, u(x,y)}{\partial n} &= 0 && \text{on remaining boundary} \end{aligned} \tag{8.1}$$

We assume that the domain is embedded in the rectangle $0 \le x \le R$ and $0 \le y \le H$. The lower edge of the conductor is at $y = h$ and $0 \le x \le r$. If $h \ll r$ we expect the gradient of u to be $1/h$ between the plates and zero away from the plates. Thus

$$\text{flux} = \iint_{\text{disk}} \vec{n} \cdot \operatorname{grad} u \; dA = 2\,\pi \int_0^R x\, \frac{\partial\, u}{\partial y}\, dx \approx 2\,\pi \int_0^r x\, \frac{1}{h}\, dx = \frac{\pi\, r^2}{h}$$

Because the electric field will not be homogeneous around the boundaries of the disk we expect deviations from the result of an idealized circular disk. With the divergence theorem (Appendix A.3) and a physical argument one can verify that the flux trough the midplane is proportional to the capacitance.

By applying the following steps we will compute the capacitance by analyzing the solution of a boundary value problem. We apply the following steps:

1. Create a mesh for the domain in question.
2. Define parameters and boundary conditions.
3. Solve the partial differential equation and visualize the solution.
4. Compute the flux through the midplane as an integral to determine the capacitance.

8.1.2 Create the mesh

According to figure 8.1 we create a mesh with the following data.

$h = 0.2$	distance between midplane and lower edge of capacitance
$r = 1.0$	radius of disk of the capacitance
$H = 0.5$	height of the enclosing rectangle
$R = 2.5$	radius of the enclosing rectangle

As input for the mesh generating code `triangle` (see [www:triangle]) we need

- the coordinates of the corner points, numbered according to figure 8.1
- a list of all the connecting edges and the type of boundary conditions to be used
- information of the desired area of the triangles to be generated

We use two different sizes of the triangles since we want to have a finer mesh between the plates, expecting large variations in the solution. Below find the listing of the file `capacitance.poly` with this information. The numbering of the nodes is also visible in figure 8.1.

```
# nodes
9 2 0 1
1 0 0 1
2 2.0 0 1
3 2.5 0 1
4 2.5 0.5 2
5 0.1 0.5 1
6 0.1 0.3 1
7 1.0 0.3 1
8 1.0 0.2 1
9 0.0 0.2 1
# segments
10 1
1 1 2 1
2 2 3 1
3 3 4 2
4 4 5 2
5 5 6 1
6 6 7 1
7 7 8 1
```

```
8 8 9 1
9 9 1 2
10 2 7 0
# holes
0
# area markers
2
1 0.1 0.01 0 0.0002
2 2.1 0.01 0 0.001
```

With the above we now use the two commands `triangle` and `CuthillMcKee`.

```
triangle -pqa capacitance.poly
CuthillMcKee -v -s0 capacitance.1
```

The first line generates the mesh. Since we need a small bandwidth of the resulting global stiffness matrix we renumber the nodes with the help of the Cuthill–McKee algorithm. More information on this algorithm is given in section 11.4. Now we may use *Octave* to load and display the generated mesh. Find the result in figure 8.2. The mesh consists of 2189 nodes, forming 4036 triangles. The resulting semi-bandwidth is smaller that 43.

Octave

```
clear *
[nodes,elem,edges]=ReadMeshTriangle("./capacitance.1");
ShowMesh(nodes,elem)
```

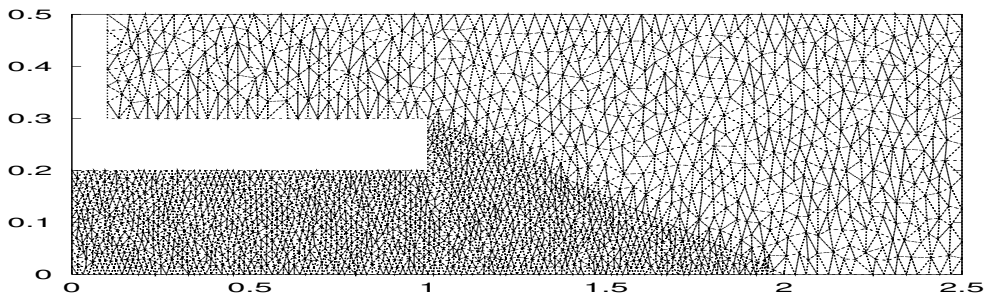


Figure 8.2: A mesh on the domain

8.1.3 Creating the functions for *Octave*

To solve the differential equation (8.1) we need a definition of the coefficient function and the Dirichlet boundary function.

Octave

```
global h=0.2;   r=1;   R=2.5;

function res = aF(xy)
  res=xy(:,1);
endfunction

function res = volt(xy)
  global h;
  [n,m]=size(xy);
  res=zeros(n,1);
```

```
  for k=1:n
    if (xy(k,2)<h/2) res(k)=0;
    else res(k)=1;endif
  endfor
endfunction
```

8.1.4 Solve the system and show the solution

Now we can set up and solve the system of linear equations. We end up with a system for 1937 unknowns and a semi-bandwidth of 41.

Octave

```
[A,b,n2d] = FEMEquation(nodes,elem,edges,'aF',0,0,'volt',0);
sprintf("The stiffnes matrix has size %i with semi-bandwidth %i",size(A))
u = FEMSolveSym(nodes,A,b,n2d,'volt');
```

Now we can generate a plot of the voltage $u(x, y)$ and its level curves by the following commands. Find the results in figures 8.3 and 8.4.

Octave

```
figure(1);
gset xrange [*:*]
gset yrange [*:*]
ShowLevelCurves(nodes,elem,u,linspace(0,1.1,12));
figure(2);
ShowSolution(nodes,elem,u);
```

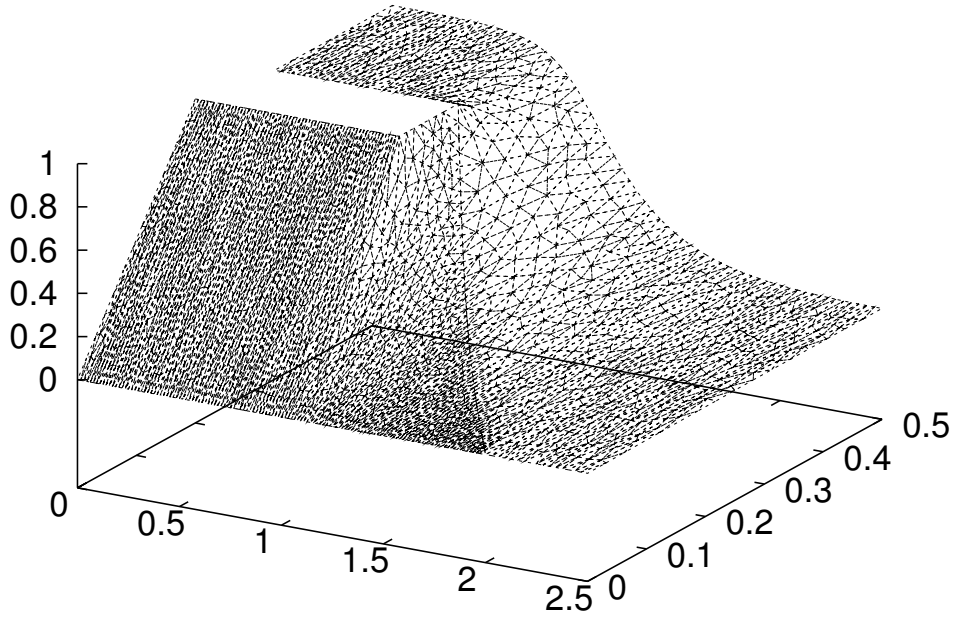


Figure 8.3: The voltage within the capacitance

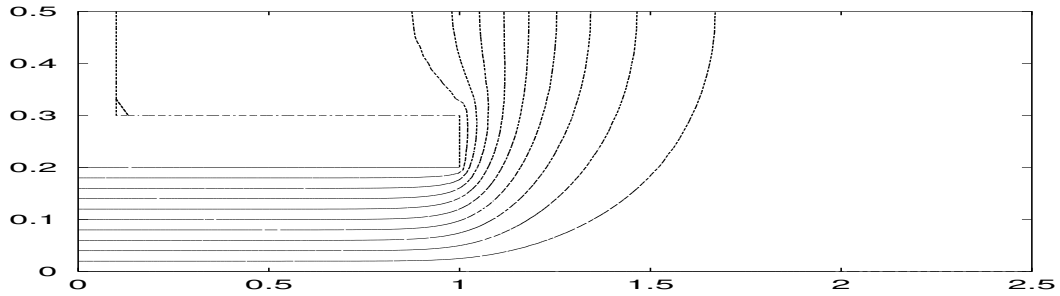


Figure 8.4: Level curves for the voltage in the capacitance

8.1.5 Compute the capacitance

It remains to compute the flux through the midplane. For this we first compute the gradient of the voltage u along the line $y = 0$. Find the plot of the normal component in figure 8.5. The graph confirms that between the plates the gradient is approximately $1/h = 1/0.2 = 5$ and vanishes away from the plate.

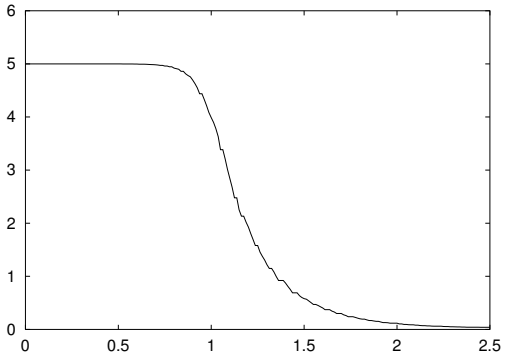


Figure 8.5: Normal component of the gradient along midplane

Then a simple trapezoidal rule is used to determine the flux accross the midplane with the integral.

$$\text{flux} = \iint_{\text{disk}} \vec{n} \cdot \operatorname{grad} u \, dA = 2\,\pi \int_0^R x \, \frac{\partial\, u}{\partial y} \, dx$$

For the chosen values of h, H, r and R we obtain a factor of 1.52 between result of the boundary value problem and the idealized approximation $\pi\, r^2/h$.

Octave

```
N=201;   # number of grid points to use for numerical integration
x=linspace(0,R,N);
y=0.0*ones(1,N);
[uVal,grad]=FEMValue([x;y]',nodes,elem,u);

plot(x,grad(:,2))

# trapezoidal integration and normalization
flux=2*pi*(x*grad(:,2)-x(1)*grad(1,2)/2-x(N)*grad(N,2)/2)*R/(N-1)*(h/(pi*r**2))
```

8.2 Heat conduction on a circuit board

Consider an elementary circuit board with one heat generating chip on the board, as shown in figure 8.6. As the chip will add thermal energy to the board it will heat up and reach a stable heat distribution. We use the following assumptions:

- The integrated circuit is the only heat generating device and the heat generated per time and area is assumed to be constant across the chip.
- The board will dissipate energy into the surrounding air with a rate proportional to the difference of the temperature on the board and the air.
- The temperature will be independent on the height z.
- We neglect the heat flux across the lateral boundary of the chip.

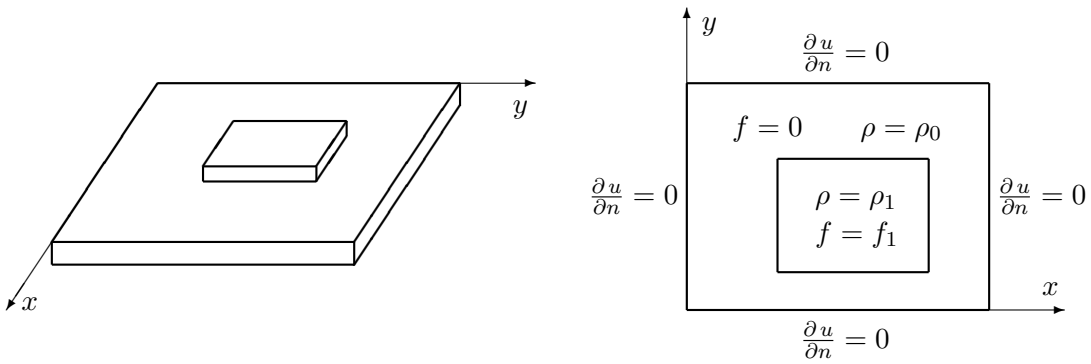


Figure 8.6: A circuit board

The dynamic heat equation (4.2) on page 65 has to be supplemented by a term representing the dissipation of heat proportional to the difference u of the temperature T on the board and the surrounding temperature. We arrive at the PDE

$$\rho \frac{\partial u}{\partial t} = c \ \mathrm{div}\,(\nabla u) - b\,u + f$$

with Neumann boundary conditions.

For the sample computations we consider the board with coordinates $0 \le x \le 4$ and $0 \le y \le 3$ and the IC is placed at $1.5 \le x \le 3$ and $1 \le y \le 2.5$. To generate a mesh we use the program `triangle` with the input file `board.poly` shown below.

```
# nodes
8 2   0   1
1 0   0   1
2 4   0   1
3 4   3   1
4 0   3   1
5 1.5 1   0
6 3   1   0
7 3   2.5 0
8 1.5 2.5 0
# segments
8 1
```

```
1 1 2 2
2 2 3 2
3 3 4 2
4 4 1 2
5 5 6 0
6 6 7 0
7 7 8 0
8 8 5 0
# holes
0
```

Then we create the mesh and renumber the nodes by

```
triangle -pqa0.01 board.poly
CuthillMcKee -v -s0 board.1
```

We arrive at a mesh with 991 nodes and a semi-bandwidth of 47.

8.2.1 The static situation

The static temperature distribution $u(x, y)$ has to solve the PDE

$$\begin{aligned} -c\ \mathrm{div}\,(\nabla u) + b\,u &= -f = \begin{cases} -f_0 & \text{in } \Omega \\ 0 & \end{cases} \\ \frac{\partial u}{\partial n} &= 0 \qquad\qquad \text{on } \Gamma \end{aligned}$$

For sake of simplicity we solve the problem with $c = b = f_0 = 1$. For real world problems those constants have to be determined using physical data.

First we provide code to compute the heating function $f(x, y)$.

Octave

```
function res=HeatFunction(xy)
  res=-ones(size(xy)(1),1);
  select=(xy(:,1)>1.5).*(xy(:,1)<3.0).*(xy(:,2)>1.0).*(xy(:,2)<2.5);
  res=res.*select;
endfunction
```

Then we read the mesh information, display the mesh and verify the correct coding for the function f .

Octave

```
% read the mesh
[nodes,elem,edges]=ReadMeshTriangle("./board.1");
% verify the mesh
ShowMesh(nodes,elem)
% disp("Hit RETURN");pause();
% verify the coding of the heat function
ShowSolution(nodes,elem,-HeatFunction(nodes));
```

Now the boundary value problem can be solved and the graph of the solution displayed, leading to the solution in figure 8.7.

Octave

```
[A,b,n2d] = FEMEquation(nodes,elem,edges,1,1,'HeatFunction',0,0);

printf("The stiffnes matrix has size %i with semi-bandwidth %i\n",size(A));
u = FEMSolveSym(nodes,A,b,n2d,0);
printf("The values of the solution vary between %f and %f\n",min(u),max(u));
```

```
% show the graph of the solution
figure(1);
ShowSolution(nodes,elem,u);
% show the level curves of the solution
figure(2);
ShowLevelCurves(nodes,elem,u,linspace(0,max(u),11));
```

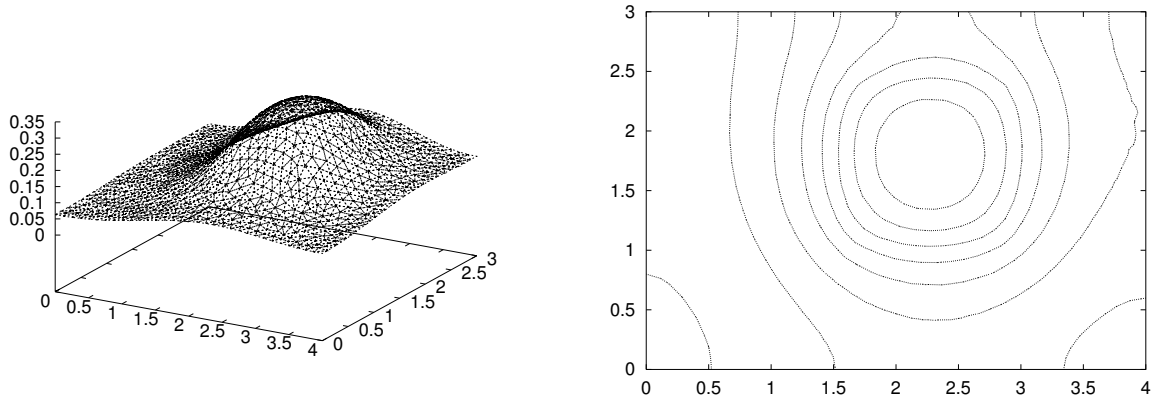


Figure 8.7: The static temperature and level curves for the circuit board

The output of the code implies that for the temperature u we have $0.063 \leq u \leq 0.344$. This implies that the temperature difference on the IC is by a factor of 6 larger than at the edges of the board.

It is interesting to observe how the temperature u depends on the values of the constant b.

8.2.2 The dynamic situation

We assume that the circuit board starts out with initial temperature $u_0 = 0$. The dynamic temperature distribution $u(t, x, y)$ has to solve the PDE

$$\begin{aligned} \rho\,\dot{u} - c\ \mathrm{div}\,(\nabla u) + b\,u &= -f = \begin{cases} -f_0 \\ 0 \end{cases} && \text{for } (t,x,y) \in \mathbb{R}_+ \times \Omega \\ \frac{\partial u}{\partial n} &= 0 && \text{for } (t,x,y) \in \mathbb{R}_+ \times \Gamma \\ u(0,x,y) &= 0 && \text{for } (x,y) \in \Omega \end{aligned}$$

For sake of simplicity we solve the problem with $c = b = f_0 = 1$. For real world problems those constants have to be determined.

Using a FEM discretization for the space variable the above is transformed into

$$\mathbf{M} \cdot \dot{\vec{u}}(t) + \mathbf{A} \cdot \vec{u}(t) = \vec{f}(t) \quad \text{with} \quad \vec{u}(0) = \vec{0}$$

Using a fully implicit approximation scheme with time step Δt as presented in section 5.4.5 we find the time step

$$\begin{aligned} \mathbf{M}\ (\vec{u}(t+\Delta t) - \vec{u}(t)) &= \Delta t\ \left(-\mathbf{A} \cdot \vec{u}(t+\Delta t) + \vec{f}(t+\Delta t)\right) \\ (\mathbf{M} + \Delta t\ \mathbf{A}) \cdot \vec{u}(t+\Delta t) &= \mathbf{M} \cdot \vec{u}(t) + \Delta t \cdot \vec{f}(t+\Delta t) \end{aligned} \tag{8.2}$$

Since a fully implicit scheme is unconditionally stable there are no restrictions on the size of the time step Δt .

We used the same values for the constants as in the static example. The density ρ of the material was assumed to be $\rho = 1$ on the board and $\rho = 2$ on the IC. First we have to define the basic functions for `FEMoctave`.

Octave

```
[nodes,elem,edges]=ReadMeshTriangle("./board.1"); % read the mesh
nn=size(nodes)(1);

function res=rho(xy)
  res=ones(size(xy)(1),1);
  select=(xy(:,1)>1.5).*(xy(:,1)<3.0).*(xy(:,2)>1.0).*(xy(:,2)<2.5);
  res=res+res.*select;
endfunction
```

Then the diagonal mass matrix $\mathbf{M}$ is created as the RHS of the correct boundary value problem[1]. Then the matrix $\mathbf{A}$ and the constant vector $\vec{f}$ are determined.

Octave

```
% determine the mass matrix M
[A,M,n2d] = FEMEquation(nodes,elem,edges,1,1,'rho',0,0);
% determine matrix A and vector b
[A,b,n2d] = FEMEquation(nodes,elem,edges,1,1,'HeatFunction',0,0);
```

To perform the time steps in equation (8.2) we choose the size of the time step Δt and then compute the matrix for the linear system to be solved and its Cholesky factorization. The used algorithm is based on the results in section 11.2 (page 262).

$$\mathbf{M} + \Delta t \cdot \mathbf{A} = \mathbf{R}^T \cdot \mathbf{D} \cdot \mathbf{R}$$

This leads to efficient code to solve the resulting systems of linear equations.

Octave

```
% time steps setup
uold=zeros(size(b)); % initialize the vectors to contain the solution.
unew=uold;
dt=0.1;
% form the new matrix M + dt*A
A *= dt;
A(:,1) += M;
R=SBFactor(A);  % determine the Cholesky decomposition
```

Now we can compute the graphs of the solutions at different time levels and create a poor man's animation on the screen. The result is shown in figure 8.8.

Octave

```
gset zrange [0:0.35]
t=0;
for kk=0:50
  eval(sprintf(...
     "gset title \" solution at time t = %1.2f, min(u)=%0.4f, max(u)=%0.3f \""...
     ,t,min(unew),max(unew)));
  ShowSolution(nodes,elem,unew);   % show the solution
  unew=SBBacksub(R,M.*uold-dt*b);  % solve one time step
  uold=unew; t +=dt;
  usleep(0.5e6)
endfor
gset zrange [*:*]
```

In figure 8.8 we observe that the solution barely changes any more for larger values of t. Most of the changes occur for small times t. To examine the solutions we choose a smaller time step and show the level curves of the solutions, where the different curves indicate a difference of 0.02 in the values of u. The code

[1] This author is fully aware that he should write code to compute the contribution separately.

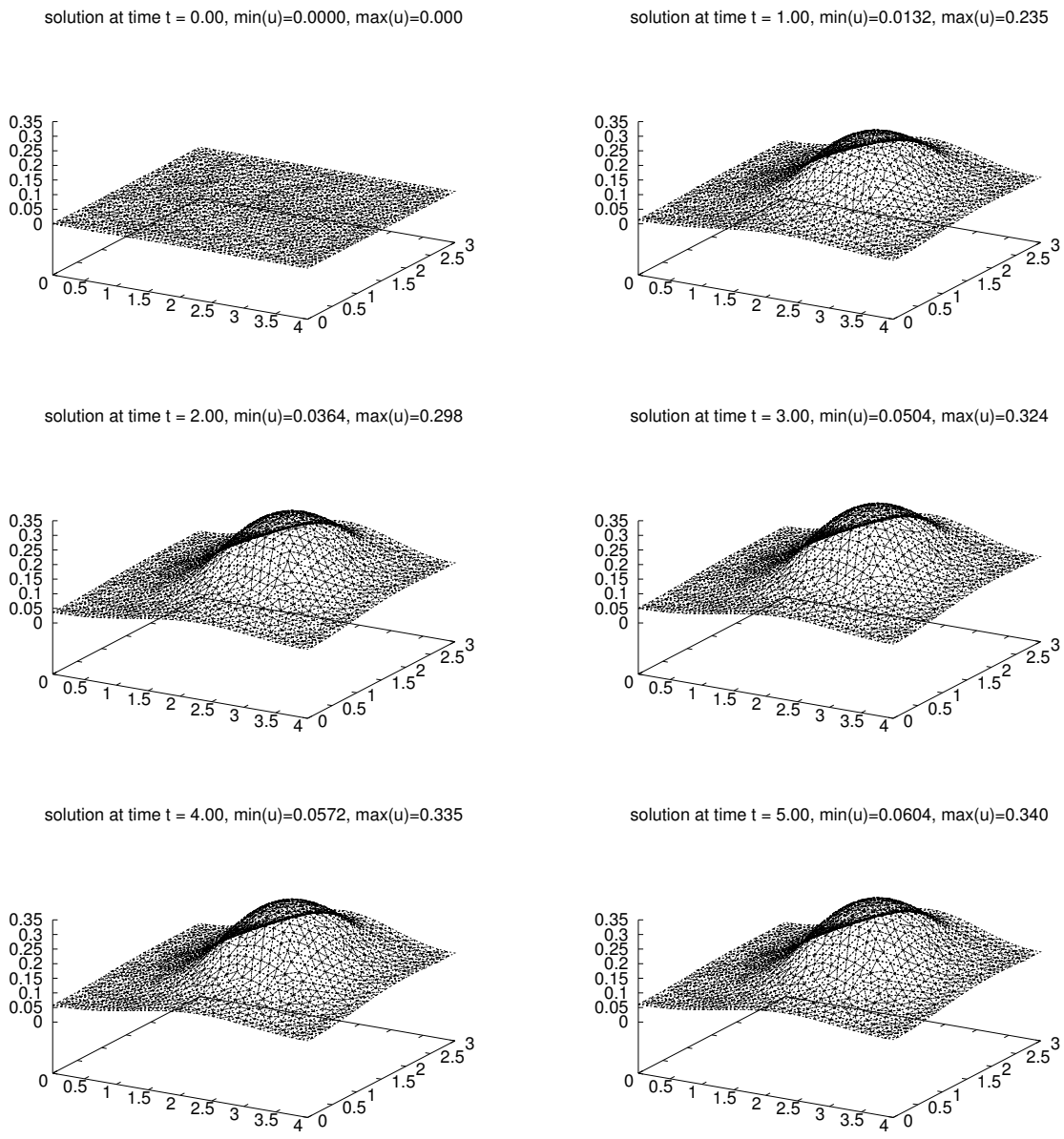


Figure 8.8: The temperatures for the dynamic solution on a circuit board

below will create the results in figure 8.9. The coding is slightly simplified, since we have no Dirichlet boundary conditions. Thus the value of the function at each node correspond to a degree of freedom. There is no need to compute the solutions at the Dirichlet nodes.

Octave

```
% time steps setup
uold=zeros(size(b));
unew=uold;
dt=0.02;

% form the new matrix M + dt*A
A *= dt;
A(:,1) += M;
R=SBFactor(A);  % determine the Cholesky decomposition

gset zrange [0:0.35]
gset xrange [0:4]
gset yrange [0:3]
t=0;
for kk=0:50
  eval(sprintf(...
     "gset title \" solution at time t = %1.2f, min(u)=%0.4f, max(u)=%0.3f \""...
     ,t,min(unew),max(unew)));
  ShowLevelCurves(nodes,elem,unew,linspace(0,0.4,21));   % show the solution
  unew=SBBacksub(R,M.*uold-dt*b);  % solve one time step
  uold=unew; t +=dt;
  usleep(0.9e6);
endfor
gset zrange [*:*]
```

Obviously the above computations are easy to repeat with different values for the parameters and physical constants. This allows to examine the effects of possible modifications to the setup, without building a new device. It is up to the user to draw the correct conclusions from the results of such simulations.

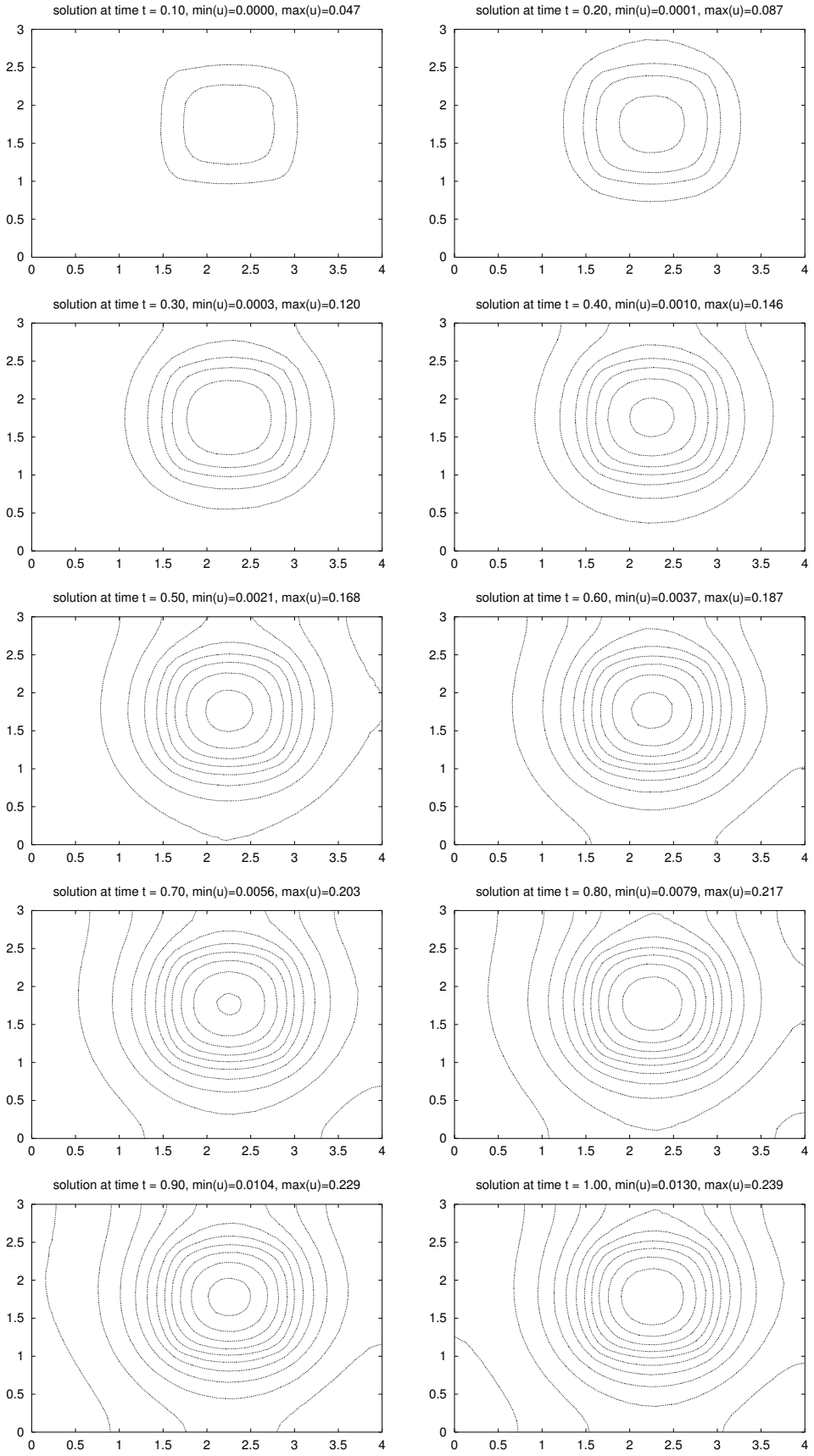


Figure 8.9: The level curves for the temperatures on a circuit board

8.3 Torsion of a shaft

In section 9.4 on page 220 the problem of torsion of shaft with constant cross is considered. Assume that the shaft extends in z–direction and the cross section is given by $\Omega \subset \mathbb{R}^2$. The domain has to be placed such that its center of gravity is at the origin. We denote the boundary by Γ and $\vec{n}$ is the outer unit normal. Then we have to solve the following boundary value problem for the warp function ϕ.

$$\begin{aligned} \operatorname{div}(\nabla\phi) &= 0 && \text{in } \Omega \subset \mathbb{R}^2 \\ \nabla\phi \cdot \vec{n} &= \vec{n} \cdot \begin{pmatrix} y \\ -x \end{pmatrix} && \text{on } \Gamma = \partial\Omega \end{aligned} \tag{8.3}$$

Then the **torsional rigidity** J is given by the integral

$$J = \iint_{\Omega} x^2 + y^2 + x\,\frac{\partial\,\phi}{\partial y} - y\,\frac{\partial\,\phi}{\partial x}\,dA \tag{8.4}$$

and thus for a shaft of length L the total change of angle β caused by a torque T is determined by

$$\beta = L \cdot \alpha = \frac{2\,(1+\nu)}{J\,E}\,L \cdot T$$

The resulting normal stresses are 0 and

$$\sigma = \pm\frac{T}{J}\sqrt{\left(x + \frac{\partial\,\phi}{\partial y}\right)^2 + \left(-y + \frac{\partial\,\phi}{\partial x}\right)^2} \tag{8.5}$$

This result is a copy of equation (9.8) on page 225. If we apply a standard torque of $J = 1$ we can then determine the stress across the section and also the maximal stress. This information may be useful for applications.

8.3.1 Torsional rigidity of a square

As a first example we consider a square section with unit area. The mesh is generated by the code below. One point of the mesh has to be declared to be a Dirichlet node since the solution of the boundary value problem (8.3) is only determined up to an additive constant. Thus we have to give the value of ϕ at one point to find a unique solution.

Octave

```
xy=[-0.5,-0.5,2; 0.5,-0.5,2; 0.5,0.5,2; -0.5,0.5,2];
CreateMeshTriangleQ("square",xy,0.0005);
[nodes,elem,edges]=ReadMeshTriangle("square.1");
nodes(1,3)=1;  % mark one point as Dirichlet node
```

The boundary Γ of the square consists of four section and we have to define a function to compute the RHS of the boundary condition in (8.3).

Octave

```
function res = gN(xy)
  [n,m]=size(xy);
  res=zeros(n,1);
  for kk=1:n
    if (xy(kk,2)>abs(xy(kk,1))) res(kk)=-xy(kk,1);
    elseif (xy(kk,1)>abs(xy(kk,2))) res(kk)=xy(kk,2);
    elseif (xy(kk,2)<-abs(xy(kk,1))) res(kk)=xy(kk,1);
    elseif (xy(kk,1)<-abs(xy(kk,2))) res(kk)=-xy(kk,2);
    endif
  endfor
endfunction
```

Now the PDE is easily solved and a graph of the warp function is shown in figure 8.10.

Octave

```
[A,b,n2d] = FEMEquation(nodes,elem,edges,1,0,0,0,'gN');
printf("The stiffnes matrix has size %i with semi-bandwidth %i\n",size(A));
u = FEMSolveSym(nodes,A,b,n2d,0);

figure(1);
ShowSolution(nodes,elem,u);
figure(2);
ShowLevelCurves(nodes,elem,u,linspace(min(u),max(u),11));
```

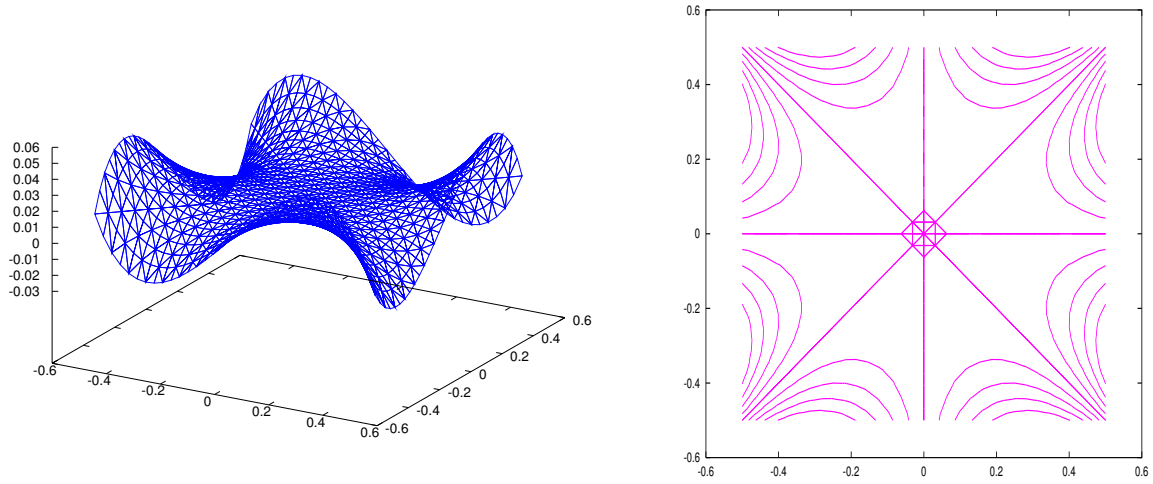


Figure 8.10: The warp function and its level curves for a square

To compute the rigidity J we use (8.4), thus we need to compute the gradient of the warp function ϕ and then find J as an integral over the domain.

$$J = \iint_{\Omega} x^2 + y^2 + x\,\frac{\partial\,\phi}{\partial y} - y\,\frac{\partial\,\phi}{\partial x}\,dA$$

Octave

```
grad=FEMGradient(nodes,elem,u);
f=nodes(:,1).^2+nodes(:,2).^2 +nodes(:,1).*grad(:,2)-nodes(:,2).*grad(:,1);
rigidity=FEMIntegrate(nodes,elem,f)
```

This leads to a result of $J \approx 0.141$.

According equation (8.5) we can compute the resulting stresses in the shaft and create figure 8.11. The maximal stress is 4.77 .

Octave

```
f=sqrt((nodes(:,1)+grad(:,2)).^2+(-nodes(:,2)+grad(:,1)).^2)/rigidity;
maximalStress=max(f)
figure(3);
ShowSolution(nodes,elem,f);
figure(4);
ShowLevelCurves(nodes,elem,f,linspace(0,max(f),11));
```

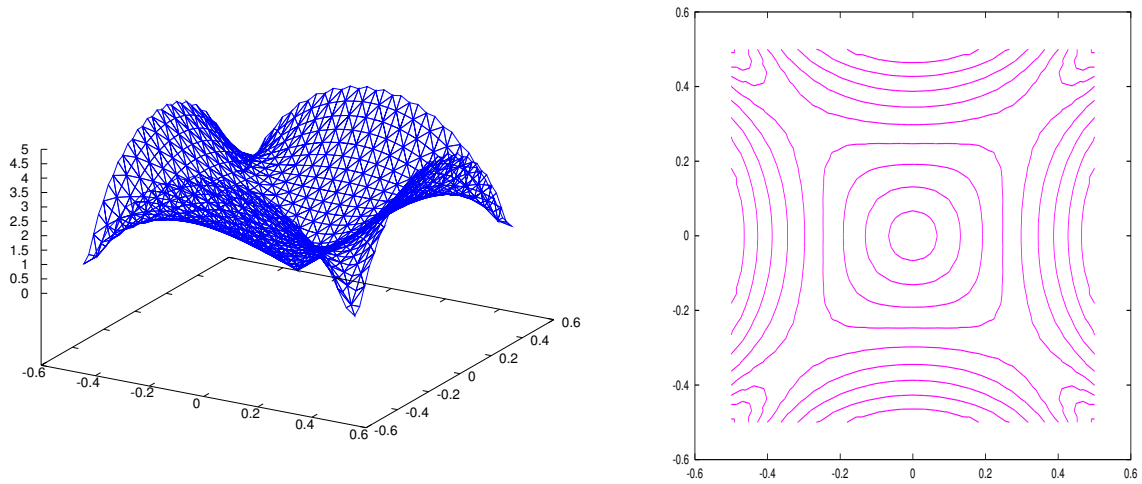


Figure 8.11: The stress function and its level curves for a square

8.3.2 Torsional rigidity of a circle and a circle with hole

If the section of the shaft is a circle with radius R we have to solve the problem

$$\begin{aligned} \operatorname{div}(\nabla\phi) &= 0 \qquad \text{for } \|\vec{x}\| < R \\ \nabla\phi \cdot \vec{n} &= 0 \qquad \text{for } \|\vec{x}\| = R \end{aligned}$$

The only solutions to this problem are $\phi(\vec{x}) = const$ and thus the torsional rigidity is given by

$$J = \int\limits_{x^2+y^2<R} \int\limits x^2 + y^2 \, dA = \frac{\pi}{2} \, R^4$$

and the resulting stresses are

$$\sigma = \pm \frac{T}{J} \sqrt{x^2 + y^2} = \frac{r}{J} = \frac{2}{\pi \, R^4}$$

For a circle with area equal to 1 we find $R = 1/\sqrt{\pi} \approx 0.564$ and thus

$$J = \frac{\pi}{2} \left(\frac{1}{\sqrt{\pi}} \right)^4 = \frac{1}{2\,\pi} \approx 0.159$$

Thus a circle is slightly more rigid than a square of the same area and also shows a smaller maximal stress of $\sigma = 3.54$.

If a hollow circular cylinder with inner radius R_1 and outer radius R_2 is considered we find

$$J = \frac{\pi}{2} \, \left(R_2^4 - R_1^4 \right)$$

and

$$\sigma = \frac{r}{J} = \frac{2\,r}{\pi \, (R_2^4 - R_1^4)}$$

By choosing $R_1 = 0.317$ and $R_2 = 0.647$ we have an area of 1 and $J = 0.260$ and a maximal tension $\sigma = 2.49$. As is to be expected we find larger rigidity and a smaller maximal stress.

8.3.3 Torsional rigidity of a rectangle

The above computations for a square can be modified to handle a rectangle. For a rectangle with width $\frac{1}{1.2}$ and height 1.2 we obtain figures 8.12 and 8.13. The rigidity of $J \approx 0.133$ is smaller than for the square and the maximal stress $\sigma \approx 5.22$ turns out to be larger.

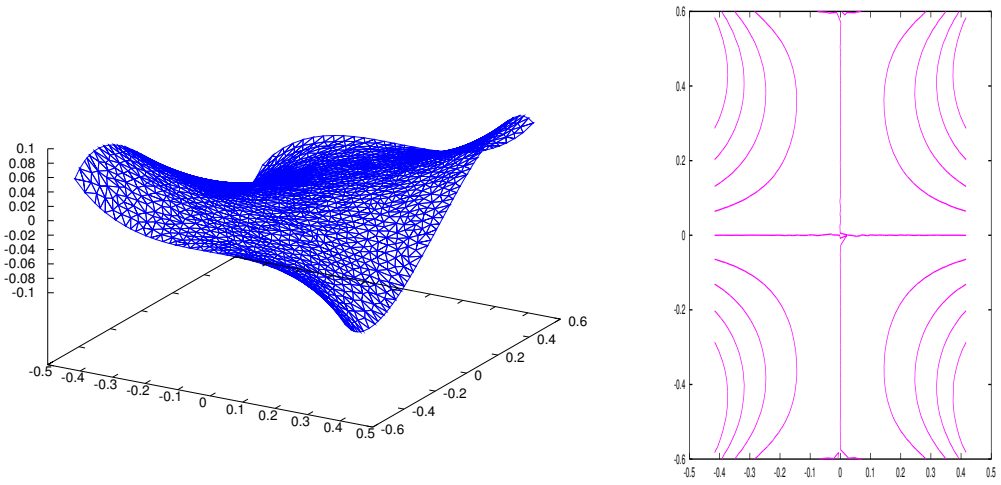


Figure 8.12: The warp function and its level curves for a rectangle

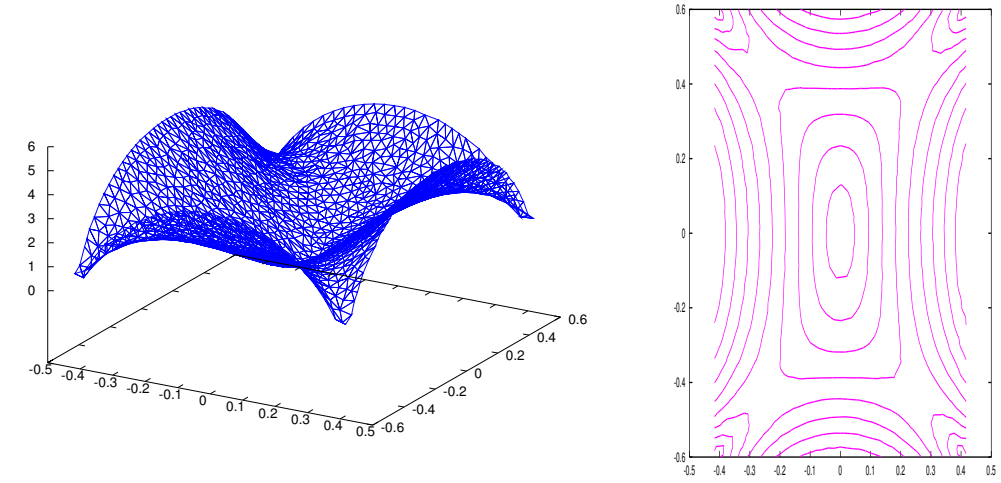


Figure 8.13: The stress function and its level curves for a rectangle

8.3.4 Torsional rigidity of a square with hole

The above computations for a square can be modified to handle a square with a hole with total area 1 again. With inner length $d_1 = 0.75$ and outer length $d_2 = 1.25$ we obtain figures 8.14 and 8.15. The rigidity of $J \approx 0.289$ is large and the maximal stress turns out to be $\sigma \approx 2.75$.

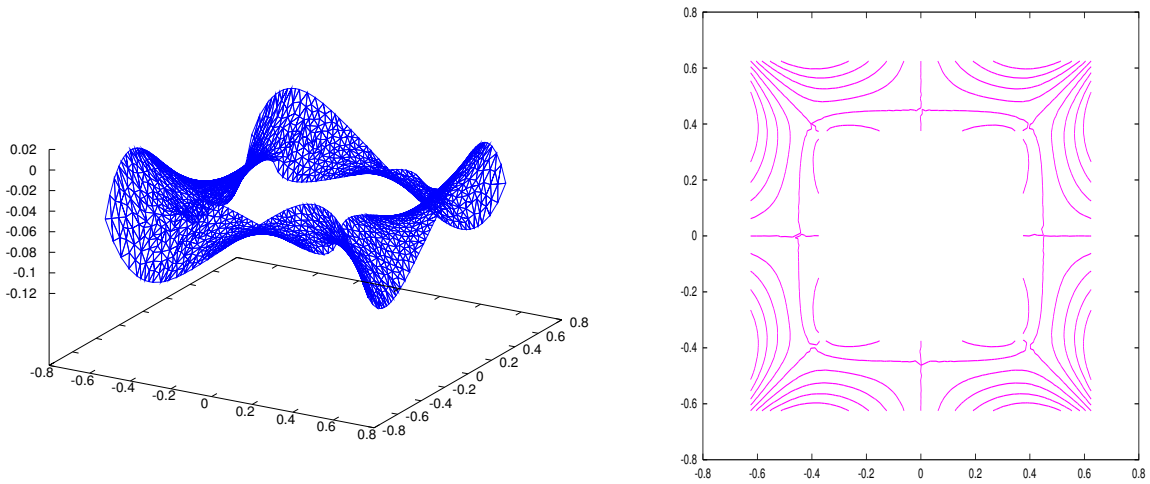


Figure 8.14: The warp function and its level curves for a square with hole

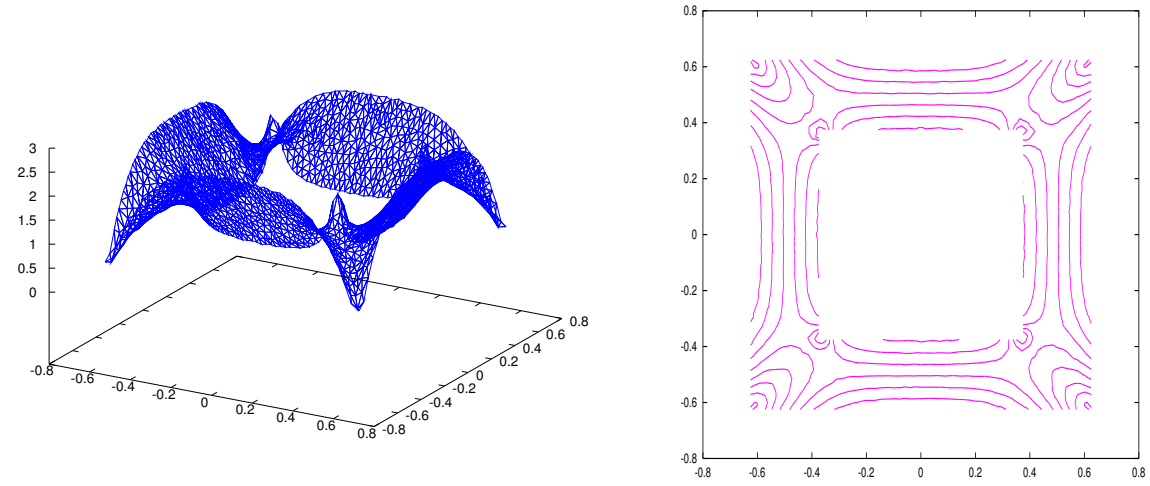


Figure 8.15: The stress function and its level curves for a square with hole

8.3.5 Comparison of different sections

In the previous section we computed the torsional rigidity of five different sections, each with area equal to 1 . On each section a standard torque of $J = 1$ was applied an then the resulting maximal stress computed. For the two hollow structures the same area was cut out, i.e. $d_1^2 = \pi\, R_1^2$. Find the results in table 8.1.

form of section, area 1 and applied torque $T = 1$	rigidity J	maximal stress σ
circle, $R = 0.564$	$J = 0.159$	$\sigma = 3.54$
square, size 1×1	$J = 0.141$	$\sigma = 4.77$
rectangle, size $1.2 \times \frac{1}{1.2}$	$J = 0.133$	$\sigma = 5.22$
square with hole, $d_1 = 0.75$, $d_2 = 1.25$	$J = 0.289$	$\sigma = 2.78$
circular hollow shaft, $R_1 = 0.317$, $R_2 = 0.647$	$J = 0.260$	$\sigma = 2.49$

Table 8.1: Comparison of torsional rigidity and maximal stress

For the above calculations we used a standard area of 1 . If the width and depth of the sections are multiplied with a factor α, then the rigidity J has to be multiplied with α^4 and the maximal tension has to be divided by α^3 .

8.4 Vibrations of a membrane

According to Table 6.1 (page 144) the vertical displacement u of membrane under tension T on the domain $\Omega \subset \mathbb{R}^2$ has to solve the boundary value problem

$$\begin{aligned} -\operatorname{div}(T\ \operatorname{grad} u) &= f \quad \text{in} \quad \Omega \\ u &= 0 \quad \text{on} \quad \Gamma = \partial\Omega \end{aligned}$$

where f represents the vertical force density (units $\left[\mathrm{N/m^2}\right]$). If no external force is applied the force density will lead to a vertical acceleration of the membrane with $f = \rho\ \ddot{u}$, where ρ is the mass per area. Thus we are lead to an wave type equation

$$-\operatorname{div}(T\ \operatorname{grad} u) = \rho\ \ddot{u} \quad \text{or} \quad -\Delta u = \frac{\rho}{T}\ \ddot{u}$$

Now consider the eigenvalue problem

$$\begin{aligned} -\Delta u &= \lambda\, u \quad \text{in} \quad \Omega \\ u &= 0 \quad \text{on} \quad \Gamma = \partial\Omega \end{aligned}$$

then the above wave equation is solved by $\cos(\omega\, t)\ u(\vec{x})$ where the angular velocity is given by

$$\omega = \sqrt{\frac{T}{\rho}}\ \sqrt{\lambda}$$

Thus for each eigenvalue λ there is a corresponding eigenfrequency ω. For a bounded domain $\Omega \subset \mathbb{R}^2$ there is a discrete set of positive eigenvalues λ and the resulting frequencies. Using *FEMoctave* numerical approximations of these eigenvalues can be computed.

For a disk $\Omega \subset \mathbb{R}^2$ with radius 1 the code below will compute four eigenvalues and eigenfunctions, leading to figure 8.16.

Octave

```
R=1; % radius of circle
nR = 40 ; % number of divisions to create circle
area=0.004;
w=linspace(0,2*pi*(1-1/nR),nR);  % angles of points on circle
xy=[R*cos(w);R*sin(w);ones(1,nR)]';
CreateMeshTriangleQ("circle",xy,area)
[nodes,elem,edges]=ReadMeshTriangle("./circle.1");

[la,vec]=FEMEig(nodes,elem,edges,1,0,1,4,1e-6);
la=la'

figure(1);ShowSolution(nodes,elem,vec(:,1))
figure(2);vmin=min(vec(:,1)); vmax=max(vec(:,1));
ShowLevelCurves(nodes,elem,vec(:,1),linspace(vmin,vmax,11))
disp("Hit RETURN"); pause();

figure(1); ShowSolution(nodes,elem,vec(:,2))
figure(2); vmin=min(vec(:,2)); vmax=max(vec(:,2));
ShowLevelCurves(nodes,elem,vec(:,2),linspace(vmin,vmax,11))
disp("Hit RETURN"); pause();

figure(1); ShowSolution(nodes,elem,vec(:,4))
figure(2); vmin=min(vec(:,4)); vmax=max(vec(:,4));
ShowLevelCurves(nodes,elem,vec(:,4),linspace(vmin,vmax,11))
```

A separation of variable argument shows that the exact eigenvalues are given by

$$\sqrt{\lambda} = z_{n,m} = \text{ m}^{th} \text{ zero of } J_n(r)$$

where $J_n(r)$ are the Bessel functions of the first kind. The eigenfunctions are

$$u(r,\phi) = J_n(r/z_{n,m}) \cdot \cos(\frac{n}{2\,\pi}\,\phi) \quad \text{and} \quad u(r,\phi) = J_n(r/z_{n,m}) \cdot \sin(\frac{n}{2\,\pi}\,\phi)$$

Thus we can compare the results of this code with the exact solution. Observe that for $n \geq 1$ we have eigenvalues of multiplicity 2. The table below compares the first 20 eigenvalues (resp. $\sqrt{\lambda}$) computed by *FEMoctave* with the exact values $z_{n,m}$.

	1	2	3	4	5	6	7	8	9	10
$\sqrt{\lambda}$	2.4089	3.8365	3.8365	5.1393	5.1390	5.5226	6.3799	6.3802	7.0120	7.0122
$z_{n,m}$	2.4048	3.8317		5.1356		5.5201	6.3802		7.0156	
n	0	1	1	2	2	0	3	3	1	1
m	1	1	1	1	1	2	1	1	2	2

	11	12	13	14	15	16	17	18	19	20
$\sqrt{\lambda}$	7.5826	7.5810	8.4047	8.4028	8.6384	8.7552	8.7555	9.7363	9.7307	9.9043
$z_{n,m}$	7.5883		8.4172		8.6537	8.7715		9.7610		9.9361
n	4	4	2	2	0	5	5	3	3	6
m	1	1	2	2	3	1	1	2	2	1

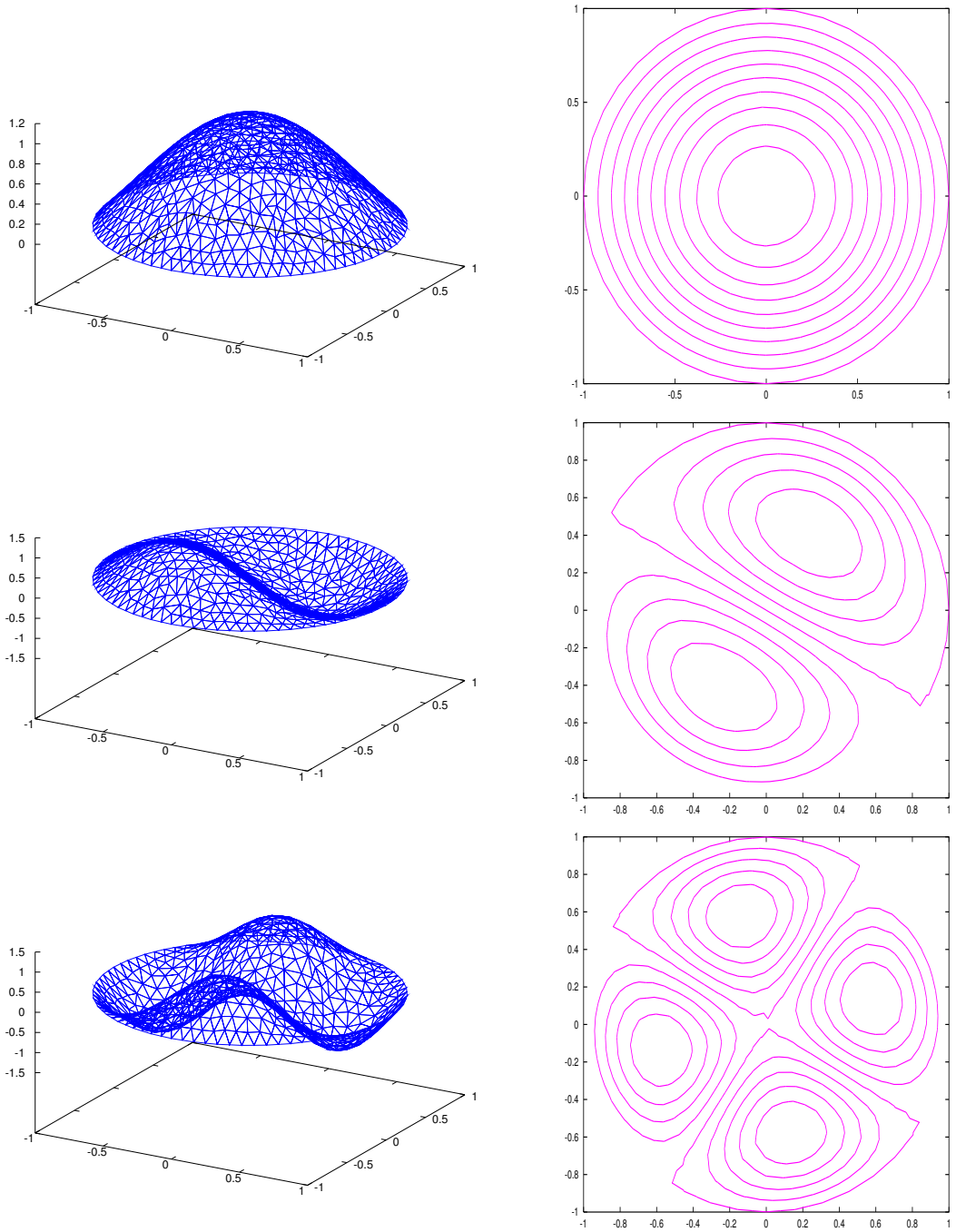


Figure 8.16: The first, second and fourth eigenfunction of a vibrating membrane

8.5 Sound in a bottle

8.5.1 The question

By blowing air across the opening of an empty bottle one can create a sound. The frequency of the sound will depend on the shape and size of the bottle. We will determine the possible frequencies. This will leads to a wave equation for the unknown pressure function $u(t, \vec{x})$. The question is then reduced to a static eigenvalue problem with the help of separation of variables. A few different, but comparable configurations are examined.

8.5.2 Finding the correct equation, based on conservation laws

With basic laws of physics we will derive the equation to be solved for the above problem. A slightly more detailed presentation is given in [GuenLee96, §1.7, §12.2]. The physical quantities to be considered are the density ρ of the gas, its pressure p and the average particle velocity $\vec{v}$, all as function of time $t \in \mathbb{R}$ and at $\vec{x} \in \mathbb{R}^3$.

- Conservation of mass
  If a gas with density ρ is moving with velocity $\vec{v}$ the change of mass in a volume $\Omega \subset \mathbb{R}^3$ with boundary $\partial\Omega$ with outer unit normal $\vec{n}$ is given by

$$\frac{d}{dt}\, M = \iiint_{\Omega} \dot{\rho}\, dV = - \oiint_{\partial\Omega} \vec{n} \cdot (\rho\, \vec{v})\, dA = - \iiint_{\Omega} \operatorname{div}(\rho\, \vec{v})\, dV$$

  Since the volume Ω is arbitrary we conclude

$$\dot{\rho} = - \operatorname{div}(\rho\, \vec{v})$$

- Newton's second law
  The only external forces acting on the gas in the volume $\Omega \subset \mathbb{R}^3$ is the pressure p on its surface. Thus the total force on the volume is

$$\vec{F} = - \oiint_{\partial\Omega} p\, \vec{n}\, dA$$

  For $i = 1, 2, 3$ we use $n_i = \vec{e}_i \cdot \vec{n}$ and then the component of the force in the direction of $\vec{e}_i$ created by the pressure is given by $p\, n_i$. Thus

$$F_i = - \oiint_{\partial\Omega} p\, \vec{e}_i \cdot \vec{n}\, dA = - \iiint_{\Omega} \operatorname{div}(p\, \vec{e}_i)\, dV = - \iiint_{\Omega} \frac{\partial}{\partial x_i}\, p\, dA$$

  Since the total momentum of the gas in the Ω is given by

$$\iiint_{\Omega} \rho\, \vec{v}\, dV$$

  Newton's second law implies

$$\frac{d}{dt} \iiint_{\Omega} \rho\, \vec{v}\, dV = - \iiint_{\Omega} \operatorname{grad} p\, dV$$

  and thus[2]

$$\frac{d}{dt}\, (\rho\, \vec{v}) = - \operatorname{grad} p$$

[2] We ignore terms due to gas moving in or out of the volume Ω, since those terms are of higher order and would be thrown away by linearization anyhow.

or writing out the components again

$$\frac{d}{dt}\,(\rho\, v_i) = -\frac{\partial}{\partial x_i}\,p \quad \text{for} \quad i = 1,2,3$$

- Linear approximations
  Now we use $\rho = \rho_0 + u$, where ρ_0 is the constant standard pressure. We assume that u (and its derivatives) are considerably smaller that ρ_0 and thus

$$\operatorname{div}(\rho\,\vec{v}) = \rho_0\ \operatorname{div}\vec{v} + u\ \operatorname{div}\vec{v} + \operatorname{grad} u\cdot\vec{v} \approx \rho_0\ \operatorname{div}\vec{v}$$

  Similarly

$$\frac{d}{dt}\,(\rho\,\vec{v}) = \dot{u}\,\vec{v} + \rho_0\,\dot{\vec{v}} \approx \rho_0\,\dot{\vec{v}}$$

  As a consequence of the above simplifications we obtain the two conservation laws

$$\begin{aligned} \dot{u} &= -\rho_0 \operatorname{div}\vec{v} \\ \rho_0\,\dot{\vec{v}} &= -\operatorname{grad} p \end{aligned}$$

  Now differentiate the first equation with respect to t and compare with the divergence of the second equation to conclude that

$$\ddot{u} = \operatorname{div}(\operatorname{grad} p)$$

  Use $p = p_0 + \beta\, u$ and thus $\operatorname{grad} p = \beta\ \operatorname{grad} u$. The coefficient β is given as $\beta = \frac{\partial p}{\partial \rho}$. We obtain the standard wave equation

$$\ddot{u} = \beta\ \operatorname{div}(\operatorname{grad} u)$$

We need to determine the boundary conditions for the above PDE. There are two different types of conditions.

- **open boundary**
  Far away we expect no forces and displacements and thus $\rho = \text{const} = \rho_0$, i.e. we find a Dirichlet boundary condition $u = 0$.

- **hard boundary**
  Along the surface of the bottle no gas transport orthogonal to the surface is possible, thus $\vec{n}\cdot\vec{v} = 0$. This implies

$$\vec{n}\cdot\dot{\vec{v}} = \frac{1}{\rho_0}\ \vec{n}\cdot\operatorname{grad} p = \frac{\beta}{\rho_0}\ \vec{n}\cdot\operatorname{grad} u = 0$$

  Thus we find a Neumann boundary condition $\vec{n}\cdot\operatorname{grad} u = 0$.

The above has to be supplemented with the initial conditions to finally find the equation to be solved.

$$\begin{aligned} \ddot{u}(t,\vec{x}) &= \beta\ \operatorname{div}\operatorname{grad} u(t,\vec{x}) && \text{in} \quad \Omega \subset \mathbb{R}^3 \\ u(t,\vec{x}) &= 0 && \text{on} \quad \Gamma_0 \\ \tfrac{\partial}{\partial n}\,u(t,\vec{x}) &= 0 && \text{on} \quad \Gamma_1 \\ u(0,\vec{x}) &= u_0(\vec{x}) && \text{in} \quad \Omega \\ \dot{u}(0,\vec{x}) &= u_1(\vec{x}) && \text{in} \quad \Omega \end{aligned} \tag{8.6}$$

The constant $\sqrt{\beta}$ represents the speed of sound for this problem.

8.5.3 Separation of variables

We seek a solution of the form

$$u(t,r,\phi,z) = T(t)\cdot\Phi(\phi)\cdot U(r,z)$$

using cylindrical coordinates.

In Appendix A.2.4 the Laplace operator is given in cylindrical coordinates

$$\Delta u = \operatorname{div}\operatorname{grad} u = \frac{1}{r}\,\frac{\partial}{\partial r}(r\,\frac{\partial\,u}{\partial r}) + \frac{1}{r^2}\frac{\partial^2\,u}{\partial\phi^2} + \frac{\partial^2\,u}{\partial z^2}$$

Thus we may rewrite the PDE a for the function $u(t,r,\phi,z)$ in the form

$$\frac{1}{\beta}\,\ddot{u} = \frac{1}{r}\,\frac{\partial}{\partial r}(r\,\frac{\partial\,u}{\partial r}) + \frac{1}{r^2}\frac{\partial^2\,u}{\partial\phi^2} + \frac{\partial^2\,u}{\partial z^2}$$

Separation time–space

Assuming that we can write the solution u as a product of a function $T(t)$, depending on time t only, and a function $V(r,\phi,z)$ we find the equation

$$\frac{1}{\beta}\,\frac{\ddot{T}}{T} = \frac{1}{V}\,\left(\frac{1}{r}\,\frac{\partial}{\partial r}(r\,\frac{\partial\,V}{\partial r}) + \frac{1}{r^2}\frac{\partial^2\,V}{\partial\phi^2} + \frac{\partial^2\,V}{\partial z^2}\right)$$

Since the LHS depends on t only and the RHS on space variables only, both sides have to equal to a constant $-\nu$ and we find the two equations

$$\begin{aligned}\ddot{T}(t) &= -\nu\,\beta\,T(t)\\ \frac{1}{r}\,\frac{\partial}{\partial r}(r\,\frac{\partial\,V}{\partial r}) + \frac{1}{r^2}\frac{\partial^2\,V}{\partial\phi^2} + \frac{\partial^2\,V}{\partial z^2} &= -\nu\,V\end{aligned}$$

If $\nu > 0$ the first equation has the obvious solution

$$T(t) = A\,\cos(\sqrt{\nu\,\beta}\,t) + B\,\sin(\sqrt{\nu\,\beta}\,t) = C\,\cos(\sqrt{\nu\,\beta}\,t+\delta)$$

and thus the final solution has angular velocity $\omega = \sqrt{\beta\,\nu}$ and is periodic with period $P = \frac{2\,\pi}{\omega}$. Thus the frequency is $f = \frac{1}{P} = \frac{\sqrt{\beta\,\nu}}{2\,\pi}$.

Separation of the angular component

The second of the above equations can be separated again $V(r,\phi,z) = \Phi(\phi)\cdot U(r,z)$ and we find the equation

$$\begin{aligned}\frac{1}{r}\,\frac{\partial}{\partial r}(r\,\frac{\partial\,V}{\partial r}) + \frac{\partial^2\,V}{\partial z^2} + \nu\,V &= -\frac{1}{r^2}\frac{\partial^2\,V}{\partial\phi^2}\\ \frac{r^2}{U}\,\left(\frac{1}{r}\,\frac{\partial}{\partial r}(r\,\frac{\partial\,U}{\partial r}) + \frac{\partial^2\,U}{\partial z^2} + \nu\,U\right) &= -\frac{1}{\Phi}\frac{\partial^2\,\Phi}{\partial\phi^2}\end{aligned}$$

Both sides have to be constant again. The radial function $\Phi(\phi)$ needs to be $2\,\pi$–periodic. The resulting ordinary differential equation

$$\frac{\partial^2}{\partial\phi^2}\,\Phi(\phi) = -\mu\,\Phi(\phi) \quad \text{with} \quad \Phi(0) = \Phi(2\,\pi) \quad \text{and} \quad \Phi'(0) = \Phi'(2\,\pi)$$

has nontrivial solutions only if $\mu = n^2$ for some $n \in \mathbb{N}_0$. The solution is $\Phi(\phi) = \sin(n\,\phi + \delta)$. The special case $n = \mu = 0$ corresponds to solutions independent on the angle ϕ.

For the remaining space variables r and z we have thus to examine the equation

$$\frac{1}{r}\,\frac{\partial}{\partial r}(r\,\frac{\partial\,U}{\partial r}) + \frac{\partial^2\,U}{\partial z^2} + \nu\,U = \frac{n^2}{r^2}\,U$$

or equivalently

$$\frac{\partial}{\partial r}(r\,\frac{\partial\,U}{\partial r}) + \frac{\partial}{\partial z}(r\,\frac{\partial\,U}{\partial z}) - \frac{n^2}{r}\,U = -\nu\,r\,U$$

If we find an eigenvalue ν for this problem, then we have the frequency and solution of the PDE (8.6) is given by

$$u(t, r, \phi, z) = C\ \cos(\sqrt{\beta\,\nu}\,t + \delta_1) \cdot \cos(n\,\phi + \delta_2) \cdot U(r, z)$$

Solutions independent on the angle ϕ

If we consider only solutions independent on ϕ (i.e. $n = 0$), then the above problem is a special case of the generalized eigenvalue problem (7.5) on page 152.

$$\begin{aligned} -\frac{\partial}{\partial r}(r\,\frac{\partial\,U}{\partial r}) - \frac{\partial}{\partial z}(r\,\frac{\partial\,U}{\partial z}) &= +\nu\,r\,U \qquad \text{in} \quad \Omega \subset \mathbb{R}^2 \\ U(r, z) &= 0 \qquad \text{on} \quad \Gamma_0 \\ \tfrac{\partial}{\partial n}\,U(r, z) &= 0 \qquad \text{on} \quad \Gamma_1 \end{aligned} \tag{8.7}$$

The finite element method will translate this boundary value problem in a generalized eigenvalue problem

$$\mathbf{A} \cdot \vec{v} = \nu\ \mathbf{B} \cdot \vec{v}$$

with symmetric, positive definite matrices $\mathbf{A}$ and $\mathbf{B}$.

8.5.4 The open organ pipe

As a first example we consider a cylinder of radius R and height H with open top. This might be an organ pipe open on one side. A section is shown in figure 8.17. In this simple situation it is possible to give an exact solution of the eigenvalue problem. A separation of the variables r and ϕ leads to

$$-\frac{1}{r\,R}\frac{\partial}{\partial r}(r\,\frac{\partial\,R}{\partial r}) - \nu\ = \frac{1}{Z}\frac{\partial}{\partial z}(\frac{\partial\,Z}{\partial z})$$

and thus the two ordinary boundary value problems

$$-\frac{\partial}{\partial z}(\frac{\partial\,Z(z)}{\partial z}) = \lambda_z\,Z(z) \quad \text{with} \quad \frac{\partial\,Z(0)}{\partial z} = 0 \quad \text{and} \quad Z(H) = 0$$

and

$$-\frac{1}{r}\frac{\partial}{\partial r}(r\,\frac{\partial\,R(r)}{\partial r}) = \lambda_r\,R(r) \quad \text{with} \quad \frac{\partial\,R(0)}{\partial r} = 0 \quad \text{and} \quad \frac{\partial\,R(R_0)}{\partial r} = 0$$

The solutions are

$$Z(z) = \sin((2\,k-1)\,\frac{\pi}{2\,H}\,(H - z)) \quad \text{and} \quad R(r) = J_0(\frac{\alpha_j}{R}\,r)$$

and the eigenvalues are given by

$$\lambda_r = \left((2\,k-1)\,\frac{\pi}{2\,H}\right)^2 \quad \text{and} \quad \lambda_r = \left(\frac{\alpha_j}{R}\right)^2$$

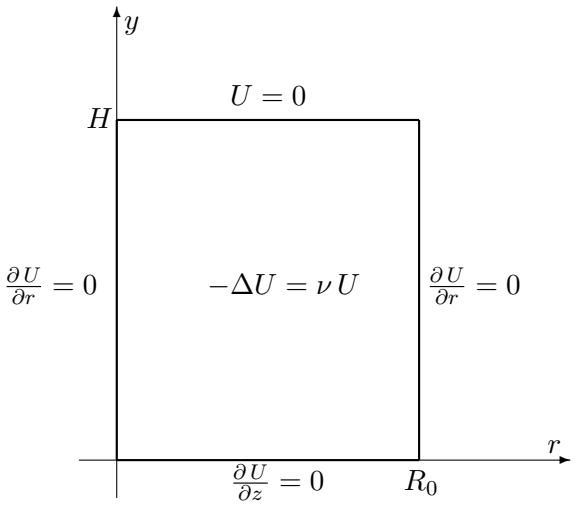


Figure 8.17: An open organ pipe

where $J_0(z)$ is a Bessel function and α_j are the zeros[3] of the derivatives of this function. The corresponding solutions of the PDE are

$$U(r,z) = \sin((2\,k-1)\,\frac{\pi}{2\,H}\,(H-z)) \cdot J_0(\frac{\alpha_j}{R}\,r)$$

with eigenvalues

$$\nu_{k,j} = \left((2\,k-1)\,\frac{\pi}{2\,H}\right)^2 + \left(\frac{\alpha_j}{R}\right)^2$$

The graph of this solution will show $k-1$ interior, local extrema in z–direction and $j-2$ interior, local extrema in r–direction. Below find a table of those eigenvalues where we choose $R_0 = 4$ and $H = 6$.

$k \setminus j$	1	2	2	4	5	6
1	0.068539	0.986162	3.144692	6.537255	11.163587	17.023642
2	0.616850	1.534473	3.693004	7.085566	11.711898	17.571954
3	1.713473	2.631096	4.789627	8.182189	12.808521	18.668576
4	3.358407	4.276030	6.434561	9.827123	14.453455	20.313510
5	5.551652	6.469276	8.627806	12.020368	16.646700	22.506756
6	8.293209	9.210832	11.369363	14.761925	19.388257	25.248313

A finite element computation with 1935 nodes leads to numerical approximations of the eigenvalues. The table also shows their identification with the exact eigenvalues by giving the indices k and j. Assuming that dimensions are given in units of cm and we use a speed of sound $\sqrt{\beta} = 330\,\frac{\mathrm{m}}{\mathrm{s}} = 3.3 \cdot 10^4\,\frac{\mathrm{cm}}{\mathrm{s}}$ we obtain resonance frequencies of $f = \frac{\sqrt{\beta\,\nu}}{2\,\pi} \approx \sqrt{\nu}\ 5252\,\mathrm{Hz}$. Observe that the second frequency equals three times the first frequency, as should be the case since the z dependence of the two modes is given by

[3] A table or mathematical software will give the values of α_j as

$$\alpha_1 = 0\,,\quad \alpha_2 = 3.8317060\,,\quad \alpha_3 = 7.0155867\,,\quad \alpha_4 = 10.1734681\,,\quad \alpha_5 = 13.3236919\,,\quad \alpha_6 = 16.4706301$$

$\sin(\frac{\pi}{2H}(H-z))$ and $\sin(3\frac{\pi}{2H}(H-z))$.

ν	k	j	frequency
0.068534	1	1	1375 Hz
0.616478	2	1	4124 Hz
0.986261	1	2	5216 Hz
1.534120	2	2	6505 Hz
1.710429	3	1	6869 Hz
2.627742	3	2	8514 Hz

(8.8)

In figure 8.18 find the solution corresponding[4] to the fourth eigenvalue.

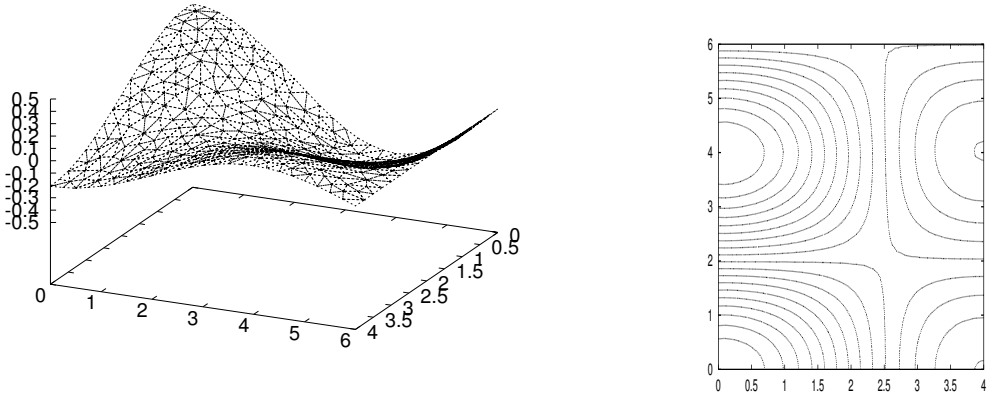


Figure 8.18: The graph of the fourth eigenfunction and its level-curves for the organ pipe problem

8.5.5 A can with a circular hole

As next example we place a lid with a circular hole on top of the organ pipe from the previous section. The setup is shown in figure 8.19 and obviously similar to the previous section. The only change to be made concerns the semi closed top. For $0 < R_1 < r < R$ we have to replace the Dirichlet condition $U = 0$ by the Neumann condition $\frac{\partial}{\partial z} U = 0$. As a consequence the separation of the variables r and z will not work and no analytical solutions are available.

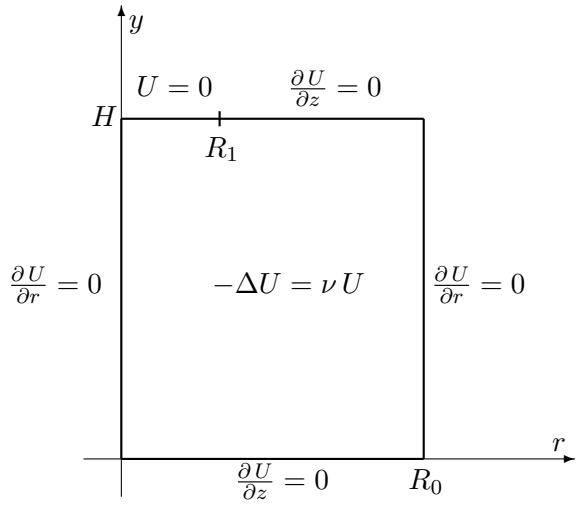
With help of FEM we can compute the first few eigenvalues of the problem and thus also the resulting frequencies. The result in figure 8.19 and has to be compared with the results in (8.8) for the open organ pipe. The values used are $R_1 = 1$, $R_0 = 4$ and $H = 6$. In figure 8.20 find the first six eigenfunctions.

8.5.6 A can with a circular neck

As next example we place circular neck on top of the can from the previous section. The setup is shown in figure 8.21 and obviously similar to the previous section. Separation of the variables is again not possible and no analytical solutions are available.

With help of FEM we can compute the first few eigenvalues of the problem and thus also the resulting frequencies. The result in figure 8.21 and has to be compared with the results in (8.8) and figure 8.19 for the

[4]The actual graph of the solution was created with fewer triangles to obtain a visually more pleasant picture.



ν	frequency
0.0168	680 Hz
0.3096	2922 Hz
0.9638	5156 Hz
1.1109	5535 Hz
1.4166	6251 Hz
2.2152	7817 Hz

Figure 8.19: A can with hole and the resulting eigenvalues and frequencies

other configurations. The values used are $R_1 = 1$, $R_0 = 4$, $H = 6$ and a neck of height 1 was added. In figure 8.22 find the first four eigenfunctions.

8.5.7 A can with a circular neck, with air gap

For all previous examples we assumed that $U = 0$ at the opening edge of the containers. This is a drastic simplification. One should actually assume that $U \approx 0$ far away from the opening. Some of the air close to the opening will certainly also vibrate. To examine this effect one can repeat the calculations of the previous example but add some air space around the opening. The boundary condition for the air section is $U = 0$. The setup and the resulting frequencies are shown in figure 8.23, together with the results of the simplified problem. The third eigenfunction and its level curves are shown in figure 8.24. The results justify the simplifying assumption. In particular we find $U \approx 0$ outside of the container.

8.5.8 Conclusion

not written yet

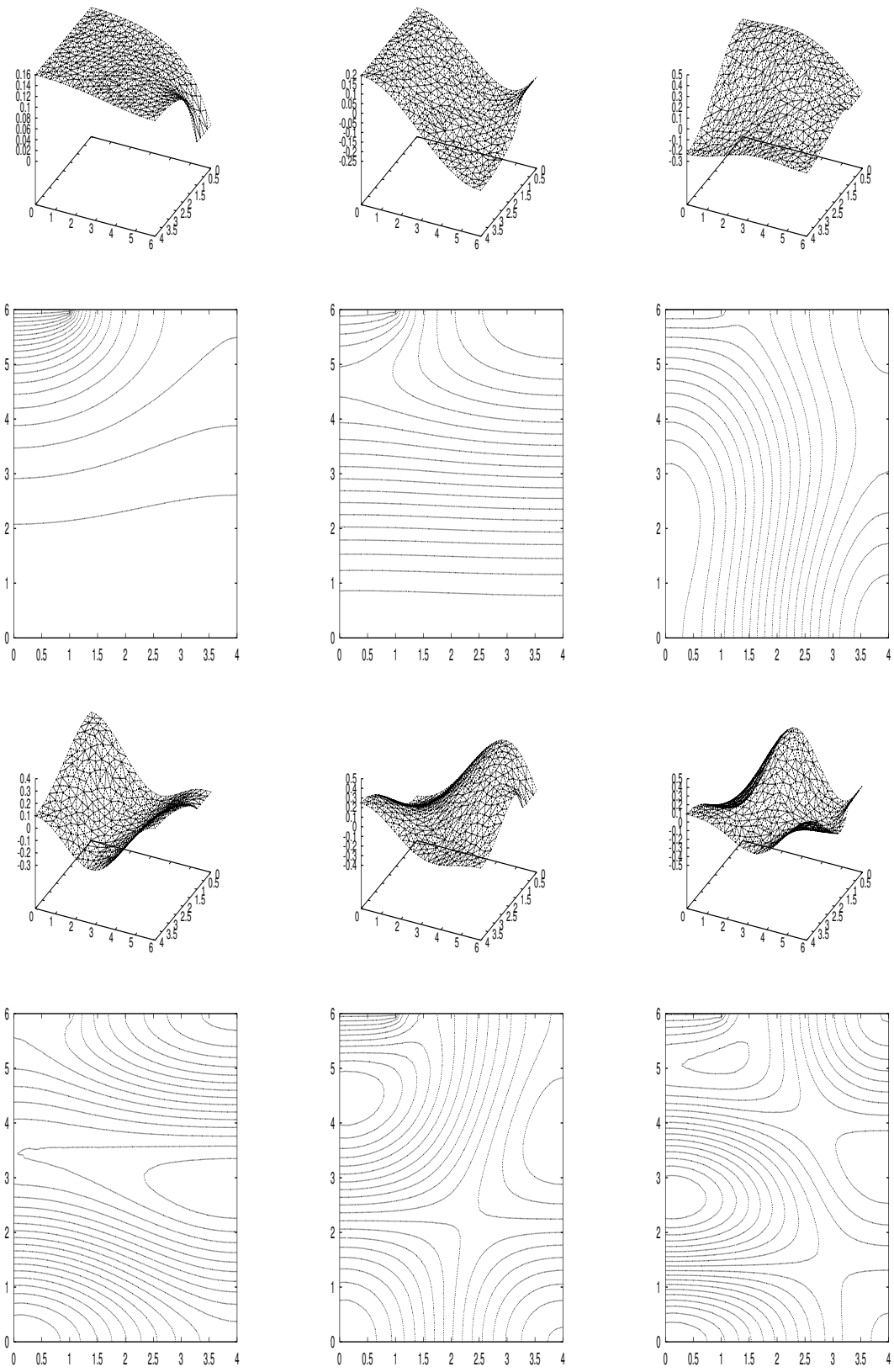
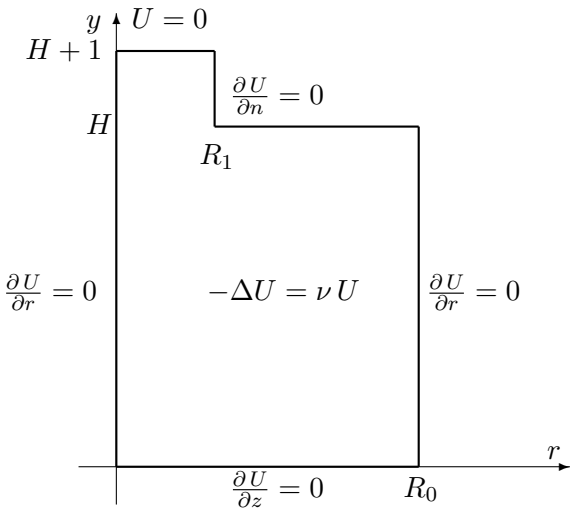


Figure 8.20: The first six eigenfunctions of a can with a hole



ν	frequency with neck	frequency no neck
0.0063	416 Hz	680 Hz
0.2861	2809 Hz	2922 Hz
0.9376	5086 Hz	5156 Hz
1.1024	5514 Hz	5535 Hz
1.2393	5847 Hz	6251 Hz
2.0326	7488 Hz	7817 Hz

Figure 8.21: A can with a neck and the resulting eigenvalues and frequencies

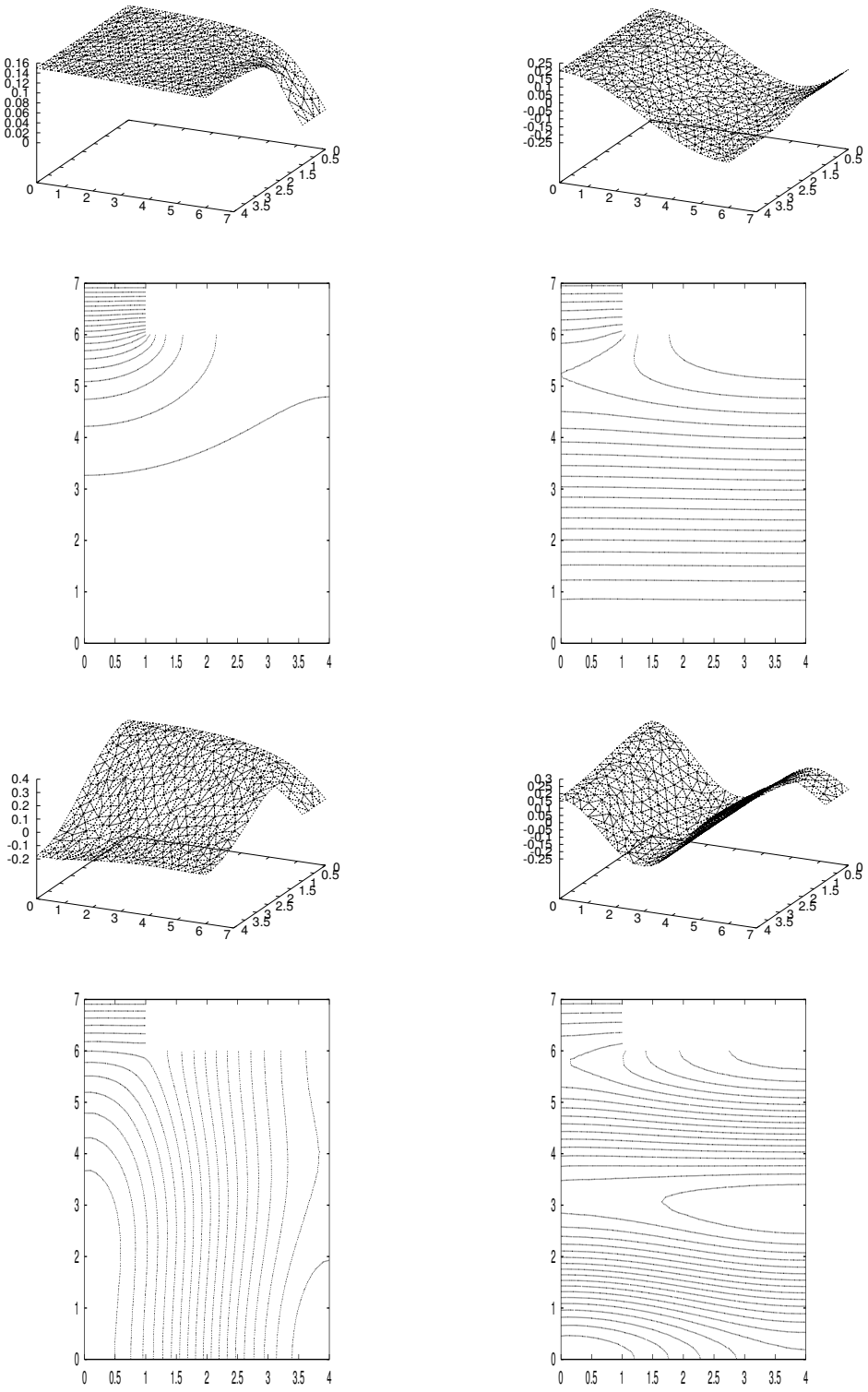
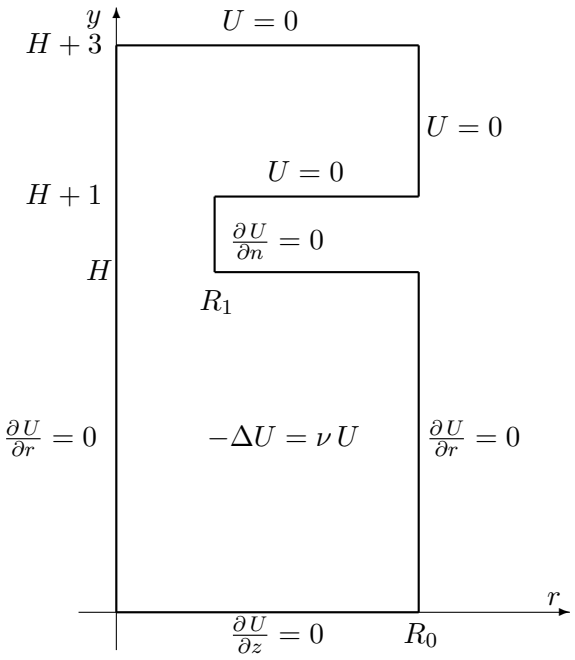


Figure 8.22: The first four eigenfunction of a can with a neck



frequency with air gap	frequency no air gap
387 Hz	416 Hz
2799 Hz	2809 Hz
5067 Hz	5086 Hz
5507 Hz	5514 Hz
5775 Hz	5847 Hz
7360 Hz	7488 Hz

Figure 8.23: Comparison of frequencies without and with the air gap

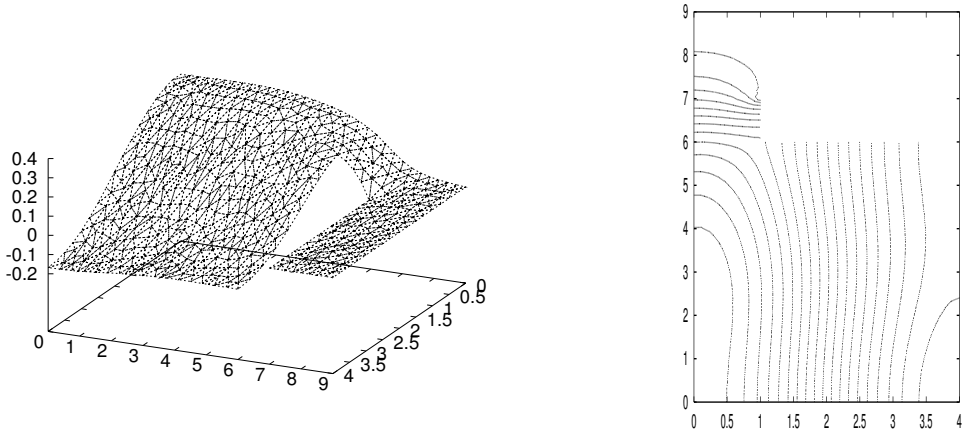


Figure 8.24: The third eigenfunction of a can with a neck and air gap

8.6 Ultrasonic distance measurements

8.7 Asparagus

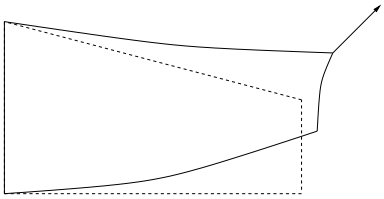
8.8 Heating a disk

Chapter 9

Linear Elasticity

9.1 Description of stress and strain

An elastic solid can be fixed at its left edge and be pulled on at the right edge by a force. Figure 9.1 shows a simple situation. The original shape (dotted line) will change into a deformed state (full line). The goal is to give a mathematical description of the deformation of the solid (strain) and the forces that will occur in the solid (stress). For a given point $\vec{x}$ in the solid is moved to $\vec{x} + \vec{u}(\vec{x})$.



$$\begin{pmatrix} x \\ y \end{pmatrix} \to \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} u_1(x,y) \\ u_2(x,y) \end{pmatrix}$$

Figure 9.1: Deformation of an elastic solid

The results will be used to give a formula for the elastic energy stored in the deformed solid. Based on this information we construct a finite element solution to the problem. For a given force we search the displacement vector field $\vec{u}(\vec{x})$.

> In order to simplify the treatment enormously we assume that the displacement of the structure are very small compared to the dimensions of the solid.

9.1.1 Description of strain

The **strain** will give us a mathematical description of the deformation of a given object. It is a purely geometrical description and at this point not related to elasticity problems.

First we examine the strain for the deformation of an object in a plane. Later we will extend the construction to object in space.

Of a large object to be deformed and moved in a plane (see figure 9.1) we consider only a small rectangle of width Δx and height Δy and examine its behavior under the deformation. The original rectangle $ABCD$ and the deformed shape $A'B'C'D'$ is shown in figure 9.2.

Since Δx and Δy are assumed to be very small the deformation is very close to an affine deformation, i.e. a linear deformation and a translation. Since the deformations are small we also know that the deformed

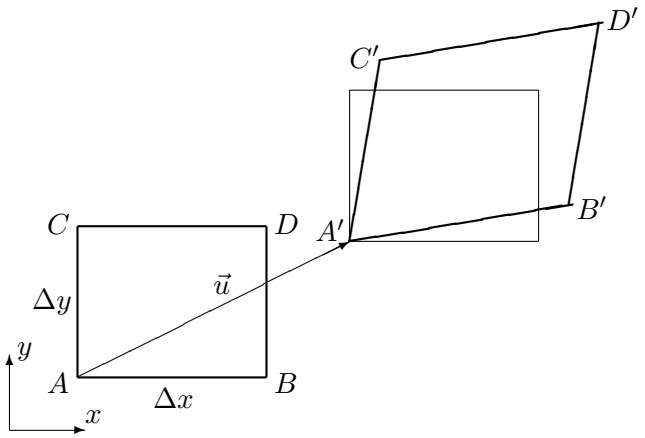


Figure 9.2: Definition of strain

rectangle has to be almost horizontal, thus figure 9.2 is correct. A straightforward Taylor approximation shows that the following behavior for the four corners of the rectangle.

$$A = \begin{pmatrix} x \\ y \end{pmatrix} \longrightarrow A' = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} u_1(x,y) \\ u_2(x,y) \end{pmatrix}$$

$$B = \begin{pmatrix} x+\Delta x \\ y \end{pmatrix} \longrightarrow B' = \begin{pmatrix} x+\Delta x \\ y \end{pmatrix} + \begin{pmatrix} u_1(x,y) \\ u_2(x,y) \end{pmatrix} + \begin{pmatrix} \frac{\partial u_1(x,y)}{\partial x}\Delta x \\ \frac{\partial u_2(x,y)}{\partial x}\Delta x \end{pmatrix}$$

$$C = \begin{pmatrix} x \\ y+\Delta y \end{pmatrix} \longrightarrow C' = \begin{pmatrix} x \\ y+\Delta y \end{pmatrix} + \begin{pmatrix} u_1(x,y) \\ u_2(x,y) \end{pmatrix} + \begin{pmatrix} \frac{\partial u_1(x,y)}{\partial y}\Delta y \\ \frac{\partial u_2(x,y)}{\partial y}\Delta y \end{pmatrix}$$

$$D = \begin{pmatrix} x+\Delta x \\ y+\Delta y \end{pmatrix} \longrightarrow D' = \begin{pmatrix} x+\Delta x \\ y+\Delta y \end{pmatrix} + \begin{pmatrix} u_1(x,y) \\ u_2(x,y) \end{pmatrix} + \begin{pmatrix} \frac{\partial u_1(x,y)}{\partial x}\Delta x + \frac{\partial u_1(x,y)}{\partial y}\Delta y \\ \frac{\partial u_2(x,y)}{\partial x}\Delta x + \frac{\partial u_2(x,y)}{\partial y}\Delta y \end{pmatrix}$$

The last equation can be rewritten as

$$\begin{aligned} \begin{pmatrix} \Delta u_1 \\ \Delta u_2 \end{pmatrix} &= \begin{pmatrix} u_1(x+\Delta x, y+\Delta y) \\ u_2(x+\Delta x, y+\Delta y) \end{pmatrix} - \begin{pmatrix} u_1(x,y) \\ u_2(x,y) \end{pmatrix} = \begin{bmatrix} \frac{\partial u_1}{\partial x} & \frac{\partial u_1}{\partial y} \\ \frac{\partial u_2}{\partial x} & \frac{\partial u_2}{\partial y} \end{bmatrix} \cdot \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \\ &= \frac{1}{2}\begin{bmatrix} \frac{\partial u_1}{\partial x} + \frac{\partial u_1}{\partial x} & \frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x} \\ \frac{\partial u_2}{\partial x} + \frac{\partial u_1}{\partial y} & \frac{\partial u_2}{\partial y} + \frac{\partial u_2}{\partial y} \end{bmatrix} \cdot \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} + \frac{1}{2}\begin{bmatrix} 0 & \frac{\partial u_1}{\partial y} - \frac{\partial u_2}{\partial x} \\ \frac{\partial u_2}{\partial x} - \frac{\partial u_1}{\partial y} & 0 \end{bmatrix} \cdot \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \\ &= \mathbf{A} \cdot \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} + \mathbf{R} \cdot \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \end{aligned}$$

Observe that the matrix $\mathbf{A}$ is symmetric and $\mathbf{R}$ is antisymmetric[1].

Since we assume that our structure is only slightly deformed we assume[2] that Δu_1 and Δu_2 are considerably smaller than Δx and Δy. Now we compute the distance of the points A' and D' in the deformed

[1] This implies $\langle \vec{v}, \mathbf{A} \cdot \vec{w} \rangle = \langle \mathbf{A}^T \cdot \vec{v}, \vec{w} \rangle = \langle \mathbf{A} \cdot \vec{v}, \vec{w} \rangle$ and $\langle \vec{v}, \mathbf{R} \cdot \vec{w} \rangle = \langle \mathbf{R}^T \cdot \vec{v}, \vec{w} \rangle = -\langle \mathbf{R} \cdot \vec{v}, \vec{w} \rangle$

[2] Due to this simplification we will later encouter a problem with rotations about large angles

body

$$\begin{aligned}
|A'D'|^2 &= (\Delta x + \Delta u_1)^2 + (\Delta y + \Delta u_2)^2 = \langle \begin{pmatrix} \Delta x + \Delta u_1 \\ \Delta y + \Delta u_2 \end{pmatrix}, \begin{pmatrix} \Delta x + \Delta u_1 \\ \Delta y + \Delta u_2 \end{pmatrix} \rangle \\
&\approx \langle \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}, \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \rangle + \langle \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}, \mathbf{A} \cdot \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} + \mathbf{R} \cdot \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \rangle + \\
&\qquad \langle \mathbf{A} \cdot \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} + \mathbf{R} \cdot \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}, \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \rangle \\
&= \langle \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}, \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \rangle + 2 \langle \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}, \mathbf{A} \cdot \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \rangle
\end{aligned}$$

Thus we observe that the term in the matrix **R** do not lead to changes of distances in the body. They correspond to rotations. Only the term in **A** are to be considered.

If we set $\Delta y = 0$ in the above formula we can compute the distance $|A'B'|$ as

$$\begin{aligned}
|A'B'|^2 &= (\Delta x)^2 + 2 \frac{u_1}{\partial x} (\Delta x)^2 \\
|A'B'| &= \sqrt{1 + 2 \frac{u_1}{\partial x}} \Delta x \approx \Delta x + \frac{u_1}{\partial x} \Delta x
\end{aligned}$$

Now we can compute the ratio of the change of length over the original length to obtain the **normal strains** ε_{xx} and ε_{yy} in the direction of the two axes.

$$\begin{aligned}
\varepsilon_{xx} &= \frac{\text{change of length in } x \text{ direction}}{\text{length in } x \text{ direction}} = \frac{\frac{\partial u_1(x,y)}{\partial x} \Delta x}{\Delta x} = \frac{\partial u_1(x,y)}{\partial x} \\
\varepsilon_{yy} &= \frac{\text{change of length in } y \text{ direction}}{\text{length in } y \text{ direction}} = \frac{\frac{\partial u_2(x,y)}{\partial y} \Delta y}{\Delta y} = \frac{\partial u_2(x,y)}{\partial y}
\end{aligned}$$

To find the geometric interpretation of the **shear strain**

$$\varepsilon_{xy} = \varepsilon_{yx} = \frac{1}{2} \left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x} \right)$$

we assume that the rectangle $ABCD$ is not rotated an in fact in the situation shown in figure 9.2. Let γ_1 be the angle formed by the line $A'B'$ with the x axis and γ_2 the angle between the line $A'C'$ and the y axis. The sign convention is such that both angles in figure 9.2 are positive. Since $\tan \phi \approx \phi$ for small angles we find

$$\begin{aligned}
\tan \gamma_1 &= \frac{\frac{\partial u_2(x,y)}{\partial x} \Delta x}{\Delta x} = \frac{\partial u_2(x,y)}{\partial x} \\
\tan \gamma_2 &= \frac{\frac{\partial u_1(x,y)}{\partial y} \Delta y}{\Delta y} = \frac{\partial u_1(x,y)}{\partial y} \\
2\, \varepsilon_{xy} &= \tan \gamma_1 + \tan \gamma_2 \approx \gamma_1 + \gamma_2
\end{aligned}$$

Thus the number ε_{xy} indicates by how much a right angle between the x and y axis would be diminished by the given deformation.

9–1 Example : It is a good exercise to compute the strain components for a few simple deformations.

- pure translation:
  If the displacement vector $\vec{u}$ is constant we have the situation of a pure translation, without deformation. Since all derivatives of u_1 and u_2 vanish we find $\varepsilon_{xx} = \varepsilon_{yy} = \varepsilon_{xy} = 0$, i.e. the strain components are all zero.

- pure rotation:
  A pure rotation by angle ϕ is given by

$$\begin{pmatrix} x \\ y \end{pmatrix} \longrightarrow \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \cos\phi\, x - \sin\phi\, y \\ \sin\phi\, x + \cos\phi\, y \end{pmatrix}$$

  and thus the displacement vector is given by

$$\begin{pmatrix} u_1(x,y) \\ u_2(x,y) \end{pmatrix} = \begin{pmatrix} \cos\phi\, x - \sin\phi\, y - x \\ \sin\phi\, x + \cos\phi\, y - y \end{pmatrix}$$

  Since the overall displacement has to be small we can only compute with small angles ϕ. This leads to

$$\begin{aligned} \varepsilon_{xx} &= \frac{\partial u_1}{\partial x} = \cos\phi - 1 \approx 0 \\ \varepsilon_{yy} &= \frac{\partial u_2}{\partial y} = \cos\phi - 1 \approx 0 \\ 2\,\varepsilon_{xy} &= \frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x} = 0 \end{aligned}$$

  Again all components of the strain vanish.

- stretching in both directions:
  the displacement

$$\begin{pmatrix} u_1(x,y) \\ u_2(x,y) \end{pmatrix} = \lambda \begin{pmatrix} x \\ y \end{pmatrix}$$

  corresponds to a stretching of the solid by the factor λ in both directions. The components of the strain are given by

$$\begin{aligned} \varepsilon_{xx} &= \frac{\partial u_1}{\partial x} = \lambda \\ \varepsilon_{yy} &= \frac{\partial u_2}{\partial y} = \lambda \\ 2\,\varepsilon_{xy} &= \frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x} = 0 \end{aligned}$$

  i.e. there is no shear stress in this situation.

- stretching in x direction only:
  the displacement

$$\begin{pmatrix} u_1(x,y) \\ u_2(x,y) \end{pmatrix} = \lambda \begin{pmatrix} x \\ 0 \end{pmatrix}$$

corresponds to a stretching by the factor λ along the x axis. The components of the strain are given by

$$\begin{aligned} \varepsilon_{xx} &= \frac{\partial u_1}{\partial x} = \lambda \\ \varepsilon_{yy} &= \frac{\partial u_2}{\partial y} = 0 \\ 2\,\varepsilon_{xy} &= \frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x} = 0 \end{aligned}$$

- stretching in $45°$ direction:
the displacement

$$\begin{pmatrix} u_1(x,y) \\ u_2(x,y) \end{pmatrix} = \frac{\lambda}{2} \begin{pmatrix} x+y \\ x+y \end{pmatrix}$$

corresponds to a stretching by the factor λ along the axis $x = y$. The straight line $y = -x$ is left unchanged. To verify this observe

$$\begin{pmatrix} u_1(x,x) \\ u_2(x,x) \end{pmatrix} = \lambda \begin{pmatrix} x \\ x \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} u_1(x,-x) \\ u_2(x,-x) \end{pmatrix} = \lambda \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

The components of the strain are given by

$$\begin{aligned} \varepsilon_{xx} &= \frac{\partial u_1}{\partial x} = \frac{\lambda}{2} \\ \varepsilon_{yy} &= \frac{\partial u_2}{\partial y} = \frac{\lambda}{2} \\ 2\,\varepsilon_{xy} &= \frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x} = \lambda \end{aligned}$$

- The two previous examples both stretch the solid in one direction by a factor λ and leave the orthogonal direction unchanged. Thus it is the same type of deformation, the difference being the coordinate system used to examine the result. Observe that the expressions

$$\begin{array}{ccc} & \varepsilon_{xx}\ ,\ \varepsilon_{yy}\ ,\ \varepsilon_{xy} & \text{depend on the coordinate system} \\ \varepsilon_{xx} + \varepsilon_{yy} \quad \text{and} & \dfrac{\partial\, u_1}{\partial y} - \dfrac{\partial\, u_2}{\partial x} & \text{do } \textbf{not depend} \text{ on the coordinate system} \end{array}$$

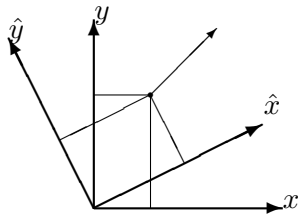
This observation will be confirmed and proven in the next result,

♢

9–2 Observation : Consider two coordinate systems, where one is generated by rotating the first coordinate axes by an angle α. The situation is shown in figure 9.3 with $\alpha = \frac{\pi}{6} = 30°$. Now we want to express a vector $\vec{u}$ (components in xy–system) also in the x', y'–system. To achieve this rotate the vector $\vec{u}$ by α and read of the components. In our example we have $\vec{u} = (1\ ,\ 1)^T$ and thus

$$\vec{u}' = \mathbf{R}^T \cdot \vec{u} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \cdot \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \approx \begin{bmatrix} 0.866 & 0.5 \\ -0.5 & 0.866 \end{bmatrix} \cdot \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1.366 \\ 0.366 \end{pmatrix}$$

The numbers are confirmed by figure 9.3 . ♢



$$\vec{x}' = \mathbf{R}^T \cdot \vec{x} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$
$$\vec{x} = \mathbf{R} \cdot \vec{x}' = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \cdot \begin{pmatrix} x' \\ y' \end{pmatrix}$$

Figure 9.3: Rotation of the coordinate system

9–3 Result : *A given strain situation is examined in two different coordinate system, as show in figure 9.3. Then we have*

$$\begin{aligned} \varepsilon_{xx} + \varepsilon_{yy} &= \varepsilon'_{x'x'} + \varepsilon'_{y'y'} \\ \frac{\partial\, u_1}{\partial y} - \frac{\partial\, u_2}{\partial x} &= \frac{\partial\, u'_1}{\partial y'} - \frac{\partial\, u'_2}{\partial x'} \end{aligned}$$

and the strain components transform according to the formula

$$\begin{bmatrix} \varepsilon'_{x'x'} & \varepsilon'_{x'y'} \\ \varepsilon'_{x'y'} & \varepsilon'_{y'y'} \end{bmatrix} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \cdot \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{xy} \\ \varepsilon_{xy} & \varepsilon_{yy} \end{bmatrix} \cdot \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}$$

◇

Proof : Since the deformations at a given point are identical we have

$$\begin{aligned} \vec{u}'(\vec{x}') &= \mathbf{R}^T \cdot \vec{u}(\vec{x}) = \mathbf{R}^T \cdot \vec{u}(\mathbf{R} \cdot \vec{x}') \\ u'_1(\vec{x}') &= \begin{bmatrix} \cos\phi & \sin\phi \end{bmatrix} \cdot \vec{u}(\mathbf{R} \cdot \vec{x}') = \cos\phi\, u_1(\vec{x}) + \sin\phi\, u_2(\vec{x}) \\ u'_2(\vec{x}') &= \begin{bmatrix} -\sin\phi & \cos\phi \end{bmatrix} \cdot \vec{u}(\mathbf{R} \cdot \vec{x}') = -\sin\phi\, u_1(\vec{x}) + \cos\phi\, u_2(\vec{x}) \\ u'_1(\vec{x}') &= \cos\phi\, u_1(\cos\phi\, x' - \sin\phi\, y', \sin\phi\, x' + \cos\phi\, y') + \\ &\quad + \sin\phi\, u_2(\cos\phi\, x' - \sin\phi\, y', \sin\phi\, x' + \cos\phi\, y') \\ u'_2(\vec{x}') &= -\sin\phi\, u_1(\cos\phi\, x' - \sin\phi\, y', \sin\phi\, x' + \cos\phi\, y') + \\ &\quad + \cos\phi\, u_2(\cos\phi\, x' - \sin\phi\, y', \sin\phi\, x' + \cos\phi\, y') \end{aligned}$$

With elementary, but lengthy computations we find

$$\begin{aligned} \frac{\partial}{\partial x'} u'_1(\vec{x}') &= \cos\phi \left(\frac{\partial u_1}{\partial x} \cos\phi + \frac{\partial u_1}{\partial y} \sin\phi \right) + \sin\phi \left(\frac{\partial u_2}{\partial x} \cos\phi + \frac{\partial u_2}{\partial y} \sin\phi \right) \\ &= \cos^2\phi \frac{\partial u_1}{\partial x} + \sin^2\phi \frac{\partial u_2}{\partial y} + \cos\phi \sin\phi \left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x} \right) \\ \frac{\partial}{\partial y'} u'_1(\vec{x}') &= \cos\phi \left(-\frac{\partial u_1}{\partial x} \sin\phi + \frac{\partial u_1}{\partial y} \cos\phi \right) + \sin\phi \left(-\frac{\partial u_2}{\partial x} \sin\phi + \frac{\partial u_2}{\partial y} \cos\phi \right) \\ &= -\cos\phi \sin\phi \frac{\partial u_1}{\partial x} + \cos\phi \sin\phi \frac{\partial u_2}{\partial y} + \cos^2\phi \frac{\partial u_1}{\partial y} - \sin^2\phi \frac{\partial u_2}{\partial x} \\ \frac{\partial}{\partial x'} u'_2(\vec{x}') &= -\sin\phi \left(\frac{\partial u_1}{\partial x} \cos\phi + \frac{\partial u_1}{\partial y} \sin\phi \right) + \cos\phi \left(\frac{\partial u_2}{\partial x} \cos\phi + \frac{\partial u_2}{\partial y} \sin\phi \right) \\ &= -\cos\phi \sin\phi \frac{\partial u_1}{\partial x} + \cos\phi \sin\phi \frac{\partial u_2}{\partial y} - \sin^2\phi \frac{\partial u_1}{\partial y} + \cos^2\phi \frac{\partial u_2}{\partial x} \end{aligned}$$

$$\begin{aligned}\frac{\partial}{\partial y'}\,u_2'(\vec{x}') &= -\sin\phi\,\left(-\frac{\partial u_1}{\partial x}\,\sin\phi+\frac{\partial u_1}{\partial y}\,\cos\phi\right)+\cos\phi\,\left(-\frac{\partial u_2}{\partial x}\,\sin\phi+\frac{\partial u_2}{\partial y}\,\cos\phi\right)\\ &= \sin^2\phi\,\frac{\partial u_1}{\partial x}+\cos^2\phi\,\frac{\partial u_1}{\partial y}-\cos\phi\,\sin\phi\,\left(\frac{\partial u_1}{\partial y}+\frac{\partial u_2}{\partial x}\right)\end{aligned}$$

Now it is easily verified that

$$\begin{aligned}\varepsilon'_{x'x'}+\varepsilon'_{y'y'}=\frac{\partial\,u_1'}{\partial x'}+\frac{\partial\,u_2'}{\partial y'} &= \frac{\partial\,u_1}{\partial x}+\frac{\partial\,u_2}{\partial y}=\varepsilon_{xx}+\varepsilon_{yy}\\ \frac{\partial\,u_1'}{\partial y'}-\frac{\partial\,u_2'}{\partial x'} &= \frac{\partial\,u_1}{\partial y}-\frac{\partial\,u_2}{\partial x}\end{aligned}$$

These two expressions are thus independent on the orientation of the coordinate system we choose.

If the matrix multiplication below is carried one step further, then the claimed transformation formula will appear.

Find names

$$\begin{aligned}&\mathbf{R}^T\cdot\begin{bmatrix}2\,\varepsilon_{xx} & 2\,\varepsilon_{xy}\\ 2\,\varepsilon_{xy} & 2\,\varepsilon_{yy}\end{bmatrix}\cdot\mathbf{R}=\\ &=\begin{bmatrix}\cos\phi & \sin\phi\\ -\sin\phi & \cos\phi\end{bmatrix}\cdot\begin{bmatrix}2\,\frac{\partial\,u_1}{\partial x} & \frac{\partial\,u_2}{\partial x}+\frac{\partial\,u_1}{\partial y}\\ \frac{\partial\,u_2}{\partial x}+\frac{\partial\,u_1}{\partial y} & 2\,\frac{\partial\,u_2}{\partial y}\end{bmatrix}\cdot\begin{bmatrix}\cos\phi & -\sin\phi\\ \sin\phi & \cos\phi\end{bmatrix}\\ &=\begin{bmatrix}\cos\phi & \sin\phi\\ -\sin\phi & \cos\phi\end{bmatrix}\cdot\begin{bmatrix}2\,\cos\phi\,\frac{\partial\,u_1}{\partial x}+\sin\phi(\frac{\partial\,u_2}{\partial x}+\frac{\partial\,u_1}{\partial y}) & -2\,\sin\phi\,\frac{\partial\,u_1}{\partial x}+\cos\phi(\frac{\partial\,u_2}{\partial x}+\frac{\partial\,u_1}{\partial y})\\ \cos\phi\,(\frac{\partial\,u_2}{\partial x}+\frac{\partial\,u_1}{\partial y})+2\,\sin\phi\,\frac{\partial\,u_2}{\partial y} & -\sin\phi\,(\frac{\partial\,u_2}{\partial x}+\frac{\partial\,u_1}{\partial y})+2\,\cos\phi\,\frac{\partial\,u_2}{\partial y}\end{bmatrix}\end{aligned}$$

□

Since the strain matrix is symmetric there always exists (see section A.1.3) an angle ϕ such that the strain matrix in the new coordinate system is diagonal, i.e.

$$\begin{bmatrix}\varepsilon'_{x'x'} & 0\\ 0 & \varepsilon'_{y'y'}\end{bmatrix}=\begin{bmatrix}\cos\phi & -\sin\phi\\ \sin\phi & \cos\phi\end{bmatrix}\cdot\begin{bmatrix}\varepsilon_{xx} & \varepsilon_{xy}\\ \varepsilon_{xy} & \varepsilon_{yy}\end{bmatrix}\cdot\begin{bmatrix}\cos\phi & \sin\phi\\ -\sin\phi & \cos\phi\end{bmatrix}$$

Thus at least close to the examined point the deformation consists of stretching the x' axis and stretching the y' axis. The displacement is given by

$$\begin{pmatrix}x'\\ y'\end{pmatrix}\longrightarrow\begin{pmatrix}x'\\ y'\end{pmatrix}+\begin{pmatrix}\varepsilon'_{x'x'}\,x'\\ \varepsilon'_{y'y'}\,y'\end{pmatrix}$$

The values of $\varepsilon'_{x'x'}$ and $\varepsilon'_{y'y'}$ can be found as eigenvalues of the original strain matrix, i.e. solutions of the equation

$$f(\lambda)=\det\begin{bmatrix}\varepsilon_{xx}-\lambda & \varepsilon_{xy}\\ \varepsilon_{xy} & \varepsilon_{yy}-\lambda\end{bmatrix}=0$$

The eigenvectors indicate the directions of pure strain, i.e. in that coordinate system you find no shear strain.

So far all calculations were made in the plane, but they can readily be adapted to solids in space. If the deformation of a solid is given by the deformation vector filed $\vec{u}$, i.e.

$$\vec{x}=\begin{pmatrix}x\\ y\\ z\end{pmatrix}\longrightarrow\vec{x}+\vec{u}=\begin{pmatrix}x\\ y\\ z\end{pmatrix}+\begin{pmatrix}u_1\\ u_2\\ u_3\end{pmatrix}$$

symbol	formula	interpretation
ε_{xx}	$\frac{\partial u_1}{\partial x}$	ratio of change of length divided by length in x direction
ε_{yy}	$\frac{\partial u_2}{\partial y}$	ratio of change of length divided by length in y direction
ε_{zz}	$\frac{\partial u_3}{\partial z}$	ratio of change of length divided by length in z direction
$\varepsilon_{xy} = \varepsilon_{yx}$	$\frac{1}{2}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right)$	the angle between the x and y axis is diminished by $2\,\varepsilon_{xy}$
$\varepsilon_{xz} = \varepsilon_{zx}$	$\frac{1}{2}\left(\frac{\partial u_1}{\partial z} + \frac{\partial u_3}{\partial x}\right)$	the angle between the x and z axis is diminished by $2\,\varepsilon_{xz}$
$\varepsilon_{yz} = \varepsilon_{zy}$	$\frac{1}{2}\left(\frac{\partial u_2}{\partial z} + \frac{\partial u_3}{\partial y}\right)$	the angle between the y and z axis is diminished by $2\,\varepsilon_{yz}$

Table 9.1: Normal and shear strains in space

then we can compute the three normal and three strain components by the formulas in table 9.1[3].

The above results about transformation of strains in a rotated coordinate system do also apply. Thus for a given strain there is a rotation of the coordinate system, given by the orthonormal matrix $\mathbf{R}$ such that

$$\begin{bmatrix} \varepsilon'_{x'x'} & 0 & 0 \\ 0 & \varepsilon'_{y'y'} & 0 \\ 0 & 0 & \varepsilon'_{z'z'} \end{bmatrix} = \mathbf{R}^T \cdot \begin{bmatrix} \varepsilon_{xx} & \varepsilon_{xy} & \varepsilon_{xz} \\ \varepsilon_{yx} & \varepsilon_{yy} & \varepsilon_{yz} \\ \varepsilon_{zx} & \varepsilon_{zy} & \varepsilon_{zz} \end{bmatrix} \cdot \mathbf{R}$$

9.1.2 Description of stress

For sake of simplicity we first consider again only planar situations and at the end of the section apply the obvious extensions to the realistic situation in space.

Consider an elastic body where all forces are parallel to the xy plane and the contour of the solid is independent on z. We assume all stress forces are parallel to the xy plane an independent on z. Now consider a small rectangular box of this solid with width Δx, height Δy and depth Δz. A cut parallel to the xy plane is shown in figure 9.4. Based on the formula

$$\text{stress} = \frac{\text{force}}{\text{area}}$$

we now examine the **normal stress** and **tangential stress** components on the surfaces of this rectangle. We assume that the small box is an a static situation and there are no body forces. Balancing all components of forces and moments leads to the conditions

$$\sigma_x^2 = \sigma_x^1 \quad , \quad \sigma_y^3 = \sigma_y^4 \quad , \quad \tau_{yx}^1 = \tau_{yx}^2 \quad , \quad \tau_{xy}^3 = \tau_{xy}^4$$

Thus the situation simplifies as shown on the right in figure 9.4.

The stress situation of a solid is described by all components of the stress, typically as functions of the location.

Normal and tangential stress in an arbitrary direction

Figure 9.5 shows an virtual cut in a sold such that the normal vector $\vec{n} = (\cos\alpha, \sin\alpha)^T$ forms an angle α with the x axis. Now examine the normal stress σ and the tangential stress τ.

[3] In part of the literature (e.g. [Prze68]) the shear stresses are defined without the division by 2. All results can be adapted accordingly

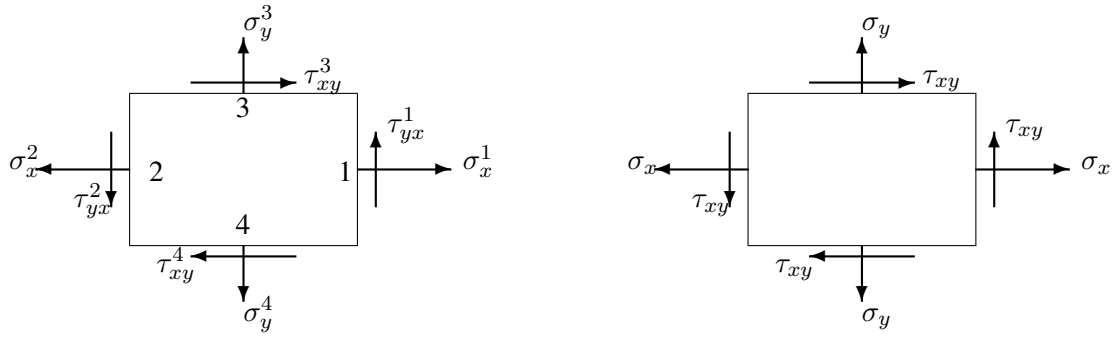


Figure 9.4: Definition of stress in a plane, initial (left) and simplified (right) situation

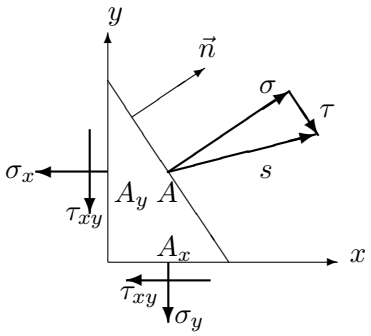


Figure 9.5: Normal and tangential stress in an arbitrary direction

Since $A_x = A\ \sin\ \alpha$ and $A_y = A\ \cos\ \alpha$ the condition of balance of force leads to

$$\begin{aligned} s_x\ A = \sigma_x\ A_y + \tau_{xy}\ A_x \quad &\Longrightarrow \quad s_x = \sigma_x\ \cos\ \alpha + \tau_{xy}\ \sin\ \alpha \\ s_y\ A = \sigma_y\ A_x + \tau_{xy}\ A_y \quad &\Longrightarrow \quad s_y = \tau_{xy}\ \cos\ \alpha + \sigma_y\ \sin\ \alpha \end{aligned}$$

where $\vec{s} = (s_x, s_y)^T$. Using matrices we may write

$$\begin{pmatrix} s_x \\ s_y \end{pmatrix} = \begin{bmatrix} \sigma_x & \tau_{xy} \\ \tau_{xy} & \sigma_y \end{bmatrix} \begin{pmatrix} \cos\ \alpha \\ \sin\ \alpha \end{pmatrix} \quad \text{or} \quad \vec{s} = \mathbf{S} \cdot \vec{n}$$

where the symmetric **stress matrix** is given by

$$\mathbf{S} = \begin{bmatrix} \sigma_x & \tau_{xy} \\ \tau_{xy} & \sigma_y \end{bmatrix}$$

The stress vector $\vec{s}$ may be decomposed in a normal component σ and a tangential component τ. We find as component of $\vec{\sigma}$ in the direction of $\vec{n}$

$$\begin{aligned} \sigma &= \langle \vec{n}\ ,\ \vec{s} \rangle = \vec{n}^T \cdot \vec{s} \\ &= (\cos\ \alpha, \sin\ \alpha) \cdot \begin{bmatrix} \sigma_x & \tau_{xy} \\ \tau_{xy} & \sigma_y \end{bmatrix} \begin{pmatrix} \cos\ \alpha \\ \sin\ \alpha \end{pmatrix} \\ \tau &= (-\sin\ \alpha, \cos\ \alpha) \cdot \begin{bmatrix} \sigma_x & \tau_{xy} \\ \tau_{xy} & \sigma_y \end{bmatrix} \begin{pmatrix} \cos\ \alpha \\ \sin\ \alpha \end{pmatrix} \end{aligned}$$

The value of τ is positive if $\vec{\tau}$ is point out of the solid and σ is positive if $\vec{\sigma}$ is pointing upward in figure 9.4.

This allows us to consider a new coordinate system, generated by rotation the xy system by an angle ϕ (see figure 9.3, page 207). We obtain

$$\begin{aligned} \sigma_{x'} &= (\cos\ \phi, \sin\ \phi) \cdot \begin{bmatrix} \sigma_x & \tau_{xy} \\ \tau_{xy} & \sigma_y \end{bmatrix} \begin{pmatrix} \cos\ \phi \\ \sin\ \phi \end{pmatrix} \\ \sigma_{y'} &= (-\sin\ \phi, \cos\ \phi) \cdot \begin{bmatrix} \sigma_x & \tau_{xy} \\ \tau_{xy} & \sigma_y \end{bmatrix} \begin{pmatrix} -\sin\ \phi \\ \cos\ \phi \end{pmatrix} \\ \tau_{x'y'} &= (-\sin\ \phi, \cos\ \phi) \cdot \begin{bmatrix} \sigma_x & \tau_{xy} \\ \tau_{xy} & \sigma_y \end{bmatrix} \begin{pmatrix} \cos\ \phi \\ \sin\ \phi \end{pmatrix} \end{aligned}$$

An elementary matrix multiplication shows that this is equivalent to

$$\begin{bmatrix} \sigma_{x'} & \tau_{x'y'} \\ \tau_{x'y'} & \sigma_{y'} \end{bmatrix} = \begin{bmatrix} \cos\ \phi & \sin\ \phi \\ -\sin\ \phi & \cos\ \phi \end{bmatrix} \cdot \begin{bmatrix} \sigma_x & \tau_{xy} \\ \tau_{xy} & \sigma_y \end{bmatrix} \cdot \begin{bmatrix} \cos\ \phi & -\sin\ \phi \\ \sin\ \phi & \cos\ \phi \end{bmatrix}$$

This transformation formula should be compared with result9–3 on page 207. It show that the behavior transformation under coordinate rotations for the stress matrix and the strain matrix is the same.

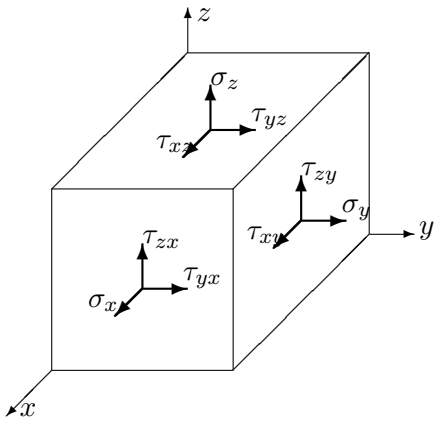


Figure 9.6: Components of stress in space

symbol	description
σ_x	normal stress at a surface orthogonal to $x = \text{const}$
σ_y	normal stress at a surface orthogonal to $y = \text{const}$
σ_z	normal stress at a surface orthogonal to $z = \text{const}$
$\tau_{xy} = \tau_{yx}$	tangential stress in y direction at surface orthogonal to $x = \text{const}$ tangential stress in x direction at surface orthogonal to $y = \text{const}$
$\tau_{xz} = \tau_{zx}$	tangential stress in z direction at surface orthogonal to $x = \text{const}$ tangential stress in x direction at surface orthogonal to $z = \text{const}$
$\tau_{yz} = \tau_{zy}$	tangential stress in z direction at surface orthogonal to $y = \text{const}$ tangential stress in y direction at surface orthogonal to $z = \text{const}$

Table 9.2: Description of normal and tangential stress in space

Normal and tangential stress in space

All the above observation can be adapted to the situation in space. Figure 9.6 shows the notational convention and table 9.2 gives a short description.

The symmetric **stress matrix** S is given by

$$\mathbf{S} = \begin{bmatrix} \sigma_x & \tau_{xy} & \tau_{xz} \\ \tau_{xy} & \sigma_y & \tau_{yz} \\ \tau_{xz} & \tau_{yz} & \sigma_z \end{bmatrix}$$

and the stress vector $\vec{s}$ at a plane orthogonal to $\vec{n}$ is given by

$$\vec{s} = \mathbf{S} \cdot \vec{n}$$

The behavior of $\mathbf{S}$ under rotation of the coordinate system $\vec{x}' = \mathbf{R} \cdot \vec{x}$ is given by

$$\mathbf{S}' = \begin{bmatrix} \sigma'_x & \tau'_{xy} & \tau'_{xz} \\ \tau'_{xy} & \sigma'_y & \tau'_{yz} \\ \tau'_{xz} & \tau'_{yz} & \sigma'_z \end{bmatrix} = \mathbf{R}^T \cdot \begin{bmatrix} \sigma_x & \tau_{xy} & \tau_{xz} \\ \tau_{xy} & \sigma_y & \tau_{yz} \\ \tau_{xz} & \tau_{yz} & \sigma_z \end{bmatrix} \cdot \mathbf{R}$$

When solving the cubic equation

$$\det(S - \lambda \mathbb{I}_3) = \det \begin{bmatrix} \sigma_x - \lambda & \tau_{xy} & \tau_{xz} \\ \tau_{xy} & \sigma_y - \lambda & \tau_{yz} \\ \tau_{xz} & \tau_{yz} & \sigma_z - \lambda \end{bmatrix} = 0$$

for the three eigenvalues $\lambda_{1,2,3}$ and the corresponding orthonormal eigenvectors $\vec{e}_1$, $\vec{e}_2$ and $\vec{e}_3$, we compute a coordinate system in which all tangential stress components vanish. We have only normal stresses, i.e. the stress matrix $\mathbf{S}'$ has the form

$$\begin{bmatrix} \sigma'_x & 0 & 0 \\ 0 & \sigma'_y & 0 \\ 0 & 0 & \sigma'_z \end{bmatrix}$$

This can be very useful to extract results out of stress computations. When asked to find the stress at a given point in a solid many different forms of answers are possible:

- Give all six components of the stress in a given coordinate system.
- Find the three normal stresses and render those as a result. One might also give the corresponding directions.
- Give the maximal normal stress.
- Give the maximal and minimal normal stress.
- Give the von Mises stress

The 'correct' form of answer depends on the context.

9.1.3 Von Mises stress

$$\begin{aligned}\sigma_M^2 &= \sigma_x^2+\sigma_y^2+\sigma_z^2-\sigma_x\,\sigma_y-\sigma_y\,\sigma_z-\sigma_z\,\sigma_x+3\,\tau_{xy}^2+3\,\tau_{yz}^2+3\,\tau_{zx}^2\\ &= \frac{1}{2}\left((\sigma_x-\sigma_y)^2+(\sigma_y-\sigma_z)^2+(\sigma_z-\sigma_x)^2\right)+3\left(\tau_{xy}^2+\tau_{yz}^2+\tau_{zx}^2\right)\end{aligned}$$

This expression is independent on orientation of coordinate system. If reduced to pure stress (no tangential stresses) we find

$$2\,\sigma_M^2=(\sigma_x'-\sigma_y')^2+(\sigma_y'-\sigma_z')^2+(\sigma_z'-\sigma_x')^2$$

Thus the von Mises stress is a measure of the **differences** among the three principal stresses.

9.2 Hooke's law and elastic energy

Using tables 9.1 (page 209) and 9.2 (page 212) we can now formulate the basic connection between the geometric deformations (strain) and the resulting forces (stress). It is a basic physical law, confirmed by many measurements. The show formulation is valid as long as all stress and strains are small. For large strains we would have to enter the field of nonlinear elasticity.

9.2.1 Hooke's law

This is the general form of **Hooke's law** for a homogeneous material. This is the foundation of linear elasticity and any book on elasticity see will show a formulation, e.g. [Prze68, §2.2][4], [Wein74, §10.1]).

$$\begin{pmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\\ \varepsilon_{xy}\\ \varepsilon_{xz}\\ \varepsilon_{yz}\end{pmatrix}=\frac{1}{E}\begin{bmatrix}1 & -\nu & -\nu & & & \\ -\nu & 1 & -\nu & & 0 & \\ -\nu & -\nu & 1 & & & \\ & & & 1+\nu & & \\ & 0 & & & 1+\nu & \\ & & & & & 1+\nu\end{bmatrix}\cdot\begin{pmatrix}\sigma_x\\ \sigma_y\\ \sigma_z\\ \tau_{xy}\\ \tau_{xz}\\ \tau_{yz}\end{pmatrix}\tag{9.1}$$

or by inverting the matrix

$$\begin{pmatrix}\sigma_x\\ \sigma_y\\ \sigma_z\\ \tau_{xy}\\ \tau_{xz}\\ \tau_{yz}\end{pmatrix}=\frac{E}{(1+\nu)\,(1-2\,\nu)}\begin{bmatrix}1-\nu & \nu & \nu & & & \\ \nu & 1-\nu & \nu & & 0 & \\ \nu & \nu & 1-\nu & & & \\ & & & 1-2\,\nu & & \\ & 0 & & & 1-2\,\nu & \\ & & & & & 1-2\,\nu\end{bmatrix}\cdot\begin{pmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\\ \varepsilon_{xy}\\ \varepsilon_{xz}\\ \varepsilon_{yz}\end{pmatrix}\tag{9.2}$$

With the obvious notation equation (9.2) may be written in the form

$$\vec{\sigma}=\mathbf{H}\cdot\vec{\varepsilon}$$

Observe that the equations decouple and we can equivalently write

[4] The missing factors 2 are due to the different definition of the shear strains

$$\begin{pmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \end{pmatrix} = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu \\ \nu & 1-\nu & \nu \\ \nu & \nu & 1-\nu \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \end{pmatrix}$$

$$\begin{pmatrix} \tau_{xy} \\ \tau_{xz} \\ \tau_{yz} \end{pmatrix} = \frac{E}{(1+\nu)} \begin{pmatrix} \varepsilon_{xy} \\ \varepsilon_{xz} \\ \varepsilon_{yz} \end{pmatrix} \tag{9.3}$$

Now we want to find the formula for the **energy density** of a deformed body. For this we consider a small block width Δx, height Δy and depth Δz, located at the fixed origin. For a fixed displacement vector $\vec{u}$ of the corner $P = (\Delta x, \Delta y, \Delta z)$ we deform the block by a sequence of affine deformations, such that P point moves along straight line. The displacement vector of P is given by the formula $t\,\vec{u}$ where t varies from 0 to 1. If the final strain is denoted by $\vec{\varepsilon}$ then the strains during the deformation are given by $t\,\varepsilon$. Accordingly the stresses are given by $t\,\sigma$ where the final tress σ can be computed by Hooke's law (e.g. equation (9.3)). Now we compute the total work needed to deform this block, using the basic formula work = force · distance. There are six different contributions:

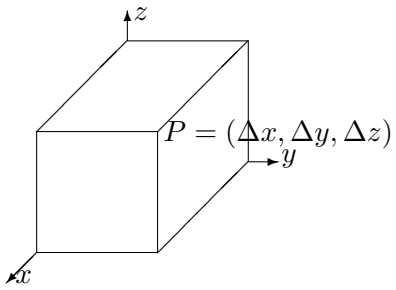


Figure 9.7: Block to be deformed to determine the elastic energy

- The face $x = \Delta x$ moves from Δx to $\Delta x\,(1 + \varepsilon_{xx})$. For a time step dt at time $0 < t < 1$ the distance is thus $t\,\varepsilon_{xx}\,\Delta x$. The force is determined by th area $\Delta y \cdot \Delta z$ and the normal strain σ_x. The first energy contribution can now be integrated by

$$\int_0^1 (\Delta y \cdot \Delta z \cdot \sigma_x) \cdot t\,\varepsilon_{xx}\,\Delta x\; dt = \Delta y \cdot \Delta z \cdot \Delta x \cdot \sigma_x \cdot \varepsilon_{xx} \int_0^1 t\; dt = \frac{1}{2}\,\Delta V \cdot \sigma_x \cdot \varepsilon_{xx}$$

- Similarly normal displacement of the faces at $y = \Delta y$ and $z = \Delta z$ lead to contributions

$$\frac{1}{2}\,\Delta V \cdot \sigma_y \cdot \varepsilon_{yy} \quad \text{and} \quad \frac{1}{2}\,\Delta V \cdot \sigma_z \cdot \varepsilon_{zz}$$

- The face $x = \Delta x$ also moves tangentially from $(1,0,0)^T \Delta x$ to $(1, \frac{\partial u_2}{\partial x}, \frac{\partial u_3}{\partial x})^T \Delta x$. The forces along this path are $t\,(0, \sigma_{yx}, \sigma_{zx})^T \Delta y \Delta z$. And again a similar integration leads to the energy contribution

$$\frac{1}{2}\,(\Delta y \cdot \Delta z \cdot \Delta x) \cdot \left(\frac{\partial\, u_2}{\partial x}\,\sigma_{yx} + \frac{\partial\, u_3}{\partial x}\,\sigma_{zx} \right) = \frac{1}{2}\,\Delta V \cdot \left(\frac{\partial\, u_2}{\partial x}\,\sigma_{yx} + \frac{\partial\, u_3}{\partial x}\,\sigma_{zx} \right)$$

- From the other two faces we obtain similar contributions

$$\frac{1}{2}\,\Delta V\cdot\left(\frac{\partial\,u_1}{\partial y}\,\sigma_{xy}+\frac{\partial\,u_3}{\partial y}\,\sigma_{zy}\right)\quad\text{and}\quad\frac{1}{2}\,\Delta V\cdot\left(\frac{\partial\,u_1}{\partial z}\,\sigma_{xz}+\frac{\partial\,u_2}{\partial z}\,\sigma_{yz}\right)$$

Adding all six contributions and then dividing by the volume ΔV we obtain the energy density

$$\begin{aligned}e &= \frac{1}{2}\,(\sigma_x\varepsilon_{xx}+\sigma_y\varepsilon_{yy}+\sigma_z\varepsilon_{zz})+\\ &\quad+\frac{1}{2}\left((\frac{\partial\,u_2}{\partial x}+\frac{\partial\,u_1}{\partial y})\,\sigma_{xy}+(\frac{\partial\,u_3}{\partial x}+\frac{\partial\,u_1}{\partial z})\,\sigma_{xz}+(\frac{\partial\,u_3}{\partial y}+\frac{\partial\,u_2}{\partial z})\,\sigma_{yz}\right)\\ &= \frac{1}{2}\,(\sigma_x\varepsilon_{xx}+\sigma_y\varepsilon_{yy}+\sigma_z\varepsilon_{zz})+(\sigma_{xy}\varepsilon_{xy}+\sigma_{xz}\varepsilon_{xz}+\sigma_{yz}\varepsilon_{yz})\end{aligned}$$

This can be written as scalar product in the form

$$e=\frac{1}{2}\,\langle\begin{pmatrix}\sigma_x\\ \sigma_y\\ \sigma_z\end{pmatrix},\begin{pmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\end{pmatrix}\rangle+\langle\begin{pmatrix}\tau_{xy}\\ \tau_{yz}\\ \tau_{zx}\end{pmatrix},\begin{pmatrix}\varepsilon_{xy}\\ \varepsilon_{yz}\\ \varepsilon_{zx}\end{pmatrix}\rangle \tag{9.4}$$

or according to Hooke's law in the form of equation (9.3) also as

$$e = \frac{1}{2}\,\frac{E}{(1+\nu)\,(1-2\,\nu)}\,\langle\begin{bmatrix}1-\nu & \nu & \nu\\ \nu & 1-\nu & \nu\\ \nu & \nu & 1-\nu\end{bmatrix}\cdot\begin{pmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\end{pmatrix},\begin{pmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\end{pmatrix}\rangle+ \tag{9.5}$$

$$+\frac{E}{(1+\nu)}\,\langle\begin{pmatrix}\varepsilon_{xy}\\ \varepsilon_{xz}\\ \varepsilon_{yz}\end{pmatrix},\begin{pmatrix}\varepsilon_{xy}\\ \varepsilon_{xz}\\ \varepsilon_{yz}\end{pmatrix}\rangle \tag{9.6}$$

9.2.2 Some exemplary situations

Now we illustrate Hooke's law by considering a few simple examples.

9–4 Example : Hooke's basic law
Consider the situation in figure 9.8 with the following assumptions:

- The solid of length L has constant cross section perpendicular to the x axis, with area $A=\Delta y\cdot\Delta z$.
- The left face is fixed in the x direction, but free to move in the other directions.
- The constant normal stress σ_x at the right face is given by $\sigma_x=\frac{F}{A}$.
- There are no forces in the y and z directions.

This leads to the consequences:

- All stresses in y and z direction vanish, i.e.

$$\sigma_y=\sigma_z=\tau_{xy}=\tau_{xz}=\tau_{yz}=0$$

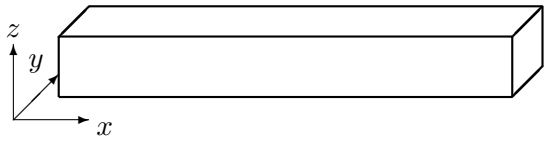


Figure 9.8: Situation for the most elementary versions of Hooke's law

- Hooke's law (9.1) implies

$$\begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \end{pmatrix} = \frac{1}{E} \begin{bmatrix} 1 & -\nu & -\nu \\ -\nu & 1 & -\nu \\ -\nu & -\nu & 1 \end{bmatrix} \cdot \begin{pmatrix} \frac{F}{A} \\ 0 \\ 0 \end{pmatrix} = \frac{1}{E} \frac{F}{A} \begin{pmatrix} 1 \\ -\nu \\ -\nu \end{pmatrix}$$

- The first component of the above equations leads to the classical, most simple form of Hooke's law

$$\varepsilon_{xx} = \frac{\Delta L}{L} = \frac{1}{E} \frac{F}{A}$$

  This is the standard definition of Young's **modulus of elasticity**. The solid is stretched by a factor $\left(1 + \frac{1}{E} \frac{F}{A}\right)$.

- In the y and z direction the solid is contracted by a factor of $\left(1 - \frac{\nu}{E} \frac{F}{A}\right)$. This is a simple interpretation of **Poisson's ratio** ν.

$$\varepsilon_{yy} = -\nu \, \varepsilon_{xx}$$

  Multiply the relative change if length in the x direction by ν to obtain the relative change if length in the y and z direction. One would expect $\nu \geq 0$.

- The energy density e can be found by equation (9.5)

$$\begin{aligned} e &= \frac{1}{2} \left\langle \begin{pmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \end{pmatrix}, \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \end{pmatrix} \right\rangle + \frac{1}{2} \left\langle \begin{pmatrix} \tau_{xy} \\ \tau_{xz} \\ \tau_{yz} \end{pmatrix}, \begin{pmatrix} \varepsilon_{xy} \\ \varepsilon_{xz} \\ \varepsilon_{yz} \end{pmatrix} \right\rangle \\ &= \frac{1}{2} \frac{1}{E} \left(\frac{F}{A}\right)^2 \left\langle \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ -\nu \\ -\nu \end{pmatrix} \right\rangle + 0 = \frac{1}{2} \frac{1}{E} \left(\frac{F}{A}\right)^2 = \frac{1}{2} E \, \varepsilon_{xx}^2 \end{aligned}$$

♢

9–5 Example : Solid under constant pressure

If a rectangular block is submitted to a constant pressure p then we know all components of the stress (assuming they are constant throughout the solid), namely

$$\sigma_x = \sigma_\tau = \sigma_z = -p \quad \text{and} \quad \tau_{xy} = \tau_{xz} = \tau_{yz} = 0$$

Hooke's law now leads to

$$\varepsilon_{xy} = \varepsilon_{xz} = \varepsilon_{yz} = 0$$

and

$$\begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \end{pmatrix} = \frac{1}{E} \begin{bmatrix} 1 & -\nu & -\nu \\ -\nu & 1 & -\nu \\ -\nu & -\nu & 1 \end{bmatrix} \cdot \begin{pmatrix} p \\ p \\ p \end{pmatrix} = -\frac{p\,(1-2\nu)}{E} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

i.e. in each direction the solid is compressed by a factor of $1 - \frac{p\,(1-2\nu)}{E}$. Since putting a solid under pressure should make it shrink the Poisson's ratio must satisfy the condition $0 \le \nu \le \frac{1}{2}$.

Since each direction is compressed by the same factor we obtain the relative change of volume

$$\frac{\Delta V}{V} = 1 + \left(1 - \frac{p\,(1-2\nu)}{E}\right)^3 \approx 3\,\frac{p\,(1-2\nu)}{E}$$

is the pressure is small enough.

The energy density e is given by

$$\begin{aligned} e &= \frac{1}{2} \left\langle \begin{pmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \end{pmatrix}, \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \end{pmatrix} \right\rangle + \frac{1}{2} \left\langle \begin{pmatrix} \tau_{xy} \\ \tau_{xz} \\ \tau_{yz} \end{pmatrix}, \begin{pmatrix} \varepsilon_{xy} \\ \varepsilon_{xz} \\ \varepsilon_{yz} \end{pmatrix} \right\rangle \\ &= -\frac{1}{2}\,\frac{p\,(1-2\nu)}{E} \left\langle \begin{pmatrix} -p \\ -p \\ -p \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \right\rangle + 0 = \frac{1}{2}\,\frac{(1-2\nu)}{E}\,3\,p^2 \end{aligned}$$

♢

9–6 Example : Shear modulus

To the block in figure 9.7 we apply a force of strength F in direction of the x axis to the top (area $\Delta x \cdot \Delta y$). No forces in the y direction are applied. The corresponding forces have to be applied to the faces at $x = 0$ and $x = \Delta x$ for the block to be in equilibrium. No other forces apply. We find

$$\tau_{xz} = \frac{F}{A} \quad \text{and} \quad \tau_{xy} = \tau_{yz} = \sigma_x = \sigma_y = \sigma_z = 0$$

Now Hooke's law (9.1) leads to

$$\varepsilon_{xx} = \varepsilon_{yy} = \varepsilon_{zz} = \varepsilon_{xy} = \varepsilon_{yz} = 0$$

and

$$\varepsilon_{xz} = \frac{1+\nu}{E}\,\frac{F}{A}$$

This is the reason why some presentations introduce the **shear modulus**[5] $G = \frac{E}{1+\nu}$. ♢

9.3 Volume and surface forces, thermoelasticity

9.3.1 Volume forces

A force applied to the volume of the solid can be introduced by means of a volume force density $\vec{f}$ (units: $\frac{\mathrm{N}}{\mathrm{m}^3}$). By adding the potential energy

$$U_{Vol} = -\iiint_{\Omega} \vec{f} \cdot \vec{u}\; dV$$

to the elastic energy and then minimizing we are lead to the correct force term.

[5] Some books define strains without the factor $\frac{1}{2}$. Then the shear modulus will be given by $G = \frac{E}{2\,(1+\nu)}$

9.3.2 Surface forces

By adding a surface potential energy, using the surface force density $\vec{g}$ (units: $\frac{\mathrm{N}}{\mathrm{m}^2}$),

$$U_{Surf} = -\iint\limits_{\partial\Omega} \vec{g}\cdot\vec{u}\,dA$$

we can also consider forces applied to the surface only.

9.3.3 Thermoelasticity

The effects of temperature changes can be introduced to the theory of linear elasticity. Let T denote the temperature change. For $T > 0$ most material will expand slightly. This material property can be described by α, the coefficient of linear thermal expansion. The stress strain relations have to be adapted accordingly. We find (e.g. [Prze68, §2.2], [Sout73, §2.8])

$$\alpha\,T\,(1\,,\,1\,,\,1\,,\,0\,,\,0\,,\,0)^T \quad \text{to be added to RHS in equation (9.1)}$$

$$\alpha\,T\,\frac{E}{1-2\,\nu}\,(1\,,\,1\,,\,1\,,\,0\,,\,0\,,\,0)^T \quad \text{to be subtracted from RHS in equation (9.2)}$$

If a material is expanded by a constant temperature change, the Hooke's law

$$\begin{pmatrix}\sigma_x\\ \sigma_y\\ \sigma_z\end{pmatrix} = \frac{E}{(1+\nu)\,(1-2\,\nu)}\begin{bmatrix}1-\nu & \nu & \nu\\ \nu & 1-\nu & \nu\\ \nu & \nu & 1-\nu\end{bmatrix}\cdot\begin{pmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\end{pmatrix} - \alpha\,T\,\frac{E}{1-2\,\nu}\begin{pmatrix}1\\1\\1\end{pmatrix}$$

leads to

$$\frac{E}{(1+\nu)\,(1-2\,\nu)}\begin{bmatrix}1-\nu & \nu & \nu\\ \nu & 1-\nu & \nu\\ \nu & \nu & 1-\nu\end{bmatrix}\cdot\begin{pmatrix}\alpha T\\ \alpha T\\ \alpha T\end{pmatrix} - \alpha\,T\,\frac{E}{1-2\,\nu}\begin{pmatrix}1\\1\\1\end{pmatrix} = \begin{pmatrix}0\\0\\0\end{pmatrix}$$

This corresponds to the fact that no external stresses apply.

The computations leading to the energy density in equation (9.4) can be modified to take the additional forces into account. In computing the energy needed to move the top, the normal strain τ_z has to be decreased by $\alpha\ T\ \frac{E}{1-2\,\nu}$, independent on the strain. This leads to an additional term $-\alpha\ T\ \frac{E}{1-2\,\nu}\ \varepsilon_{xx}$ in the energy. Adding up the terms from all sides of the box we obtain an energy density, due to the temperature change of

$$-\alpha\,T\,\frac{E}{1-2\,\nu}\ (\varepsilon_{xx}+\varepsilon_{yy}+\varepsilon_{zz})$$

This can be integrated to find the contribution to the total energy

$$U_{Thermo} = -\iiint\limits_{\Omega} \alpha\,T\,\frac{E}{1-2\,\nu}\ (\varepsilon_{xx}+\varepsilon_{yy}+\varepsilon_{zz})\ dV \tag{9.7}$$

As another simple example consider a rectangular block, heated to temperature $T > 0$. The total energy of the block is (assuming constant strains and stresses and $\varepsilon_{xy} = \varepsilon_{xz} = \varepsilon_{yz} = 0$) given by

$$\frac{E}{V} = \frac{1}{2}\,\frac{E}{(1+\nu)\,(1-2\,\nu)}\left\langle\begin{bmatrix}1-\nu & \nu & \nu\\ \nu & 1-\nu & \nu\\ \nu & \nu & 1-\nu\end{bmatrix}\cdot\begin{pmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\end{pmatrix},\begin{pmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\end{pmatrix}\right\rangle -$$

$$-\frac{\alpha\,T\,E}{1-2\,\nu}\left\langle\begin{pmatrix}\varepsilon_{xx}\\ \varepsilon_{yy}\\ \varepsilon_{zz}\end{pmatrix},\begin{pmatrix}1\\1\\1\end{pmatrix}\right\rangle$$

This expression is minimal (as function of $\vec{\varepsilon}$) iff

$$\frac{1}{(1+\nu)} \begin{bmatrix} 1-\nu & \nu & \nu \\ \nu & 1-\nu & \nu \\ \nu & \nu & 1-\nu \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \end{pmatrix} = \alpha\, T \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

According to Hooke's law in the form (9.2) (modified for the contribution of the heat expansion) this leads to a situation with vanishing stresses, i.e. no external forces applied. Thus if no extrenal forces are applied, the deformation is such that the total energy is minimal. This is another example of the principle of least energy.

9.4 Torsion of a shaft

In this section we discuss the torsion of a shaft with constant cross section. Based on a few assumtions we will determine the deformation of the shaft under torsion. The problem is treated in [Sout73, §12].

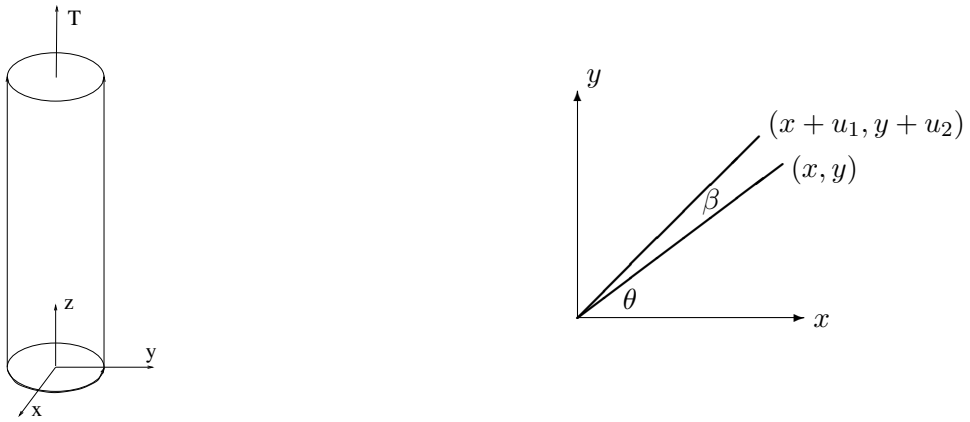


Figure 9.9: Torsion of a shaft

9.4.1 Basic description

Consider a vertical shaft with constant cross section. The centers of gravity of the cross section are along the z axis and the bottom of the shaft is fixed. The top surface is twisted by a total torque T. The situation of a circular cross section is shown in figure 9.9. We do not specify exactly how the forces and twisting moments are applied to the two ends. Based on Saint-Venant principle (see [Sout73, §5.6]) we assume that the stress distribution in the cross sections does not depend on z, except very close to the two ends. We assume that the twisting leads to a rotation of each cross section by an angle β where

$$\beta = z \cdot \alpha$$

The constant α is a measure of the change of angle per unit length of the shaft. Its value α has to be determined, using the moment T. Based on this we determine the horizontal displacements for small angles β by the right part of figure 9.9 and a linear approximation

$$\begin{aligned} u_1(x,y) &= r\,\cos(\beta+\theta) - r\,\cos(\theta) \approx -\beta\, r\,\sin\theta = -y\,\beta = -y\,z\,\alpha \\ u_2(x,y) &= r\,\sin(\beta+\theta) - r\,\sin(\theta) \approx +\beta\, r\,\cos\theta = +x\,\beta = +x\,z\,\alpha \end{aligned}$$

We also assume that the vertical displacement is independent of z and given by a warping function $\phi(x, y)$. This leads to the displacements

$$u_1 = -y\,z\,\alpha \quad , \quad u_2 = x\,z\,\alpha \quad , \quad u_3 = \alpha\,\phi(x,y)$$

and thus the strain components

$$\varepsilon_{xx} = \varepsilon_{yy} = \varepsilon_{zz} = \varepsilon_{xy} = 0 \quad , \quad \varepsilon_{xz} = -\frac{1}{2}\,\alpha\,y + \frac{1}{2}\,\alpha\,\frac{\partial\,\phi}{\partial x} \quad , \quad \varepsilon_{yz} = \frac{1}{2}\,\alpha\,x + \frac{1}{2}\,\alpha\,\frac{\partial\,\phi}{\partial y}$$

Using Hooke's law we find the stress components

$$\sigma_x = \sigma_y = \sigma_z = \tau_{xy} = 0 \quad , \quad \tau_{xz} = \frac{E\,\alpha}{2\,(1+\nu)}\,(-y + \frac{\partial\,\phi}{\partial x}) \quad , \quad \tau_{yz} = \frac{E\,\alpha}{2\,(1+\nu)}\,(x + \frac{\partial\,\phi}{\partial y})$$

Using the stresses we can determine the horizontal forces and the torsion along a hypothetical horizontal cross section. Since the origin is the center of gravity of the cross section Ω the first moments vanish and we find

$$\begin{aligned}
F_x &= \iint\limits_\Omega \tau_{xz}\,dA = \frac{E\,\alpha}{2\,(1+\nu)} \iint\limits_\Omega -y + \frac{\partial\,\phi}{\partial x}\,dA = 0 \\
F_y &= \iint\limits_\Omega \tau_{yz}\,dA = \frac{E\,\alpha}{2\,(1+\nu)} \iint\limits_\Omega x + \frac{\partial\,\phi}{\partial y}\,dA = 0 \\
T &= \iint\limits_\Omega x\,\tau_{yz} - y\,\tau_{yz}\,dA \\
&= \frac{E\,\alpha}{2\,(1+\nu)} \iint\limits_\Omega x\,(x + \frac{\partial\,\phi}{\partial y}) - y\,(-y + \frac{\partial\,\phi}{\partial x})\,dA \\
&= \frac{E\,\alpha}{2\,(1+\nu)} \iint\limits_\Omega x^2 + y^2 + x\,\frac{\partial\,\phi}{\partial y} - y\,\frac{\partial\,\phi}{\partial x}\,dA = \frac{E\,\alpha}{2\,(1+\nu)}\,J
\end{aligned}$$

Using the **torsional rigidity** J with

$$J = \iint\limits_\Omega x^2 + y^2 + x\,\frac{\partial\,\phi}{\partial y} - y\,\frac{\partial\,\phi}{\partial x}\,dA$$

we can now determine the constant α by

$$\alpha = \frac{2\,(1+\nu)}{J\,E}\,T$$

and thus for a shaft of height H the total change of angle β as

$$\beta = H \cdot \alpha = \frac{2\,(1+\nu)}{J\,E}\,H \cdot T$$

The only difficult part is to determine the function ϕ, then J is determined by an integration.

9.4.2 Deriving the differential equation, using calculus of variations

The above computations allow to compute the energy E in one cross section Ω as

$$\begin{aligned}
E &= \iint\limits_\Omega \varepsilon_{xz}\,\tau_{xz} + \varepsilon_{yz}\,\tau_{yz}\,dA = \frac{E\,\alpha}{4\,(1+\nu)} \iint\limits_\Omega (x + \frac{\partial\,\phi}{\partial y})^2 + (-y + \frac{\partial\,\phi}{\partial x})^2\,dA \\
&= \frac{E\,\alpha}{4\,(1+\nu)} \iint\limits_\Omega (\frac{\partial\,\phi}{\partial x})^2 + (\frac{\partial\,\phi}{\partial y})^2 - 2\,y\,\frac{\partial\,\phi}{\partial x} + 2\,x\,\frac{\partial\,\phi}{\partial y} + x^2 + y^2\,dA
\end{aligned}$$

Thus minimizing the total energy is equivalent to minimizing the functional

$$\begin{aligned} F(\phi) &= \iint_\Omega \frac{1}{2}\,\|\nabla\phi\|^2 - y\,\frac{\partial\,\phi}{\partial x} + x\,\frac{\partial\,\phi}{\partial y}\;dA = \iint_\Omega \frac{1}{2}\,\langle\nabla\phi\,,\,\nabla\phi\rangle + \langle\begin{pmatrix} -y \\ x \end{pmatrix}\,,\,\nabla\phi\rangle\;dA \\ &= \iint_\Omega \frac{1}{2}\,\langle\nabla\phi\,,\,\nabla\phi\rangle - 0\;dA + \oint_{\partial\Omega}\begin{pmatrix} -y \\ x \end{pmatrix}\cdot\vec{n}\;\phi\;ds \end{aligned}$$

For the last step we used Green's formula in Appendix A.3

$$\iint_\Omega f\;(\operatorname{div}\vec{v})\;dA \;=\; \oint_{\partial\Omega} f\,\vec{v}\cdot\vec{n}\;ds - \iint_\Omega (\operatorname{grad} f)\cdot\vec{v}\;dA$$

with $f = \psi$ and $\vec{v} = \nabla\phi + \begin{pmatrix} -y \\ x \end{pmatrix}$.

A necessary condition for this functional $F(\phi)$ to be minimized at ϕ is

$$\begin{aligned} 0 &= \iint_\Omega \langle\nabla\phi\,,\,\nabla\psi\rangle\;dA + \oint_{\partial\Omega}\begin{pmatrix} -y \\ x \end{pmatrix}\cdot\vec{n}\;\psi\;ds \\ &= -\iint_\Omega \operatorname{div}(\nabla\phi)\;\psi\;dA + \oint_{\partial\Omega}\left(\nabla\phi + \begin{pmatrix} -y \\ x \end{pmatrix}\right)\cdot\vec{n}\;\psi\;ds \quad \text{for all functions} \quad \psi \end{aligned}$$

Since this expression has to vanish for arbitrary testfunctions ψ we conclude

$$\begin{aligned} \operatorname{div}(\nabla\phi) = \nabla\nabla\phi &= 0 && \text{in the cross section} \quad \Omega \\ \nabla\phi\cdot\vec{n} &= \begin{pmatrix} y \\ -x \end{pmatrix}\cdot\vec{n} && \text{on the boundary} \quad \partial\Omega \end{aligned}$$

Since the stress components are given by

$$\sigma_x = \sigma_y = \sigma_z = \tau_{xy} = 0 \quad , \quad \tau_{xz} = \frac{E\,\alpha}{2\,(1+\nu)}\,(-y + \frac{\partial\,\phi}{\partial x}) \quad , \quad \tau_{yz} = \frac{E\,\alpha}{2\,(1+\nu)}\,(x + \frac{\partial\,\phi}{\partial y})$$

the boundary condition can be written as

$$\begin{pmatrix} \tau_{xz} \\ \tau_{yz} \end{pmatrix}\cdot\vec{n} = 0$$

This equation implies that there is no stress on the lateral surface of the shaft. This condition is consistent with the mechanical setup.

9.4.3 Uniqueness and existence of the solution

This leads to an elliptic partial differential equation of the type of equation (6.3) (page 142) in section 6.3

$$\begin{aligned} \nabla\cdot\nabla\phi &= 0 && \text{for} \quad (x,y)\in\Omega \\ \vec{n}\cdot\nabla\phi &= g_2 && \text{for} \quad (x,y)\in\partial\Omega \end{aligned}$$

where $g_2(x,y) = y\,n_1(x,y) - x\,n_2(x,y)$ is a known function along the boundary of the domain. Since we have no Dirichlet condition the solution can only be determined up to an additive constant. This corresponds to the obvious fact that we can move the shaft along the z axis, without changing the deformation. In a numerical calculation we add the condtion that the function ϕ has to vanish at an arbitrary chosen point.

If ϕ_1 and ϕ_2 are two solutions to the differential equation with identical RHS g_2, then

$$\begin{aligned} \nabla \cdot \nabla(\phi_1 - \phi_2) &= 0 \qquad \text{for} \quad (x,y) \in \Omega \\ \vec{n} \cdot \nabla(\phi_1 - \phi_2) &= 0 \qquad \text{for} \quad (x,y) \in \partial\Omega \end{aligned}$$

Multiplying this equation by $(\phi_1 - \phi_2)$ and applying Green's formula lead to

$$\iint\limits_{\Omega} \|\nabla(\phi_1 - \phi_2)\|^2 \, dA = 0$$

Since the expression under the integral is nonnegative we conclude that $\nabla(\phi_1 - \phi_2) =$ and thus the two solutions differ by a constant[6].

If ϕ is in fact a solution of the above problem, then the divergence theorem implies

$$0 = \iint\limits_{\Omega} 0 \, dA = \iint\limits_{\Omega} \operatorname{div} \nabla\phi \, dA = \oint\limits_{\partial\Omega} \vec{n} \cdot \nabla\phi \, ds = \oint\limits_{\partial\Omega} g_2 \, ds$$

Thus we have the necessary condition that the integral of g_2 along the boundary has to vanish. The divergence theorem implies

$$\oint\limits_{\partial\Omega} \vec{n} \cdot \begin{pmatrix} -y \\ x \end{pmatrix} ds = \iint\limits_{\Omega} \operatorname{div} \begin{pmatrix} -y \\ x \end{pmatrix} dA = \iint\limits_{\Omega} 0 \, da = 0$$

and thus the condition is satisfied. It can be shown that this condition implies that the problem does have a solution.

9.4.4 Torsion of a shaft with circular cross section

In the situation of a circular cross section with radius R we find

$$g_2 = \vec{n} \cdot \begin{pmatrix} y \\ -x \end{pmatrix} = \frac{1}{R} \begin{pmatrix} x \\ y \end{pmatrix} \cdot \begin{pmatrix} y \\ -x \end{pmatrix} = 0$$

and thus the partial differential equation has only the trivial solution $\phi = 0$. The torsional rigidity is given by

$$J = \iint\limits_{\Omega} x^2 + y^2 \, dA = 2\,\pi \int_0^R r^2 \, r \, dr = \frac{\pi}{2} R^4$$

and thus

$$\alpha = \frac{T\ 2\ (1+\nu)}{J\ E} = \frac{T\ 4\ (1+\nu)}{\pi\ E\ R^4}$$

The stress matrix has the structure

$$\begin{bmatrix} \sigma_x & \tau_{xy} & \tau_{xz} \\ \tau_{xy} & \sigma_y & \tau_{yz} \\ \tau_{xz} & \tau_{yz} & \sigma_z \end{bmatrix} = \frac{E\,\alpha}{2\,(1+\nu)} \begin{bmatrix} 0 & 0 & -y + \frac{\partial\phi}{\partial x} \\ 0 & 0 & x + \frac{\partial\phi}{\partial y} \\ -y + \frac{\partial\phi}{\partial x} & x + \frac{\partial\phi}{\partial y} & 0 \end{bmatrix} = \frac{T}{J} \begin{bmatrix} 0 & 0 & -y \\ 0 & 0 & x \\ -y & x & 0 \end{bmatrix}$$

[6] We quietly assume that the solutions are smooth enough and the domain is connected.

Since the symmetric matrix

$$\begin{bmatrix} 0 & 0 & a \\ 0 & 0 & b \\ a & b & 0 \end{bmatrix}$$

has the eigenvalues and eigenvectors

$$\lambda_1 = 0 \quad , \quad \vec{e}_1 = \begin{pmatrix} -b \\ a \\ 0 \end{pmatrix} \quad \text{and} \quad \lambda_{2,3} = \pm\sqrt{a^2+b^2} \quad , \quad \vec{e}_{2,3} = \begin{pmatrix} \pm a \\ \pm b \\ \sqrt{a^2+b^2} \end{pmatrix}$$

we find that the shaft is stress free in the direction $(x, y, 0)^T$ at the points $(x, y, z)^T$ and is streched with resulting stress

$$\lambda_2 = \frac{T}{J}\sqrt{x^2+y^2} = \frac{T}{J}\, r \quad \text{in the direction of} \quad \begin{pmatrix} -y \\ x \\ r \end{pmatrix}$$

and compressed with resulting stress

$$\lambda_3 = -\frac{T}{J}\sqrt{x^2+y^2} = -\frac{T}{J}\, r \quad \text{in the direction of} \quad \begin{pmatrix} y \\ -x \\ r \end{pmatrix}$$

Both stress directions form an angle of $45°$ with the xy plane and are orthogonal to $(x, y, 0)^T$.

9.4.5 Torsion of a shaft with square cross section

For a shaft with square cross section we consider the domain in figure 9.10. In this situation the boundary function g_2 does not vanish and we have to solve the equation for the warping function ϕ. The mesh for the finite element method is generated by `EasyMesh`. Observe that we mark point 3 as a Dirichlet point to assure a unique solution of the differential equation. Point 5 is used as a material marker and is not useful in this simple example.

Then we can use the *Mathematica* package `BVP2.m` to solve the equation.

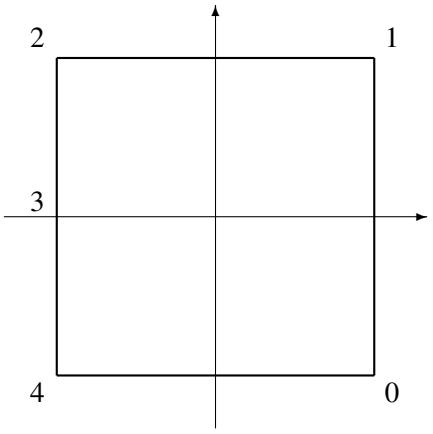
Mathematica

```
<<./BVP2.m
{nodes,elements,segments}=ReadMesh["./Torsion"];
gD[x_,y_]=0;
n[x_,y_]=Which[ y==-1,{0,-1}, y==1,{0,1},x==1,{1,0},x==-1,{-1,0}];
a=Function[{x,y},1];
b=Function[{x,y},0];
f=Function[{x,y},0];
gDirichlet=Function[{x,y},gD[x,y]];
gNeumann=Function[{x,y}, n[x,y].{y,-x}];
points=FEMSolve[a,b,f,gDirichlet,gNeumann,nodes,elements,segments];
```

and generate the plots of the warping function ϕ and its level curves in figure 9.11.

Mathematica

```
g=FEMSurface[points,elements];
Show[g,Axes->True,AxesLabel->{"x","y","u"},AspectRatio->1,
       PlotRange->All,ViewPoint->{1.582,-2.9,0.647}];
```




```
6
0:   1   -1   0.1   2
1:   1    1   0.1   2
2:  -1    1   0.1   2
3:  -1    0   0.1   1
4:  -1   -1   0.1   2
5:   0    0   1     1
5
0:   0    1   2
1:   1    2   2
2:   2    3   2
3:   3    4   2
4:   4    0   2
```

Figure 9.10: A square cross section and its description in *EasyMesh*

```
minVal=Min[Transpose[points][[3]]];
maxVal=Max[Transpose[points][[3]]];
levels=Table[j,{j,minVal,maxVal,(maxVal-minVal)/20}];
glevel=LevelGraphics2D[points,elements,levels];
Show[glevel,AspectRatio->Automatic];
```

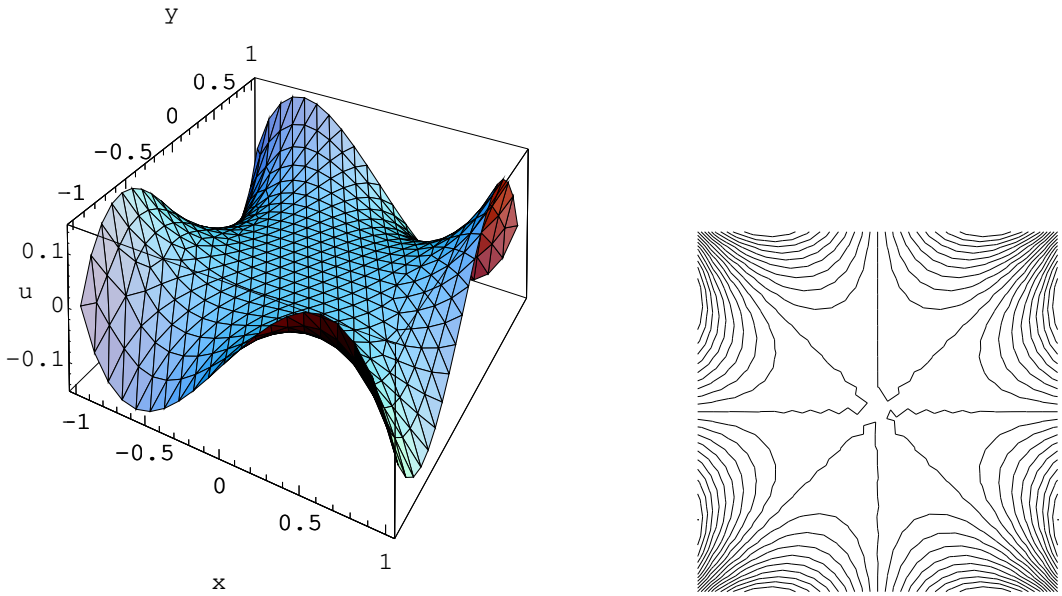


Figure 9.11: Warping functions and its level curves for a square cross section

As in the previous section the resulting stress is given by

$$\frac{E\,\alpha}{2\,(1+\nu)}\sqrt{\left(x+\frac{\partial\,\phi}{\partial y}\right)^2+\left(-y+\frac{\partial\,\phi}{\partial x}\right)^2} \tag{9.8}$$

We can use *Mathematica* again to compute the stress and create the figure 9.12. It can be seen that the stress is maximal at the center of the faces. The stress distribution is far from being uniform.

Mathematica

```
grad=FEMGradient[points,elements];
stress=Table[Flatten[{points[[k,{1,2}]],
         Sqrt[( points[[k,1]]+grad[[k,2]])^2 +
              (-points[[k,2]]+grad[[k,1]])^2]}],
         {k,Length[points]} ];
g=FEMSurface[stress,elements];
stressfig=Show[g,Axes->True,AxesLabel->{"x","y","u"},AspectRatio->1,
    PlotRange->All,ViewPoint->{1.582,-2.9,1.647}];
```

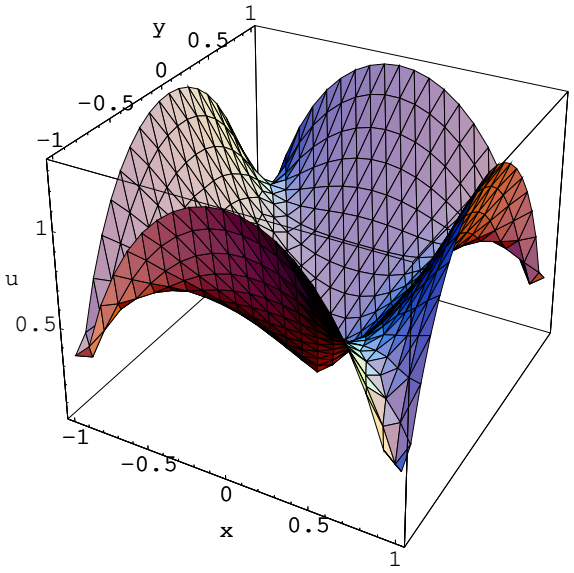


Figure 9.12: The stress distribution in a square shaft, subject to torsion

9.4.6 Using the Prandtl stress function

An alternative approach is using the Prandtl stress funktion χ.

$$\frac{\partial \chi}{\partial y} = -y + \frac{\partial \phi}{\partial x} = \frac{2\,(1+\nu)}{E\,\alpha}\,\tau_{xz} \quad \text{and} \quad -\frac{\partial \chi}{\partial x} = x + \frac{\partial \phi}{\partial y} = \frac{2\,(1+\nu)}{E\,\alpha}\,\tau_{yz}$$

By differentiating the above equations by y (resp. x) and subtracting we find

$$\Delta\chi = \frac{\partial^2 \chi}{\partial x^2} + \frac{\partial^2 \chi}{\partial y^2} = -2$$

Since we consider a shaft we use again $\sigma_x = \sigma_y = \sigma_z = \tau_{xy} = 0$. To find boundary conditions for χ we use fact that there are no external forces on the boundary.

$$\begin{pmatrix} \tau_{xz} \\ \tau_{yz} \end{pmatrix} \cdot \vec{n} = 0 \qquad \Longrightarrow \qquad \begin{pmatrix} \frac{\partial \chi}{\partial y} \\ -\frac{\partial \chi}{\partial x} \end{pmatrix} \cdot \vec{n} = \nabla\chi \cdot \vec{t} = 0$$

where $\vec{t}$ is a tangential vector of the boundary curve. This implies that χ is constant on each component of the curve. Since only information about the derivatives of χ is given we are free to choose the value of χ

on the outer boundary. We choose $\chi = C_0 = 0$ on Γ_0. To determine the values C_i on an interior section Γ_i we use that the warp function is well defined and thus the following integral over a closed section of the boundary has to vanish.

$$\begin{aligned} 0 &= \oint_{\Gamma_i} \nabla\phi \cdot \vec{ds} = \oint_{\Gamma_i} \begin{pmatrix} \frac{\partial\phi}{\partial y} \\ -\frac{\partial\phi}{\partial x} \end{pmatrix} \cdot \vec{n}\, ds = \oint_{\Gamma_i} \begin{pmatrix} -x - \frac{\partial\chi}{\partial x} \\ -y - \frac{\partial\chi}{\partial y} \end{pmatrix} \cdot \vec{n}\, ds \\ \oint_{\Gamma_i} \nabla\chi \cdot \vec{n}\, ds &= \oint_{\Gamma_i} \begin{pmatrix} -x \\ -y \end{pmatrix} \cdot \vec{n}\, ds = \oint_{\Gamma_i} \begin{pmatrix} -y \\ x \end{pmatrix} \cdot \vec{ds} = -2\, A_i \end{aligned}$$

where A_i is the area of the enclosed section. Observe that the orientation of Γ_i as boundary of A_i is negative. The situation is illustrated in Figure 9.13.

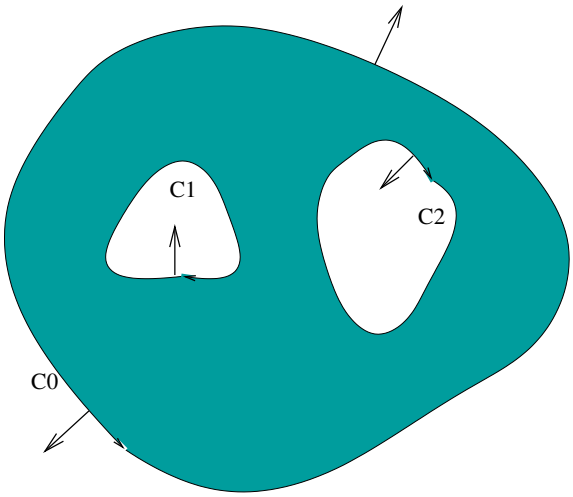


Figure 9.13: A cross section with holes

Thus we arrive at the boundary value problem

$$\begin{aligned} \operatorname{div}(\nabla\chi) = \nabla\nabla\chi &= -2 \qquad \text{in the cross section} \quad \Omega \\ \chi &= C_i \qquad \text{on the boundary sections} \quad \Gamma_i \end{aligned}$$

On the outer boundary Γ_0 we have $C_0 = 0$. The constants C_i on the other sections Γ_i of the boundary have to be determined such that

$$\oint_{\Gamma_i} \nabla\chi \cdot \vec{n}\, ds = -2\, A_i$$

These additional conditions allow to determine a unique solution of the above boundary value problem.

Using equation (9.8) we find the resulting stresses as

$$\begin{aligned} \sigma &= \pm\frac{E\,\alpha}{2\,(1+\nu)} \sqrt{\left(x + \frac{\partial\phi}{\partial y}\right)^2 + \left(-y + \frac{\partial\phi}{\partial x}\right)^2} \\ &= \pm\frac{E\,\alpha}{2\,(1+\nu)} \sqrt{\left(\frac{\partial\chi}{\partial x}\right)^2 + \left(\frac{\partial\chi}{\partial y}\right)^2} = \pm\frac{E\,\alpha}{2\,(1+\nu)}\, \|\nabla\chi\| \end{aligned}$$

The energy in one cross section is given by

$$\frac{1}{2}\,\frac{E\,\alpha}{4\,(1+\nu)}\,\iint\limits_{\Omega}(x+\frac{\partial\,\phi}{\partial y})^2+(-y+\frac{\partial\,\phi}{\partial x})^2\,dA=\frac{1}{2}\,\frac{E\,\alpha}{4\,(1+\nu)}\,\iint\limits_{\Omega}(\frac{\partial\,\chi}{\partial x})^2+(\frac{\partial\,\chi}{\partial y})^2\,dA$$

If the center of gravity of the section is at the origin then the torsional rigidity is given by

$$\begin{aligned}J &= \iint\limits_{\Omega} x^2+y^2+x\,\frac{\partial\,\phi}{\partial y}-y\,\frac{\partial\,\phi}{\partial x}\,dA\\ &= \iint\limits_{\Omega} x^2+y^2+x\,(-x-\frac{\partial\,\chi}{\partial x})-y\,(y+\frac{\partial\,\chi}{\partial y})\,dA\\ &= -\iint\limits_{\Omega} x\,\frac{\partial\,\chi}{\partial x}+y\,\frac{\partial\,\chi}{\partial y}\,dA\end{aligned}$$

With the help of the divergence theorem we may rewrite this intgeral.

$$\begin{aligned}\operatorname{div}(\chi\begin{pmatrix}x\\y\end{pmatrix}) &= \nabla\chi\cdot\begin{pmatrix}x\\y\end{pmatrix}+2\,\chi\\ J &= -\iint\limits_{\Omega}\nabla\chi\cdot\begin{pmatrix}x\\y\end{pmatrix}\,dA=\iint\limits_{\Omega}2\,\chi-\operatorname{div}(\chi\begin{pmatrix}x\\y\end{pmatrix})\,dA\\ &= 2\iint\limits_{\Omega}\chi\,dA-\int_{\partial\Omega}\chi\begin{pmatrix}x\\y\end{pmatrix}\cdot\vec{n}\,ds=2\iint\limits_{\Omega}\chi\,dA-\sum_{i\geq 1}C_i\,\int_{\Gamma_i}\begin{pmatrix}x\\y\end{pmatrix}\cdot\vec{n}\,ds\\ &= 2\iint\limits_{\Omega}\chi\,dA+2\,\sum_{i\geq 1}C_i\,A_i\end{aligned}$$

If the section Ω is simply connected (no holes) the additional constraints on the boundary are of no importance and we have a rather simple boundary values problem and the torsional rigidity is given by the integral

$$J=2\,\iint\limits_{\Omega}\chi\,dA$$

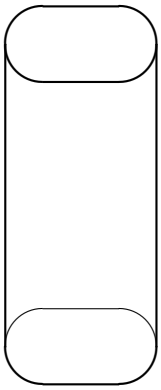
We conclude that the Prandtl stress function leads to an easier formulation if the domain is simply connected. If the domain has holes we can choose between the non-homogeneous boundary condition for the warp function and the additional conditions for the interiour boundary values for the Prandtl stress function.

9.5 Plane strain

In these only problems depending on two variables are solved by the finite element method. Thus to consider elasticity problems we have to simplify the situation such that only two independent variables x and y come into play. There are two important setups leading to this situation: plane strain and plane stress. In both cases a solid with a constant cross section ω (parallel to the xy plane) is considered and horizontal forces are applied to the solid. If the solid is long (in the z direction) we have the situation of plane strain. If the solid is thin we have a plane stress situation. This is illustrated in figure 9.14.

Consider a situation where the z component of the displacement vector is a constant

$$u_3 \quad \text{independent on } x,\ y \text{ and } z$$



all external forces act horizontally

plane strain

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

$\tau_{xz} = \tau_{yz} = 0$

but $\sigma_z \neq 0$

plane stress

$\sigma_z = \tau_{xz} = \tau_{yz} = 0$

$\varepsilon_{xz} = \varepsilon_{yx} = 0$

but $\varepsilon_{zz} \neq 0$

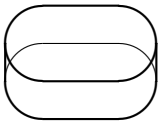


Figure 9.14: Plane strain and plane stress

and

$$u_1 = u_1(x, y) \quad , \quad u_2 = u_2(x, y) \quad \text{independent on } z$$

This leads to vanishing strains in z direction

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

and thus this is called a **plane strain** situation. It can be realized by a long solid in the direction of the z axis with constant cross section and a force distribution parallel to the xy plane, independent on z. The two ends are to be fixed. Due to Saint–Venants's principle (see e.g. [Sout73, §5.6] the boundary effects at the two far ends can safely be ignored.

Hooke's law in the form (9.3) implies

$$\begin{pmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \end{pmatrix} = \frac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & \nu \\ \nu & 1-\nu & \nu \\ \nu & \nu & 1-\nu \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ 0 \end{pmatrix} \quad \text{and} \quad \begin{aligned} \tau_{xy} &= \tfrac{E}{(1+\nu)}\, \varepsilon_{xy} \\ \tau_{xz} &= \tfrac{E}{(1+\nu)}\, 0 \\ \tau_{yz} &= \tfrac{E}{(1+\nu)}\, 0 \end{aligned}$$

or equivalently

$$\begin{pmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{pmatrix} = \tfrac{E}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 1-2\nu \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \tag{9.9}$$

$$\sigma_z = \frac{E\,\nu\,(\varepsilon_{xx} + \varepsilon_{yy})}{(1+\nu)(1-2\nu)} \quad , \quad \tau_{xz} = \tau_{yz} = 0$$

The energy density can be found by equation (9.4) in the form

$$e = \frac{1}{2}\,\langle \vec{\sigma}\,,\, \vec{\varepsilon} \rangle = \frac{1}{2}\,\frac{E}{(1+\nu)(1-2\nu)}\,\left\langle \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 2(1-2\nu) \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} , \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right\rangle \tag{9.10}$$

As unknown functions we consider the two components of the displacement vector $\vec{u} = (u_1, u_2)^T$, as function of x and y. The components of the strain can be computed as derivatives of $\vec{u}$. Thus if $\vec{u}$ is known, all other expressions can be computed.

If the volume and surface forces are parallel to the xy plane and independent on z then the corresponding energy contributions[7] can be written as integrals over the domain $\Omega \subset \mathbb{R}^2$, resp. the boundary $\partial\Omega$. We obtain the total energy as a **functional** of the yet unknown function $\vec{u}$.

$$\begin{aligned} U(\vec{u}) &= U_{elast} + U_{Vol} + U_{Surf} \\ &= \iint_\Omega \frac{1}{2}\,\frac{E}{(1+\nu)\,(1-2\,\nu)} \left\langle \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 2\,(1-2\,\nu) \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix}, \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right\rangle dx\,dy - \\ &\quad - \iint_\Omega \vec{f}\cdot\vec{u}\,dx\,dy - \oint_{\partial\Omega} \vec{g}\cdot\vec{u}\,ds \end{aligned}$$

As in many other situations we use again a principle of least energy to find the equilibrium states of a deformed solid:

> If the above solid is in equilibrium, then the displacement function $\vec{u}$ is a minimizer of the above energy functional, subject to the given boundary conditions.

This is the basis for a finite element solution to plane strain problems.

9–7 Example : Consider a horizontal plate, stretched in x direction by a force F applied to its right edge. We assume that all strains are constant and ε_{xx} is given. Now ε_{yy} and ε_{xy} can be determined by minimizing the energy density. From equation (9.10) we obtain

$$e = \frac{1}{2}\,\frac{E}{(1+\nu)\,(1-2\,\nu)}\,\left((1-\nu)\,\varepsilon_{xx}^2 + 2\,\nu\,\varepsilon_{xx}\,\varepsilon_{yy} + (1-\nu)\,\varepsilon_{yy}^2 + (1-2\,\nu)\,\varepsilon_{xy}^2\right)$$

As a necessary condition for a minimum the partial derivatives with respect to ε_{yy} and ε_{xy} have to vanish. This leads to

$$+2\,\nu\,\varepsilon_{xx} + 2\,(1-\nu)\,\varepsilon_{yy} = 0 \quad \text{and} \quad \varepsilon_{xy} = 0$$

This leads to a modified Poisson's ratio ν^* for the plane strain situation.

$$\varepsilon_{yy} = -\frac{\nu}{1-\nu}\,\varepsilon_{xx} = -\nu^*\,\varepsilon_{xx}$$

The energy density is the given by

$$\begin{aligned} e &= \frac{1}{2}\,\frac{E}{(1+\nu)\,(1-2\,\nu)}\,\left((1-\nu)\,\varepsilon_{xx}^2 - 2\,\frac{\nu^2}{1-\nu}\,\varepsilon_{xx}^2 + \frac{\nu^2\,(1-\nu)}{(1-\nu)^2}\,\varepsilon_{xx}^2\right) \\ &= \frac{1}{2}\,\frac{E}{(1+\nu)\,(1-2\,\nu)}\,\left(\frac{1-2\,\nu+\nu^2}{1-\nu} - \frac{2\,\nu^2}{1-\nu} + \frac{\nu^2}{1-\nu}\right)\,\varepsilon_{xx}^2 \\ &= \frac{1}{2}\,\frac{E}{1-\nu^2}\,\varepsilon_{xx}^2 \end{aligned}$$

By comparing this situation with the situation of a simple stretched shaft (example 9–4, page 216) we find a modified modulus of elasticity

$$E^* = \frac{1}{1-\nu^2}\,E$$

[7] Observe that we quietly switch from a domain in $\Omega \times [0, H] \subset \mathbb{R}^3$ to the planar domain $\Omega \subset \mathbb{R}^2$. The 'energy' U actually denotes the 'energy divided by height H'.

and the force density needed to stretch the plate is given by

$$\frac{F}{A} = E^* \, \frac{\Delta L}{L} = E^* \, \varepsilon_{xx}$$

Similarly modified constants are used in [Sout73, p. 87] to formulate the partial differential equations governing this situation. ♢

9.5.1 From the minimization formulation to a system of PDE's

The displacement vector u has to minimize to total energy of the system, given by

$$\begin{aligned} U(\vec{u}) &= U_{elast} + U_{Vol} + U_{Surf} \\ &= \iint\limits_{\Omega} \frac{1}{2} \, \frac{E}{(1+\nu)\,(1-2\,\nu)} \, \langle \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 2\,(1-2\,\nu) \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} , \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \rangle \, dx\, dy - \\ &\quad - \iint\limits_{\Omega} \vec{f} \cdot \vec{u} \, dx\, dy - \oint_{\partial\Omega} \vec{g} \cdot \vec{u} \, ds \end{aligned}$$

This can be used to derive a system of partial differential equations that are solved by the actual displacement function. Use the abreviation

$$k = \frac{1}{2} \, \frac{E}{(1+\nu)\,(1-2\,\nu)}$$

to find the main expression for the elastic energy given by

$$\begin{aligned} U_{elast} &= \iint\limits_{\Omega} \langle \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} , k \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 2\,(1-2\,\nu) \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \rangle \, dx\, dy \\ &= \iint\limits_{\Omega} \langle \begin{pmatrix} \frac{\partial u_1}{\partial x} \\ \frac{\partial u_2}{\partial y} \\ \frac{1}{2}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right) \end{pmatrix} , k \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 2\,(1-2\,\nu) \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \rangle \, dx\, dy \\ &= \iint\limits_{\Omega} \langle \begin{pmatrix} \frac{\partial u_1}{\partial x} \\ \frac{\partial u_1}{\partial y} \end{pmatrix} , k \begin{bmatrix} 1-\nu & \nu & 0 \\ 0 & 0 & 1-2\,\nu \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \rangle \, dx\, dy \\ &\quad + \iint\limits_{\Omega} \langle \begin{pmatrix} \frac{\partial u_2}{\partial x} \\ \frac{\partial u_2}{\partial y} \end{pmatrix} , k \begin{bmatrix} 0 & 0 & 1-2\,\nu \\ \nu & 1-\nu & 0 \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \rangle \, dx\, dy \end{aligned}$$

Using the divergence theorem (Appendix A.3) on the two integrals we find

$$\begin{aligned} U_{elast} &= - \iint\limits_{\Omega} u_1 \ \mathrm{div} \left(k \begin{bmatrix} 1-\nu & \nu & 0 \\ 0 & 0 & 1-2\,\nu \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right) dx\, dy \\ &\quad + \oint_{\partial\Omega} u_1 \, \langle \vec{n} \, , k \begin{bmatrix} 1-\nu & \nu & 0 \\ 0 & 0 & 1-2\,\nu \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \rangle \, ds \end{aligned}$$

$$
\begin{aligned}
&- \iint_{\Omega} u_2 \ \operatorname{div} \left(k \begin{bmatrix} 0 & 0 & 1-2\,\nu \\ \nu & 1-\nu & 0 \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right) dx\,dy \\
&+ \oint_{\partial\Omega} u_2 \, \langle \vec{n} \, , \, k \begin{bmatrix} 0 & 0 & 1-2\,\nu \\ \nu & 1-\nu & 0 \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \rangle \, ds
\end{aligned}
$$

Using a calculus of variations argument with perturbations of u_1 vanishing on the boundary we conclude

$$
\begin{aligned}
\operatorname{div} \left(\frac{E}{(1+\nu)\,(1-2\,\nu)} \begin{bmatrix} 1-\nu & \nu & 0 \\ 0 & 0 & 1-2\,\nu \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right) &= -f_1 \\
\operatorname{div} \left(\frac{E}{(1+\nu)\,(1-2\,\nu)} \begin{pmatrix} (1-\nu)\,\frac{\partial\,u_1}{\partial x} + \nu\,\frac{\partial\,u_2}{\partial y} \\ \frac{1-2\,\nu}{2} \left(\frac{\partial\,u_1}{\partial y} + \frac{\partial\,u_2}{\partial x} \right) \end{pmatrix} \right) &= -f_1
\end{aligned}
$$

and similarly

$$
\operatorname{div} \left(\frac{E}{(1+\nu)\,(1-2\,\nu)} \begin{pmatrix} \frac{1-2\,\nu}{2} \left(\frac{\partial\,u_1}{\partial y} + \frac{\partial\,u_2}{\partial x} \right) \\ \nu\,\frac{\partial\,u_1}{\partial x} + (1-\nu)\,\frac{\partial\,u_2}{\partial y} \end{pmatrix} \right) = -f_2
$$

We have a system of second order partial differential equations (PDE) for the unknown displacement vector function $\vec{u}$. If the coefficients E and ν are constant we can juggle with these equations and arrive at different formulations. The first equation may be rewritten as

$$
\begin{aligned}
\frac{E}{2\,(1+\nu)} \left(\frac{2\,(1-\nu)}{(1-2\,\nu)} \frac{\partial^2\,u_1}{\partial x^2} + \frac{2\,\nu}{(1-2\,\nu)} \frac{\partial^2\,u_2}{\partial y\,\partial x} + \frac{\partial^2\,u_1}{\partial y^2} + \frac{\partial^2\,u_2}{\partial x\,\partial y} \right) &= -f_1 \\
\frac{E}{2\,(1+\nu)} \left(\frac{1+(1-2\,\nu)}{(1-2\,\nu)} \frac{\partial^2\,u_1}{\partial x^2} + \frac{\partial^2\,u_1}{\partial y^2} + \frac{1}{(1-2\,\nu)} \frac{\partial^2\,u_2}{\partial y\,\partial x} \right) &= -f_1 \\
\frac{E}{2\,(1+\nu)} \left(\frac{\partial^2\,u_1}{\partial x^2} + \frac{\partial^2\,u_1}{\partial y^2} + \frac{1}{(1-2\,\nu)} \frac{\partial}{\partial x} \left(\frac{\partial\,u_1}{\partial x} + \frac{\partial\,u_2}{\partial y} \right) \right) &= -f_1
\end{aligned}
$$

By rewriting the second differential equation in a similar fashion we arrive at a formulation given in [Sout73, p. 87].

$$
\begin{aligned}
\frac{E}{2\,(1+\nu)} \left(\Delta\,u_1 + \frac{1}{(1-2\,\nu)} \frac{\partial}{\partial x} \left(\frac{\partial\,u_1}{\partial x} + \frac{\partial\,u_2}{\partial y} \right) \right) &= -f_1 \\
\frac{E}{2\,(1+\nu)} \left(\Delta\,u_2 + \frac{1}{(1-2\,\nu)} \frac{\partial}{\partial y} \left(\frac{\partial\,u_1}{\partial x} + \frac{\partial\,u_2}{\partial y} \right) \right) &= -f_2
\end{aligned}
$$

With the usual definitions of the operators $\vec{\nabla}$ and Δ this can be written in the dense form

$$
\frac{E}{2\,(1+\nu)} \left(\Delta\,\vec{u} + \frac{1}{1-2\,\nu} \, \vec{\nabla} \left(\vec{\nabla} \cdot \vec{u} \right) \right) = -\vec{f} \tag{9.11}
$$

9.5.2 Boundary conditions

There are different types of useful boundary conditions. We only examine the most important situations.

Prescribed displacement

If on a section Γ_1 of the boundary $\partial\Omega$ the displacement vector $\vec{u}$ is known we can use this as a boundary condition on the section Γ_1. Thus we find Dirichlet conditions on this section of the boundary.

Given boundary forces, no constraints

If on a section Γ_2 of the boundary $\partial\Omega$ the displacement $\vec{u}$ is free, then we use calculus of variations again and have to examine all contributions of the integral over the boundary section Γ_2 in the total energy $U_{elast} + U_{Vol} + U_{Surf}$

$$\begin{aligned}
\int_{\Gamma_2} \dots \, ds &= \int_{\Gamma_2} u_1 \, \langle \vec{n}\,, k \begin{bmatrix} 1-\nu & \nu & 0 \\ 0 & 0 & 1-2\,\nu \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \rangle \, ds \\
&\quad + \int_{\Gamma_2} u_2 \, \langle \vec{n}\,, k \begin{bmatrix} 0 & 0 & 1-2\,\nu \\ \nu & 1-\nu & 0 \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \rangle \, ds - \int_{\Gamma_2} \vec{g} \cdot \vec{u} \, ds \\
&= \int_{\Gamma_2} u_1 \, \langle \vec{n}\,, k \cdot \begin{pmatrix} (1-\nu)\,\varepsilon_{xx} + \nu\,\varepsilon_{yy} \\ (1-2\,\nu)\,\varepsilon_{xy} \end{pmatrix} \rangle \, ds - \int_{\Gamma_2} g_1 \, u_1 \, ds \\
&\quad + \int_{\Gamma_2} u_2 \, \langle \vec{n}\,, k \cdot \begin{pmatrix} (1-2\,\nu)\,\varepsilon_{xy} \\ \nu\,\varepsilon_{xx} + (1-\nu)\,\varepsilon_{yy} \end{pmatrix} \rangle \, ds - \int_{\Gamma_2} g_2 \, u_2 \, ds
\end{aligned}$$

This leads to the two boundary conditions

$$\begin{aligned}
\frac{E}{(1+\nu)\,(1-2\,\nu)} \, \langle \vec{n}\,, \begin{pmatrix} (1-\nu)\,\varepsilon_{xx} + \nu\,\varepsilon_{yy} \\ (1-2\,\nu)\,\varepsilon_{xy} \end{pmatrix} \rangle &= g_1 \\
\frac{E}{(1+\nu)\,(1-2\,\nu)} \, \langle \vec{n}\,, \begin{pmatrix} (1-2\,\nu)\,\varepsilon_{xy} \\ \nu\,\varepsilon_{xx} + (1-\nu)\,\varepsilon_{yy} \end{pmatrix} \rangle &= g_2
\end{aligned}$$

Using Hooke's law (equation (9.3)) we can also reformulate these conditions in terms of external stresses. This leads to

$$\begin{aligned}
\langle \vec{n}\,, \begin{pmatrix} \sigma_x \\ \tau_{xy} \end{pmatrix} \rangle = n_x\,\sigma_x + n_y\,\tau_{xy} &= g_1 \\
\langle \vec{n}\,, \begin{pmatrix} \tau_{xy} \\ \sigma_y \end{pmatrix} \rangle = n_y\,\sigma_y + n_x\,\tau_{xy} &= g_2
\end{aligned}$$

This allows a mechanical verification of the equations.

The above boundary conditions have to be written in terms of the unknown displacement vector $\vec{u}$ and we find

$$\begin{aligned}
\frac{E}{(1+\nu)\,(1-2\,\nu)} \left(n_x \left((1-\nu)\,\frac{\partial\,u_1}{\partial x} + \nu\,\frac{\partial\,u_2}{\partial y} \right) + n_y\,\frac{1-2\,\nu}{2} \left(\frac{\partial\,u_1}{\partial y} + \frac{\partial\,u_2}{\partial x} \right) \right) &= g_1 \\
\frac{E}{(1+\nu)\,(1-2\,\nu)} \left(n_y \left((1-\nu)\,\frac{\partial\,u_2}{\partial y} + \nu\,\frac{\partial\,u_1}{\partial x} \right) + n_x\,\frac{1-2\,\nu}{2} \left(\frac{\partial\,u_1}{\partial y} + \frac{\partial\,u_2}{\partial x} \right) \right) &= g_2
\end{aligned}$$

9.5.3 Thermoelasticity

If the solid is at a temperature T which may depend on the position, then thermal stresses will be created. This leads to an additional energy contribution. Based on equation (9.7) (page 219) we find in the plain strain setup ($\varepsilon_{zz} = 0$) the energy contribution

$$\begin{aligned} U_{Thermo} &= -\iint_\Omega \alpha\, T\, \frac{E}{1-2\,\nu}\, (\varepsilon_{xx} + \varepsilon_{yy})\; dV \\ &= -\iint_\Omega \alpha\, T\, \frac{E}{1-2\,\nu} \left(\frac{\partial\, u_1}{\partial x} + \frac{\partial\, u_2}{\partial y} \right)\; dV \\ &= -\iint_\Omega \alpha\, T\, \frac{E}{1-2\,\nu}\, \operatorname{div} \vec{u}\; dV \\ &= -\oint_{\partial\Omega} \alpha\, T\, \frac{E}{1-2\,\nu}\, \langle \vec{n}\,,\, \vec{u} \rangle\; ds + \iint_\Omega \operatorname{grad}(\alpha\, T\, \frac{E}{1-2\,\nu}) \cdot \vec{u}\; dV \end{aligned}$$

Thus we find the system of partial differential equations

$$\frac{E}{2\,(1+\nu)} \left(\Delta\, \vec{u} + \frac{1}{(1-2\,\nu)}\, \vec{\nabla} \left(\vec{\nabla} \cdot \vec{u} \right) \right) = -\vec{f} + \frac{\alpha\, E}{1-2\,\nu}\, \vec{\nabla} T$$

and additional contributions on the boundary Γ_2

$$\begin{aligned} \frac{E}{(1+\nu)\,(1-2\,\nu)}\, \langle \vec{n}\,, \begin{pmatrix} (1-\nu)\,\varepsilon_{xx} + \nu\,\varepsilon_{yy} \\ (1-2\,\nu)\,\varepsilon_{xy} \end{pmatrix} \rangle &= g_1 + n_x\, \alpha\, T\, \frac{E}{1-2\,\nu} \\ \frac{E}{(1+\nu)\,(1-2\,\nu)}\, \langle \vec{n}\,, \begin{pmatrix} (1-2\,\nu)\,\varepsilon_{xy} \\ \nu\,\varepsilon_{xx} + (1-\nu)\,\varepsilon_{yy} \end{pmatrix} \rangle &= g_2 + n_y\, \alpha\, T\, \frac{E}{1-2\,\nu} \end{aligned}$$

Thus we can treat the thermal stress with the help of an extra external force based on the gradient of the temperature and an additional boundary term.

9–8 Example : If we consider a plate with $0 \le x \le L$ and a constant temperature distribution $T(x, y) = T$ and no external forces. On the left boundary at $x = 0$ we require $u_1(0, y) = 0$. On the lower edge ($y = 0$) we ask for $u_2 = 0$ and no forces apply to the upper and right edge.

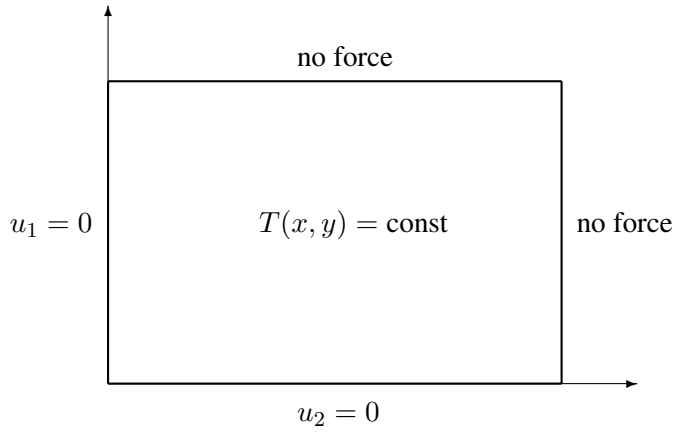


Figure 9.15: Heat stress in plain strain problem, free boundary

Thus we find the differential equations

$$\begin{aligned}\frac{E}{2\,(1+\nu)}\,\left(\frac{\partial^2 u_1}{\partial x^2}+\frac{\partial^2 u_1}{\partial y^2}+\frac{1}{(1-2\,\nu)}\,\frac{\partial}{\partial x}\,\left(\frac{\partial u_1}{\partial x}+\frac{\partial u_2}{\partial y}\right)\right) &= 0\\ \frac{E}{2\,(1+\nu)}\,\left(\frac{\partial^2 u_2}{\partial x^2}+\frac{\partial^2 u_2}{\partial y^2}+\frac{1}{(1-2\,\nu)}\,\frac{\partial}{\partial y}\,\left(\frac{\partial u_1}{\partial x}+\frac{\partial u_2}{\partial y}\right)\right) &= 0\end{aligned}$$

and along the right edge the boundary conditions

$$\begin{aligned}\frac{E}{(1+\nu)\,(1-2\,\nu)}\,\left((1-\nu)\,\frac{\partial u_1}{\partial x}+\nu\,\frac{\partial u_2}{\partial y}\right) &= \alpha\,T\,\frac{E}{1-2\,\nu}\\ \frac{E}{2\,(1+\nu)}\,\left(\frac{\partial u_1}{\partial y}+\frac{\partial u_2}{\partial x}\right) &= 0\end{aligned}$$

Along the upper edge we find

$$\begin{aligned}\frac{E}{2\,(1+\nu)}\,\left(\frac{\partial u_1}{\partial y}+\frac{\partial u_2}{\partial x}\right) &= 0\\ \frac{E}{(1+\nu)\,(1-2\,\nu)}\,\left(\nu\,\frac{\partial u_1}{\partial x}+(1-\nu)\,\frac{\partial u_2}{\partial y}\right) &= \alpha\,T\,\frac{E}{1-2\,\nu}\end{aligned}$$

All the above conditions are satisfied by the functions

$$u_1(x,y)=(1+\nu)\,\alpha\,T\,x \quad \text{and} \quad u_2(x,y)=(1+\nu)\,\alpha\,T\,y$$

Thus the solid will not expand by a factor $\alpha\,T$ but with a larger expansion factor in x and y direction. This is caused by the plain strain condition, i.e. no displacement allowed in z direction. ♢

9–9 Example : In the previous example we require in addition along the upper edge that $u_2=0$ and thus there will be no displacement in y direction in the solid.

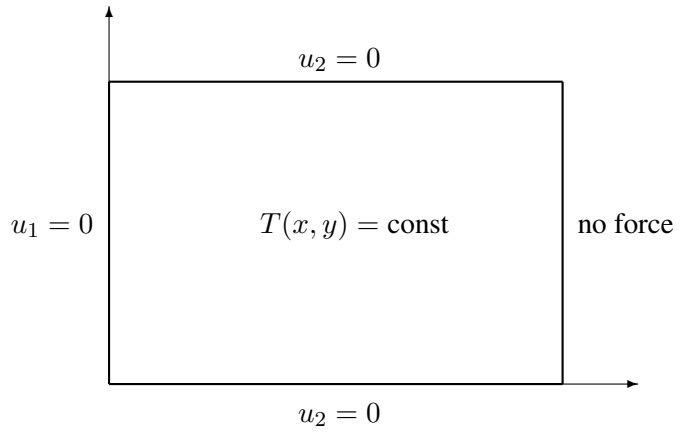


Figure 9.16: Heat stress in plain strain problem, clamped

The displacement u_1 will depend on x only and we find the differential equations

$$\begin{aligned}\frac{E}{2\,(1+\nu)}\,\left(\frac{\partial^2 u_1}{\partial x^2}+\frac{1}{(1-2\,\nu)}\,\frac{\partial}{\partial x}\,\left(\frac{\partial u_1}{\partial x}+0\right)\right) &= 0\\ \frac{E}{2\,(1+\nu)}\,\left(0+\frac{1}{(1-2\,\nu)}\,\frac{\partial}{\partial y}\,\left(\frac{\partial u_1}{\partial x}+0\right)\right) &= 0\end{aligned}$$

and along the right edge the boundary conditions

$$\begin{aligned} \frac{E}{(1+\nu)(1-2\nu)}\left((1-\nu)\frac{\partial u_1}{\partial x}+\nu\,0\right) &= \alpha\,T\,\frac{E}{1-2\nu} \\ \frac{E}{2(1+\nu)}\left(\frac{\partial u_1}{\partial y}+0\right) &= 0 \end{aligned}$$

The solution is given by

$$u_1(x,y)=\frac{\alpha\,T\,(1+\nu)}{1-\nu}\,x \quad \text{and} \quad u_2(x,y)=0$$

Thus the solid will expand by an even larger expansion factor. This is again caused by the constraints. ♢

9–10 Example : If we consider the plate in the above example with a temperature distribution $T(x,y)=\beta\,x$ and no external forces in the solid.

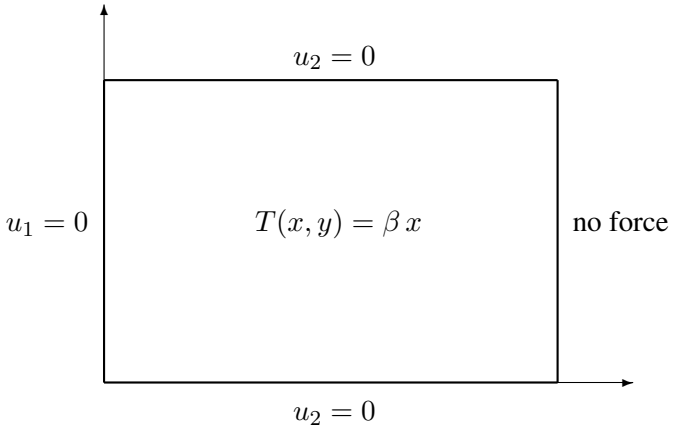


Figure 9.17: Heat stress in plain strain problem with variable temperature

We conclude that u_1 will not depend on y and u_2 will be a constant. Thus we find an ordinary differential equation for $u_1(x)$

$$\begin{aligned} \frac{E}{2(1+\nu)}\left(\frac{\partial^2 u_1}{\partial x^2}+\frac{1}{(1-2\nu)}\frac{\partial}{\partial x}\left(\frac{\partial u_1}{\partial x}+\frac{\partial u_2}{\partial y}\right)\right) &= +\frac{\alpha\,E}{1-2\nu}\,\beta \\ \left(1+\frac{1}{1-2\nu}\right)\frac{\partial^2 u_1}{\partial x^2} &= \frac{\alpha\,2(1+\nu)}{1-2\nu}\,\beta \\ \left(\frac{2-2\nu}{1-2\nu}\right)\frac{\partial^2 u_1}{\partial x^2} &= \frac{\alpha\,2(1+\nu)}{1-2\nu}\,\beta \\ \frac{\partial^2 u_1}{\partial x^2} &= \frac{\alpha\,(1+\nu)}{1-\nu}\,\beta \end{aligned}$$

and the boundary conditions at $x=0$ with $u_1(0)=0$ and at $x=L$ lead to the equation

$$\begin{aligned} \frac{E}{(1+\nu)(1-2\nu)}\left((1-\nu)\frac{\partial u_1(L)}{\partial x}+\nu\,0\right) &= \alpha\,T\,\frac{E}{1-2\nu}=\alpha\,\beta\,L\,\frac{E}{1-2\nu} \\ \frac{\partial u_1(L)}{\partial x} &= \alpha\,\beta\,L\,\frac{1+\nu}{1-\nu} \end{aligned}$$

The solution is given by

$$u_1(x)=\frac{\alpha\,\beta}{2}\,\frac{1+\nu}{1-\nu}\,x^2$$

and we arrive at the strain function

$$\varepsilon_{xx}(x) = u_1'(x) = \alpha\,\beta\,\frac{1+\nu}{1-\nu}\,x$$

Thus the strain created by the temperature gradient increases from left to right with the maximal value given by

$$\varepsilon_{xx}(L) = u_1'(L) = \alpha\,\beta\,\frac{1+\nu}{1-\nu}\,L$$

This result may be verified with a FEM calculation for this situation. ♢

9.6 Plane stress

Consider the situation of a thin (thickness h) plate in the plane $\Omega \subset \mathbb{R}^2$. There are no external stresses on the top and bottom surface and no vertical forces within the plate. Thus we assume that $\sigma_z = 0$ within the plate and $\tau_{xz} = \tau_{yz} = 0$, i.e all stress components in z direction vanish. Thus this is called a **plane stress** situation.

$$\sigma_z = \tau_{xz} = \tau_{yz} = 0$$

Hooke's law in the form (9.1) implies

$$\begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \varepsilon_{xy} \\ \varepsilon_{xz} \\ \varepsilon_{yz} \end{pmatrix} = \frac{1}{E} \begin{bmatrix} 1 & -\nu & -\nu & & & \\ -\nu & 1 & -\nu & & 0 & \\ -\nu & -\nu & 1 & & & \\ & & & 1+\nu & 0 & 0 \\ & 0 & & 0 & 1+\nu & 0 \\ & & & 0 & 0 & 1+\nu \end{bmatrix} \cdot \begin{pmatrix} \sigma_x \\ \sigma_y \\ 0 \\ \tau_{xy} \\ 0 \\ 0 \end{pmatrix}$$

or by eliminating vanishing terms

$$\begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} = \frac{1}{E} \begin{bmatrix} 1 & -\nu & 0 \\ -\nu & 1 & 0 \\ 0 & 0 & 1+\nu \end{bmatrix} \begin{pmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{pmatrix} \quad \text{and} \quad \begin{aligned} \varepsilon_{zz} &= \tfrac{-\nu}{E}\,(\sigma_x + \sigma_y) \\ \varepsilon_{xz} &= 0 \\ \varepsilon_{yz} &= 0 \end{aligned}$$

This matrix can be inverted and we arrive at

$$\begin{pmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{pmatrix} = \frac{E}{1-\nu^2} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & 1-\nu \end{bmatrix} \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \quad \text{and} \quad \begin{aligned} \varepsilon_{zz} &= \tfrac{-\nu}{1-\nu}\,(\varepsilon_{xx} + \varepsilon_{yy}) \\ \varepsilon_{xz} &= 0 \\ \varepsilon_{yz} &= 0 \end{aligned}$$

The energy density can be found by equation (9.4) as

$$e = \frac{1}{2}\left\langle \begin{pmatrix} \sigma_x \\ \sigma_y \\ 2\,\tau_{xy} \end{pmatrix}, \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right\rangle = \frac{1}{2}\,\frac{E}{(1-\nu^2)}\left\langle \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & 2\,(1-\nu) \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix}, \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right\rangle \tag{9.12}$$

This equation is very similar to the expression for a plane strain situation in equation (9.10) (page 229). The only difference is in the coefficients. As a starting point for a finite element solution of a plane stress

problem we will minimize the energy

$$\begin{aligned} U(\vec{u}) &= U_{elast} + U_{Vol} + U_{Surf} \\ &= \iint\limits_{\Omega} \frac{1}{2}\, \frac{E}{(1-\nu^2)} \left\langle \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & 2\,(1-\nu) \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix}, \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right\rangle dx\, dy - \\ &\quad - \iint\limits_{\Omega} \vec{f} \cdot \vec{u}\, dx\, dy - \oint_{\partial\Omega} \vec{g} \cdot \vec{u}\, ds \end{aligned}$$

Using the divergnce theorem we may rewrite the elastiv energy as

$$\begin{aligned} U_{elast} &= \frac{1}{2} \iint\limits_{\Omega} \frac{E}{(1-\nu^2)} \left\langle \begin{pmatrix} \frac{\partial u_1}{\partial x} \\ \frac{\partial u_2}{\partial y} \\ \frac{1}{2}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right) \end{pmatrix}, \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & 2\,(1-\nu) \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right\rangle dx\, dy \\ &= \frac{E}{2\,(1-\nu^2)} \iint\limits_{\Omega} \left\langle \begin{pmatrix} \frac{\partial u_1}{\partial x} \\ \frac{\partial u_1}{\partial y} \end{pmatrix}, \begin{bmatrix} 1 & \nu & 0 \\ 0 & 0 & 1-\nu \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right\rangle dx\, dy \\ &\quad + \frac{E}{2\,(1-\nu^2)} \iint\limits_{\Omega} \left\langle \begin{pmatrix} \frac{\partial u_2}{\partial x} \\ \frac{\partial u_2}{\partial y} \end{pmatrix}, \begin{bmatrix} 0 & 0 & 1-\nu \\ \nu & 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right\rangle dx\, dy \\ &= -\frac{E}{2\,(1-\nu^2)} \iint\limits_{\Omega} u_1\ \mathrm{div} \left(\begin{bmatrix} 1 & \nu & 0 \\ 0 & 0 & 1-\nu \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right) dx\, dy \\ &\quad + \frac{E}{2\,(1-\nu^2)} \oint_{\partial\Omega} u_1 \left\langle \vec{n}\,, \begin{bmatrix} 1 & \nu & 0 \\ 0 & 0 & 1-\nu \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right\rangle ds \\ &\quad - \frac{E}{2\,(1-\nu^2)} \iint\limits_{\Omega} u_2\ \mathrm{div} \left(\begin{bmatrix} 0 & 0 & 1-\nu \\ \nu & 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right) dx\, dy \\ &\quad + \frac{E}{2\,(1-\nu^2)} \oint_{\partial\Omega} u_2 \left\langle \vec{n}\,, \begin{bmatrix} 0 & 0 & 1-\nu \\ \nu & 1 & 0 \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right\rangle ds \end{aligned}$$

Reconsidering the calculations for the plane strain situation we will only have to make a few minor changes to adapt the results to the above plane stress situation to arrive at the system of partial differential equations.

$$\begin{aligned} \mathrm{div} \left(\frac{E}{1-\nu^2} \begin{pmatrix} \frac{\partial u_1}{\partial x} + \nu \frac{\partial u_2}{\partial y} \\ \frac{1-\nu}{2}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right) \end{pmatrix} \right) &= -f_1 \\ \mathrm{div} \left(\frac{E}{1-\nu^2} \begin{pmatrix} \frac{1-\nu}{2}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right) \\ \nu \frac{\partial u_1}{\partial x} + \frac{\partial u_2}{\partial y} \end{pmatrix} \right) &= -f_2 \end{aligned}$$

Using elementary, tedious operations we find

$$\begin{aligned}\frac{E}{2\,(1+\nu)}\left(\frac{\partial^2 u_1}{\partial x^2}+\frac{\partial^2 u_1}{\partial y^2}+\frac{1+\nu}{1-\nu}\frac{\partial}{\partial x}\left(\frac{\partial u_1}{\partial x}+\frac{\partial u_2}{\partial y}\right)\right) &= -f_1\\ \frac{E}{2\,(1+\nu)}\left(\frac{\partial^2 u_2}{\partial x^2}+\frac{\partial^2 u_2}{\partial y^2}+\frac{1+\nu}{1-\nu}\frac{\partial}{\partial y}\left(\frac{\partial u_1}{\partial x}+\frac{\partial u_2}{\partial y}\right)\right) &= -f_2\end{aligned}$$

or with a shorter notation

$$\frac{E}{2\,(1+\nu)}\left(\Delta\vec{u}+\frac{1+\nu}{1-\nu}\vec{\nabla}\left(\vec{\nabla}\vec{u}\right)\right)=-\vec{f} \tag{9.13}$$

This has a structure similar to the equations (9.11) for the plain strain situation. If we set

$$\nu^\star=\frac{\nu}{1-\nu}$$

then we find

$$\frac{1+\nu^\star}{1-\nu^\star}=\frac{1+\frac{\nu}{1-\nu}}{1-\frac{\nu}{1-\nu}}=\frac{1}{1-2\,\nu}$$

And thus the plain strain equations (9.11) take the form

$$\frac{E}{2\,(1+\nu)}\left(\Delta\vec{u}+\frac{1+\nu^\star}{1-\nu^\star}\vec{\nabla}\left(\vec{\nabla}\vec{u}\right)\right)=-\vec{f}$$

and thus are very similar to the plain stress equations (9.13).

9.6.1 Boundary conditions

Again we consider only two types of boundary conditions:

- On a section Γ_1 of the boundary we assume that the displacement vector $\vec{u}$ is known and thus we find Dirichlet boundary conditions.
- On the section Γ_2 the displacement $\vec{u}$ is not submitted to constraints, but we apply and external force $\vec{g}$. Again we use a calculus of variations argument to find the resulting boundary conditions.

The contributions of the integral over the boundary section Γ_2 in the total energy $U_{elast}+U_{Vol}+U_{Surf}$ are given by

$$\begin{aligned}\int_{\Gamma_2}\ldots\,ds &= \frac{E}{2\,(1-\nu^2)}\int_{\Gamma_2}u_1\,\langle\vec{n}\,,\begin{bmatrix}1&\nu&0\\0&0&1-\nu\end{bmatrix}\cdot\begin{pmatrix}\varepsilon_{xx}\\\varepsilon_{yy}\\\varepsilon_{xy}\end{pmatrix}\rangle\,ds\\ &\quad+\frac{E}{2\,(1-\nu^2)}\int_{\Gamma_2}u_2\,\langle\vec{n}\,,\begin{bmatrix}0&0&1-\nu\\\nu&1&0\end{bmatrix}\cdot\begin{pmatrix}\varepsilon_{xx}\\\varepsilon_{yy}\\\varepsilon_{xy}\end{pmatrix}\rangle\,ds-\int_{\Gamma_2}\vec{g}\cdot\vec{u}\,ds\\ &= \frac{E}{2\,(1-\nu^2)}\int_{\Gamma_2}u_1\,\langle\vec{n}\,,\begin{pmatrix}\varepsilon_{xx}+\nu\,\varepsilon_{yy}\\(1-\nu)\,\varepsilon_{xy}\end{pmatrix}\rangle\,ds\\ &\quad+\frac{E}{2\,(1-\nu^2)}\int_{\Gamma_2}u_2\,\langle\vec{n}\,,\begin{pmatrix}(1-\nu)\,\varepsilon_{xy}\\\nu\,\varepsilon_{xx}+\varepsilon_{yy}\end{pmatrix}\rangle\,ds-\int_{\Gamma_2}g_1\,u_1+g_2\,u_2\,ds\end{aligned}$$

This leads to the two boundary conditions

$$\begin{aligned}\frac{E}{1-\nu^2}\,\langle\vec{n}\,,\begin{pmatrix}\varepsilon_{xx}+\nu\,\varepsilon_{yy}\\(1-\nu)\,\varepsilon_{xy}\end{pmatrix}\rangle &= g_1\\ \frac{E}{1-\nu^2}\,\langle\vec{n}\,,\begin{pmatrix}(1-\nu)\,\varepsilon_{xy}\\\nu\,\varepsilon_{xx}+\varepsilon_{yy}\end{pmatrix}\rangle &= g_2\end{aligned}$$

The above boundary conditions have to be written in terms of the unknown displacement vector $\vec{u}$ and we find

$$\begin{aligned}\frac{E}{1-\nu^2}\left(n_x\left(\frac{\partial\,u_1}{\partial x}+\nu\,\frac{\partial\,u_2}{\partial y}\right)+n_y\,\frac{1-\nu}{2}\left(\frac{\partial\,u_1}{\partial y}+\frac{\partial\,u_2}{\partial x}\right)\right) &= g_1\\ \frac{E}{1-\nu^2}\left(n_y\left(\frac{\partial\,u_2}{\partial y}+\nu\,\frac{\partial\,u_1}{\partial x}\right)+n_x\,\frac{1-\nu}{2}\left(\frac{\partial\,u_1}{\partial y}+\frac{\partial\,u_2}{\partial x}\right)\right) &= g_2\end{aligned}$$

9.6.2 Thermoelasticity

We have to adapt Hooke's law (9.1) to the plain stress situation and the thermal expansion. This leads to

$$\begin{pmatrix}\varepsilon_{xx}\\\varepsilon_{yy}\\\varepsilon_{zz}\\\varepsilon_{xy}\\\varepsilon_{xz}\\\varepsilon_{yz}\end{pmatrix}=\frac{1}{E}\begin{bmatrix}1&-\nu&-\nu&&&\\-\nu&1&-\nu&&0&\\-\nu&-\nu&1&&&\\&&&1+\nu&0&0\\&0&&0&1+\nu&0\\&&&0&0&1+\nu\end{bmatrix}\cdot\begin{pmatrix}\sigma_x\\\sigma_y\\0\\\tau_{xy}\\0\\0\end{pmatrix}+\alpha\,T\begin{pmatrix}1\\1\\1\\0\\0\\0\end{pmatrix}$$

or by eliminating vanishing terms

$$\begin{pmatrix}\varepsilon_{xx}\\\varepsilon_{yy}\\\varepsilon_{xy}\end{pmatrix}=\frac{1}{E}\begin{bmatrix}1&-\nu&0\\-\nu&1&0\\0&0&1+\nu\end{bmatrix}\begin{pmatrix}\sigma_x\\\sigma_y\\\tau_{xy}\end{pmatrix}+\alpha\,T\begin{pmatrix}1\\1\\0\end{pmatrix}\quad\text{and}\quad\begin{aligned}\varepsilon_{zz}&=\tfrac{-\nu}{E}\,(\sigma_x+\sigma_y)+\alpha\,T\\ \varepsilon_{xz}&=0\\ \varepsilon_{yz}&=0\end{aligned}$$

This matrix can be inverted and we arrive at

$$\begin{pmatrix}\sigma_x\\\sigma_y\\\tau_{xy}\end{pmatrix}=\frac{E}{1-\nu^2}\begin{bmatrix}1&\nu&0\\\nu&1&0\\0&0&1-\nu\end{bmatrix}\begin{pmatrix}\varepsilon_{xx}\\\varepsilon_{yy}\\\varepsilon_{xy}\end{pmatrix}-\alpha\,T\,\frac{E}{1-\nu}\begin{pmatrix}1\\1\\0\end{pmatrix}$$

and

$$\begin{aligned}\varepsilon_{zz} &= \frac{-\nu}{E}\,(\sigma_x+\sigma_y)+\alpha\,T=\frac{-\nu}{E}\,\frac{E\,(1+\nu)}{1-\nu^2}\,(\varepsilon_{xx}+\varepsilon_{yy})+\frac{2\,\nu\,\alpha\,T}{E}\,\frac{E}{1-\nu}+\alpha\,T\\ &= \frac{-\nu}{1-\nu}\,(\varepsilon_{xx}+\varepsilon_{yy})+\frac{2\,\nu\,\alpha\,T}{1-\nu}+\alpha\,T=\frac{-\nu}{1-\nu}\,(\varepsilon_{xx}+\varepsilon_{yy})+\frac{1+\nu}{1-\nu}\,\alpha\,T\end{aligned}$$

9–11 Example : As a elementary example we consider the situation of constant termperature T and thus $\varepsilon_{xx} = \varepsilon_{yy} = \alpha\, T$. This leads to

$$\begin{pmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{pmatrix} = \frac{E}{1-\nu^2} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & 1-\nu \end{bmatrix} \begin{pmatrix} \alpha\, T \\ \alpha\, T \\ 0 \end{pmatrix} - \alpha\, T\, \frac{E}{1-\nu} \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\varepsilon_{zz} = \frac{-\nu}{1-\nu}\ (\varepsilon_{xx} + \varepsilon_{yy}) + \frac{1+\nu}{1-\nu}\, \alpha\, T = \frac{-\nu}{1-\nu}\ (\alpha\, T + \alpha\, T) + \frac{1+\nu}{1-\nu}\, \alpha\, T = \alpha\, T$$

Thus we confirm that there are no external stresses and $\varepsilon_{zz} = \alpha\, T$. ♢

If the solid is at a temperature T which may depend on the position, then thermal stresses will be created. This leads to an additional energy contribution. Based on equation (9.7) (page 219) we find in the plain stress setup the energy contribution

$$\begin{aligned} U_{Thermo} &= -\iint_\Omega \alpha\, T\, \frac{E}{1-2\,\nu}\ (\varepsilon_{xx} + \varepsilon_{yy} + \varepsilon_{zz})\ dV \\ &= -\iint_\Omega \alpha\, T\, \frac{E}{1-2\,\nu} \left(\varepsilon_{xx} + \varepsilon_{yy} - \frac{\nu}{1-\nu}\, (\varepsilon_{xx} + \varepsilon_{yy}) + \frac{1+\nu}{1-\nu}\, \alpha\, T \right)\ dV \\ &= K(T) - \iint_\Omega \alpha\, T\, \frac{E}{1-\nu}\ (\varepsilon_{xx} + \varepsilon_{yy})\ dV \\ &= K(T) - \iint_\Omega \alpha\, T\, \frac{E}{1-\nu} \left(\frac{\partial\, u_1}{\partial x} + \frac{\partial\, u_2}{\partial y} \right)\ dV = K(T) - \iint_\Omega \alpha\, T\, \frac{E}{1-\nu}\ \operatorname{div} \vec{u}\ dV \\ &= K(T) - \oint_{\partial\Omega} \alpha\, T\, \frac{E}{1-\nu}\ \langle \vec{n}\,,\, \vec{u} \rangle\ ds + \iint_\Omega \operatorname{grad}(\alpha\, T\, \frac{E}{1-\nu}) \cdot \vec{u}\ dV \end{aligned}$$

Thus we find the system of partial differential equations

$$\frac{E}{2\,(1+\nu)} \left(\Delta \vec{u} + \frac{1+\nu}{1-\nu}\, \vec{\nabla} \left(\vec{\nabla} \vec{u} \right) \right) = -\vec{f} + \alpha\, \frac{E}{1-\nu}\, \vec{\nabla} T$$

and additional contributions on the boundary Γ_2

$$\begin{aligned} \frac{E}{1-\nu^2} \left(n_x \left(\frac{\partial\, u_1}{\partial x} + \nu\, \frac{\partial\, u_2}{\partial y} \right) + n_y\, \frac{1-\nu}{2} \left(\frac{\partial\, u_1}{\partial y} + \frac{\partial\, u_2}{\partial x} \right) \right) &= g_1 + n_x\, \alpha\, \frac{E}{1-\nu}\, T \\ \frac{E}{1-\nu^2} \left(n_y \left(\frac{\partial\, u_2}{\partial y} + \nu\, \frac{\partial\, u_1}{\partial x} \right) + n_x\, \frac{1-\nu}{2} \left(\frac{\partial\, u_1}{\partial y} + \frac{\partial\, u_2}{\partial x} \right) \right) &= g_2 + n_y\, \alpha\, \frac{E}{1-\nu}\, T \end{aligned}$$

9–12 Example : If we consider a plate with $0 \le x \le L$ and a constant temperature distribution $T(x, y) = T$ and no external forces. On the left boundary at $x = 0$ we require $u_1(0, y) = 0$. On the lower edge ($y = 0$) we ask for $u_2 = 0$ and no forces apply to the upper and right edge.

Thus we find the differential equations

$$\begin{aligned} \frac{E}{2\,(1+\nu)} \left(\frac{\partial^2\, u_1}{\partial x^2} + \frac{\partial^2\, u_1}{\partial y^2} + \frac{1+\nu}{(1-\nu)}\, \frac{\partial}{\partial x} \left(\frac{\partial\, u_1}{\partial x} + \frac{\partial\, u_2}{\partial y} \right) \right) &= 0 \\ \frac{E}{2\,(1+\nu)} \left(\frac{\partial^2\, u_2}{\partial x^2} + \frac{\partial^2\, u_2}{\partial y^2} + \frac{1+\nu}{(1-\nu)}\, \frac{\partial}{\partial y} \left(\frac{\partial\, u_1}{\partial x} + \frac{\partial\, u_2}{\partial y} \right) \right) &= 0 \end{aligned}$$

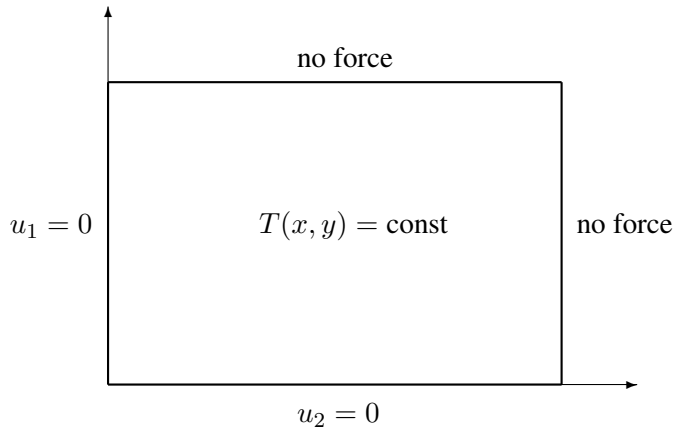


Figure 9.18: Heat stress in plain stress problem, free boundary

and along the right edge the boundary conditions

$$\begin{aligned} \frac{E}{1-\nu^2}\left(\frac{\partial u_1}{\partial x} + \nu\frac{\partial u_2}{\partial y}\right) &= \alpha\frac{E}{1-\nu}T \\ \frac{E}{2\,(1+\nu)}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right) &= 0 \end{aligned}$$

Along the upper edge we find

$$\begin{aligned} \frac{E}{2\,(1+\nu)}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right) &= 0 \\ \frac{E}{1-\nu^2}\left(\frac{\partial u_2}{\partial y} + \nu\frac{\partial u_1}{\partial x}\right) &= \alpha\frac{E}{1-\nu}T \end{aligned}$$

All the above conditions are satisfied by the functions

$$u_1(x, y) = \alpha\, T\, x \quad \text{and} \quad u_2(x, y) = \alpha\, T\, y$$

This leads to

$$\varepsilon_{xx} = \varepsilon_{yy} = \alpha\, T \quad , \quad \varepsilon_{zz} = \frac{-\nu}{1-\nu}\,(\varepsilon_{xx} + \varepsilon_{yy}) + \frac{1+\nu}{1-\nu}\,\alpha\, T = \alpha\, T$$

and thus the solid will expand by a factor $\alpha\, T$ in all directions. One may verify, using Hooke's law, that all stesses vanish. ♢

9–13 Example : Next we consider a plate in Figure 9.19 with a temperature distribution $T(x, y) = \beta\, x$ and no external forces in the solid.

We conclude that u_1 will not depend on y and u_2 will be a constant. Thus we find an ordinary differential equation for $u_1(x)$

$$\begin{aligned} \frac{E}{2\,(1+\nu)}\left(\frac{\partial^2 u_1}{\partial x^2} + \frac{1+\nu}{1-\nu}\,\frac{\partial}{\partial x}\left(\frac{\partial u_1}{\partial x} + \frac{\partial u_2}{\partial y}\right)\right) &= +\frac{\alpha\, E}{1-\nu}\,\beta \\ \left(1 + \frac{1+\nu}{1-\nu}\right)\frac{\partial^2 u_1}{\partial x^2} &= \frac{\alpha\, 2\,(1+\nu)}{1-\nu}\,\beta \\ \frac{\partial^2 u_1}{\partial x^2} &= \alpha\,(1+\nu)\,\beta \end{aligned}$$

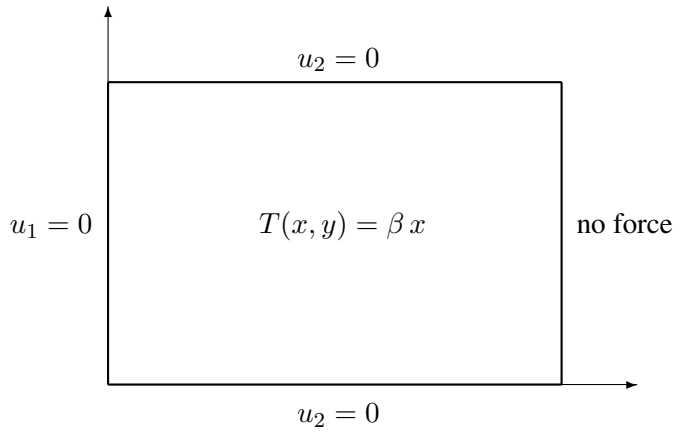


Figure 9.19: Heat stress in plain stress problem with variable temperature

and the boundary conditions at $x = 0$ with $u_1(0) = 0$ and at $x = L$ lead to the equation

$$\begin{aligned} \frac{E}{1-\nu^2}\left(\frac{\partial\, u_1(L)}{\partial x} + 0\right) &= \alpha\, T\, \frac{E}{1-\nu} = \alpha\, \beta\, L\, \frac{E}{1-\nu} \\ \frac{\partial\, u_1(L)}{\partial x} &= \alpha\, \beta\, L\, (1+\nu) \end{aligned}$$

The solution is given by

$$u_1(x) = \frac{\alpha\, \beta}{2}\, (1+\nu)\, x^2$$

and we arrive at the strain function

$$\varepsilon_{xx}(x) = u_1'(x) = \alpha\, \beta\, (1+\nu)\, x$$

Thus the strain created by the temperature gradient increases from left to right with the maximal value given by

$$\varepsilon_{xx}(L) = u_1'(L) = \alpha\, \beta\, (1+\nu)\, L$$

This result may be verified with a FEM calculation for this situation. ♢

9.7 FEM solution for plane strain problems

First we consider the contributions to the total energy from an individual triangular element and from a individual boundary segment. We assume that there are no constraints on the displacement vector $\vec{u}$. Then we focus on possible boundary constraints.

9.7.1 A single element contribution

For given linear functions $u_1(x,y)$ and $u_2(x,y)$ on a triangle with corners at (x_i, y_i) we want to compute the components $\varepsilon_{xx} = \frac{\partial u_1}{\partial x}$, $\varepsilon_{yy} = \frac{\partial u_2}{\partial y}$ and $\varepsilon_{xy} = \frac{1}{2}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right)$ of the strain. We denote the two comonents of $\vec{u}$ at the node i with $u_{1,i}$ and $u_{2,i}$. Based on equation (7.2) (page 148) we find the first partial derivatives.

$$\begin{pmatrix} \frac{\partial u_1}{\partial x} \\ \frac{\partial u_1}{\partial y} \end{pmatrix} = \frac{-1}{2\,A} \begin{bmatrix} (y_3 - y_2) & (y_1 - y_3) & (y_2 - y_1) \\ (x_2 - x_3) & (x_3 - x_1) & (x_1 - x_2) \end{bmatrix} \cdot \begin{pmatrix} u_{1,1} \\ u_{1,2} \\ u_{1,3} \end{pmatrix}$$

$$\begin{pmatrix} \frac{\partial u_2}{\partial x} \\ \frac{\partial u_2}{\partial y} \end{pmatrix} = \frac{-1}{2\,A} \begin{bmatrix} (y_3 - y_2) & (y_1 - y_3) & (y_2 - y_1) \\ (x_2 - x_3) & (x_3 - x_1) & (x_1 - x_2) \end{bmatrix} \cdot \begin{pmatrix} u_{2,1} \\ u_{2,2} \\ u_{2,3} \end{pmatrix}$$

and thus

$$\begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} = \begin{pmatrix} \frac{\partial u_1}{\partial x} \\ \frac{\partial u_2}{\partial y} \\ \frac{1}{2}\left(\frac{\partial u_1}{\partial y} + \frac{\partial u_2}{\partial x}\right) \end{pmatrix}$$

$$= \frac{-1}{2\,A} \begin{bmatrix} y_3 - y_2 & 0 & y_1 - y_3 & 0 & y_2 - y_1 & 0 \\ 0 & x_2 - x_3 & 0 & x_3 - x_1 & 0 & x_1 - x_2 \\ \frac{x_2 - x_3}{2} & \frac{y_3 - y_2}{2} & \frac{x_3 - x_1}{2} & \frac{y_1 - y_3}{2} & \frac{x_1 - x_2}{2} & \frac{y_2 - y_1}{2} \end{bmatrix} \begin{pmatrix} u_{1,1} \\ u_{2,1} \\ u_{1,2} \\ u_{2,2} \\ u_{1,3} \\ u_{2,3} \end{pmatrix}$$

$$= \frac{-1}{2\,A}\,\mathbf{M} \cdot \vec{U}_\Delta$$

Since the derivatives of the displacement are constant, this expression can easily be integrated over the triangle, leading to

$$U_{elast} = \iint_\Delta \frac{1}{2}\,\frac{E}{(1+\nu)\,(1-2\,\nu)} \left\langle \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 2\,(1-2\,\nu) \end{bmatrix} \cdot \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix}, \begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} \right\rangle dx\,dy$$

$$= \frac{1}{2}\,\frac{E}{4\,A\,(1+\nu)\,(1-2\,\nu)} \left\langle \mathbf{M} \cdot \vec{U}_\Delta\,, \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 2\,(1-2\,\nu) \end{bmatrix} \cdot \mathbf{M} \cdot \vec{U}_\Delta \right\rangle$$

The element stiffness matrix for this problem is thus given by

$$\mathbf{A}_\Delta = \frac{E}{4\,A\,(1+\nu)\,(1-2\,\nu)}\,\mathbf{M}^T\cdot\begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 2\,(1-2\,\nu) \end{bmatrix}\cdot\mathbf{M}$$

$\mathbf{A}_\Delta$ is a positive semidefinte, symmetric 6×6 matrix[8].

The volume force expression can be approximated by

$$\begin{aligned} U_{Vol} &= -\iint_\Delta \vec{f}\cdot\vec{u}\,dx\,dy \\ &\approx -\frac{A}{3}\,(f_{1,1}\,u_{1,1}+f_{2,1}\,u_{2,1}+f_{1,2}\,u_{1,2}+f_{2,2}\,u_{2,2}+f_{1,3}\,u_{1,3}+f_{2,3}\,u_{2,3}) = -\frac{A}{3}\,\langle\vec{F}\,,\,\vec{U}_\Delta\rangle \end{aligned}$$

and thus

$$\vec{b}_\Delta = -\frac{A}{3}\,(f_{1,1}\,,\,f_{2,1}\,,\,f_{1,2}\,,\,f_{2,2}\,,\,f_{1,3}\,,\,f_{2,3})^T$$

Now the energy contribution of one triangular element with area A is given by

$$\frac{1}{2}\,\langle\mathbf{A}_\Delta\,\vec{U}_\Delta\,,\,\vec{U}_\Delta\rangle + \langle\vec{b}_\Delta\,,\,\vec{U}_\Delta\rangle$$

9.7.2 Edge segment contribution

We evaluate the vector function $\vec{g}$ at the node i and obtain the result $(g_{1,i},g_{2,i})^T$. The integration along an edge can be approximated by (trapezoidal rule)

$$-\int_{\text{edge}}\vec{g}\cdot\vec{u}\,ds \approx -\frac{L}{2}\,(u_{1,1}\,g_{1,1}+u_{2,1}\,g_{2,1}+u_{1,2}\,g_{1,2}+u_{2,2}\,g_{2,2}) = -\frac{L}{2}\,\langle\begin{pmatrix} u_{1,1}\\ u_{2,1}\\ u_{1,2}\\ u_{2,2}\end{pmatrix}\,,\,\begin{pmatrix} g_{1,1}\\ g_{2,1}\\ g_{1,2}\\ g_{2,2}\end{pmatrix}\rangle$$

where $L=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$ is the length of the segment.

But again (as in section 7.1.3, page 149) it is better solution to use a Gauss integration along an edge. To do so we use the Gauss integration points

$$\vec{p}_1 = \frac{1}{2}\,(\vec{x}_1+\vec{x}_2)+\frac{1}{2\,\sqrt{3}}\,(\vec{x}_1-\vec{x}_2)\quad\text{and}\quad\vec{p}_2=\frac{1}{2}\,(\vec{x}_1+\vec{x}_2)-\frac{1}{2\,\sqrt{3}}\,(\vec{x}_1-\vec{x}_2)$$

By linear interpolation between the points $\vec{x}_1$ and $\vec{x}_2$ we find the values of the function $\vec{u}$ at the Gauss points to be

$$\vec{u}(\vec{p}_1) = (1-w)\,\vec{u}_1+w\,\vec{u}_2\quad\text{and}\quad\vec{u}(\vec{p}_2)=w\,\vec{u}_1+(1-w)\,\vec{u}_2$$

[8] Observe that with $a=(1+\sqrt{1-2\nu})/2$ we find

$$\begin{bmatrix} a & 1-a & 0\\ 1-a & a & 0\\ 0 & 0 & \sqrt{1-2\nu}\end{bmatrix}^2 = \begin{bmatrix} 1-\nu & \nu & 0\\ \nu & 1-\nu & 0\\ 0 & 0 & 1-2\,\nu\end{bmatrix}$$

This identity may be usefull when implementing the computations.

where $w = \frac{1-1/\sqrt{3}}{2} \approx 0.211325$. This leads to the approximation

$$-\int_{\text{edge}} \vec{g}\,\vec{u}\,ds \quad \approx \quad -\frac{L}{2}\,\langle \begin{pmatrix} u_{1,1} \\ u_{2,1} \\ u_{1,2} \\ u_{2,2} \end{pmatrix}, \begin{pmatrix} (1-w)\,g_{1,1} + w\,g_{1,2} \\ (1-w)\,g_{2,1} + w\,g_{2,2} \\ w\,g_{1,1} + (1-w)\,g_{1,2} \\ w\,g_{2,1} + (1-w)\,g_{2,2} \end{pmatrix} \rangle$$

where $(g_{1,i}, g_{2,i})^T$ is the value of $\vec{g}$ at $\vec{p}_i$.

9.7.3 Boundary constraints

Along the boundary of the domain different type of conditions have to be considered. We consider three types of boundary conditions:

- Type 1: prescribed displacement
- Type 2: displacement along a straight line and possibly a surface force
- Type 3: no constraints, but possibly a surface force

Type 1 : prescribed displacement

If the displacement at a node is given by $u_1 = c_1$ and $u_2 = c_2$, then these degrees of freedom are not available and the corresponding rows in the stiffness matrix have to be eliminated. The contributions in the other rows can be moved to the element vector. A very similar procedure was used in section 7.1.5 on page 151. The energy is given by

$$\frac{1}{2}\sum_i \sum_j a_{i,j}\,u_i\,u_j + \sum_i b_i\,u_i$$

For this expression to be minimized the partial derivative with respect to u_k has to vanish, iff u_k is a fre variable. Since $a_{i,j} = a_{j,i}$ this leads to

$$\frac{1}{2}\sum_i \sum_j (a_{i,j}\,\delta_{i,k}\,u_j + a_{i,j}\,u_i\,\delta_{j,k}) + \sum_i f_i\,\delta_{i,k} = \sum_i a_{i,k}\,u_i + b_k = 0$$

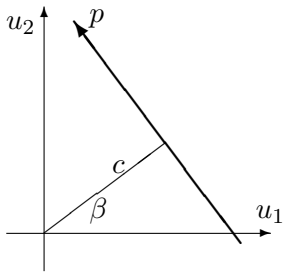
If the values of u_i for $i \in I_1$ are free and $u_i = c_i$ for $i \in I_2$ then we find

$$\sum_{i \in I_1} a_{i,k}\,u_i = b_k - \sum_{i \in I_2} a_{i,k}\,c_i$$

Instead of adding the term $a_{i,k}$ to the global stiffness matrix we have to subtract $a_{i,k}c_k$ from the global RHS vector.

Type 2 : constrained along a straight line

We consider a node whose displacement has to move along a given straight line. This line can be given by an angle β and the distance c of the line from the origin. Thus the node has to move a distance c in with an angle β against the x axis. The situation is shown in figure 9.20. Instead of two degrees of freedom (u_1 and u_2) we work with the single degree of freedom p.



$$
\begin{aligned}
c &= \cos\beta\, u_1 + \sin\beta\, u_2 \\
\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} &= c \begin{pmatrix} \cos\beta \\ \sin\beta \end{pmatrix} + p \begin{pmatrix} -\sin\beta \\ \cos\beta \end{pmatrix}
\end{aligned}
$$

Figure 9.20: Linear constraint on a node

This operation can also be considered as a rotation into a new coordinate system with coordinates c (fixed) and p (free). In fact we find

$$
\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{bmatrix} \cos\beta & -\sin\beta \\ \sin\beta & \cos\beta \end{bmatrix} \cdot \begin{pmatrix} c \\ p \end{pmatrix}
$$

To illustrate the effect of this transformation let us assume that the first node in a triangle is submitted to such a constraint and the other two nodes are free to move.

$$
\begin{aligned}
\begin{pmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{pmatrix} &= \frac{-1}{2\,A}\,\mathbf{M} \begin{pmatrix} u_{1,1} \\ u_{2,1} \\ u_{1,2} \\ u_{2,2} \\ u_{1,3} \\ u_{2,3} \end{pmatrix} = \frac{-1}{2\,A}\,\mathbf{M} \begin{bmatrix} \cos\beta & -\sin\beta & 0 & 0 & 0 & 0 \\ \sin\beta & \cos\beta & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} c \\ p \\ u_{1,2} \\ u_{2,2} \\ u_{1,3} \\ u_{2,3} \end{pmatrix} \\
&= \frac{-1}{2\,A}\,\mathbf{M}\,\mathbf{T}\,\tilde{\vec{U}}_\Delta = \frac{-1}{2\,A}\,\overline{\mathbf{M}}\,\tilde{\vec{U}}_\Delta
\end{aligned}
$$

This leads to a slightly modified element stiffness matrix

$$
\overline{\mathbf{A}}_\Delta = \frac{E}{4\,A\,(1+\nu)\,(1-2\,\nu)}\,\overline{\mathbf{M}}^T \cdot \begin{bmatrix} 1-\nu & \nu & 0 \\ \nu & 1-\nu & 0 \\ 0 & 0 & 1-2\,\nu \end{bmatrix} \cdot \overline{\mathbf{M}}
$$

For the new matrix the first row/column corresponds to c and thus has to treated like a fixed value when generating the global stiffness matrix. The second row/column corresponds to p and is an actual degree of freedom. The implementation is similar to the Type 1 constraints.

Similar procedures apply to the volume force and edge contributions. As an example we consider an edge whose first corner is free to move along a straight line. Thus instead of $u_{1,1}$ and $u_{2,1}$ we have the degree of freedom p, the value of c is prescribed. Since

$$
\begin{pmatrix} u_{1,1} \\ u_{2,1} \end{pmatrix} = \begin{bmatrix} \cos\beta & -\sin\beta \\ \sin\beta & \cos\beta \end{bmatrix} \begin{pmatrix} c \\ p \end{pmatrix}
$$

we find

$$
-\int_{\text{edge}} \vec{g}\,\vec{u}\,ds \;\approx\; -\frac{L}{2}\,\langle \begin{pmatrix} u_{1,1} \\ u_{2,1} \\ u_{1,2} \\ u_{2,2} \end{pmatrix},\, \begin{pmatrix} (1-w)\,g_{1,1} + w\,g_{1,2} \\ (1-w)\,g_{2,1} + w\,g_{2,2} \\ w\,g_{1,1} + (1-w)\,g_{1,2} \\ w\,g_{2,1} + (1-w)\,g_{2,2} \end{pmatrix} \rangle
$$

$$
\begin{aligned}
&= -\frac{L}{2}\left\langle \begin{bmatrix} \cos\beta & -\sin\beta & 0 & 0 \\ \sin\beta & \cos\beta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} c \\ p \\ u_{1,2} \\ u_{2,2} \end{pmatrix}, \begin{pmatrix} (1-w)\, g_{1,1} + w\, g_{1,2} \\ (1-w)\, g_{2,1} + w\, g_{2,2} \\ w\, g_{1,1} + (1-w)\, g_{1,2} \\ w\, g_{2,1} + (1-w)\, g_{2,2} \end{pmatrix} \right\rangle \\
&= -\frac{L}{2}\left\langle \begin{pmatrix} c \\ p \\ u_{1,2} \\ u_{2,2} \end{pmatrix}, \begin{bmatrix} \cos\beta & \sin\beta & 0 & 0 \\ -\sin\beta & \cos\beta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} (1-w)\, g_{1,1} + w\, g_{1,2} \\ (1-w)\, g_{2,1} + w\, g_{2,2} \\ w\, g_{1,1} + (1-w)\, g_{1,2} \\ w\, g_{2,1} + (1-w)\, g_{2,2} \end{pmatrix} \right\rangle
\end{aligned}
$$

Thus for the degree of freedom p we find the contribution

$$
-\frac{L}{2}\, p\ \left(-\sin\beta\ \left((1-w)\, g_{1,1} + w\, g_{1,2}\right) + \cos\beta\ \left((1-w)\, g_{2,1} + w\, g_{2,2}\right)\right)
$$

???

Type 3 : unconstrained boundary

If there are no fixed constraints on the displacement at a node, but a possible surface force is applied, then the results in section 9.7.2 apply.

Chapter 10

Matlab PDE–Toolbox

The mathematical software MATLAB does have a nice add-on, called **PDE toolbox** to solve partial differential equations depending on two independent variables. We present a few examples and instructions on how to solve the equations. This chapter can by no means replace the manual [PDEToolbox95].

The previous chapters should have prepared the reader to understand the steps MATLAB is doing behind the scene when the individual commands are applied.

10.1 Starting the toolbox and demos

The first step is certainly to start MATLAB on the system. By typing `help pde` you will get a very brief description of the commands in the PDE toolbox. The demonstrations `pdedemo1` to `pdedemo8` will give illustrative examples on how to use the toolbox, but no details are given.[1]

To start the graphical user interface type `pdetool`. A window with many buttons and menus should pop up. This will be used to solve our model problem. The toolbox also has an efficient and flexible command line interface which is documented in [PDEToolbox95] and the included demo's can serve as examples.

10.2 A heat conduction problem

As a first example we consider a heat conduction problem in a u–shaped profile, see figure 10.1. The temperature is set to 0 along the boundary, except at the lower edge where we have a thermal insulation. There is internal heating in a rectangle inside the profile. As a result we search the temperature curve and the heat flux within the profile. The deviation of the corresponding equation (4.2) is given in section 4.1.3 on page 64. We consider only the static situation, i.e. we wait long enough for the temperature to be independent on the time.

We have to solve the linear partial differential equation

$$\begin{aligned} \nabla \cdot \nabla u &= f(x,y) && \text{for} \quad (x,y) \in \Omega \\ u &= 0 && \text{for} \quad (x,y) \in \Gamma_1 \\ \frac{\partial u}{\partial \vec{n}} u &= 0 && \text{for} \quad (x,y) \in \Gamma_2 \end{aligned}$$

where $\Omega \subset \mathbb{R}^2$ corresponds to the profile, Γ_2 is the lower edge and Γ_1 the remaining part of the boundary. The function f is given by

$$f(x,y) = \begin{cases} 1 & \text{if } (x,y) \text{ in rectangle} \\ 0 & \text{if } (x,y) \text{ not in rectangle} \end{cases}$$

[1] At the HTA Biel the source files for these examples are in the directory `/tools/matlab/toolbox/pde/` and can be copied and modified.

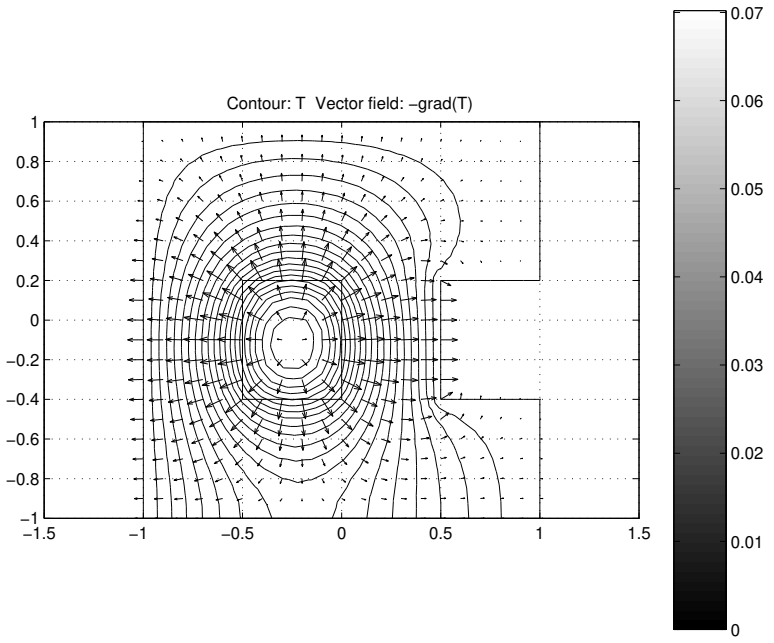


Figure 10.1: Solution of a heat problem with MATLAB

Use the menue Options → Application → Heat Transfer to specify the type of problem at hand. MATLAB will use a notation adapted to the physical context, e.g. T for the temperature instead of u.

10.2.1 Setting up the domain

The buttons at the upper left will allow you to construct rectangle, ellipses and some combinations. As an example construct the domain in figure 10.1. The boundary of the domain Ω is determined by straight lines connecting the points $(-1,-1)$, $(1,-1)$, $(1,-0.4)$, $(0.5,-0.4)$, $(0.5,0.2)$, $(1,0.2)$, $(1,1)$, and $(-1,1)$. Put the MATLAB interface in draw mode by clicking on the polygon button and then enter the above list of points with the mouse. As the last point reenter the first point to close the polygon. The the rectangle inside is generated by pushing the rectangle button and specifying the lower left corner at $(-0.5,-0.4)$ and the upper right corner at $(0,0.2)$. Now the domain is completely specified.

10.2.2 Specifying boundary conditions

Use Boundary → Boundary Mode to specify the boundary conditions.

No flux condition at the lower edge

Along the lower edge we have want

$$\frac{\partial u(x,y)}{\partial \vec{n}} = 0$$

1. click on this section of the boundary, it will turn black.
2. choose Boundary → Specify Boundary Conditions and a menu will pop up. We use Neumann conditions. The equation has the form $\vec{n}\, k\nabla T + q\, T = g$, where the functions q and g can be specified. In our case we want $q = 0$ and $g = 0$.
3. confirm you choice with OK.

Prescribed temperature at the boundary

Along the other parts of the boundary we want

$$u(x, y) = 0$$

1. click on all remaining parts of the boundary (keep `shift` pressed) to select it.
2. choose `Boundary` $\rightarrow$ `Specify Boundary Conditions` and a menu will pop up. Here we want Dirichlet conditions with $h = 1$ and $r = 0$.

Now the lower edge should be blue and the other parts red to indicate the different type of boundary conditions.

10.2.3 Specifying the differential equation

Click on the button `PDE` to get to the menu to specify the differential equation in the form

$$-\operatorname{div}(c * \operatorname{grad}(u)) + a * u = f$$

or with the `MATLAB` notation for heat problems

$$-\operatorname{div}(k * \operatorname{grad}(T)) = Q + h\,(T_{ext} - T)$$

with functions k, Q and T_{ext} to be chosen.

First choose the menue `PDE` $\rightarrow$ `PDE Mode` and the click inside the small rectangle. Here we need to set $k(x, y) = 1$, $h = 0$, $T_{ext} = 0$ and $f = 1$. Choose the menue `PDE` $\rightarrow$ `PDE Specification` and modify the functions, then click `OK`. Now click within the profile, but outside of the rectangle and go the PDE Specification menue again. The only change concerns the function Q, which has to be $Q = 0$ here. Now the differential equations are set up.

10.2.4 Setting up the mesh

Now a mesh for the **Finite Element Method** has to be generated. There are two options to generate this mesh.

- Click on the button with the large triangle to generate the mesh and on the button with four triangles to refine the mesh.
- Choose the menu `Mesh` and appropriate subcommands.

10.2.5 Solving the differential equation and plotting the solution

Now either push the button = or apply commands in the menu `Solve` to solve the equation and generate a first plot, see figure 10.1. With the button showing the meshed surface or the menu `Plot` more elaborate figures can be generated. Play with the options and consult the manual [PDEToolbox95].

10.3 A partial differential equation in polar coordinates

10.3.1 The equation to be solved

The domain $\Omega \subset \mathbb{R}^3$ is a cylinder with radius 2 and height 3. As a model problem we solve the following differential equation with boundary conditions.

$$\begin{aligned} -\Delta u &= 4 - \sin\tfrac{3}{\pi} z && \text{in} \quad \Omega \\ u\,(x, y, 0) &= 1 - x^2 - y^2 && \text{for} \quad z = 0 \quad \text{and} \quad z = 2 \\ u\,(x, y, z) &= 1 - x^2 - y^2 && \text{if} \quad x^2 + y^2 = 2^2 \end{aligned}$$

Observe that the equation and solution will depend on the radius r and the hight z only. Thus polar coordinates should be used. The exact solution is given by

$$u(r,z) = 1 - x^2 - y^2 - \left(\frac{\pi}{3}\right)^2 \sin\frac{3}{\pi}z$$

10.3.2 Using cylindrical coordinates

In cylindrical coordinates the Laplace operator can be expressed with derivatives with respect to the variables r, θ and z. We have

$$\Delta u = \frac{1}{r}\left(\frac{\partial}{\partial r}\left(r\,\frac{\partial\,u}{\partial r}\right) + \frac{1}{r}\,\frac{\partial^2\,u}{\partial\theta^2} + r\,\frac{\partial^2\,u}{\partial z^2}\right)$$

Thus if a function f and the solution u are know not to depend on the angle θ we can rewrite the equation $-\Delta u = f$ as follows

$$-\frac{\partial}{\partial r}\left(r\,\frac{\partial\,u}{\partial r}\right) - r\,\frac{\partial^2\,u}{\partial z^2} = r\,f\,(r,z)$$

Along the edge $r = 0$ the solution has to satisfy the Neumann boundary condition $\frac{\partial\,u}{\partial r} = 0$. Now the original problem can be formulated as a partial differential equation on the rectangle $0 < r < 2$, $0 < z < \pi$ with four parts of the boundary.

$$\begin{array}{rcll}
-\frac{\partial}{\partial r}\left(r\,\frac{\partial\,u}{\partial r}\right) - r\,\frac{\partial^2\,u}{\partial z^2} & = & r\,(4 - \sin\frac{3}{\pi}z) & \text{in} \quad \Omega = [0,2]\times[0,3] \\
u\,(r,0) & = & 1 - r^2 & \text{for} \quad r \in [0,2] \\
u\,(r,\pi) & = & 1 - r^2 & \text{for} \quad r \in [0,2] \\
u\,(2,z) & = & -3 - \sin\frac{3}{\pi}z & \text{for} \quad z \in [0,3] \\
\frac{\partial}{\partial r}\,u\,(0,z) & = & 0 & \text{for} \quad z \in [0,3]
\end{array}$$

10.3.3 Setting up the domain

The rectangular domain for our problem is $0 < r < 2$ and $0 < z < 3$, but the variables in the toolbox need to have the names x and y. We translate $r \to x$ and $z \to y$. With the menu `Options` $\to$ `Axes Limits` the visible domain can be adjusted. If you want to chose only points on a grid you can toggle the switch `Options` $\to$ `Grid`. Now click on the rectangle on the very left, then move the mouse pointer to the point $(0,0)$, hold the left mouse button, move to the point $(2,3)$ and the release the mouse button. Now you should see a grey rectangle on which we will solve the problem.

10.3.4 Specifying boundary conditions

Use `Boundary` $\to$ `Boundary Mode` to specify the boundary conditions.

Right edge

Along the right edge we have to specify

$$u(2,y) = -3 - \sin\frac{3}{\pi}y$$

1. click on the right boundary, it will turn black.
2. choose `Boundary` $\to$ `Specify Boundary Conditions` and a menu will pop up. The equation has the form `h*u=r`, where the functions `h`and `r` can be specified. In our case we want $h = 1$ and $r = -3 - \sin(3 * y/\pi)$
3. confirm you choice with `OK`.

Lower and upper edge

Along the lower and upper edge we have to specify

$$u(x,0) = u(x,3) = 1 - x^2$$

1. click on the lower boundary, it will turn black. Press the `shift` key and click on the upper boundary. Both should turn black.
2. choose `Boundary` → `Specify Boundary Conditions` and a menu will pop up. Here we want $h = 1$ and $r = 1 - x^2$. The precise syntax for the second function is `1-x.^2` or also `1-x.*x`. With these commands the elements in the vector x will be multiplied component by component.
3. confirm you choice with `OK`.

Left edge

Along the left edge we have the Neumann condition

$$\frac{\partial}{\partial x}\, u(x,0) = 0$$

1. click on the left boundary, it will turn black.
2. choose `Boundary` → `Specify Boundary Conditions` and a menu will pop up. Now we have to switch to the Neumann condition with the button on the left. The condition has to be specified in the form
$$\vec{n} \cdot c\nabla u + q \cdot u = g$$
with the functions g and q to be chosen. The function c has to be given when specifying the differential equation. Here we want $q = 0$ and $g = 0$
3. confirm you choice with `OK`. The left edge should now be blue, to indicate the different type of boundary condition.

10.3.5 Specifying the differential equation

Click on the button `PDE` to get to the menu to specify the differential equation

$$-\frac{\partial}{\partial r}\left(r\,\frac{\partial\, u}{\partial r}\right) - r\,\frac{\partial^2 u}{\partial z^2} = r\,(4 - \sin z)$$

Our problem is of elliptic type and thus the appropriate box has to be marked. The `MATLAB` notation for the differential equation is

$$-\operatorname{div}(c * \operatorname{grad}(u)) + a * u = f$$

with functions c, a and f to be chosen. If we set $c(x,y) = x$, $a = 0$ and $f(x,y) = x \cdot (4 - \sin y)$ the we have

$$\operatorname{div}\begin{pmatrix} c\,\frac{\partial\, u}{\partial x} \\ c\,\frac{\partial\, u}{\partial y} \end{pmatrix} + a\,u = \frac{\partial}{\partial x}\left(x\,\frac{\partial\, u}{\partial x}\right) + \frac{\partial}{\partial y}\left(x\,\frac{\partial\, u}{\partial y}\right) = x\,(4 - \sin y)$$

Thus the differential equation is set up.

10.3.6 Setting up the mesh

Now a mesh for the **Finite Element Method** has to be generated. There are two options to generate this mesh.

- Click on the button with the large triangle to generate the mesh and on the button with four triangles to refine the mesh.
- Choose the menu `Mesh` and appropriate subcommands.

10.3.7 Solving the differential equation and plotting the solution

Now either push the button = or apply commands in the menu `Solve` to solve the equation and generate a first plot. With the button showing the meshed surface or the menu `Plot` more elaborate figures can be generated. Play with the options and consult the manual [PDEToolbox95].

10.4 A two dimensional fluid flow problem

Consider a laminar flow between two plates with an obstacle between the two plates. We assume that the situation is independent on one of the spatial variables and consider a cross section only shown in the figure 10.2. The goal is to find the velocity field $\vec{v}$ of the fluid.

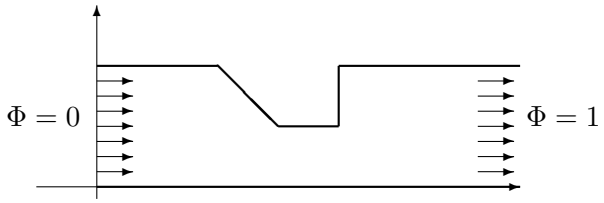


Figure 10.2: Fluid flow between two plates, the setup

According to table 6.1 on page 144 we can introduce a velocity potential $\Phi(x, y)$. The velocity vector $\vec{v}$ is then given by

$$\vec{v} = \begin{pmatrix} v_x \\ v_y \end{pmatrix} = \begin{pmatrix} \frac{\partial \Phi}{\partial x} \\ \frac{\partial \Phi}{\partial y} \end{pmatrix}$$

The flow is assumed to be uniform far away from the obstacle. Thus we set the potential to $\Phi = 0$ (resp. $\Phi = 1$) at the left (resp. right) end of the plates. Since the fluid can not flow through the plates we know that the normal component of the velocity has to vanish at the upper and lower boundary. The differential equation to be satisfied by Φ is

$$\Delta\Phi = \operatorname{div}(\operatorname{grad}\Phi) = 0$$

This boundary value problem can be solved by `MATLAB` and a possible result[2] is shown in figure 10.3 After the numerical solution is found you may export the information about the mesh and the solution to the `MATLAB` environment and find a velocity profile along a given height $y = 0.5$, only the x–component of the velocity is computed. The code below and leads to figure 10.4. Documentation can be found in the online help or in [PDEToolbox95].

[2]Since `MATLAB` plots $-\operatorname{grad}\Phi$ we actually set $\Phi = 1$ on the left border and $\Phi = 0$ on the right border. Thus the fluid will move from left to right.

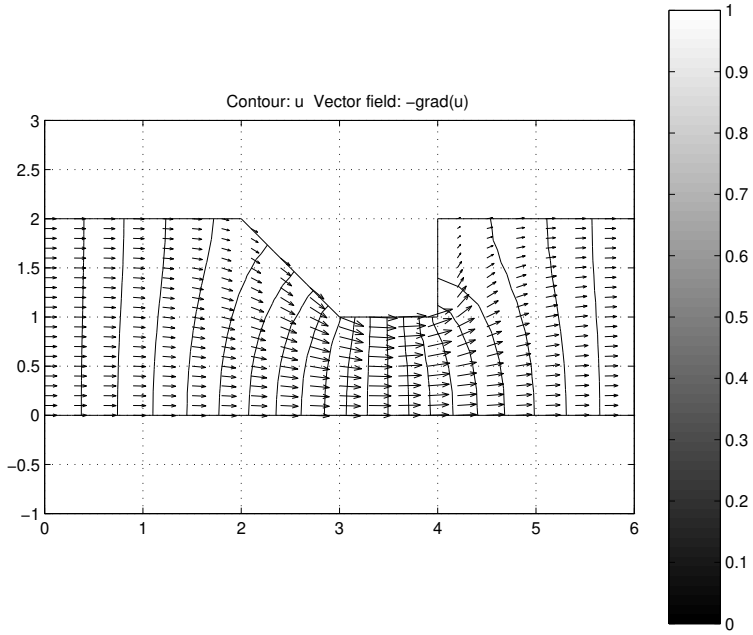


Figure 10.3: Fluid flow between two plates, the result

Matlab

```
% export mesh to       p e t
% export solution to   u
x=linspace(0,6,100);
y=0.5;
phix=tri2grid(p,t,u,x,y);

dx=x(2)-x(1);
xcenter=x(1:length(x)-1)+dx/2;
vx=-diff(phix)/dx;
plot(xcenter,vx)
axis([0 6 0 0.3]);
```

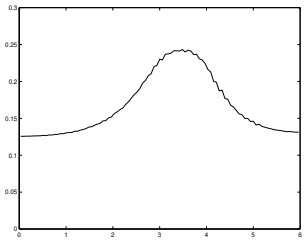


Figure 10.4: Fluid flow between two plates, a speed profile

Chapter 11

Some matrix computations

The bible for matrix computations is [GoluVanLoan96] and this presentation is based on this standard reference. Other useful references are [Demm97] or [Axel94].

In this chapter we present some results from numerical linear analysis needed to solve the systems arising from the type of FEM problems considered in these notes. Thus we restrict the results to positive definite symmetric matrices and will exploit the band structure of the matrices to obtain efficient algorithms to solve the system of equations and find a few eigenvalues. Then the conjugate gradient will be examined as a typical and important iterative solution method.

11.1 A few basic definitions for matrices

When solving systems of linear equations errors will occur. Matrix norms and conditions numbers can be used to examine and control the errors.

11–1 Definition : The **norm**[1] of a real $n \times n$ matrix is defined as

$$\|\mathbf{A}\| = \max_{\|\vec{x}\|=1} \|\mathbf{A}\ \vec{x}\| = \max_{\vec{x}\in\mathbb{R}^n} \frac{\|\mathbf{A}\ \vec{x}\|}{\|\vec{x}\|}$$

If the matrix $\mathbf{A}$ is invertible then the number

$$\kappa = \|\mathbf{A}\| \cdot \|\mathbf{A}^{-1}\|$$

is called the **condition number** of the matrix.

The definition of a matrix norm implies

$$\|\mathbf{A}\ \vec{x}\| \leq \|\mathbf{A}\|\ \|\vec{x}\| \quad \text{for all} \quad \vec{x} \in \mathbb{R}^n$$

The norm of a matrix $\mathbf{A}$ is the factor by which a vector will be stretched at most if multiplied by $\mathbf{A}$. Thus it is no surprise that the eigenvalues are related to the norm of a matrix.

11–2 Result : *If $\mathbf{A}$ is a symmetric matrix with eigenvalues λ_i then*

$$\|\mathbf{A}\| = \max_i |\lambda_i|$$

◇

[1] We only consider the matrix norm based on the Euclidean norm on $\mathbb{R}^n$, i.e. $\|\vec{x}\| = \sqrt{\sum_i x_i^2}$

This result applies only to symmetric matrices as the example

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$$

clearly illustrates.

Proof : Use the diagonalization $\mathbf{A} = \mathbf{Q}\,\mathbf{D}\,\mathbf{Q}^T$ with an orthogonal matrix $\mathbf{Q}$. The eigenvalues λ_i form the diagonal of $\mathbf{D}$. Based on exercise 11–3 and $\|\mathbf{Q}^T\,\vec{x}\| = \|\vec{x}\|$ we obtain

$$\|\mathbf{A}\,\vec{x}\| = \|\mathbf{Q}\,\mathbf{D}\,\mathbf{Q}^T\,\vec{x}\| = \|\mathbf{Q}\,\mathbf{D}\,\vec{y}\| = \|\mathbf{D}\,\vec{y}\|$$

where $\vec{y} = \mathbf{Q}^T\,\vec{x}$. This implies $\kappa(\mathbf{A}) = \kappa(\mathbf{D})$, using exercise 11–1. □

Since $\mathbf{A}^{-1} = \left(\mathbf{Q}\,\mathbf{D}\,\mathbf{Q}^T\right)^{-1} = \mathbf{Q}\,\mathbf{D}^{-1}\,\mathbf{Q}^T$ we obtain

$$\|\mathbf{A}^{-1}\| = \min_i |\lambda_i|$$

Now we can characterize the condition number of a symmetric matrix. Computing κ is a different story though, as the eigenvalues are not readily computed.

11–3 Result : *The condition number of a symmetric, invertible matrix with eigenvalues λ_1 is given by*

$$\kappa = \frac{\max_i |\lambda_i|}{\min_i |\lambda_i|}$$

◇

Now we show that the condition number contains information about the precision of the solution of a system of linear equations.

11–4 Result : *Consider a system of linear equations $\mathbf{A}\,\vec{x} = \vec{b}$. But instead of the exact values of the RHS $\vec{b}$ a perturbed vector $\vec{b} + \vec{\Delta b}$ is used, leading to a perturbed solution $\vec{x} + \vec{\Delta x}$. Then we find for the* ***relative error***

$$\frac{\|\vec{\Delta x}\|}{\|\vec{x}\|} \leq \kappa\,\frac{\|\vec{\Delta b}\|}{\|\vec{b}\|}$$

i.e. the relative error is (at worst) multiplied by the condition number. ◇

As an example consider a matrix with condition number $\kappa = 1'000 = 10^3$ and an input vector $\vec{b}$ with 5 digits known to be correct. Then the result might have only $5 - 3 = 2$ correct digits. This problem can not be avoided, it is inherent to the equation to be solved. A stable algorithm will not worsen the problem. The above is clearly a worst case scenario. For many FEM problems the actual error will be considerably smaller. A good heuristic explanation for this, based on eigenvalues and eigenvectors, is given in [Axel94, p. 606]. Exercise 11–2 also illustrates the effect. Nonetheless the condition number is a good indicator for possible precision problems. One can show that the Cholesky algorithm, without pivoting, is stable, i.e. does not lead to unnecessary errors, see e.g. [Wilk63] or [GoluVanLoan96]. Find a description of the essential steps in section 11.2.3.

Proof : Since the equation is linear we find $\mathbf{A}\,\vec{\Delta x} = \vec{\Delta b}$ and thus

$$\|\vec{\Delta x}\| = \|\mathbf{A}^{-1}\,\vec{\Delta b}\| \leq \|\mathbf{A}^{-1}\|\;\|\vec{\Delta b}\|$$

and since $\|\vec{b}\| = \|\mathbf{A}\,\vec{x}\| \leq \|\mathbf{A}\|\;\|\vec{x}\|$ we find

$$\|\vec{x}\| \geq \frac{1}{\|\mathbf{A}\|}\;\|\vec{b}\|$$

Now divide the two inequalities

$$\frac{\|\vec{\Delta x}\|}{\|\vec{x}\|} \leq \|\mathbf{A}\|\;\|\mathbf{A}^{-1}\|\;\frac{\|\vec{\Delta b}\|}{\|\vec{b}\|} = \kappa\;\frac{\|\vec{\Delta b}\|}{\|\vec{b}\|}$$

□

The above result can be extended to the situation where the matrix $\mathbf{A}$ is perturbed by a matrix $\Delta\mathbf{A}$, i.e. we solve

$$(\mathbf{A}+\Delta\mathbf{A})\;(\vec{x}+\vec{\Delta x}) = \vec{b}+\vec{\Delta b}$$

Then the relative error can be estimated by the following computation, if $\|\mathbf{A}^{-1}\,\Delta\mathbf{A}\| < 1$.

$$\begin{aligned}
(\mathbf{A}+\Delta\mathbf{A})\;(\vec{x}+\vec{\Delta x}) &= \vec{b}+\vec{\Delta b}\\
\mathbf{A}\,\vec{\Delta x} &= -\Delta\mathbf{A}\;\;(\vec{x}+\Delta\vec{x}) + \vec{\Delta b}\\
\vec{\Delta x} &= -\mathbf{A}^{-1}\,\Delta\mathbf{A}\;\;(\vec{x}+\Delta\vec{x}) + \mathbf{A}^{-1}\,\vec{\Delta b}\\
\frac{\|\vec{\Delta x}\|}{\|\vec{x}\|} &\leq \|\mathbf{A}^{-1}\,\Delta\mathbf{A}\|\;\frac{\|\vec{x}+\Delta\vec{x}\|}{\|\vec{x}\|} + \|\mathbf{A}^{-1}\|\frac{\|\mathbf{A}\,\vec{x}\|}{\|\vec{x}\|}\frac{\|\vec{\Delta b}\|}{\|\vec{b}\|}\\
&\leq \|\mathbf{A}^{-1}\,\Delta\mathbf{A}\|\;\left(1+\frac{\|\Delta\vec{x}\|}{\|\vec{x}\|}\right) + \kappa(\vec{x})\;\frac{\|\vec{\Delta b}\|}{\|\vec{b}\|}\\
\left(1-\|\mathbf{A}^{-1}\,\Delta\mathbf{A}\|\right)\frac{\|\vec{\Delta x}\|}{\|\vec{x}\|} &\leq \|\mathbf{A}^{-1}\,\Delta\mathbf{A}\| + \kappa(\vec{x})\;\frac{\|\vec{\Delta b}\|}{\|\vec{b}\|}\\
\frac{\|\vec{\Delta x}\|}{\|\vec{x}\|} &\leq \frac{1}{1-\|\mathbf{A}^{-1}\,\Delta\mathbf{A}\|}\;\left(\|\mathbf{A}^{-1}\,\Delta\mathbf{A}\| + \kappa(\vec{x})\;\frac{\|\vec{\Delta b}\|}{\|\vec{b}\|}\right)
\end{aligned}$$

Using

$$\|\mathbf{A}^{-1}\,\Delta\mathbf{A}\| \leq \|\mathbf{A}^{-1}\|\;\|\Delta\mathbf{A}\| = \|\mathbf{A}\|\;\|\mathbf{A}^{-1}\|\;\frac{\|\Delta\mathbf{A}\|}{\|\mathbf{A}\|} = \kappa\;\frac{\|\Delta\mathbf{A}\|}{\|\mathbf{A}\|}$$

and $\kappa(\vec{x}) \leq \kappa$ (see exercise 11–2) this leads to the bound on the relative error. Again the condition number κ plays an important role.

$$\frac{\|\vec{\Delta x}\|}{\|\vec{x}\|} \leq \frac{\kappa}{1-\|\mathbf{A}^{-1}\,\Delta\mathbf{A}\|}\;\left(\frac{\|\Delta\mathbf{A}\|}{\|\mathbf{A}\|} + \frac{\|\vec{\Delta b}\|}{\|\vec{b}\|}\right) \tag{11.1}$$

11–5 Definition : A symmetric, real matrix $\mathbf{A}$ is called **positive definite** if and only if

$$\langle \mathbf{A}\cdot\vec{x}\,,\,\vec{x}\rangle = \langle \vec{x}\,,\,\mathbf{A}\cdot\vec{x}\rangle > 0 \quad \text{for all} \quad \vec{x} \neq \vec{0}$$

The matrix is called **positive semidefinite** if and only if

$$\langle \mathbf{A}\cdot\vec{x}\,,\,\vec{x}\rangle = \langle \vec{x}\,,\,\mathbf{A}\cdot\vec{x}\rangle \geq 0 \quad \text{for all} \quad \vec{x}$$

11–6 Result : *If the matrix* $\mathbf{A} = (a_{i,j})_{1 \le i,j \le n}$ *is positive definite then*

- $a_{i,i} > 0$ *for* $1 \le i \le n$*, i.e. the numbers on the diagonal are positive.*
- $\max |a_{i,j}| = \max a_{i,i}$*, i.e. the maximal value has to be on the diagonal.*

◇

Proof :

- Choose $\vec{x} = \vec{e}_i$ and compute $\langle \vec{e}_i \, , \, \mathbf{A} \cdot \vec{e}_i \rangle = a_{i,i} > 0$
- Assume $\max |a_{i,j}| = \max a_{k,l}$ with $k \neq l$. Choose $\vec{x} = \vec{e}_k - \text{sign}(a_{k,l}) \; \vec{e}_k$ and compute $\langle \vec{x} \, , \, \mathbf{A} \cdot \vec{x} \rangle = a_{k,k} + a_{l,l} - 2 \, |a_{k,l}| \le 0$, contradicting positive definiteness.

□

The above allows to verify quickly that a matrix is not positive definite, but it does not contain a criterion to quickly decide that **A** is positive definite. The eigenvalues contain all information about definiteness of a symmetric matrix.

11–7 Result : *The matrix* **A** *is positive definite iff all eigenvalues are strictly positive. The matrix* **A** *is positive semidefinite iff all eigenvalues are positive or zero.* ◇

Proof : This is a direct consequence of the diagonalization result $\mathbf{A} = \mathbf{Q} \, \mathbf{D} \, \mathbf{Q}^T$ in section A.1.3 □

This result is of little help to decide whether a given large matrix is positive definite or not. Finding all eigenvalues is not an option, as it is computationally rather expensive. A positive answer can be given using diagonal dominance and reducible matrices, see e.g. [Axel94, §4].

11–8 Definition : Consider a symmetric $n \times n$ matrix **A**.

- **A** is called **strictly diagonally dominant** iff $|a_{i,i}| > \sigma_i$ for all $1 \le i \le n$, where
$$\sigma_i = \sum_{j \neq i \, , \, 1 \le j \le n} |a_{i,j}|$$
Along each column/row the sum of the off-diagonal elements is smaller than the diagonal element.
- **A** is called **diagonally dominant** iff $|a_{i,i}| \ge \sigma_i$ for all $1 \le i \le n$.
- **A** is called **reducible** if the exists a permutation matrix **P** and square matrices $\mathbf{B}_1$, $\mathbf{B}_2$ and a matrix $\mathbf{B}_3$ such that
$$\mathbf{P} \cdot \mathbf{A} \cdot \mathbf{P}^T = \begin{bmatrix} \mathbf{B}_1 & \mathbf{B}_3 \\ \mathbf{0} & \mathbf{B}_2 \end{bmatrix}$$
Since **A** is symmetric the matrix $\mathbf{P} \cdot \mathbf{A} \cdot \mathbf{P}^T$ is also symmetric and the block $\mathbf{B}_3$ has to vanish, i.e. we have the condition
$$\mathbf{P} \cdot \mathbf{A} \cdot \mathbf{P}^T = \begin{bmatrix} \mathbf{B}_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{B}_2 \end{bmatrix}$$
This leads to an easy interpretation of a reducible matrix **A**: the system of linear equation $\mathbf{A} \, \vec{u} = \vec{b}$ can be decomposed into two smaller systems $\mathbf{B}_1 \, \vec{u}_1 = \vec{b}_1$ and $\mathbf{B}_2 \, \vec{u}_2 = \vec{b}_2$. To arrive at this situation all one has to do is renumber the equations and variables.
- **A** is called **irreducible** if it is not reducible.
- **A** is called **irreducibly diagonally dominant** if **A** is irreducible and
  - $|a_{i,i}| \ge \sigma_i$ for all $1 \le i \le n$
  - $|a_{i,i}| > \sigma_i$ for at least one $1 \le i \le n$

11–9 Observation : If the matrix **A** is generated by a finite element problem, then there is a simple criterion to decide where **A** is irreducible. The method is based on **graph theory**. A symmetric $n \times n$ matrix leads to a **graph** by the following procedure:

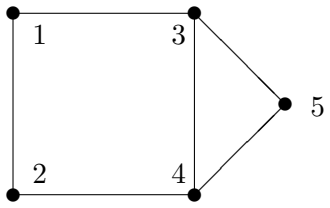
- For each row (number i) in the matrix **A** draw a point P_i in the plane.
- Two points P_i and P_j are directly connected if $a_{i,j} = a_{j,i} \neq 0$. In this case draw a line from P_i to P_j.
- The resulting graph is said to be **strongly connected** if for each pair of points there is a path connection the points.

If the matrix is generated by a finite element problem, based on linear interpolation on a given mesh then we already have the graphical representation of the graph: the mesh. The matrix is strongly connected if the mesh consists of one piece, i.e. we can walk from any point to any other point on the mesh. For a given problem this is often very easy to verify by looking at the mesh.

It can be shown ([Axel94, Theorem4.3]) that a symmetric matrix is irreducible if and only if its graph is connected. ♢

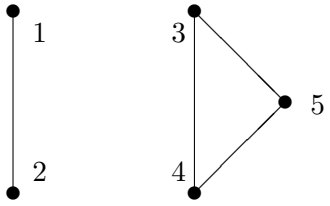
11–10 Example : The matrix below is strongly connected, as can be seen by a possible representation of its graph on the right.

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$



A modification leads to a matrix that is not strongly connected.

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$



For the above two example one can quickly decide whether the matrices are reducible or not. But a renumbering of the rows and columns will lead to matrices where it is not as easy to decide. The graphs on the right will not change by reordering rows and columns. ♢

11–11 Example : A discretization of the boundary value problem $u''(x) = f(x)$ for $0 < x < 1$ and $u(0) = u(1) = 0$ leads to the matrix

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}$$

See also Example 4–1 on page 73. This matrix is diagonally dominant, but not strictly diagonally dominant. The matrix is irreducible and thus irreducibly diagonally dominant.

Most global stiffness matrices for FEM problems satisfy similar properties. This is based on the fact that the element stiffness matrices are diagonally dominant and positive semidefinite, see also exercise 7–1. ♢

11–12 Example : A positive definite matrix need not be diagonally dominant. As an example consider the square of the matrix $\mathbf{A}$ above

$$\mathbf{A}^2 = \begin{bmatrix} 5 & -4 & 1 & & & & & \\ -4 & 6 & -4 & 1 & & & & \\ 1 & -4 & 6 & -4 & 1 & & & \\ & 1 & -4 & 6 & -4 & 1 & & \\ & & \ddots & \ddots & \ddots & \ddots & \ddots & \\ & & & 1 & -4 & 6 & -4 & 1 \\ & & & & 1 & -4 & 6 & -4 \\ & & & & & 1 & -4 & 5 \end{bmatrix}$$

Since the eigenvalues of $\mathbf{A}^2$ are the squares of the eigenvalues of $\mathbf{A}$ this matrix is positive definite. But it is clearly not diagonally dominant. ♢

11–13 Result : *(see e.g. [Axel94, Theorem 4.9])*
Consider a real symmetric matrix $\mathbf{A}$ with positive numbers along the diagonal. If $\mathbf{A}$ is strictly diagonally dominant or irreducibly diagonally dominant, then $\mathbf{A}$ is positive definite. ♢

This result is a consequence of Gershgorin's theorem and we omit its proof. The following results are useful too to decide whether matrices arising from FEM problems will be positive definite.

11–14 Result : *Consider the positive definite, symmetric matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ and $\mathbf{B} \in \mathbb{R}^{n\times m}$ is such that $\mathbf{B}\,\vec{x} \neq \vec{0}$ if $\vec{x} \neq \vec{0}$, i.e. $\ker \mathbf{B} = \{\vec{0}\}$. Then $\mathbf{C} = \mathbf{B}^T \cdot \mathbf{A} \cdot \mathbf{B} \in \mathbb{R}^{m\times m}$ is also symmetric and positive definite. If the condition on $\mathbf{B}$ is not satisfied or $\mathbf{A}$ is only positive semidefinite then $\mathbf{C}$ is positive semidefinite.* ♢
Proof : If $\vec{x} \neq \vec{0}$ then $\vec{y} = \mathbf{B}\,\vec{x} \neq \vec{0}$ and thus

$$\langle \vec{x}\,,\, \mathbf{C}\,\vec{x}\rangle = \langle \vec{x}\,,\, \mathbf{B}^T \cdot \mathbf{A} \cdot \mathbf{B}\,\vec{x}\rangle = \langle \mathbf{B}\,\vec{x}\,,\, \mathbf{A} \cdot \mathbf{B}\,\vec{x}\rangle = \langle\, \vec{y}\,,\, \mathbf{A}\,\vec{y}\rangle > 0$$

This implies all of the claimed results. □

The following two results are verified by elementary computations.

11–15 Result : *The sum of positive semidefinite matrices is positive semidefinite.* ♢

11–16 Result : *If for a symmetric, positive semidefinite matrix $\mathbf{A}$ the equation $\mathbf{A}\,\vec{x} = \vec{0}$ has only the trivial solution, i.e. 0 is not an eigenvalue, then $\mathbf{A}$ is positive definite.* ♢

11.2 The Cholesky decomposition

11–17 Result : *If a symmetric matrix* $\mathbf{A}$ *is strictly positive definite, then there exists a diagonal matrix* $\mathbf{D}$ *with positive entries only and a upper triangular unity matrix* $\mathbf{R}$ *(numbers 1 on the diagonal) such that*

$$\mathbf{A} = \mathbf{R}^T \cdot \mathbf{D} \cdot \mathbf{R}$$

This is called the ***Cholesky decomposition***[2] *of the matrix* $\mathbf{A}$*. If an above decomposition exists for a symmetric matrix* $\mathbf{A}$ *and the diagonal matrix* $\mathbf{D}$ *has only positive entries, then* $\mathbf{A}$ *is positive definite.* $\diamond$

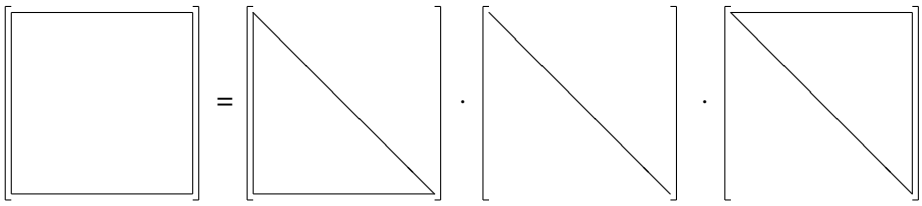


Figure 11.1: The Cholesky decomposition of a symmetric matrix

The factorization of the full matrix $\mathbf{A}$ is visualized in Figure 11.1. If we provide an algorithm to find this factorization then we have a method to decide whether $\mathbf{A}$ is positive definite or not. In section 11.2.3 it is shown that if $\mathbf{A}$ is positive definite, then the algorithm is stable and no pivoting is necessary. If $\mathbf{A}$ is not positive definite then the algorithm shown below might not be stable and some pivoting scheme should be used. Possible methods can be found in [GoluVanLoan96, §4.4].

Since $\mathbf{A}$ and $\mathbf{D}$ have the same number of positive, zero and negative eigenvalues ([Axel94, Theorem 3.20]) we know the number of positive eigenvalues of $\mathbf{A}$, once $\mathbf{D}$ is known.

Instead of solving the system $\mathbf{A}\,\vec{x} = \vec{b}$ one can then solve three much simpler problems.

$$\mathbf{A}\,\vec{x} = \vec{b} \quad \Longleftrightarrow \quad \begin{array}{lrcll} (1) & \mathbf{R}^T\,\vec{y} & = & \vec{b} & \text{from top to bottom} \\ (2) & \mathbf{D}\,\vec{z} & = & \vec{y} & \text{divisions} \\ (3) & \mathbf{R}\,\vec{x} & = & \vec{z} & \text{from bottom to top} \end{array}$$

11.2.1 The algorithm of Cholesky for a 3×3 matrix

This Cholesky decomposition can be generated by applying row and column operations to the matrix $\mathbf{A}$ to transform it into diagonal form. To examine this we consider a simple example, observing that the ideas and operations also apply to the general case.

[2] The standard notation for the Cholesky decomposition is $\mathbf{A} = \mathbf{R}_1^T \cdot \mathbf{R}_1$ where $\mathbf{R}_1$ is an upper triangular matrix and does not need to have numbers 1 on the diagonal. Letting $\mathbf{R}_1 = \sqrt{\mathbf{D}} \cdot \mathbf{R}$ one may generate the standard presentation using the result given here. The chosen algorithm has the advantage that also some non positive definite problems can be solved, as long as not pivoting is necessary. In [GoluVanLoan96] the presented result is called a LDL^T factorization.

11–18 Example : For the simple 3×3 matrix $\mathbf{A}$

$$\mathbf{A} = \begin{bmatrix} 1 & 3 & -4 \\ 3 & 11 & 0 \\ -4 & 0 & 10 \end{bmatrix} = \mathbf{A}_1$$

we apply two sets of row and column operations to transform $\mathbf{A}$ into diagonal form.

- First generate zeros in the first column, below the diagonal. Then treat the first row accordingly.
  - Subtract 3 times the first row from the second row.
  - Add 4 times the first row to the third row.
  - Subtract 3 times the new first column from the second column.
  - Add 4 times the new first column to the third column.

  All the above operations can be written as matrix multiplications with elementary matrices. We obtain.

$$\mathbf{C}_1^T \cdot \mathbf{A}_1 \cdot \mathbf{C}_1 = \begin{bmatrix} 1 & 0 & 0 \\ -3 & 1 & 0 \\ +4 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 3 & -4 \\ 3 & 11 & 0 \\ -4 & 0 & 10 \end{bmatrix} \cdot \begin{bmatrix} 1 & -3 & +4 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 12 \\ 0 & 12 & -6 \end{bmatrix} = \mathbf{A}_2$$

- Now subtract 6 times the second row from the third row and do the corresponding operation on the columns. You find a matrix $\mathbf{A_3} = \mathbf{D}$ with nonzero entries in the diagonal only.

$$\begin{aligned} \mathbf{C}_2^T \cdot \mathbf{A}_2 \cdot \mathbf{C}_2 &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -6 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 12 \\ 0 & 12 & -6 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -6 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & -78 \end{bmatrix} \\ &= \mathbf{A}_3 = \mathbf{D} \end{aligned}$$

- The above two sets of row and column operations can thus be written as

$$\mathbf{A}_3 = \mathbf{C}_2^T \cdot \mathbf{A}_2 \cdot \mathbf{C}_2 = \mathbf{C}_2^T \cdot \mathbf{C}_1^T \cdot \mathbf{A}_1 \cdot \mathbf{C}_1 \cdot \mathbf{C}_2 = (\mathbf{C}_1 \cdot \mathbf{C}_2)^T \cdot \mathbf{A}_1 \cdot (\mathbf{C}_1 \cdot \mathbf{C}_2)$$

  or equivalently as

$$\mathbf{A}_1 = \left((\mathbf{C}_1 \cdot \mathbf{C}_2)^T\right)^{-1} \cdot \mathbf{A}_3 \cdot (\mathbf{C}_1 \cdot \mathbf{C}_2)^{-1} = \mathbf{R}^T \cdot \mathbf{A}_3 \cdot \mathbf{R} = \mathbf{R}^T \cdot \mathbf{D} \cdot \mathbf{R}$$

  It remains to verify that $\mathbf{R}$ is in fact an upper triangular matrix. For this we have to realize that R can be produced by a sequence of column operations applied to the identity matrix

$$\begin{aligned} \mathbf{R} &= (\mathbf{C}_1 \cdot \mathbf{C}_2)^{-1} = \mathbf{C}_2^{-1} \cdot \mathbf{C}_1^{-1} \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -6 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} 1 & -3 & 4 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}^{-1} \\ &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & +6 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 3 & -4 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & +3 & -4 \\ 0 & 1 & +6 \\ 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

- Observe that the entries in the matrix $\mathbf{R}$ are the numbers used while transforming $\mathbf{A}$ into diagonal form $\mathbf{D}$. Thus we can store the numbers in $\mathbf{R}$ while performing the row/column operations on $\mathbf{A}$.

As a final result we obtain the factorization

$$\mathbf{A} = \begin{bmatrix} 1 & 3 & -4 \\ 3 & 11 & 0 \\ -4 & 0 & 10 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ +3 & 1 & 0 \\ -4 & +6 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & -78 \end{bmatrix} \cdot \begin{bmatrix} 1 & +3 & -4 \\ 0 & 1 & +6 \\ 0 & 0 & 1 \end{bmatrix} = \mathbf{R}^T \cdot \mathbf{D} \cdot \mathbf{R}$$

♢

11–19 Example : Instead of solving the system of three linear equations

$$\begin{bmatrix} 1 & 3 & -4 \\ 3 & 11 & 0 \\ -4 & 0 & 10 \end{bmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

we can use the Cholesky decomposition and solve three simple systems of equations.

1. Solve the equation $\mathbf{R}^T \vec{y} = \vec{b}$ from top to bottom, i.e. first solve the first equation for y_1, then the second for y_2 and finally the third for y_3.

$$\begin{bmatrix} 1 & 0 & 0 \\ +3 & 1 & 0 \\ -4 & +6 & 1 \end{bmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} \qquad \Longrightarrow \qquad y_1 = 1 \quad , \quad y_2 = -1 \quad , \quad y_3 = 13$$

2. The system $\mathbf{D}\,\vec{z} = \vec{y}$ can be solved very easily, divide the values of y_i by the corresponding number in $\mathbf{D}$.

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & -78 \end{bmatrix} \cdot \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \\ 13 \end{pmatrix} \qquad \Longrightarrow \qquad z_1 = 1 \quad , \quad z_2 = \frac{-1}{2} \quad , \quad z_3 = \frac{13}{-78} = \frac{-1}{6}$$

3. Solve the equation $\mathbf{R}\,\vec{x} = \vec{z}$ from bottom to top, i.e. first solve the third equation for x_3, then the second for x_2 and finally the first for x_1.

$$\begin{bmatrix} 1 & +3 & -4 \\ 0 & 1 & +6 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ \frac{-1}{2} \\ \frac{-1}{6} \end{pmatrix} \qquad \Longrightarrow \qquad x_3 = \frac{-1}{6} \quad , \quad x_2 = \frac{1}{2} \quad , \quad x_1 = \frac{-7}{6}$$

Observe that each of the three systems is easily solvable. ♢

11–20 Example : As the entry -78 in the diagonal matrix $\mathbf{D}$ is negative we know that the original matrix $\mathbf{A}$ is not positive definite. To find a vector $\vec{x}$ such that $\langle \vec{x}\,,\, \mathbf{A}\,\vec{x} \rangle < 0$ we may solve the system

$$\mathbf{R}\,\vec{x} = \begin{bmatrix} 1 & +3 & -4 \\ 0 & 1 & +6 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

This leads to

$$\langle \vec{x}\,,\, \mathbf{A}\,\vec{x}\rangle = \langle \vec{x}\,,\, \mathbf{R}^T\cdot\mathbf{D}\cdot\mathbf{R}\,\vec{x}\rangle = \langle \mathbf{R}\,\vec{x}\,,\, \mathbf{A}\cdot\mathbf{R}\,\vec{x}\rangle = \langle \begin{pmatrix} 0\\0\\1\end{pmatrix}\,,\, \begin{bmatrix} 1 & 0 & 0\\ 0 & 2 & 0\\ 0 & 0 & -78\end{bmatrix}\begin{pmatrix} 0\\0\\1\end{pmatrix}\rangle = -78$$

and thus **A** is not positive definite. ♢

11.2.2 The algorithm and an implementation in *Octave*

The example above leads to the Cholesky algorithm for the factorization of a positive matrix **A**, as shown in Table 11.1. This algorithm can be implemented in any programming language. The code below shows the straightforward code in *Octave* (a MATLAB clone).

```
for each row:                                    for k=1:n
  for each row below the current row               for j=k+1:n
    find the factor for the row operation            R(k,j) = A(j,k)/A(k,k);
    do the row operation                             A(j,:) = A(j,:) - R(k,j)*A(k,:);
    do the column operation                          A(:,j) = A(:,j) - R(k,j)*A(:,k);
```

Table 11.1: Algorithm of Cholesky

Octave

```
function [R,D] = choleskyDiag(A)
% [R,D] = choleskyDiag(A) if A is a symmetric positive definite matrix
%         returns a upper triangular matrix R and a diagonal matrix D
%         such that  A = R'*D*R

% this code can only be used for didactical purposes
% it has some major flaws!

[n,m] = size(A);
D=zeros(n);
R=zeros(n);

for k=1:n-1
  R(k,k)=1;
  for j=k+1:n
     R(k,j) = A(j,k)/A(k,k);
     % row operations
     A(j,:) = A(j,:) - R(k,j)*A(k,:);
     % column operations
     A(:,j) = A(:,j) - R(k,j)*A(:,k);
  endfor
R(n,n)=1;
endfor
D=diag(diag(A));
```

The above code has some serious flaws

- It does not check for correct size of the input.
- It does not check for a possible division by 0.

- As we go through the algorithm the coefficients in **R** can replace the coefficients in **A** which will not be used any more. This cuts the memory requirement in half.
- If we do all computations in the upper right part of **A**, we already know that the result in the lower left part has to be the same. Thus we can do only half of the calculations.
- As we already know that the numbers in the diagonal of **R** have to be 1, we do not need to return them. One can use the diagonal of **R** to return the coefficients of the diagonal matrix **D**.

If we implement most[3] of the above points we obtain an improved algorithm, shown below.

Octave

```
function R = cholesky(A)
% R = cholesky(A) if A is a symmetric positive definite matrix
%         returns a upper triangular matrix R and a diagonal matrix D
%         such that  A = R1'*D*R1
%         R1 has all diagonal entries equals 1
%         the values of D are returned on the diagonal of R

TOL=1e-10; %% there certainly are better tests than this!!

[n,m] = size(A);
if (n!=m)   error ("cholesky: matrix has to be square ") endif

for k=1:n-1
  if ( abs(A(k,k)) <= TOL) error ("cholesky:might be a singular matrix")
    endif
  for j=k+1:n
     A(j,k) = A(k,j)/A(k,k);
     % row operations only
     A(j,j:n) = A(j,j:n) - A(j,k)*A(k,j:n);
  endfor
endfor
if ( abs(A(n,n)) <= TOL) error ("cholesky:might be a singular matrix")
  endif

% return the lower triangular part of A.
% Transpose it to obtain an upper triangular matrix
R=tril(A)';
```

This code can be used for an operation count, i.e. how many operations does a computer have to perform to complete the algorithm. We are only interested in cases where n is rather large, where $n < 100$ is certainly considered small. We consider one **flop** (floating point operation) as a unit of

- one addition/subtraction and one multiplication of floating point numbers.
- two memory fetches.

The inner most loop (index `(j:n)` in MATLAB notation) performs $n - j + 1$ such operations. The second loop (variable `k`) is performed $n - k$ times. Thus the two inner loops consume a total of[4]

$$\sum_{j=k+1}^{n} (n - j + 1) = \sum_{i=1}^{n-k} i \approx \frac{1}{2} (n - k)^2 \quad \text{flops}$$

[3] The memory requirements can be made considerably smaller

[4] We used the simple approximation idea

$$\sum_{j=0}^{n} j^m \approx \int_0^n x^m \, dx = \frac{1}{m+1} n^{m+1}$$

The outer most loop has $n-1$ cycles. This leads to a total flop count of

$$\sum_{k=1}^{n-1} \frac{1}{2}\,(n-k)^2 = \frac{1}{2}\,\sum_{i=1}^{n-1} i^2 \approx \frac{1}{6}\,n^3$$

leads to a total computational effort of

$$\text{Flop}_{\text{Chol}} = \frac{1}{6}\,n^3$$

This information is useful to estimate the computation time of the algorithm on a given computer, it should be proportional to n^3.

The above code finds the Cholesky factorization of the matrix, but does not solve a system of linear equations. It has to be supplemented with the corresponding back-substitution algorithm.

Octave

```
function x = choleskySolver(R,b)
% x = choleskySolver(R,b) solves A x = b
%     R has te be generated by R=cholesky(A)

[n,m] = size(R);
if (n!=length(b))
  error ("choleskySover: matrix and vector do not have same dimension ") endif

% forward substitution of R' y = b
y=zeros(size(b));
y(1)=b(1);
for k=2:n
  y(k) = b(k);
  for j=1:k-1  y(k) = y(k) - R(j,k)*y(j);endfor
endfor

% solve diagonal system
for k=1:n y(k) = y(k)/R(k,k);endfor

% forward substitution of R' y = b
x=zeros(size(b));
x(n)=y(n);
for k=1:n-1
  x(n-k) = y(n-k);
  for j=n-k+1:n  x(n-k) = x(n-k) - R(n-k,j)*x(j);endfor
endfor
```

Now we can solve the numerical example on the previous pages by

Octave

```
A=[1 3 -4; 3 11 0; -4 0 10];
R = cholesky(A)
b=[1; 2; 3];
x=choleskySolver(R,b)'
.

R =
```

An exact inequality is given by

$$\frac{(n-1)^{m+1}}{m+1} = \int_0^{n-1} x^m\,dx < \sum_{j=1}^{n} j^m < \int_1^n x^m\,dx = \frac{n^{m+1}-1}{m+1}$$

```
     1    3   -4
     0    2    6
     0    0  -78

 x =  -1.16667  0.50000  -0.16667
```

The operation count of this algorithm is given by

$$\text{Flop}_{\text{Solve}} = n^2$$

For large n this is small compared to the $n^3/6$ operations for the factorization.

11.2.3 Stability of the Cholesky algorithm

When running the algorithm of Cholesky on a computer we will certainly encounter numerical rounding errors. The effect of these have be extensively studied, e.g. [Wilk63], [High96]. The good news is that for positive definite, symmetric matrices the effect of rounding errors can be controlled ([GoluVanLoan96, Theorem 4.2.4 and §4.7.2]). For given $\mathbf{A}$ and $\vec{b}$ we will find a Cholesky factorization, to be used for forward and backward substitution to determine an approximate solution $\hat{\vec{x}}$ of the system $\mathbf{A}\,\vec{x} = \vec{b}$. This approximate solution is an exact solution of an approximate equation

$$(\mathbf{A} + \mathbf{E})\,\hat{\vec{x}} = \vec{b}$$

where the size of the entries in the matrix $\mathbf{E}$ can be controlled, i.e. we have **backward stability** of the algorithm. The concept of backward stability is explained in [Wilk63]. The above does not imply that the effect of the roundoff errors is always small. If the condition number κ is large we may in fact find large roundoff errors. But combined with the error estimate (11.1) (page 258) this is the best result we can hope for.

To show that the Cholesky algorithm is stable for positive definite system two essential ingredients are used

- Show that the entries in the factorization $\mathbf{R}$ and $\mathbf{D}$ are bounded by the entries in $\mathbf{A}$. This is only correct for positive definite matrices.
- Keep track of rounding errors for the algebraic operations to be executed during the algorithm of Cholesky.

The entries of R and D are bounded

For a symmetric, positive definite matrix we have the factorization

$$\mathbf{A} = \mathbf{R}^T \cdot \mathbf{D} \cdot \mathbf{R}$$

By multiplying out the diagonal elements we obtain

$$a_{i,i} = d_i + \sum_{k=1}^{i-1} r_{k,i}\, d_k\, r_{k,i} = d_i + \sum_{k=1}^{i-1} d_k\, r_{k,i}^2$$

Thus we find bounds on the coefficients in $\mathbf{R}$ and $\mathbf{D}$ in terms of $\mathbf{A}$.

$$d_i \le a_{i,i} \quad \text{and} \quad \sum_{k=1}^{i-1} d_k\, r_{k,i}^2 \le a_{i,i}$$

Using this and the Cauchy–Schwartz inequality we now obtain an estimate for the result of the matrix multiplication below, where the entries in $|\mathbf{R}|$ are given by the absolute values of the entries in $\mathbf{R}$. Estimates of this type are needed to keep track of the 'worst case' situation for rounding errors and the algorithm.

$$\begin{aligned}\left(|\mathbf{R}|^T \cdot \mathbf{D} \cdot |\mathbf{R}|\right)_{i,j} &= \sum_{k=1}^{n} |r_{k,i}|\, d_k\, |r_{k,j}| \\ &\leq \sqrt{\sum_{k=1}^{n} d_k\, r_{k,i}^2} \cdot \sqrt{\sum_{k=1}^{n} d_k\, r_{k,j}^2} \\ &\leq \sqrt{a_{i,i}} \cdot \sqrt{a_{j,j}} \leq \max_{1\leq j\leq n} a_{j,j} \end{aligned} \tag{11.2}$$

Example 11–23 shows that the above is false if $\mathbf{A}$ is not positive definite.

Rounding errors while solving

When finite precision arithmetic is used to perform matrix operation round off errors will invariably occur. We use the **unit round off error** ε. When a number of size comparable to 1 has to be rounded of an error of the size ε will occur. As an example we consider the computation of a scalar product, see [GoluVanLoan96]. For $\vec{x}, \vec{y} \in \mathbb{R}^n$ we perform the operations to compute $\langle \vec{x}\,,\, \vec{y} \rangle$ in a programming language.

Octave

```
function s=scalarproduct(x,y)
  s=0
  for i=1:size(x) s = s +  x(i)*y(i)
endfunction
```

and call the result $s = fl(\langle \vec{x}\,,\, \vec{y} \rangle)$. One can verify that

$$|fl(\langle \vec{x}\,,\, \vec{y} \rangle) - \langle \vec{x}\,,\, \vec{y} \rangle| \leq n\,\varepsilon\, \langle |\vec{x}|\,,\, |\vec{y}| \rangle + O(\varepsilon^2)$$

Based on this results on the effect or round of errors during Cholesky factorizations can be controlled, leading to the following two results.

11–21 Result : *(Modification of [GoluVanLoan96, Theorem 3.3.1])*
Assume that for a positive definite, symmetric $n \times n$ matrix $\mathbf{A}$ the algorithm of Cholesky leads to an approximate factorization

$$\hat{\mathbf{R}}^T \cdot \hat{\mathbf{D}} \cdot \hat{\mathbf{R}} = \mathbf{A} + \mathbf{H}$$

Then the error matrix $\boldsymbol{H}$ satisfies

$$|\mathbf{H}| \leq 3\,(n-1)\,\varepsilon\, \left(|\mathbf{A}| + |\mathbf{R}|^T \cdot |\mathbf{D}| \cdot |\mathbf{R}|\right) + O(\varepsilon^2)$$

◇

The estimate (11.2) for a positive definite $\mathbf{A}$ now implies

$$|\mathbf{H}| \leq 6\,(n-1)\,\varepsilon\, \max_i a_{i,i}$$

11–22 Result : *(Modification of [GoluVanLoan96, Theorem 3.3.2])*
Let $\hat{\mathbf{R}}$ and $\hat{\mathbf{D}}$ be the computed factors of the Cholesky factorization of the $n \times n$ matrix $\mathbf{A}$. Then forward and back substitution are used to solve $\hat{\mathbf{D}} \cdot \hat{R}^T\ \vec{y} = \vec{b}$ with computed solution $\hat{\vec{y}}$ and solve $\hat{\mathbf{R}}\,\vec{x} = \hat{\vec{y}}$ with computed solution $\hat{\vec{x}}$. Then

$$(\mathbf{A} + \mathbf{E})\,\hat{\vec{x}} = \vec{b} \quad \textit{with} \quad |\mathbf{E}| \leq n\,\varepsilon\, \left(3\,|\mathbf{A}| + 5\,|\mathbf{R}|^T \cdot |\mathbf{D}| \cdot |\mathbf{R}|\right) + O(\varepsilon^2)$$

◇

The estimate (11.2) for a positive definite **A** now implies

$$|\mathbf{E}| \leq 8\, n\, \varepsilon\, \max_i a_{i,i}$$

i.e. the result of the numerical computations is the exact solution of a slightly modified equations. The modification is small compared to the maximal coefficient in the original problem.

11–23 Example : If the matrix is not positive definite the effect of roundoff errors may be large, even if the matrix has a condition number close to 1. Consider the matrix

$$\begin{bmatrix} 0.0001 & 1 \\ 1 & 0.0001 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Exact arithmetic leads to the factorization

$$\begin{bmatrix} 0.0001 & 1 \\ 1 & 0.0001 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 10000 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0.0001 & 0 \\ 0 & -9999.9999 \end{bmatrix} \cdot \begin{bmatrix} 1 & 10000 \\ 0 & 1 \end{bmatrix}$$

The condition number is $\kappa = 1.0002$ and thus we expect almost no loss of precision. The exact solution is $\vec{x} = (0.99990001\,,\, 0.99990001)^T$. Since all numbers in **A** and $\vec{b}$ are smaller than 1 one might hope for an error of the order of machine precision. The bounds on the entries in **R** and **D** in (11.2) are clearly violated, e.g.

$$\left(|\mathbf{R}|^T \cdot \mathbf{D} \cdot |\mathbf{R}|\right)_{2,2} = |r_{1,2}|\, d_1\, |r_{1,2}| + |r_{2,2}|\, d_2\, |r_{2,2}| = 10^8 \cdot 10^{-4} + 9999.9999 \approx 20000$$

Using floating point arithmetic with $\varepsilon \approx 10^{-8}$ we obtain a factorization

$$\begin{bmatrix} 1 & 0 \\ 10000 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0.0001 & 0 \\ 0 & -10000 \end{bmatrix} \cdot \begin{bmatrix} 1 & 10000 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.0001 & 1 \\ 1 & 0 \end{bmatrix}$$

and the solution is $\hat{\vec{x}} = (1.0\,,\, 0.9999)^T$. Thus the relative error of the solution is 10^{-4}. This is by magnitudes larger than the machine precision $\varepsilon \approx 10^{-8}$. The effect is generated by the large numbers in the factorization. This can not occur if the matrix **A** is positive definite since we have the bound (11.2). To overcome this type of problem a good pivoting scheme has to be used when the matrix is not positive definite, see e.g. [GoluVanLoan96, §4.4]. ♢

11.3 Banded matrices

For systems of linear equations generated by a finite element approximation the corresponding matrix has most entries equals 0. Matrices of this type are called **sparse**. As a typical example consider section 7.3 where all nonzero entries in matrix **A** are along the main diagonal, 2 upper and 2 lower diagonals. **Iterative methods** take advantage of this property, e.g. [Axel94]. We present the most elementary direct method using the band structure of the matrix **A**. This approach is practical if the degrees of freedom in a finite element problem are numbered to minimize the bandwidth of the matrix.

11.3.1 The algorithm of Cholesky for banded matrices

If a symmetric matrix **A** has all nonzero numbers close to diagonal, then it is called a **banded matrix**. If $a_ij = 0$ for $|i - j| >= b$ then the integer B is called the **semibandwidth** of **A**. For a tridiagonal matrix we find $b = 2$, the main diagonal and one off-diagonal. As the algorithm of Cholesky is based on row and column operation we can apply it to a banded matrix and as long as no pivoting is done the band structure of

the matrix is maintained. Thus we can factor a positive definite symmetric matrix $\mathbf{A}$ with semibandwidth b as

$$\mathbf{A} = \mathbf{R}^T \cdot \mathbf{D} \cdot \mathbf{R}$$

where $\mathbf{R}$ is an upper triangular unity matrix with semibandwidth b and $\mathbf{D}$ is a diagonal matrix with positive entries. This situation is visualized in Figure 11.2.

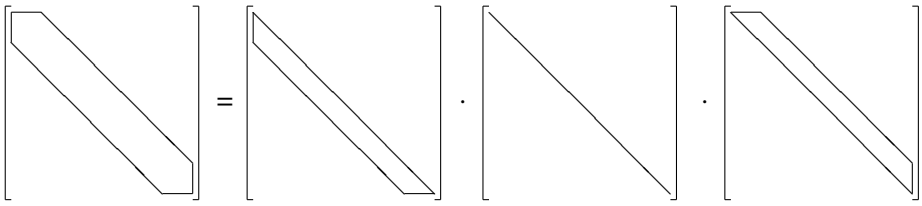


Figure 11.2: The Cholesky decomposition for a banded matrix

For a $n \times n$ matrix $\mathbf{A}$ we are interested in the situation

$$1 < b \ll n$$

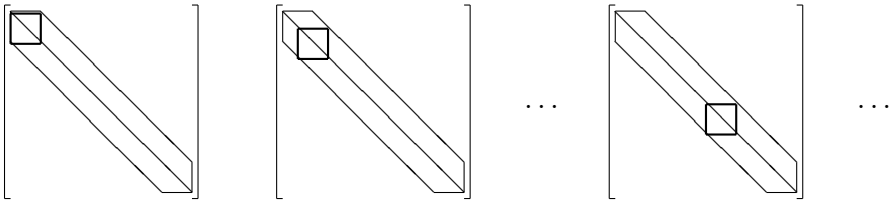


Figure 11.3: Cholesky steps for a banded matrix. The active area is marked

When implementing the algorithm of Cholesky (see Table 11.1 on page 265) one works along the diagonal, top to bottom. For each step only a block of size $b \times b$ of numbers is worked on, i.e. has to be quickly accessible. Three of these situations are shown in Figure 11.3. Each of those steps needs approximately $b^2/2$ flops and there are about n of those, thus we find an approximate[5] operational count of

$$\text{Flop}_{\text{CholBand}} \approx \frac{1}{2}\, b^2\, n$$

The additional cost to solve a system by back substitution is approximately

$$\text{Flop}_{\text{SolveBand}} \approx 2\, b\, n$$

We need $n \cdot b$ numbers to store the complete matrix $\mathbf{A}$. As the algorithm proceeds along the diagonal in $\mathbf{A}$ (or its reduction $\mathbf{R}$) in each step only the next b rows will be worked on. As we go to the next row the previous top row will not be used any more but a new row bottom will will be needed, see also Figure 11.3. Thus we have only $b \cdot b$ active entries at any time. If these numbers can be placed in **fast memory** then the implementation will run faster than in regular memory. Thus for good performance we like to store b^2 numbers in fast memory. This have to be taken in consideration when setting up the data structure for a banded matrix together with the memory and cache architecture of the computer to be used. Table 11.2 shows the types of fast and regular memory relevant for most problems and some typical sizes of matrices. If not enough fast memory is available the algorithm will still generate the result but not as fast. This fact will be examined in the next sections, using a C++ implementation.

[5] We ignored the effect that the first row in each diagonal steps is left unchanged and we also do not take into account that in the lower right corner fewer computations are needed. Both effects are of lower order.

size	band	fast memory (MB)	memory (MB)
regular problem		cache	RAM
huge problem		RAM	hard disk
100	10	0.0008	0.008
200	20	0.0032	0.032
1'000	100	0.08	0.8
10'000	100	0.08	8
100'000	100	0.08	80
100'000	200	0.32	160
100'000	500	2	400
100'000	1'000	8	800

Table 11.2: Memory requirements for the Cholesky algorithm for banded matrices

11.3.2 An implementation in C++

The ideas of the previous sections can be implemented in any good programming language. Here parts of a C++ implementation are shown and the performance of a few slightly different implementations is examined. The goal is to have reasonably optimized code.

Data structure

For a symmetric $n \times n$ matrix $\mathbf{A}$ with semibandwidth b a $n \times b$ matrix will be able to store all numbers, as only the upper right half has to be considered. Due to the observations in the previous section we store the matrix row by row, but only the b numbers on and to the right of the main diagonal. The example below illustrates this fact where $n = 5$ and $b = 3$.

$$\mathbf{A} = \begin{bmatrix} 11 & 2 & 3 & 0 & 0 \\ 2 & 17 & 0 & 4 & 0 \\ 3 & 0 & 12 & 4 & 4 \\ 0 & 1 & 1 & 10 & 3 \\ 0 & 0 & 1 & 3 & 19 \end{bmatrix} \quad \longrightarrow \quad \begin{bmatrix} 11 & 2 & 3 \\ 17 & 0 & 1 \\ 12 & 1 & 1 \\ 10 & 3 & \\ 19 & & \end{bmatrix}$$

To access this memory we use two different methods. A block of memory to store $n \times b$ double precision numbers is dynamically allocated. The pointer `R` indicates the position of the first number and we can access all entries in $\mathbf{A}$ by `R[i]`.

```
R -> [11, 2, 3, 17, 0, 1, 12, 1, 1, 10, 3, 0, 19, 0, 0]
```

As an example[6] `R[6]` will return 12. The same block of memory can also be accessed by an array of pointers `Rp` to get to each row in the matrix $\mathbf{A}$.

```
Rp[0] -> [11, 2, 3]
Rp[1] -> [17, 0, 1]
Rp[2] -> [12, 1, 1]
```

[6] Observe that the array numbering starts with 0 and not with 1. Thus to access the seventh number we have to use `R[6]`. This is a common source of programming problems.

```
Rp[3] -> [10, 3, 0]
Rp[4] -> [19, 0, 0]
```

Here the number 12 is returned by `Rp[2][0]` .

A first implementation

Using the above memory structure we can now implement the basic Cholesky algorithm. The `C++` class uses the variables `length` and `band` to indicate size and semibandwidth of the matrix. On calling the routine the memory block `R` contains the values of the matrix **A**. On return these values are replaced by the values of the factorization $\mathbf{R}^T \cdot \mathbf{D} \cdot \mathbf{R}$. The first column of `Rp` contains the diagonal elements of **D** and the values of the upper triangular unit matrix **R** are stored in the other columns. The values of the original matrix **A** will be overwritten.

```
 10  int jMax; double tmp; double* tmprow;
 20  tmprow = new double[band];
 30  for(int k= 0; k < length-1; k++) {
 40    if ( fabs(Rp[k][0]) <= TOL ) { return message }
 50    jMax = min(band,length-k);
 60    for (int j=0;j < jMax; j++){tmprow[j]=Rp[k][j];}
 70    for (int j=1;j < jMax; j++){
 80      tmp = Rp[k][j] =tmprow[j]/tmprow[0];
 90      for(int i=0;i<(jMax-j);i++) Rp[k+j][i] -= tmp*tmprow[j+i];
100    }; //for j flops: band^2/2
110  }; //for k   length iterations
120  delete[] tmprow;
```

Table 11.3: First implementation of the Cholesky algorithm

The code in Table 11.3 is explained below line by line.

10: define temporary variables

20: allocate the memory for temporary storage of one row of the matrix.

30: The counter `k` indicates the current row/column. With the help of this row all rows with a higher index will be modified. But only `jMax` row will actually change, due to the band structure of the matrix.

40: If the diagonal element is to close to 0 the matrix may be singular, if the element is negative the matrix is not positive definite.

50: The variable `jMax` indicates how many elements are present in the current row. The result is typically `band`, unless we are in the lower right corner.

60: the current row is copied into the temporary row `tmprow` .

70: Row operations have to be applied to `jMax` rows below the current row. The index `j` indicates that we are working on row `k+j` .

80: Determine the factor `tmp` and store it in `Rp[k][j]`. This is part of the result to be returned.

90: Subtract the multiple of the temporary row. This is the inner most loop and will consume most of the computational time. One loop typically requires `band-j` flops.

100: End of the `j`-loop. As `j` runs from 1 to `band` we use about $\texttt{band}^2/2$ flops in each loop.

110: End of the `k`-loop. As `k` runs from 1 to `length` we use about $\texttt{length}\cdot\texttt{band}^2/2$ flops in the algorithm.

120: Deallocate the temporary memory of `tmprow` .

A second implementation

The above code does solve the problem, but we might try to make some modifications to obtain faster code, i.e. tune the code. Ideas on how to proceed can be found in [DowdSeve98]. The numbering of the code lines is taken from 11.3 to show similarities.

```
 10  int jMax; double tmp; double* tmprow; int kj;
 20  tmprow = new double[band];
 30  for(int k= 0; k < length-1; k++) {
 40    if ( fabs(Rp[k][0]) <= TOL ) { return message }
 50    jMax = min(band,length-k);
 60    for (int j=jMax-1;j >= 0; j--){tmprow[j]=Rp[k][j];}
 65    kj=(k+jMax)*band; // initialise Rp[k+jMax][0] = R[(k+jMax)*band]
 70    for (int j=jMax-1; j>0; j--){
 80      tmp = Rp[k][j] = tmprow[j]/tmprow[0];
 85      kj -= band;
 90      for(int i= jMax-j-1 ; i>=0; i--) R[kj+i] -= tmp*tmprow[j+i];
100    }; //for j flops: band^2/2
110  }; //for k   length iterations
120  delete[] tmprow;
```

Table 11.4: Second implementation of the Cholesky algorithm

Some of the changes in Table 11.4 might lead to faster code.

60: Instead of increasing to counter `j++` we decrease `j--`. Thus the test `j>=0` in the loop is against 0 and does not need an additional variable `jMax` on each test.

70: This loop is also used with decreasing counter.

90: Again the counter `i` is decreased.
The reference `Rp[k+j][i]` is replaced by `R[kj+i]`. Thus the compiler does not have to work with a two dimensional array, but with a simple array. This simplifies the address computation for the compiler. As a tradeoff we have to explicitly initialize the row counter in line 65 and decrement it on line 85 .

11.3.3 Performance tests on different computers

As a test problem we consider the finite element (or finite difference) problem from section 7.3. We vary the number of interior grid points in x and y direction, i.e. nx and ny. The corresponding symmetric, positive definite matrix has size $nx \cdot ny$ and semi-bandwidth $nx + 1$. All computations are performed with double precision and as each number requires 8 byte of storage we can compute the total memory needed ($\approx nx^2 \cdot ny \cdot 8$ Byte) and the need for fast memory ($\approx nx^2 \cdot 8$ Byte). A set of 10 different values for nx and ny was used, chosen such that the time needed to complete the computations did not stretch the authors patience too much. The numbers are shown in Table 11.5.

The tests were run with three different implementations of the algorithm of Cholesky.

nx	ny	size	memory (MB)	fast memory (MB)
64	1000	64000	32.8	0.03
96	296	28416	21.8	0.07
144	87	12528	14.4	0.17
216	25	5400	9.3	0.37
324	7	2268	5.9	0.84
486	4	1944	7.6	1.89
729	4	2916	17.0	4.25
1093	4	4372	38.2	9.56
1639	4	6556	86.0	21.50
1693	4	6772	91.7	22.93

Table 11.5: Data for the test problems for the Cholesky algorithm

A The code was copied from a public source and run without major modifications. This implementation does not require temporary storage for one row but needs more floating point operations. An improved version is discussed in exercise 11–6.

B The code in Table 11.3.

C The code in Table 11.4.

The timing was done using standard Unix system calls and indicate the time the system spent to compute the Cholesky factorization. As a result we show the number of flops per second as function of nx.

The results below illustrate that one not only has to choose a good algorithm for a given problem but a careful implementation is important too to achieve good performance on a given platform. Some aspects of basic code tuning are discussed in [Bent00], Appendix 4 gives some basic rules for tuning. Readers interested in this topic might consider reading *High Performance Computing* [DowdSeve98]. This book presents many important aspect of computing on personal computers, it explains many aspects of modern CPU and memory architectures. For professionals interested in scientific computing or numerical analysis this book is a 'must read' (and understand).

Results for a Pentium III based computer

This[7] is dual Pentium III 600MHz PC with 512KB cash per CPU and 512M RAM. The operating system is Linux 2.2.12-20smp and the compiler gcc 2.95.2 with the switches `-O3 -ffast-math`. The system was only very lightly loaded by other processes. The results are shown in Figure 11.4.

- The cache is 512KB and thus if $nx = 250$ a block of $nx \cdot nx$ numbers will fill the cache memory completely. The drop-of in performance as nx crosses this limit is clearly visible for all implementations used.
- Implementation C gives the best performance, followed by B.
- Implementation B was run again (B2) with doubled values for ny. The results are identical to B. This indicates that the performance depends mainly on the fast memory requirements.

[7] provided by the Department of Mathematics, University of Utah

These results point towards implementation C as being the most efficient.

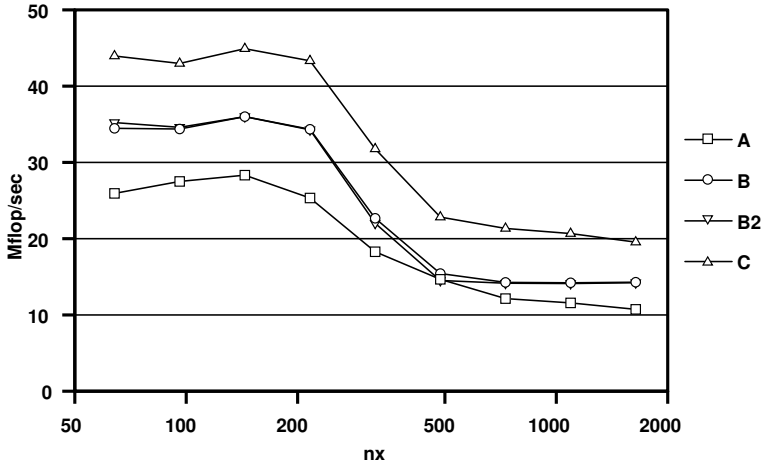


Figure 11.4: Performance of Cholesky algorithm on a Intel Pentium III Linux system

Results for an Alpha 21164 based computer

This[8] is a DEC Alpha 4100-4/466 , 4 CPU 21164/466, 8K instruction cache, 8K data cache, 96KB secondary cache per CPU, 4MB on board cache per CPU, 4GB RAM, Tru64 UNIX and the compiler gcc 2.95.2 with the switches `-O3 -ffast-math`. The system was only very lightly loaded by other processes. The results are shown in Figure 11.5.

- The cache is 4MB and thus if $nx = 707$ a block of $nx \cdot nx$ numbers will fill the cache memory completely. The drop of in performance as nx crosses this limit is visible for all implementations used.
- Implementation B gives the best performance, followed by C.
- Implementation C was modified slightly, leading to (C2). The inner most loop was using references `Rp[kj][i]` (as in implementation B) instead of `R[kj+i]`. The results are identical to B. This indicates that on this computer the array referencing for two dimensional arrays is best left to the compiler.

These results point towards implementation B or C2 as being the most efficient.

Results for an Alpha 21264 based computer

This[9] is a DEC Alpha ES40 server. It has 4 Alpha 21264 EV67 CPUs clocked at 667MHz with 8MB L2 cache. As operation system Linux RedHat 6.2 is used. Here we used two compilers: a Compaq compiler (`cxx -lm -arch ev67 -O3`) and the GNU complier (`g++ -O3 -Wall -ffast-math`). Implementation C was tested, but B leads to almost identical results, shown in Figure 11.6. It has to be noted that this system was heavily loaded during the test by other processes, thus the results on a lightly loaded system might improve, since more cache is available for one process.

- The cache is 8MB and thus if $nx = 1000$ a block of $nx \cdot nx$ numbers will fill the cache memory completely. The drop of in performance as nx crosses this limit is visible for all compilers used.

[8] provided by the Department of Mathematics, University of Utah
[9] provided by Compaq at `http://www.testdrive.compaq.com`

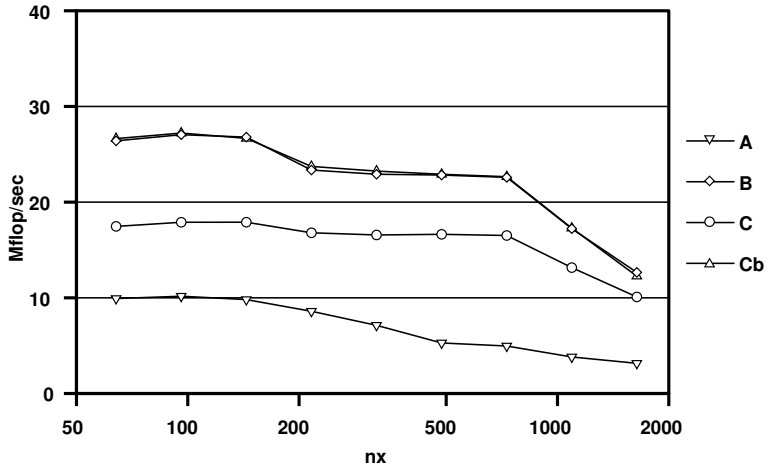


Figure 11.5: Performance of Cholesky algorithm on a DEC Alpha 21164 OFS1 Unix system

- The Compaq compiler leads to the best performance and illustrates that a good optimizing compiler is clearly desirable.

The performance of this hardware with the optimizing compiler is outstanding.

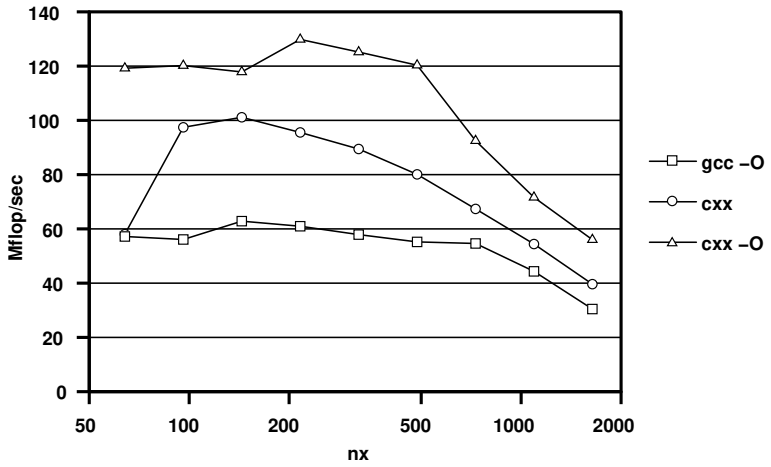


Figure 11.6: Performance of Cholesky algorithm on a Alpha 21264 Linux system

11.4 The algorithm of Cuthill and McKee to reduce bandwidth

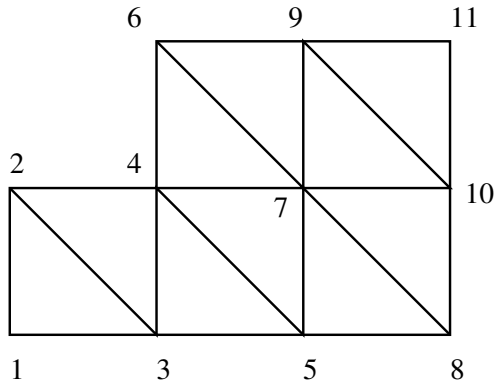
The numbering of the nodes of a mesh created on a given domain will determine the bandwidth of the resulting matrix **A** for the given differential equation to be solved by the FEM. For linear elements on triangles each node leads to one degree of freedom, the value of the function at this node. We find $a_{i,j} \neq 0$ if the nodes with number i and j share a common triangle. In view of the result in section 11.3 we should aim for a numbering leading to a small bandwidth. One possible (and rather efficient) algorithm is known as the Cuthill–McKee algorithm.

There are different criterions on how to choose an optimal first node. Tests show that nodes with few neighbors are often good stating nodes. Thus one may choose nodes with the minimal number of neighbors.

```
choose a starting node and give it the number 1
while there are unnumbered nodes
  pick the next node
  find all its neighbors not yet numbered
  sort them, using the number of connections to
             unnumbered nodes as sorting criterion
  give them the next free numbers
endwhile
```

Table 11.6: Algorithm of Cuthill–McKee

Also good candidates are nodes at extremal points of the discretized domain. A more detailed description of the Cuthill-McKee algorithm and how to choose starting points is given in [LascTheo87].



	1	2	3	4	5	6	7	8	9	10	11
1	*	*	*								
2	*	*	*	*							
3	*	*	*	*	*						
4		*	*	*	*	*	*				
5			*	*	*		*	*			
6				*		*	*		*		
7				*	*	*	*	*	*	*	
8					*		*	*		*	
9						*	*		*	*	*
10							*	*	*	*	*
11									*	*	*

Figure 11.7: Numbering of a simple mesh by Cuthill–McKee

The algorithm is illustrated by numbering the simple mesh in Figure 11.7. On the right the structure of the nonzero elements in the resulting stiffness matrix is shown. The band structure is clearly recognizable.

- The first node is chosen, since it has only two neighbors and is at one end of the domain.
- Node 1 has two neighbors, number 2 is given to the node above, since it has only one free neighbor. The node on the right (two free neighbors) of 1 will be number 3 .
- Node 2 has only one free node with number 4 .
- Node 3 now has also only one free node left, number 5 .
- Of the two free neighbors of node 4, the one above has fewer free nodes and thus will receive number 6. The node on the right will be number 7 .
- The only free neighbor of node 5 will now receive number 8 .
- The only free neighbor of node 6 will now receive number 9 .
- The only free neighbor of node 7 will now receive number 10 .
- The last node will be number 11 .

On the authors web site [www:sha] a possible implementation of the above algorithm is done in C++. The code will read output of the mesh generating code `triangle` [www:triangle] and renumber the nodes to aim for a small bandwidth of the resulting matrix. As an example consider the domain in Figure 11.8, whose mesh was generated by `triangle`. The mesh has 518 nodes and the original numbering leads to a semi–bandwidth of 515, i.e. no band structure. Nonetheless we have a **sparse** matrix, since only 3368 entries are nonzero (i.e. 1.25%). The nonzero elements in the matrix **A** are shown in Figure 11.9, before and after applying the Cuthill–McKee algorithm. The new semibandwidth is 28. If finer meshes (more nodes) are used then the improvements due to a good renumbering of the nodes will be even larger.

Within the band only 21% of the entries are not zero, i.e. we still have a certain sparsity within the band. The algorithm of Cholesky can not take advantage of this stucture, but iterative methods can, see section 11.7.

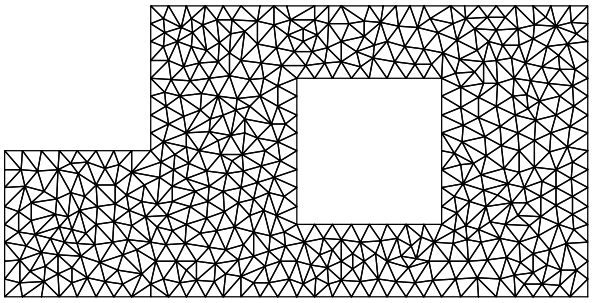


Figure 11.8: Mesh generated by `triangle`

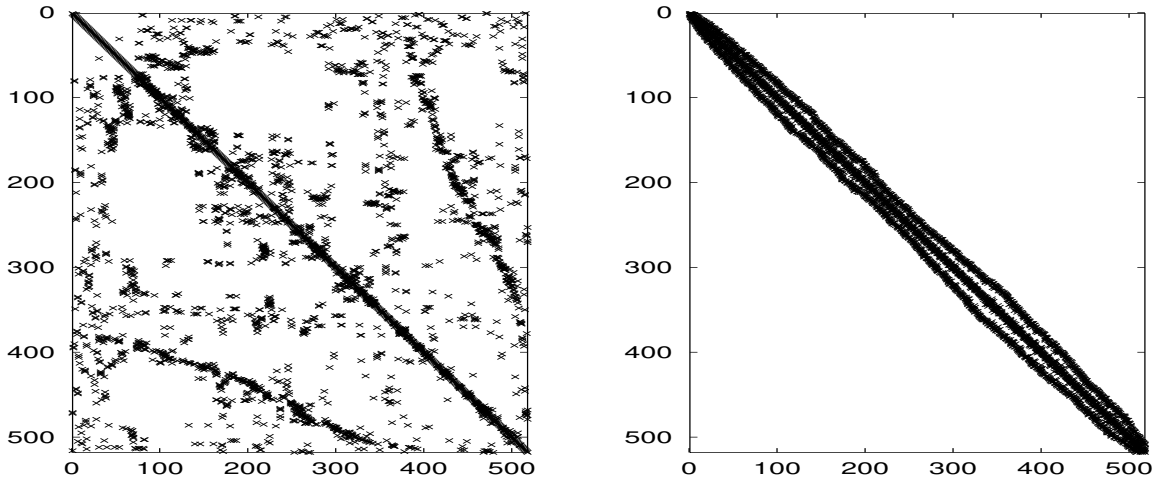


Figure 11.9: Original numbering and after renumbering by Cuthill–McKee

11.5 Eigenvalues and eigenvectors

Eigenvalues and vectors of symmetric, real matrices have a few very useful properties that serve as a foundation for computational methods. For sake of completeness we list some of the results. The main goal of this section is to develop an algorithm to determine a few of the smallest or largest eigenvalues. These contain often useful information about the problem to be examined. As an example take the vibrating beam problem in section 4.8 where the eigenvalues are closely related to the resonance frequencies of the beam.

11.5.1 Basic facts on eigenvalues of symmetric matrices

11–24 Result : *Let* $\mathbf{A}$ *be a symmetric, real* $n \times n$ *matrix.*

- *The eigenvalues* λ_i *are real.*
- *if* $\lambda_i \neq \lambda_j$ *then* $\langle \vec{e}_i \, , \, \vec{e}_j \rangle = 0$*, i.e. the eigenvectors of different eigenvalues are orthogonal.*

$\diamond$

Proof :

- The eigenvalues are zeros of the characteristic polynomial $\det(\mathbf{A} - \lambda \, \mathbb{I})$. Thus if λ is a zero of this real polynomial then $\bar{\lambda}$ is also a zero. Since $\mathbf{A} = \bar{\mathbf{A}}$ we know that $\mathbf{A} \, \vec{e} = \lambda \, \vec{e}$ implies $\mathbf{A} \, \bar{\vec{e}} = \bar{\lambda} \, \bar{\vec{e}}$. Now the calculation

$$\lambda \, \langle \vec{e}, \, \vec{e} \rangle = \langle \vec{e}, \, \lambda \, \vec{e} \rangle = \langle \vec{e}, \, \mathbf{A} \, \vec{e} \rangle = \langle \mathbf{A} \, \vec{e}, \, \vec{e} \rangle = \langle \lambda \, \vec{e}, \, \vec{e} \rangle = \bar{\lambda} \, \langle \vec{e}, \, \vec{e} \rangle$$

  shows that $\lambda = \bar{\lambda}$, i.e. the eigenvalues are real.

- Since $\lambda_i - \lambda_j \neq 0$ the computation

$$\begin{aligned} (\lambda_i - \lambda_j) \, \langle \vec{e}_i \, , \, \vec{e}_j \rangle &= \lambda_i \, \langle \vec{e}_i \, , \, \vec{e}_j \rangle - \lambda_j \, \langle \vec{e}_i \, , \, \vec{e}_j \rangle = \langle \lambda_i \, \vec{e}_i \, , \, \vec{e}_j \rangle - \langle \vec{e}_i \, , \, \lambda_j \, \vec{e}_j \rangle \\ &= \langle \mathbf{A} \, \vec{e}_i \, , \, \vec{e}_j \rangle - \langle \vec{e}_i \, , \, \mathbf{A} \, \vec{e}_j \rangle = \langle \mathbf{A} \, \vec{e}_i \, , \, \vec{e}_j \rangle - \langle \mathbf{A} \, \vec{e}_i \, , \, \vec{e}_j \rangle = 0 \end{aligned}$$

  implies $\langle \vec{e}_i \, , \, \vec{e}_j \rangle = 0$.

$\square$

Using the result in section A.1.3 the above result can be sharpened. For multiple eigenvalues we find the same number of orthonormal eigenvectors. This leads to the factorization result

$$\mathbf{Q}^T \cdot \mathbf{A} \cdot \mathbf{Q} = \mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix}$$

or

$$\mathbf{A} = \mathbf{Q} \cdot \mathbf{\Lambda} \cdot \mathbf{Q}^T$$

The elements λ_i on the diagonal of $\mathbf{\Lambda}$ are **eigenvalues** of the symmetric matrix $\mathbf{A}$ and the column vectors of $\mathbf{Q}$ are normalized **eigenvectors** of $\mathbf{A}$. The eigenvectors are pairwise orthogonal, leading to the orthogonal matrix $\mathbf{Q}$, satisfying $\mathbf{Q}^{-1} = \mathbf{Q}^T$.

Another characterization of the eigenvalues can be based on the Rayleigh quotient

$$\rho(\vec{x}) = \frac{\langle \vec{x} \, , \, \mathbf{A} \, \vec{x} \rangle}{\langle \vec{x} \, , \, \vec{x} \rangle} \tag{11.3}$$

Assume that the eigenvalues $\lambda_1 \leq \lambda_2 \leq \lambda_3 \leq \ldots \leq \lambda_n$ are sorted. When looking for an extremum of the function $\langle \vec{x} \, , \, \mathbf{A} \, \vec{x} \rangle$, subject to the constraint $\|\vec{x}\| = 1$ use the Lagrange multiplier theorem and

$$\vec{\nabla} \langle \vec{x} \, , \, \mathbf{A} \, \vec{x} \rangle = 2 \, \mathbf{A} \, \vec{x} \quad \text{and} \quad \vec{\nabla} \langle \vec{x} \, , \, \vec{x} \rangle = 2 \, \vec{x}$$

to conclude that $\mathbf{A} \, \vec{x} = \lambda \, \vec{x}$ for some factor λ. Using $\langle \vec{x} \, , \, \mathbf{A} \, \vec{x} \rangle = \langle \vec{x} \, , \, \lambda \, \vec{x} \rangle = \lambda \, \|\vec{x}\|^2$ we conclude

$$\lambda_1 = \min_{\|\vec{x}\|=1} \langle \vec{x} \, , \, \mathbf{A} \, \vec{x} \rangle \quad \text{and} \quad \lambda_n = \max_{\|\vec{x}\|=1} \langle \vec{x} \, , \, \mathbf{A} \, \vec{x} \rangle$$

If we are looking for other eigenvalues we need a slight modification of this result. If $\vec{e}_1$ is an eigenvector to the first eigenvalue we know that the eigenvector to strictly larger eigenvalues are orthogonal to $\vec{e}_1$. This leads to a method to determine λ_2 by

$$\lambda_2 = \min\{\langle \vec{x}\,,\, \mathbf{A}\,\vec{x}\rangle \quad : \quad \|\vec{x}\,\| = 1 \quad \text{and} \quad \vec{x} \perp \vec{e}_1\}$$

This result can be extended in the obvious way. The other eigenvalues can also be characterized by looking at subspaces by the **Courant-Fischer Minimax Theorem**, see [GoluVanLoan96, Theorem 8.1.2] or [Axel94, Lemma 3.13].

$$\lambda_k = \max_{\dim S = n-k} \min_{\vec{x} \in S \setminus \{\vec{0}\}} \frac{\langle \vec{x}\,,\, \mathbf{A}\,\vec{x}\rangle}{\langle \vec{x}\,,\, \vec{x}\rangle}$$

11.5.2 Power iteration

For general matrices there is no closed formula to find the eigenvalues. Otherwise we would have a formula to determine zeros of a polynomial of high degree and it can be shown that there exists no such formula. Thus we must search for an iterative method to determine good approximations of the eigenvalues.

The goal is to find the largest eigenvalue of a symmetric, positive definite matrix $\mathbf{A}$. The method is based on a simple observation: if an eigenvector $\vec{e}_i$ is multiplied by the matrix $\mathbf{A}$ it is stretched by the factor λ_i. Thus the eigenvector $\vec{e}_n$ will be stretched by the largest factor. Now we repeatedly multiply an initial vector $\vec{x}_0$ by $\mathbf{A}$, hoping for the component in the direction of the largest eigenpair finally to dominate all other contributions. The basic algorithm is thus very simple

$$\text{choose arbitrary} \quad \vec{x}_0 \in \mathbb{R}^n \quad \text{then} \quad \vec{x}_k = \mathbf{A}\,\vec{x}_{k-1} \quad \text{for} \quad k = 1, 2, 3, \ldots$$

To analyze the behavior of the algorithm we use the eigenvector $\vec{e}_i$ as basis and write the initial vector as a linear combination of the eigenvectors.

$$\vec{x}_0 = \sum_{i=1}^{n} c_i\,\vec{e}_i = c_1\,\vec{e}_1 + c_2\,\vec{e}_2 + c_3\,\vec{e}_3 + \ldots + c_{n-1}\,\vec{e}_{n-1} + c_n\,\vec{e}_n$$

Then the effect of the iteration is easily traced

$$\begin{aligned}
\vec{x}_1 &= \mathbf{A}\,\vec{x}_0 = \sum_{i=1}^{n} c_i\,\mathbf{A}\,\vec{e}_i = \sum_{i=1}^{n} c_i\,\lambda_i\,\vec{e}_i \\
\vec{x}_2 &= \mathbf{A}\,\vec{x}_1 = \sum_{i=1}^{n} c_i\,\lambda_i^2\,\vec{e}_i \\
\vec{x}_k &= \mathbf{A}\,\vec{x}_{k-1} = \sum_{i=1}^{n} c_i\,\lambda_i^k\,\vec{e}_i = \lambda_n^k \sum_{i=1}^{n} c_i \left(\frac{\lambda_i}{\lambda_n}\right)^k \vec{e}_i \\
&= \lambda_n^k \left(c_n\,\vec{e}_n + c_{n-1}\left(\frac{\lambda_{n-1}}{\lambda_n}\right)^k \vec{e}_{n-1} + c_{n-2}\left(\frac{\lambda_{n-2}}{\lambda_n}\right)^k \vec{e}_{n-2} + \ldots + c_1\left(\frac{\lambda_1}{\lambda_n}\right)^k \vec{e}_1 \right)
\end{aligned}$$

If $\lambda_{n-1} < \lambda_n$ then $\left(\frac{\lambda_{n-1}}{\lambda_n}\right)^k$ converges to zero as $k \to \infty$ and thus

$$\vec{x}_n = \mathbf{A}^n\,\vec{x}_0 \to \lambda_n^k\,c_n\,\vec{e}_n \quad \text{as} \quad k \to \infty$$

And the relative error is of the order $\left(\frac{\lambda_{n-1}}{\lambda_n}\right)^k$. Thus he have a (hopefully good) approximation of the eigenvector $\vec{e}_n$ by normalizing $\vec{x}_k$. A good approximation of the eigenvalue can then be given by $\|\mathbf{A}\,\vec{x}_k\| \,/\, \|\vec{x}_k\|$ or by

$$\lambda^{(k)} = \langle \vec{x}_k\,,\, \mathbf{A}\,\vec{x}_k\rangle$$

Since the factor λ_n^k can become very large it is a good idea to normalize the vector $\vec{x}_k$ during each iteration (or at least every few iterations). This leads to the **power iteration method** to determine the largest eigenvalue.

```
choose initial vector x
for k=1,2,3,...
   x = A x
   lambda = norm(x)
   x = x/norm(x)
endfor
lambda = < x, A x >
```

The algorithm fails if $c_n = 0$ and it converges slowly if λ_{n-1} is very close to λ_n. If θ_k denotes the angle between $\vec{x}_k$ and $\vec{e}_n$ the one can verify that ([GoluVanLoan96, Theorem 8.2.1])

$$\begin{aligned} |\sin\theta_k| &\leq \tan\theta_0 \left|\frac{\lambda_{n-1}}{\lambda_n}\right|^k \\ |\lambda^{(k)} - \lambda_n| &\leq |\lambda_n - \lambda_1| \tan^2\theta_0 \left|\frac{\lambda_{n-1}}{\lambda_n}\right|^{2k} \end{aligned}$$

These are **à priori estimates**, i.e. we have an estimate for the error before actually performing the calculations. The estimate do not lead to a reliable bound on the error, since we can not determine the values of the coefficients c_i and thus can not control θ_0. To compute θ_0 we need c_n, which can be computed if we knew $\vec{e}_n$. But this is the vector to be determined by the algorithm.

When deriving the above algorithm and its error estimates we built on eigenvalues and eigenvectors, but to run the algorithm we **do not need** the eigenvalues and eigenvectors. As a result we obtain approximations for the largest eigenvalue and its eigenvector.

11.5.3 The Rayleigh quotient and an à posteriori estimate

If β and $\vec{x}$ (with $\|\vec{x}\| = 1$) are an approximate eigenpair of the matrix $\mathbf{A}$ then the **residual vector**

$$\vec{r} = \mathbf{A}\,\vec{x} - \beta\,\vec{x}$$

should be small. Using the diagonalization $\mathbf{A} = \mathbf{Q}\,\mathbf{D}\,\mathbf{Q}^T$ with an orthogonal matrix $\mathbf{Q}$ and $\mathbf{D} = \mathrm{diag}(\lambda_i) = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$ one finds

$$\vec{r} = \mathbf{Q}\,\mathbf{D}\,\mathbf{Q}^T\,\vec{x} - \beta\,\mathbf{Q}\,\mathbf{Q}^T\vec{x}$$

and thus

$$\mathbf{Q}^T\,\vec{r} = \mathbf{D}\,\mathbf{Q}^T\,\vec{x} - \beta\,\mathbf{Q}^T\vec{x} = \mathrm{diag}(\lambda_i - \beta)\;\mathbf{Q}^T\vec{x}$$

Now two cases may occur

- $\beta = \lambda_i$ for some eigenvalues λ_i. In this case we found an exact eigenvalue.
- $\beta \neq \lambda_i$ for all eigenvalues λ_i. Multiply the above equation with $(\mathrm{diag}(\lambda_i - \beta))^{-1}$

$$\begin{aligned} \mathbf{Q}^T\,\vec{x} &= (\mathrm{diag}(\lambda_i - \beta))^{-1}\;\mathbf{Q}^T\vec{r} \\ \|\mathbf{Q}^T\,\vec{x}\| &\leq \|\,(\mathrm{diag}(\lambda_i - \beta))^{-1}\,\|\;\|\mathbf{Q}^T\vec{r}\| \\ 1 &\leq \max_i(|\lambda_i - \beta|^{-1})\;\|\vec{r}\| \\ \min_i|\lambda_i - \beta| &\leq \|\vec{r}\| \end{aligned}$$

Thus the distance between the approximate eigenvalue β and the closest eigenvalue is at most $\|\vec{r}\|$.

This is an **à posteriori estimate**, i.e. we estimate the error of an already known result[10]. This bound can be used to assure accuracy of a computed solution.

The theorem below ([Demm97, Theorem 5.5]) shows that the error bound on the approximate eigenvalue β with approximate eigenvector $\vec{x}$ can be improved, using the Rayleigh quotient (11.3).

11–25 Theorem : *Let $\mathbf{A}$ be a symmetric matrix and $\vec{x}$ a normalized vector and β be a scalar. Then $\mathbf{A}$ has an eigenpair $\mathbf{A}\,\vec{v} = \alpha\,\vec{v}$ satisfying $|\alpha - \beta| \leq \|\mathbf{A}\,\vec{x} - \beta\,\vec{x}\|$. Given $\vec{x} \in \mathbb{R}^n$ the choice $\beta = \rho(\vec{x})$ minimizes $\|\mathbf{A}\,\vec{x} - \beta\,\vec{x}\|$. Thus*

$$\min_i |\lambda_i - \beta| \leq \|\vec{r}\| = \|\mathbf{A}\,\vec{x} - \beta\,\vec{x}\|$$

where λ_i are the exact eigenvalues of $\mathbf{A}$.

If we have more information on the eigenvalues then the estimate can be sharpened. Let $\vec{r} = \mathbf{A}\,\vec{x} - \rho(\vec{x})\,\vec{x}$ and let λ_i be the eigenvalue of $\mathbf{A}$ closest to $\rho(\vec{x})$. The expression

$$gap = \min_{j \neq i} |\lambda_j - \rho(\vec{x})|$$

measures the distance of $\rho(\vec{x})$ to the other eigenvalues of $\mathbf{A}$. Let θ be the acute angle between $\vec{x}$ and $\vec{v}$. Then

$$\sin\theta \leq \frac{\|\vec{r}\|}{gap} \quad \textit{and} \quad |\lambda_i - \rho(\vec{x})| \leq \frac{\|\vec{r}\|^2}{gap}$$

◇

This theorem shows that if we have an approximate eigenvalue β with normalized eigenvector $\vec{x}$ we can get a better approximation of the true eigenvalue λ_i by the Rayleigh quotient $\rho(\vec{x})$. The difference can be significant since the error of a Rayleigh approximation is proportional to the square of the norm of the residual vector, while the error of β is proportional to the norm. The result can be extended to systems of eigenpairs, see [Demm97, Theorem 7.1].

11.5.4 Inverse power iteration

With the power iteration method we could compute the largest eigenvalue λ_n of a symmetric, positive definite matrix. Observing that the smallest eigenvalue λ_1 of $\mathbf{A}$ becomes the largest eigenvalue $1/\lambda_1$ of the inverse matrix we can now give an algorithm to determine λ_1 and $\vec{e}_1$.

$$\text{choose an arbitrary} \quad \vec{x}_0 \in \mathbb{R}^n \quad \text{, then solve} \quad \mathbf{A}\,\vec{x}_k = \vec{x}_{k-1} \quad \text{for} \quad k = 1, 2, 3, \ldots$$

This algorithm is called **inverse power iteration**.

Then the effect of the iteration is very similar to a regular power iteration.

$$\begin{aligned}
\vec{x}_1 &= \mathbf{A}^{-1}\,\vec{x}_0 = \sum_{i=1}^{n} c_i\,\mathbf{A}^{-1}\,\vec{e}_i = \sum_{i=1}^{n} c_i\,\frac{1}{\lambda_i}\,\vec{e}_i \\
\vec{x}_k &= \mathbf{A}^{-1}\,\vec{x}_{k-1} = \sum_{i=1}^{n} c_i\,\frac{1}{\lambda_i^k}\,\vec{e}_i = \frac{1}{\lambda_1^k}\,\sum_{i=1}^{n} c_i\left(\frac{\lambda_1}{\lambda_i}\right)^k \vec{e}_i \\
&= \frac{1}{\lambda_1^k}\left(c_1\,\vec{e}_1 + c_2\left(\frac{\lambda_1}{\lambda_2}\right)^k \vec{e}_2 + c_3\left(\frac{\lambda_1}{\lambda_3}\right)^k \vec{e}_3 + \ldots + c_n\left(\frac{\lambda_1}{\lambda_n}\right)^k \vec{e}_n\right)
\end{aligned}$$

The rate of convergence is determined by the factor λ_1/λ_2 and the iteration converges if $\lambda_1 < \lambda_2$ (and $c_1 \neq 0$). There are à priori bounds similar to the power iteration method. If we use the Rayleigh quotient the we find the following basic algorithm

[10] In [GoluVanLoan96, p. 64] a remark by J. H. Wilkinson is quoted. It states different merits and goals of à priori and à posteriori estimates. A very brief summary is: à priori estimates help to understand the algorithm and à posteriori bounds help to control the errors of concrete problems.

```
choose initial vector x
for k=1,2,3,...
   solve A y  =  x
   x = y/norm(y)
endfor
lambda = < x, A x >
```

For the power iteration we have to compute $\mathbf{A}\,\vec{x}$ repeatedly, for the inverse power iteration we compute $\mathbf{A}^{-1}\,\vec{x}$, i.e. we have to solve a system of linear equations. This seems to be computationally more expensive. But observe that we are working with one matrix only. Thus we compute the Cholesky factorization $\mathbf{A} = \mathbf{R}^T \cdot \mathbf{D} \cdot \mathbf{R}$ once and then work with forward and backward substitutions. The computational effort for one iteration is now comparable to a matrix multiplication.

There are modifications of this basic algorithm to improve its performance. In particular one might work with $\mathbf{A} - \lambda\mathbb{I}$ for well chosen values of λ to accelerate convergence. One possible result is the **Rayleigh quotient iteration method**, e.g. [GoluVanLoan96, §8.2.3].

11.5.5 Inverse power iteration for subspaces

In the previous sections we found algorithms to determine the smallest and largest eigenvalue of a symmetric, positive definite matrix. In many applications a few of the smallest or largest eigenvalues are of interest. As an example consider the vibrating beam problem in section4.8. We present the a possible algorithm for the few smallest eigenvalues, using the basic idea of inverse power iteration.

Consider m different random initial vectors $\vec{v}_{1,0}$, $\vec{v}_{2,0}$, $\ldots\vec{v}_{m,0}$. The goal is to make them convege towards the eigenvectors belonging to the first m eigenvalues. If $m = 1$ we have the situation of inverse power iteration in section 11.5.4. Thus we know that under most circumstances the algorithm

Solve the system $\vec{v}_{1,k} = \mathbf{A}^{-1}\,\vec{v}_{1,k-1}$

Normalize $\vec{v}_{1,k} = \vec{v}_{1,k}/\|\vec{v}_{1,k}\|$

leads to a sequence of vectors with $\vec{v}_{1,k} \longrightarrow \vec{e}_1$. Since the eigenvector belonging to λ_2 has to be orthogonal to $\vec{e}_1$ we only search in this subspace. To achieve this we make sure that $\vec{v}_{2,k} \perp \vec{v}_{1,k}$ by the following algorithm

Solve the system $\vec{v}_{2,k} = \mathbf{A}^{-1}\,\vec{v}_{2,k-1}$

Subtract the component of $\vec{v}_{2,k}$ in direction of $\vec{v}_{1,k}$ from $\vec{v}_{2,k}$. To do so set $\vec{v}_{2,k} = \vec{v}_{2,k} - \langle\vec{v}_{2,k}\,,\,\vec{v}_{1,k}\rangle\,\vec{v}_{1,k}$ to assure[11] $\vec{v}_{2,k} \perp \vec{v}_{1,k}$.

Normalize $\vec{v}_{2,k} = \vec{v}_{2,k}/\|\vec{v}_{2,k}\|$

This leads to a sequence of normalized vectors $\vec{v}_{2,k}$, orthogonal to $\vec{v}_{1,k}$ and the component in the direction of $\vec{e}_2$ is multiplied by the largest factor by the inverse power iteration. Thus we expect $\vec{v}_{2,k} \longrightarrow \vec{e}_2$. Now we have approximations for two eigenvalues and eigenvectors. For more eigenvectors we proceed similarly:

Solve the system $\vec{v}_{j,k} = \mathbf{A}^{-1}\,\vec{v}_{j,k-1}$

Subtract the components of $\vec{v}_{j,k}$ in directions of $\vec{v}_{1,k}$, $\vec{v}_{1,k}$, $\ldots$ $\vec{v}_{j-1,k}$, from $\vec{v}_{j,k}$.

Normalize $\vec{v}_{j,k} = \vec{v}_{j,k}/\|\vec{v}_{j,k}\|$

[11] Use $\langle\vec{v}_{1,k}\,,\,\vec{v}_{2,k} - \langle\vec{v}_{2,k}\,,\,\vec{v}_{1,k}\rangle\,\vec{v}_{1,k}\rangle = \langle\vec{v}_{1,k}\,,\,\vec{v}_{2,k}\rangle - \langle\vec{v}_{2,k}\,,\,\vec{v}_{1,k}\rangle\,\langle\vec{v}_{1,k}\,,\,\vec{v}_{1,k}\rangle = 0$

One can show that the vectors $\vec{v}_{j,k}$ $(1 \leq j \leq m)$ converge in fact towards $\vec{e}_j$, unless one has poorly chosen initial vectors, e.g. exercise 11–8. The rate of convergence is $(\lambda_m/\lambda_{m+1})^k$. With these approximation we can then use

$$\lambda_j = \langle \vec{e}_j \, , \, \mathbf{A}\, \vec{e}_j \rangle \approx \langle \vec{v}_{j,k} \, , \, \mathbf{A}\, \vec{v}_{j,k} \rangle$$

to determine the eigenvalues.

A concrete implementation of the above idea should use the modified **Gram-Schmidt** algorithm to ortho-normalize the vectors. Consider an $n \times m$ matrix $\mathbf{V}$ with $m \leq n$. The goal is to ortho-normalize the columns $V(:, j)$ of this matrix.

```
for j=1:m
  factor = norm( V(:,j) )
  V(:,j) = V(:,j) /factor                  %% normalize column j
  for k=j+1:m
     factor = < V(:,j) , V(:,k) >
     V(:,k) =  V(:,k) - factor* V(:,j)     %% subtract multiples of columns
  endfor
endfor
```

A precise description of this algorithm can be found in [GoluVanLoan96, §5.2.8] or [Axel94, p. 71–72].

Now we can formulate an algorithm to determine the m smallest eigenvalues of a symmetric, positive definite matrix $\mathbf{A}$

- Create an $n \times m$ matrix $\mathbf{V}_0$ with the initial vectors $\vec{v}_{j,0}$ as its columns.
- repeat until desired precision is reached
  - solve the matrix equation $\mathbf{A} \cdot \mathbf{V}_k = \mathbf{V}_{k-1}$ or $\mathbf{V}_k = \mathbf{A}^{-1} \cdot \mathbf{V}_{k-1}$
  - ortho-normalize the columns of $\mathbf{V}_k$, using Gram-Schmidt
- for $j = 1, 2 \ldots m$ compute $\beta_j = \langle \mathbf{V}(:, j) \, , \, \mathbf{A} \cdot \mathbf{V}(:, j) \rangle$

The resulting values β_j should be close to the eigenvalues λ_j. There is an à priori error estimate of the form $(\lambda_m/\lambda_{m+1})^{2k}$, unless the corresponding component of the initial vector happens to vanish exactly. The à posteriori estimates of the above sections can be used to control the errors.

The above algorithm is a variation of the **QR** iteration . It should only be used if the system of linear equations can be solved efficiently. This is certainly the case if $\mathbf{A}$ is a matrix with a narrow band and its Cholesky factorization is already known. If one needs all (or many) of the eigenvalues then there are considerably better algorithms to be used.

11.6 The generalized eigenvalue problem

For many applications it is easier to consider a **generalized eigenvalue** problem

$$\mathbf{A}\, \vec{e} = \lambda\, \mathbf{B}\, \vec{e}$$

instead of the corresponding standard problem $\mathbf{B}^{-1} \cdot \mathbf{A}\, \vec{x} = \lambda \vec{x}$. This allows to keep more of the structures of the matrices $\mathbf{A}$ and $\mathbf{B}$[12]. Even if both matrices are symmetric, their product generally is not. Even if $\mathbf{B}$ has a band structure its inverse might be a full matrix. In many applications $\mathbf{B}$ is a diagonal matrix, e.g. the problem in section 4.8.7 where frequencies of a vibrating beam are to be determined.

[12] Another approach would be to use the Cholesky factorization $\mathbf{B} = \mathbf{R}^T \cdot \mathbf{R}$ of $\mathbf{B}$. The symmetric matrix $\mathbf{C} = (\mathbf{R}^T)^{-1} \cdot \mathbf{A} \cdot \mathbf{R}^{-1}$ has the same eigenvalues as $\mathbf{A}$ and the multiplication by $\mathbf{R}^{-1}$ may be implemented using backsubstitution

We consider the case of a symmetric, positive definite $n \times n$ matrix $\mathbf{B}$ and a symmetric, positive semidefinite $n \times n$ matrix $\mathbf{A}$. This implies that the equation $\mathbf{A}\,\vec{e} = \lambda\,\mathbf{B}\,\vec{e}$ has exactly n real solutions λ_i. Since $\mathbf{B}$ is invertible we have n solutions of $\mathbf{B}^{-1}\,\mathbf{A}\,\vec{e} = \lambda\,\vec{e}$ and based on the calculation

$$\lambda\,\langle \vec{e},\, \mathbf{B}\,\vec{e}\rangle = \langle \vec{e},\, \lambda\,\mathbf{B}\,\vec{e}\rangle = \langle \vec{e},\, \mathbf{A}\,\vec{e}\rangle = \langle \mathbf{A}\,\vec{e},\, \vec{e}\rangle = \langle \lambda\,\mathbf{B}\,\vec{e},\, \vec{e}\rangle = \bar{\lambda}\,\langle \vec{e},\, \mathbf{B}\,\vec{e}\rangle$$

we conclude that the eigenvalues are real.

The smallest (or largest) generalized eigenvalue can be characterized by an extremal problem. Using Lagrange multipliers on verifies that λ_1 is given as the solution of

$$\text{minimzie} \quad \langle \vec{x}\,,\, \mathbf{A}\,\vec{x}\rangle \quad \text{subject to the constraint} \quad \langle \vec{x}\,,\, \mathbf{B}\,\vec{x}\rangle = 1$$

As in the case of the standard eigenvalue problem we find the eigenvectors to be pairwise orthogonal, but with respect to the generalized scalar product

$$\langle \vec{x}\,,\, \vec{x}\rangle_B = \langle \vec{x}\,,\, \mathbf{B}\,\vec{x}\rangle$$

To verify this use

$$\begin{aligned}(\lambda_i - \lambda_j)\,\langle \vec{e}_i\,,\, \vec{e}_j\rangle_B &= \lambda_i\,\langle \vec{e}_i\,,\, \vec{e}_j\rangle_B - \lambda_j\,\langle \vec{e}_i\,,\, \vec{e}_j\rangle_B = \langle \lambda_i\,\mathbf{B}\,\vec{e}_i\,,\, \vec{e}_j\rangle - \langle \vec{e}_i\,,\, \lambda_j\,\mathbf{B}\,\vec{e}_j\rangle \\ &= \langle \mathbf{A}\,\vec{e}_i\,,\, \vec{e}_j\rangle - \langle \vec{e}_i\,,\, \mathbf{A}\,\vec{e}_j\rangle = \langle \mathbf{A}\,\vec{e}_i\,,\, \vec{e}_j\rangle - \langle \mathbf{A}\,\vec{e}_i\,,\, \vec{e}_j\rangle = 0\end{aligned}$$

to conclude that $\langle \vec{e}_i\,,\, \vec{e}_j\rangle_B = 0$. The result show that there exists a complete set of eigenpairs with orthonormal eigenvectors in the above sense.

11–26 Result : *$\mathbf{A}$ and $\mathbf{B}$ can be diagonalised simultaneously, i.e. there exists a nonsingular matrix $\mathbf{X} = [\vec{x}_1\ \vec{x}_2, \ldots, \vec{x}_n]$ such that*

$$\mathbf{X}^T \cdot \mathbf{A} \cdot \mathbf{X} = \operatorname{diag}(\lambda_1,\, \lambda_2, \ldots, \lambda_n) \quad \textit{and} \quad \mathbf{X}^T \cdot \mathbf{B} \cdot \mathbf{X} = \mathbb{I}_n$$

Moreover $\mathbf{A}\,\vec{x}_i = \lambda_i\,\mathbf{B}\,\vec{x}_i$. ◇

Proof : [GoluVanLoan96, Theorem 8.7.1]
Since $\mathbf{B}$ is positive definite we obtain an eigenvalue factorization

$$\mathbf{Q}^T \cdot \mathbf{B} \cdot \mathbf{Q} = \operatorname{diag}(b_i)$$

with $b_i > 0$. Thus we can compute square roots and set

$$\mathbf{X}_1 = \mathbf{Q} \cdot \operatorname{diag}(1/\sqrt{b_i}) \quad \text{and} \quad \mathbf{A_1} = \mathbf{X}_1^T \cdot \mathbf{A} \cdot \mathbf{X}_1$$

This matrix $\mathbf{A}_1$ is symmetric and thus there exists a factorization

$$\mathbf{Q_1}^T \cdot \mathbf{A_1} \cdot \mathbf{Q_1} = \operatorname{diag}(\lambda_1)$$

Now the choice

$$\mathbf{X} = \mathbf{X}_1 \cdot \mathbf{Q}_1 = \mathbf{Q} \cdot \operatorname{diag}(1/\sqrt{b_i}) \cdot \mathbf{Q}_1$$

leads to

$$\begin{aligned}\mathbf{X}^T \cdot \mathbf{B} \cdot \mathbf{X} &= \mathbf{Q}_1^T \cdot \mathbf{X}_1^T \cdot \mathbf{B} \cdot \mathbf{X}_1 \cdot \mathbf{Q}_1 \\ &= \mathbf{Q}_1^T \cdot \operatorname{diag}(1/\sqrt{b_i}) \cdot \mathbf{Q}^T \cdot \mathbf{B} \cdot \mathbf{Q} \cdot \operatorname{diag}(1/\sqrt{b_i}) \cdot \mathbf{Q}_1 \\ &= \mathbf{Q}_1^T \cdot \operatorname{diag}(1/\sqrt{b_i}) \cdot \operatorname{diag}(b_i) \cdot \operatorname{diag}(1/\sqrt{b_i}) \cdot \mathbf{Q}_1 = \mathbb{I}_n\end{aligned}$$

and

$$\mathbf{X}^T \cdot \mathbf{A} \cdot \mathbf{X} = \mathbf{Q}_1^T \cdot \mathbf{X}_1^T \cdot \mathbf{A} \cdot \mathbf{X}_1 \cdot \mathbf{Q}_1 = \mathbf{Q}_1^T \cdot \mathbf{A}_1 \cdot \mathbf{Q}_1 = \operatorname{diag}(\lambda_i)$$

Juggling with the above equations we obtain

$$\begin{aligned}\mathbf{B} \cdot \mathbf{X} \cdot \operatorname{diag}(\lambda_i) &= \mathbf{B} \cdot (\mathbf{X}_1 \cdot \mathbf{Q}_1) \cdot (\mathbf{Q_1}^T \cdot \mathbf{A_1} \cdot \mathbf{Q_1}) = \mathbf{B} \cdot \mathbf{X}_1 \cdot \mathbf{A_1} \cdot \mathbf{Q_1} \\ &= \mathbf{B} \cdot \mathbf{X}_1 \cdot \mathbf{X}_1^T \cdot \mathbf{A} \cdot \mathbf{X}_1 \cdot \mathbf{Q_1} \\ &= \mathbf{B} \cdot (\mathbf{Q} \cdot \operatorname{diag}(1/\sqrt{b_i})) \cdot (\operatorname{diag}(1/\sqrt{b_i}) \cdot \mathbf{Q}^T) \cdot \mathbf{A} \cdot \mathbf{X} = \mathbf{A} \cdot \mathbf{X}\end{aligned}$$

and thus conclude $\mathbf{A}\,\vec{x}_i = \lambda_i\,\mathbf{B}\,\vec{x}_i$. □

Inverse power iteration applies again but now ortho-normalized with respect to the generalized scalar product and the equation to be solved is modified. The à priori estimates are comparable to the standard inverse power iteration. The algorithm below generates the m smallest eigenvalues.

- Create an $n \times m$ matrix $\mathbf{V}_0$ with the initial vectors $\vec{v}_{j,0}$ as its columns.
- repeat until desired precision is reached
  - solve the matrix equation $\mathbf{A} \cdot \mathbf{V}_k = \mathbf{B} \cdot \mathbf{V}_{k-1}$ or $\mathbf{V}_k = \mathbf{A}^{-1} \cdot \mathbf{B} \cdot \mathbf{V}_{k-1}$
  - ortho-normalize the columns of $\mathbf{V}_k$, using the generalized Gram-Schmidt algorithm shown below.
- for $j = 1, 2 \ldots m$ compute $\beta_j = \langle \mathbf{V}(:,j)\,,\ \mathbf{A} \cdot \mathbf{V}(:,j) \rangle$. Then β_j should be good approximations to the eigenvalues.

Since the generalized eigenvalues are orthogonal with respect to the modified scalar product we have to modify the Gram-Schmidt algorithm to assure $\langle \vec{e}_i\,,\ \mathbf{B}\,\vec{e}_j \rangle = \delta_{i,j}$.

```
for j=1:m
  factor = sqrt( < V(:,j), B*V(:,j)> )
  V(:,j) = V(:,j) /factor                  %% normalize column j
  for k=j+1:m
     factor = < V(:,j) , B*V(:,k) >
     V(:,k) =  V(:,k) - factor* V(:,j)     %% subtract multiples of columns
  endfor
endfor
```

It remains to find an à posteriori estimate and to verify that the Rayleigh quotient $\beta_j = \langle \mathbf{V}(:,j)\,,\ \mathbf{A} \cdot \mathbf{V}(:,j) \rangle$ leads to the best possible solution. To achieve this we use steps similar to section 11.5.3 and we use the notation of the proof of result 11–26 with

$$\mathbf{Y} = \mathbf{X}^{-1} = \mathbf{Q}_1^T \cdot \operatorname{diag}(\sqrt{b_i}) \cdot \mathbf{Q}^T$$

Now we compute

$$\begin{aligned}
\vec{r} &= \mathbf{A} \cdot \vec{x} - \beta\,\mathbf{B}\,\vec{x} = \mathbf{Y}^T \cdot \operatorname{diag}(\lambda_i) \cdot \mathbf{Y}\,\vec{x} - \beta\,\mathbf{Y}^T \cdot \mathbf{Y}\,\vec{x} \\
&= \mathbf{Y}^T \cdot \operatorname{diag}(\lambda_i - \beta) \cdot \mathbf{Y}\,\vec{x} \\
\mathbf{X}^T \cdot \vec{r} &= \operatorname{diag}(\lambda_i - \beta) \cdot \mathbf{Y}\,\vec{x} \\
\mathbf{Y}\,\vec{x} &= \operatorname{diag}(\lambda_i - \beta)^{-1} \cdot \mathbf{X}^T\,\vec{r} \\
\mathbf{Q}_1^T \cdot \operatorname{diag}(\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{x} &= \operatorname{diag}(\lambda_i - \beta)^{-1} \cdot \mathbf{Q}_1^T \cdot \operatorname{diag}(1/\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{r} \\
\|\mathbf{Q}_1^T \cdot \operatorname{diag}(\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{x}\| &\leq \max(|\lambda_i - \beta|^{-1})\ \|\mathbf{Q}_1^T \cdot \operatorname{diag}(1/\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{r}\| \\
\min(|\lambda_i - \beta|) &\leq \|\operatorname{diag}(1/\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{r}\| / \|\operatorname{diag}(\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{x}\|
\end{aligned}$$

For the denominator we find

$$\begin{aligned}
\|\operatorname{diag}(\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{x}\|^2 &= \langle \operatorname{diag}(\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{r},\ \operatorname{diag}(\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{r} \rangle \\
&= \langle \vec{r},\ \mathbf{Q} \cdot \operatorname{diag}(b_i) \cdot \mathbf{Q}^T\,\vec{r} \rangle = \langle \vec{x},\ \mathbf{B}\,\vec{x} \rangle
\end{aligned}$$

For a given $\vec{x}$ we want to find β such that the numerator $\|\operatorname{diag}(1/\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{r}\|$ is minimal, i.e. minimize the expression

$$\begin{aligned}
\|\operatorname{diag}(1/\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{r}\|^2 &= \langle \operatorname{diag}(1/\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{r},\ \operatorname{diag}(1/\sqrt{b_i}) \cdot \mathbf{Q}^T\,\vec{r} \rangle \\
&= \langle \vec{r},\ \mathbf{Q} \cdot \operatorname{diag}(1/b_i) \cdot \mathbf{Q}^T\,\vec{r} \rangle = \langle \vec{r},\ \mathbf{B}^{-1}\,\vec{r} \rangle
\end{aligned}$$

To minimize the error of the eigenvalue we find the necessary condition

$$\begin{aligned} 0 &= \frac{d}{d\beta}\,\langle \vec{r}\,,\, \mathbf{B^{-1}}\,\vec{r}\rangle = \frac{d}{d\beta}\,\langle \mathbf{A}\,\vec{x} - \beta\,\mathbf{B}\,\vec{x}\,,\, \mathbf{B^{-1}}\,(\mathbf{A}\,\vec{x} - \beta\,\mathbf{B})\,\vec{x}\rangle \\ &= -\langle \mathbf{B}\,\vec{x}\,,\, \mathbf{B}^{-1}\cdot\mathbf{A}\,\vec{x} - \beta\,\vec{x}\rangle - \langle \mathbf{A}\,\vec{x} - \beta\,\mathbf{B}\,\vec{x}\,,\, \vec{x}\rangle \\ &= -\langle \vec{x}\,,\, \mathbf{A}\,\vec{x} - \beta\,\mathbf{B}\,\vec{x}\rangle - \langle \mathbf{A}\,\vec{x} - \beta\,\mathbf{B}\,\vec{x}\,,\, \vec{x}\rangle \\ &= -2\,\langle \vec{x}\,,\, \mathbf{A}\,\vec{x}\rangle + 2\,\beta\,\langle \vec{x}\,,\, \mathbf{B}\,\vec{x}\rangle \end{aligned}$$

This equation can be solved for the optimal value of β .

$$\beta = \frac{\langle \vec{x}\,,\, \mathbf{A}\,\vec{x}\rangle}{\langle \vec{x}\,,\, \mathbf{B}\,\vec{x}\rangle}$$

Since the generalized Gram-Schmidt step assures $\langle \vec{x}\,,\, \mathbf{B}\,\vec{x}\rangle = 1$ we find the optimal solution with $\beta = \langle \vec{x}\,,\, \mathbf{A}\,\vec{x}\rangle$ and we have the à posteriori error bound

$$\min(|\lambda_i - \beta|) \le \frac{\|\operatorname{diag}(1/\sqrt{b_i})\cdot\mathbf{Q}^T\,\vec{r}\|}{\|\operatorname{diag}(\sqrt{b_i})\cdot\mathbf{Q}^T\,\vec{x}\|} = \sqrt{\langle \vec{r}\,,\, \mathbf{B}^{-1}\,\vec{r}\rangle}$$

This is a generalization of the first estimate in result 11–25. An alternative proof is given in exercise 11–11.

11.7 Iterative methods

The finite element method leads to linear systems of the form $\mathbf{A}\,\vec{x} + \vec{b} = \vec{0}$, where only very few entries of the large matrix $\mathbf{A}$ are different from zero, i.e. we have a **sparse matrix**. The Cholesky algorithm for banded matrices is using only some of this sparsity. Due to the sparsity the computational effort to compute a matrix product $\mathbf{A}\,\vec{x}$ is minimal, compared to the number of operations to solve the above system with a direct method. One is lead to search for an algorithm to solve the linear system, using matrix multiplications only. This leads to **iterative methods**. The previously considered algorithms of Gauss and Cholesky are both **direct methods**, since both methods will lead to the solution of the linear system using a known, finite number of operations.

11.7.1 Basic definitions

For an invertible $N \times N$–matrix $\mathbf{A}$ and a vector $\vec{b}$ we have the exact solution $\vec{x}$ of $\mathbf{A}\,\vec{x} + \vec{b} = \vec{0}$. For a mapping $\Phi : \mathbb{R}^N \to \mathbb{R}^N$ we choose an initial vector $\vec{x}_0$ and then compute $\vec{x}_1 = \Phi(\vec{x}_0)$, $\vec{x}_2 = \Phi(\vec{x}_1)$ or

$$\vec{x}_k = \Phi^k(\vec{x}_0)$$

Φ is called an **iterative method** with **linear convergence factor** $q < 1$ if the error after k steps is bounded by

$$\|\vec{x}_k - \vec{x}\| \le c\,q^k$$

If we wish to improve the accuraccy of the inital guess $\vec{x}_0$ by D digits we need $q^k \le 10^{-D}$. This is satisfied if

$$\begin{aligned} k\,\log q &\le -D \\ k &\ge \frac{-D}{\log q} = \frac{-D\,\ln 10}{\ln q} > 0 \end{aligned}$$

For most applications the factor $q < 1$ will be very close to 1. Thus we write $q = 1 - q_1$ and use the Taylor approximation $\ln q = \ln(1 - q_1) \approx -q_1$. Then the above computations leads to an estimate for the number of iterations necessary to decrease the error by D digits.

$$k \ge \frac{D\,\ln 10}{q_1} \tag{11.4}$$

This implies that the numbers of desired correct digits is proportional to the number of required iterations and inversely proportional to the deviation q_1 of the factor $q = 1 - q_1$ from 1.

11.7.2 A model problem

As a model problem we use the example in section 7.3 (page 154)

$$\begin{aligned} u_{xx} + u_{yy} &= f(x,y) \qquad \text{for} \quad (x,y) \in \Omega = (0,1) \times (0,1) \\ u(x,y) &= 0 \qquad\quad\ \ \text{for} \quad (x,y) \text{ on boundary } \partial\Omega \end{aligned}$$

and a regular grid with $n+2$ points on each edge. Thus we will have n^2 interior nodes and the resulting matrix $\mathbf{A}$ will be of size $N = n^2 \times n^2$ with a semi-bandwidth of $n+1 \approx n$. In each row/column of the matrix only 5 entries are nonzero. The size of the triangles is given by $h = \frac{1}{n+1} \approx \frac{1}{n}$. The eigenvalues of this matrix are

$$\lambda_{i,j} = \frac{2}{h^2}\ (2 - \cos(i\,\pi\,h) - \cos(j\,\pi\,h)) = \frac{4}{h^2}\ \left(\sin^2(i\,\pi\,h/2) + \sin^2(j\,\pi\,h/2)\right) \quad \text{for} \quad 1 \le i,j \le n$$

This implies $\lambda_{min} = \lambda_{1,1} \approx 2\,\pi^2$ and $\lambda_{max} = \lambda_{n,n} \approx \frac{8}{h^2} \approx 8\,n^2$ and thus we find the condition number

$$\kappa = \frac{\lambda_{max}}{\lambda_{min}} \approx \frac{4}{\pi^2}\ n^2$$

When using a banded Cholesky algorithm to solve $\mathbf{A}\,\vec{x} + \vec{b} = \vec{0}$ we need

- storage for $n \cdot n^2 = n^3$ numbers
- approximately $\frac{1}{2}\,n^2\,n^2 = \frac{1}{2}\,n^4$ floating point instructions

An iterative method will have to do better than this to be considered useful. To multiply the matrix $\mathbf{A}$ with a vector we need about $5\,n^2$ multiplications.

For the similar three dimensional problem we find a matrix $\mathbf{A}$ of size $N = n^3$ and each row has approximately $nz = 7$ nonzero entries. The semi-bandwidth of the matrix is n^2. Thus the banded Cholesky solver requires approximately $\frac{1}{2}\ n^3 \cdot n^4$ floating point operations. The condition number is identical to the 2-D situation.

11.7.3 Steepest descent iteration

For a symmetric, positive definite matrix $\mathbf{A}$ the solution of the linear system $\mathbf{A}\,\vec{x} + \vec{b} = \vec{0}$ is given by the location of the minimum of the function

$$f(\vec{x}) = \frac{1}{2}\ \langle \vec{x}\,,\ \mathbf{A}\,\vec{x} \rangle + \langle \vec{x}\,,\ \vec{b} \rangle$$

A possible graph of such a function and its levels curves are shown in Figure 11.10. The gradient of this function is given by

$$\nabla f(\vec{x}) = \mathbf{A}\,\vec{x} + \vec{b}$$

A given point $\vec{x}_k$ is assumed to be a good approximation of the exact solution $\vec{x}$. The error is given by the **residual vector**

$$\vec{r}_k = \mathbf{A}\,\vec{x}_k + \vec{b}$$

It is well known that the direction of steepest descent is given by

$$\vec{d}_k = -\nabla f(\vec{x}_k) = -\mathbf{A}\,\vec{x}_k - \vec{b} = -\vec{r}_k$$

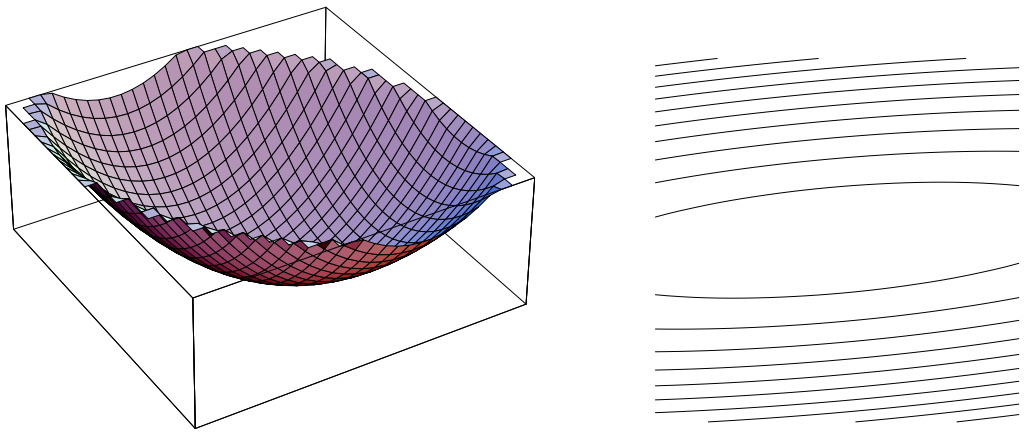


Figure 11.10: Graph of a function to be minimized and its level curves

This is the reason for the name **stepest descent** or **gradient method**. Thus we search for a better solution in the direction $\vec{d}_k$, i.e. we have to determine the coefficient $\alpha \in \mathbb{R}$ such that the value of the function

$$h(\alpha) = f(\vec{x}_k + \alpha\, \vec{d}_k)$$

is minimal. This leads to the condition

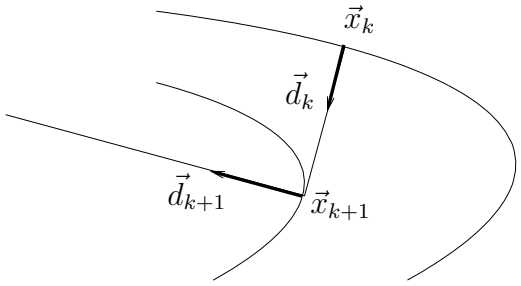


Figure 11.11: One step of a gradient iteration

$$\begin{aligned} 0 &= \frac{d}{d\,\alpha}\, h(\alpha) = \langle \nabla f(\vec{x}_k + \alpha\, \vec{d}_k)\,,\, \vec{d}_k \rangle = \langle \mathbf{A}\,(\vec{x}_k + \alpha\, \vec{d}_k) + \vec{b}\,,\, \vec{d}_k \rangle \\ &= \langle \vec{r}_k + \alpha\, \mathbf{A}\, \vec{d}_k\,,\, \vec{d}_k \rangle \\ \alpha &= -\frac{\langle \vec{r}_k\,,\, \vec{d}_k \rangle}{\langle \mathbf{A}\, \vec{d}_k\,,\, \vec{d}_k \rangle} \end{aligned}$$

and thus the next approximation point

$$\vec{x}_{k+1} = \vec{x}_k + \alpha\, \vec{d}_k = \vec{x}_k - \frac{\langle \vec{r}_k\,,\, \vec{d}_k \rangle}{\langle \mathbf{A}\, \vec{d}_k\,,\, \vec{d}_k \rangle}\, \vec{d}_k$$

One step of this iteration is shown in Figure 11.11 and a pseudo code for the algorithm is shown in Table 11.7.

The computational effort for one step in the algorithm seems to be: 2 matrix/vector multiplications, 2 scalar products and 2 vector additions. But the resudial vector $\vec{r}_k$ and the direction vector $\vec{d}_k$ differ only in their sign. Since

$$\vec{r}_{k+1} = \mathbf{A}\, \vec{x}_{k+1} + \vec{b} = \mathbf{A}\,(\vec{x}_k + \alpha_k\, \vec{d}_k) + \vec{b} = \mathbf{A}\, \vec{x}_k + \vec{b} + \alpha_k\, \mathbf{A}\, \vec{d}_k = \vec{r}_k + \alpha_k\, \mathbf{A}\, \vec{d}_k$$

choose initial point $\vec{x}_0$
$k = 0$
while $\|\vec{r}_k\| = \|\mathbf{A}\,\vec{x}_k + \vec{b}\|$ too large
$\quad \vec{d}_k = -\vec{r}_k$
$\quad \alpha = -\dfrac{\langle \vec{r}_k\,,\,\vec{d}_k\rangle}{\langle \mathbf{A}\,\vec{d}_k\,,\,\vec{d}_k\rangle}$
$\quad \vec{x}_{k+1} = \vec{x}_k + \alpha\,\vec{d}_k$
$\quad k = k + 1$
endwhile

Table 11.7: A first gradient algorithm to solve $\mathbf{A}\,\vec{x} + \vec{b} = \vec{0}$

choose initial point $\vec{x}$
$\vec{r} = \mathbf{A}\,\vec{x}_0 + \vec{b}$
while $\rho = \|\vec{r}\|^2$ too large
$\quad \vec{d} = \mathbf{A}\,\vec{r}$
$\quad \alpha = -\dfrac{\rho}{\langle \vec{d}\,,\,\vec{r}\rangle}$
$\quad \vec{x} = \vec{x} + \alpha\,\vec{r}$
$\quad \vec{r} = \vec{r} + \alpha\,\vec{d}$
endwhile

Table 11.8: The gradient algorithm to solve $\mathbf{A}\,\vec{x} + \vec{b} = \vec{0}$

the necessary computations for one step of the iteration can be reduced. The improved algorithm in Table 11.8 requires

- one matrix–vector product and two scalar products
- two vector additions of the type $\vec{x} = \vec{x} + \alpha\,\vec{r}$
- storage for the sparse matrix and 3 vectors

If each row of the matrix $\mathbf{A}$ has on average nz nonzero entries the we determine that each iteration requires approximately $(4 + nz)\,N$ flops (multiplication/addition pairs).

Since the matrix $\mathbf{A}$ is positive definite we have

$$\frac{d^2}{d\alpha^2}\,h(\alpha) = \langle \mathbf{A}\,\vec{d}_k\,,\,\vec{d}_k\rangle > 0$$

unless $-\vec{d}_k = \mathbf{A}\,\vec{x}_k + \vec{b} = \vec{0}$. Thus we found a minimum of the function $h(\alpha)$ and consequently $f(\vec{x}_{k+1}) < f(\vec{x}_k)$, unless $\vec{x}_k$ equals the exact solution of $\mathbf{A}\,\vec{x} + \vec{b} = \vec{0}$. Since $\vec{d}_k = -\vec{r}_k$ we conclude that $\alpha \geq 0$, i.e. we actually made a step of positive length in the direction of the negative gradient.

The algorithm does not perform well if we search the minimal value in a narrow valley, as illustrated in Figure 11.12. Instead of *going down* the valley, the algorithm *jumps across* and it requires many steps to get close to the lowest point. This is reflected by the error estimation for this algorithm. One can show that (e.g. [LascTheo87, p. 496], [KnabAnge00, p. 212], [AxelBark84, Theorem 1.8])

$$\|\vec{x}_k - \vec{x}\|_A \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)^k \|\vec{x}_0 - \vec{x}\|_A \approx \left(1 - \frac{2}{\kappa}\right)^k \|\vec{x}_0 - \vec{x}\|_A$$

where we use the norm.

$$\|\vec{y}\|_A^2 = \langle \vec{y}\,,\,\mathbf{A}\,\vec{y}\rangle$$

For most matrices based on finite element problems we know that $\|\vec{y}\| \leq \alpha\,\|\vec{y}\|_A$ and thus

$$\|\vec{x}_k - \vec{x}\| \leq c\,\left(\frac{\kappa - 1}{\kappa + 1}\right)^k \approx c\,\left(1 - \frac{2}{\kappa}\right)^k$$

where

$$\kappa = \frac{\lambda_{max}}{\lambda_{min}} = \text{ condition number of } \mathbf{A}$$

Thus if the ratio of the largest and smallest eigenvalue of the matrix $\mathbf{A}$ is large then the algorithm converges slowly. Unfortunately this is most often the case, thus Figure 11.12 shows rather the typical situation than the exception.

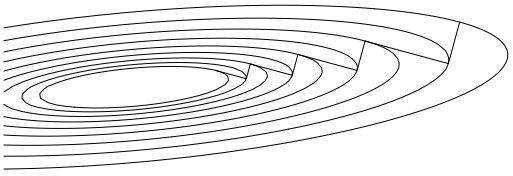


Figure 11.12: The gradient algorithm for a large condition number

Performance on the model problem

For the problem in section 11.7.2 we find $\kappa \approx \frac{4}{\pi^2}\, n^2$ and thus

$$q = 1 - q_1 = 1 - \frac{2}{\kappa} \approx 1 - \frac{\pi^2}{2\,n^2}$$

Then equation (11.4) implies that we need

$$k \geq \frac{D\,\ln 10}{q_1} = \frac{2\,D\,\ln 10}{\pi^2}\, n^2$$

iterations to increase the precision by D digits. Based on the estimate for the operations necessary to multiply the matrix with a vector we estimate the total number of flops as

$$9\,n^2\,k \approx \frac{18\,D\,\ln 10}{\pi^2}\, n^4$$

This is not substantially better than a banded Cholesky algorithm.

11.7.4 Conjugate gradient iteration

The **conjugate gradient method** will improve the above mentioned problem of the original gradient method. Instead of searching for the minimum of the function $f(\vec{x})$ in the direction of steepest descent we combine this direction with the previous search direction and aim to reach the minimal value of the function $f(\vec{x})$ in this plane with one step only.

Conjugate directions

The left section in Figure 11.13 shows elliptical level curves of the function $g(\vec{x}) = \langle \vec{x}\,,\, \mathbf{A}\,\vec{x} \rangle$. A first vector $\vec{a}$ is tangential to a given level curve at a point. A second vector $\vec{b}$ is connecting this point to the origin. The two vectors represent two subsequent search directions. When applying the transformation

$$\vec{u} = \begin{pmatrix} u \\ v \end{pmatrix} = \mathbf{A}^{1/2} \begin{pmatrix} x \\ y \end{pmatrix} = \mathbf{A}^{1/2}\,\vec{x}$$

we obtain

$$g(\vec{x}) = \langle \vec{x}\,,\, \mathbf{A}\,\vec{x} \rangle = \langle \mathbf{A}^{1/2}\,\vec{x}\,,\, \mathbf{A}^{1/2}\,\vec{x} \rangle = \langle \vec{u}\,,\, \vec{u} \rangle = h(\vec{u})$$

and the level curves of the function h in a (u, v) system will be circles, shown on the right in Figure 11.13. The two vectors $\vec{a}$ and $\vec{b}$ shown on in the left part will transform according to the same transformation rule. The resulting images will be orthogonal and thus

$$0 = \langle \mathbf{A}^{1/2}\vec{a}\,,\, \mathbf{A}^{1/2}\vec{b} \rangle = \langle \mathbf{A}\,\vec{a}\,,\, \vec{b} \rangle$$

The vectors $\vec{a}$ and $\vec{b}$ are said to be **conjugate**.

The basic conjugate gradient algorithm

The direction vectors $\vec{d}_{k-1}$ and $\vec{d}_k$ of two subsequent steps of the conjugate gradient algorithm should behave like the two vectors in the left part of Figure 11.13. The new direction vector $\vec{d}_k$ is assumed to be a linear combination of the gradient $\nabla f(\vec{x}_k) = \mathbf{A}\,\vec{x}_k + \vec{b} = \vec{r}_k$ and the old direction $\vec{d}_{k-1}$, i.e.

$$\vec{d}_k = -\vec{r}_k + \beta\,\vec{d}_{k-1} \quad \text{where} \quad \vec{r}_k = \mathbf{A}\,\vec{x}_k + \vec{b}$$

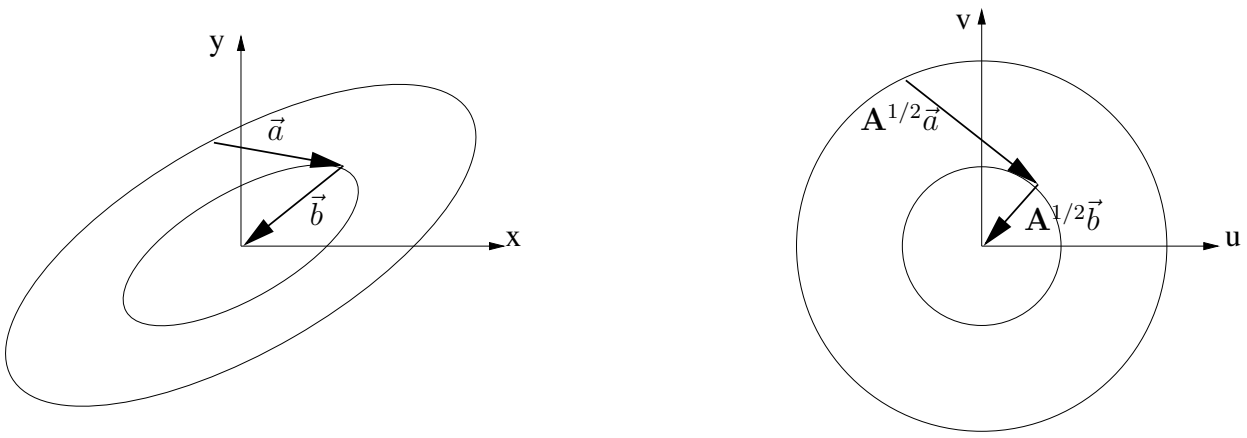


Figure 11.13: Ellipse and circle to illustrate conjugate directions

Since the two directions $\vec{d}_k$ and $\vec{d}_{k-1}$ have to be conjugate we conclude

$$\begin{aligned} 0 &= \langle \vec{d}_k \, , \, \mathbf{A}\, \vec{d}_{k-1} \rangle = \langle -\vec{r}_k + \beta\, \vec{d}_{k-1} \, , \, \mathbf{A}\, \vec{d}_{k-1} \rangle \\ \beta &= \frac{\langle \vec{r}_k \, , \, \mathbf{A}\, \vec{d}_{k-1} \rangle}{\langle \vec{d}_{k-1} \, , \, \mathbf{A}\, \vec{d}_{k-1} \rangle} \end{aligned}$$

Then the optimal value of α_k to minimize $h(\alpha) = f(\vec{x}_k + \alpha_k\, \vec{d}_k)$ can be determined with a calculation identical to the standard gradient method, i.e.

$$\alpha_k = -\frac{\langle \vec{r}_k \, , \, \vec{d}_k \rangle}{\langle \mathbf{A}\, \vec{d}_k \, , \, \vec{d}_k \rangle}$$

and we obtain a better approximation of the solution of the linear system as $\vec{x}_{k+1} = \vec{x}_k + \alpha_k\, \vec{d}_k$. This algorithm spelled out on the left in Table 11.9 and its result is illustrated in Figure 11.14. Just as in the standard gradient algorithm we find $\frac{d^2}{d\alpha^2}\, h(\alpha) = \langle \mathbf{A}\, \vec{d}_k \, , \, \vec{d}_k \rangle$ and we find that

- either the algorithm terminates, i.e. we found the optimal solution at this point
- or $\alpha_k > 0$.

This allows for division by α_k in the analysis of the algorithm.

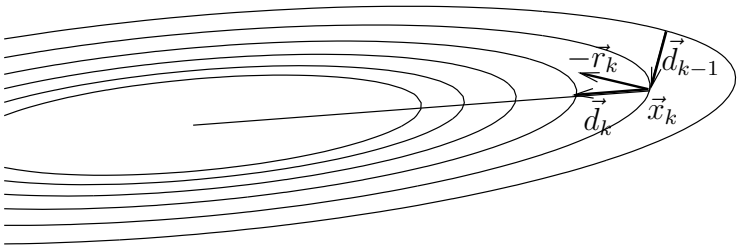


Figure 11.14: One step of a conjugate gradient iteration

choose initial point $\vec{x}_0$

$\vec{r}_0 = \mathbf{A}\,\vec{x}_0 + \vec{b}$

$\vec{d}_0 = -\vec{r}_0$

$\alpha_0 = -\dfrac{\langle \vec{r}_0\,,\,\vec{d}_0\rangle}{\langle \mathbf{A}\,\vec{d}_0\,,\,\vec{d}_0\rangle}$

$\vec{x}_1 = \vec{x}_0 + \alpha_0\,\vec{d}_0$

$k = 1$

while $\|\vec{r}_k\| = \|\mathbf{A}\,\vec{x}_k + \vec{b}\|$ too large

$\quad \beta_k = \dfrac{\langle \vec{r}_k\,,\,\mathbf{A}\,\vec{d}_{k-1}\rangle}{\langle \vec{d}_{k-1}\,,\,\mathbf{A}\,\vec{d}_{k-1}\rangle}$

$\quad \vec{d}_k = -\vec{r}_k + \beta_k\,\vec{d}_{k-1}$

$\quad \alpha_k = -\dfrac{\langle \vec{r}_k\,,\,\vec{d}_k\rangle}{\langle \mathbf{A}\,\vec{d}_k\,,\,\vec{d}_k\rangle}$

$\quad \vec{x}_{k+1} = \vec{x}_k + \alpha_k\,\vec{d}_k$

$\quad k = k + 1$

endwhile

choose initial point $\vec{x}$

$\vec{r} = \mathbf{A}\,\vec{x} + \vec{b}$

$\rho_0 = \|\vec{r}\|^2$

$\vec{d} = -\vec{r}$

$\vec{p} = \mathbf{A}\,\vec{d}$

$\alpha = \dfrac{\rho_0}{\langle \vec{p},\,\vec{d}\rangle}$

$\vec{x} = \vec{x} + \alpha\,\vec{d}$

$\vec{r} = \vec{r} + \alpha\,\vec{p}$

$k = 1$

while $\rho_k = \|\vec{r}\|^2$ too large

$\quad \beta = \dfrac{\rho_k}{\rho_{k-1}}$

$\quad \vec{d} = -\vec{r} + \beta\,\vec{d}$

$\quad \vec{p} = \mathbf{A}\,\vec{d}$

$\quad \alpha = \dfrac{\rho_k}{\langle \vec{p},\,\vec{d}\rangle}$

$\quad \vec{x} = \vec{x} + \alpha\,\vec{d}$

$\quad \vec{r} = \vec{r} + \alpha\,\vec{p}$

$\quad k = k + 1$

endwhile

Table 11.9: The conjugate gradient algorithm to solve $\mathbf{A}\,\vec{x} + \vec{b} = \vec{0}$ and an efficient implementation

Orthogonality properties

We define the subspace

$$K(k, \vec{d}_0) = \text{span}\{\vec{d}_0\,,\, \mathbf{A}\,\vec{d}_0\,,\, \mathbf{A}^2\,\vec{d}_0\,,\, \ldots,\, \mathbf{A}^{k-1}\,\vec{d}_0\,,\, \mathbf{A}^k\,\vec{d}_0\}$$

Since $\vec{r}_{k+1} = \vec{r}_k + \alpha_k\,\mathbf{A}\,\vec{d}_k$ and $\vec{d}_k = -\vec{r}_k + \beta_k\,\vec{d}_{k-1}$ we conclude

$$\vec{r}_i \in K(k, \vec{d}_0) \quad , \quad \vec{d}_i \in K(k, \vec{d}_0) \quad \text{and} \quad \vec{x}_i \in \vec{x}_0 + K(k, \vec{d}_0) \quad \text{for} \quad 0 \leq i \leq k$$

The above is correct for any choice of the parameters β_k. Now we examine the algorithm in Table 11.9 with the optimal choice for α_k, but the values of β_k in $\vec{d}_k = -\vec{r}_k + \beta_k\,\vec{d}_{k-1}$ are to be determined by a new criterion.

The theorem below shows that we minimized the function $f(\vec{x})$ on the $k+1$ dimensional affine subspace $K(k, \vec{d}_0)$, and not only on the two dimensional plane spanned by the last two search directions.

11–27 Theorem : *Consider given values of $k \in \mathbb{N}$, $\vec{x}_0$ and $\vec{r}_0 = \mathbf{A}\,\vec{x}_0 + \vec{b}$. Choose the vector $\vec{x} \in \vec{x}_0 + K(k, \vec{d}_0)$ such that the function $g(\vec{x})$ is minimized on the affine subspace $\vec{x}_0 + K(k, \vec{d}_0)$. The subspace $K(k, \vec{d}_0)$ has dimension $k+1$. The following orthogonality properties are correct*

$$\begin{aligned} \langle \vec{r}_j\,,\, \vec{r}_i \rangle = 0 \qquad & \textit{for all} \qquad 0 \leq i \neq j < k \\ \langle \vec{d}_j\,,\, \mathbf{A}\,\vec{d}_i \rangle = 0 \qquad & \textit{for all} \qquad 0 \leq i \neq j \leq k \\ \langle \vec{r}_k\,,\, \vec{y} \rangle = \langle \vec{x}_k - \vec{x}\,,\, \mathbf{A}\,\vec{y} \rangle = 0 \qquad & \textit{for all} \qquad \vec{y} \in K(k, \vec{d}_0) \end{aligned}$$

The values

$$\beta_k = \frac{\langle \vec{r}_k\,,\, \mathbf{A}\,\vec{d}_{k-1} \rangle}{\langle \vec{d}_{k-1}\,,\, \mathbf{A}\,\vec{d}_{k-1} \rangle}$$

will generate the optimal solution with the algorithm on the left in Table 11.9. ◇

Proof : If we choose the vector $\vec{x} \in \vec{x}_0 + K(k, \vec{d}_0)$ such that the function $f(\vec{x})$ is minimized on the affine subspace $\vec{x}_0 + K(k, \vec{d}_0)$ then its gradient has to be orthogonal on the subspace $K(k, \vec{d}_0)$, i.e.

$$\langle \mathbf{A}\,\vec{x} + \vec{b}\,,\, \vec{h} \rangle = \langle \vec{r}\,,\, \vec{h} \rangle = 0 \quad \text{for all} \quad \vec{h} \in K(k, \vec{d}_0)$$

Since $\vec{r}_{k+1} = \mathbf{A}\,\vec{x} + \vec{b}$ this leads to

$$\langle \vec{r}_{k+1}\,,\, \vec{r}_i \rangle = \langle \vec{r}\,,\, \vec{r}_i \rangle = 0 \quad \text{for all} \quad 0 \leq i \leq k$$

and $K(k, \vec{d}_0)$ is a strict subspace of $K(k+1, \vec{d}_0)$. This implies $\dim(K(k, \vec{d}_0)) = k+1$.

Using $\vec{r}_{k+1} = \vec{r}_k + \alpha_k\,\mathbf{A}\,\vec{d}_k$ and $\vec{d}_i = -\vec{r}_k + \beta_i\,\vec{d}_{i-1}$ we conclude by recursion

$$\begin{aligned} \langle \vec{d}_i\,,\, \mathbf{A}\,\vec{d}_k \rangle &= \frac{1}{\alpha_k}\,\langle -\vec{r}_i + \beta_i \vec{d}_{i-1}\,,\, \vec{r}_{k+1} - \vec{r}_k \rangle \\ &= \frac{\beta_i}{\alpha_k} \langle \vec{d}_{i-1}\,,\, \vec{r}_{k+1} - \vec{r}_k \rangle = \frac{1}{\alpha_k} \left(\prod_{j=1}^{i} \beta_j \right)\, \langle \vec{d}_0\,,\, \vec{r}_{k+1} - \vec{r}_k \rangle \\ &= \frac{-1}{\alpha_k} \left(\prod_{j=1}^{i} \beta_j \right)\, \langle \vec{r}_0\,,\, \vec{r}_{k+1} - \vec{r}_k \rangle = 0 \end{aligned}$$

The above is correct for all possible choices of β_j and also implies

$$0 = \langle \vec{d}_k\,,\, \mathbf{A}\,\vec{d}_{k-1} \rangle = \langle -\vec{r}_k + \beta_k\,\vec{d}_{k-1}\,,\, \mathbf{A}\,\vec{d}_{k-1} \rangle = -\langle \vec{r}_k\,,\, \mathbf{A}\,\vec{d}_{k-1} \rangle + \beta_k\,\langle \vec{d}_{k-1}\,,\, \mathbf{A}\,\vec{d}_{k-1} \rangle$$

Thus the optimal values for β_k are as shown in the theorem. □

11–28 Corollary :

- *Since* $\dim(K(k, \vec{d}_0)) = k+1$ *the conjugate gradient algorithm with exact arithmetic will terminate after at most* N *steps. Due to rounding errors this will not be of relevance for large matrices. In addition the number of steps might be prohibitively large. We use the conjugate gradient algorithm as an iterative method.*
- *Using the orthogonalities in the above theorem we conclude*

$$\begin{aligned}
\langle \vec{r}_k \,,\, \vec{d}_k \rangle &= \langle \vec{r}_k \,,\, -\vec{r}_k + \beta_k \, \vec{d}_{k-1} \rangle = -\|\vec{r}_k\|^2 \\
\langle \vec{r}_k \,,\, \mathbf{A}\, \vec{d}_{k-1} \rangle &= \beta_k \, \langle \vec{d}_{k-1} \,,\, \mathbf{A}\, \vec{d}_{k-1} \rangle \\
\vec{r}_{k+1} &= \vec{r}_k + \alpha_k \, \mathbf{A}\, \vec{d}_k \\
\langle \vec{r}_{k+1} \,,\, \mathbf{A}\, \vec{d}_k \rangle &= \frac{1}{\alpha_k} \, \langle \vec{r}_{k+1} \,,\, \vec{r}_{k+1} - \vec{r}_k \rangle = \frac{1}{\alpha_k} \, \|\vec{r}_{k+1}\|^2 \\
\langle \vec{d}_k \,,\, \mathbf{A}\, \vec{d}_k \rangle &= \frac{1}{\alpha_k} \, \langle \vec{d}_k \,,\, \vec{r}_{k+1} - \vec{r}_k \rangle = \frac{1}{\alpha_k} \, \|\vec{r}_k\|^2 \\
\beta_k &= \frac{\langle \vec{r}_k \,,\, \mathbf{A}\, \vec{d}_{k-1} \rangle}{\langle \vec{d}_{k-1} \,,\, \mathbf{A}\, \vec{d}_{k-1} \rangle} = \frac{\alpha_{k-1} \, \|\vec{r}_k\|^2}{\alpha_{k-1} \, \|\vec{r}_{k-1}\|^2} = \frac{\|\vec{r}_k\|^2}{\|\vec{r}_{k-1}\|^2}
\end{aligned}$$

◇

The above properties allow a more efficient implementation of the conjugate gradient algorithm. The algorithm on the right in Table 11.9 is taken from [GoluVanLoan96]. This improved implementation of the algorithm requires for each iteration

- one matrix–vector product and two scalar products
- three vector additions of the type $\vec{x} = \vec{x} + \alpha \, \vec{r}$
- storage for the sparse matrix and 4 vectors

If each row of the matrix $\mathbf{A}$ has on average nz nonzero entries then we determine that each iteration requires approximately $(5 + nz)\, N$ flops (multiplication/addition pairs).

Convergence estimate

Assume that the exact solution is given by $\vec{x}$, i.e. $\mathbf{A}\, \vec{z} + \vec{b} = \vec{0}$. Use the notation $\vec{r} = \mathbf{A}\, \vec{y} + \vec{b}$, resp. $\vec{y} = \mathbf{A}^{-1}(\vec{r} - \vec{b})$ to conclude that $\vec{y} - \vec{x} = \mathbf{A}^{-1}\, \vec{r}$. Then consider the following function

$$g(\vec{y}) = \|\vec{y} - \vec{x}\|_A^2 = \langle \vec{y} - \vec{x} \,,\, \mathbf{A}(\vec{y} - \vec{x}) \rangle = \langle \vec{r} \,,\, \mathbf{A}^{-1} \vec{r} \rangle$$

and verify that

$$\begin{aligned}
\frac{1}{2} \, \|\vec{x} - \vec{z}\|_A^2 &= \frac{1}{2} \, \langle \vec{x} - \vec{z} \,,\, \mathbf{A}(\vec{x} - \vec{z}) \rangle = \frac{1}{2} \, \langle \vec{x} + \mathbf{A}^{-1}\, \vec{b} \,,\, \mathbf{A}\, \vec{x} + \vec{b} \rangle \\
&= \frac{1}{2} \, \langle \vec{x} \,,\, \mathbf{A}\, \vec{x} + \vec{b} \rangle + \langle \vec{x} \,,\, \vec{b} \rangle + \frac{1}{2} \, \langle \mathbf{A}^{-1}\, \vec{b} \,,\, \vec{b} \rangle \\
&= f(\vec{x}) + \frac{1}{2} \, \langle \mathbf{A}^{-1}\, \vec{b} \,,\, \vec{b} \rangle
\end{aligned}$$

Thus the conjugate gradient algorithm minimized this norm on the subspaces. It should be no surprise that the error estimate can be expresses in this norm. Find the result and proofs e.g in [LascTheo87], [KnabAnge00, p. 218] or [AxelBark84].

$$\|\vec{x}_k - \vec{x}\|_A \leq 2 \, \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k \, \|\vec{x}_0 - \vec{x}\|_A \approx 2 \, \left(1 - \frac{2}{\sqrt{\kappa}} \right)^k \, \|\vec{x}_0 - \vec{x}\|_A$$

and this leads to

$$\|\vec{x}_k - \vec{x}\| \leq c \left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k \approx c \left(1 - \frac{2}{\sqrt{\kappa}}\right)^k$$

This is considerably better than the estimate for the steepest descent method, since κ is replaced by $\sqrt{\kappa} \ll \kappa$.

Performance on the model problem

For the problem in section 11.7.2 we find $\sqrt{\kappa} \approx \frac{2}{\pi}\, n$ and thus

$$q = 1 - q_1 = 1 - \frac{2}{\sqrt{\kappa}} \approx 1 - \frac{\pi}{n}$$

Then equation (11.4) implies that we need

$$k \geq \frac{D\, \ln 10}{q_1} = \frac{D\, \ln 10}{\pi}\, n$$

iterations to increase the precision by D digits. Based on the estimate for the operations necessary to multiply the matrix with a vector we estimate the total number of flops as

$$(5+5)\, n^2\, k \approx 10\, \frac{D\, \ln 10}{\pi}\, n^3$$

This is considerably better than a banded Cholesky algorithm, since the number of operations is proportional to n^3 instead of n^4. For large values of n the conjugate gradient method is clearly preferable.

Table 11.10 shows the required storage and the number of necessary flops to solve the 2–D and 3–D model problem with n free grid points in each direction. The results are illustrated[13] in Figure 11.15. Observe that one operation for the gradient algorithms requires more time than one operation of the Cholesky algorithm, due to the multiplication of the sparse matrix with a vector.

We may draw the following conclusions from Table 11.10 and the corresponding Figure 11.15. Table 11.11 lists approximate computation times for a computer capable of performing 10^8 flops per second.

- The iterative methods require less memory than the direct solver. For 3–D problem this difference is accentuated.
- For 2–D problems with small resolution the banded Cholesky algorithm is more efficient than the conjugate gradient method. For larger 2–D problems conjugate gradient will perform better.
- For 3–D problems one should always use conjugate gradient, even for small problems.
- For small 3–D problems banded Cholesky might be able to give results within a reasonable time frame.
- The method of steepest descent is never competitive.

11.7.5 Preconditioned conjugate gradient iteration

to be written

[13] We required the accuracy to be improved by 6 digits.

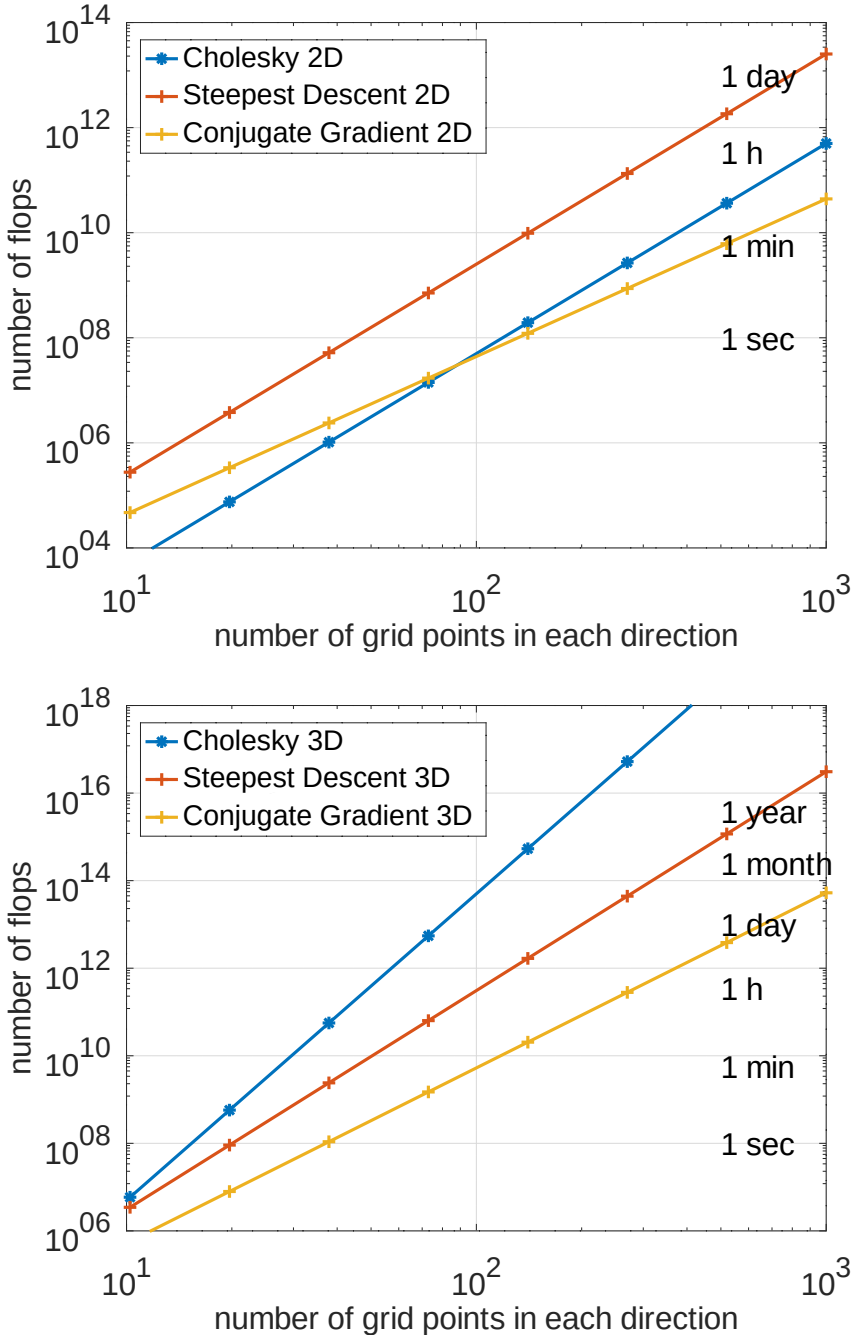


Figure 11.15: Number of operations of banded Cholesky, steepest descent and conjugate gradient on the model problem

	2–D		3–D	
	storage	flops	storage	flops
Cholesky, banded	$\frac{1}{2}\, n^3$	$\frac{1}{2}\, n^4$	$\frac{1}{2}\, n^5$	$\frac{1}{2}\, n^7$
Steepest Descent	$8\, n^2$	$9\, \frac{2\, D\, \ln 10}{\pi^2}\, n^4$	$10\, n^3$	$11\, \frac{2\, D\, \ln 10}{\pi^2}\, n^5$
Conjugate Gradient	$9\, n^2$	$10\, \frac{D\, \ln 10}{\pi}\, n^3$	$11\, n^3$	$12\, \frac{D\, \ln 10}{\pi}\, n^4$

Table 11.10: Comaprison of algorithms for the model problem

flops	10^8	10^9	10^{10}	10^{11}	10^{12}	10^{14}	10^{16}	10^{18}
Time required	1 sec	10 sec	1.7 min	17 min	2.8 h	11.6 days	3.2 years	320 years

Table 11.11: Time required to complete a given number of flops

11.8 Exercises

• Exercise 11–1:
Examine a diagonal matrix $\mathbf{D} = \mathrm{diag}(d_1,\, d_2,\, d_3, \ldots d_n)$. Verify

$$\|\mathbf{D}\| = \max_i |d_i| \quad \text{and} \quad \kappa = \frac{\max_i |d_i|}{\min_i |d_i|}$$

• Exercise 11–2:
The condition number can also be defined depending on the matrix $\mathbf{A}$ and the vector $\vec{x}$ by

$$\kappa(\vec{x}) = \|\mathbf{A}^{-1}\| \frac{\|\mathbf{A}\, \vec{x}\|}{\|\vec{x}\|}$$

(a) Verify that the definition number in definition 11–1 is related to the above definition by

$$\kappa(\vec{x}) \leq \kappa$$

(b) Verify the modification of result 11–4, i.e.

$$\frac{\|\vec{\Delta x}\|}{\|\vec{x}\|} \leq \kappa(\vec{x})\, \frac{\|\vec{\Delta b}\|}{\|\vec{b}\|}$$

(c) Consider a symmetric matrix $\mathbf{A}$ with eigenvalues $0 < \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n$ and a vector $\vec{b}$ which is a linear combination of eigenvectors with eigenvalues smaller then Λ, i.e.

$$\vec{b} \in \mathrm{span}\{\vec{e}_i \;:\; \lambda_i \leq \Lambda\}$$

If $\mathbf{A}\, \vec{x} = \vec{b}$ then

$$\kappa(\vec{x}) \leq \frac{\Lambda}{\lambda_1}$$

This has to be compared with result 11–3. The situation occurs for FEM problems if the RHS $\vec{b}$ is dominated by small eigenvalue components.

• Exercise 11–3:
Verify that for an orthogonal matrix $\mathbf{Q}$ (i.e. $\mathbf{Q}^{-1} = \mathbf{Q}^T$) we find $\|\mathbf{Q}\, \vec{x}\| = \|\vec{x}\|$ and the condition number of $\mathbf{Q}$ is $\kappa = 1$.

• **Exercise 11–4:**
Find the Cholesky factorization of

$$\mathbf{A} = \begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix}$$

The answer can be found with brain-power, paper and pencil or with brain-power and some software.

• **Exercise 11–5:**
In linear elasticity problems the matrix

$$\begin{bmatrix} 1-\nu & \nu & \nu \\ \nu & 1-\nu & \nu \\ \nu & \nu & 1-\nu \end{bmatrix}$$

is used to formulate Hooke's law (equation (9.3) on page 215). Use the algorithm of Cholesky to verify that this matrix is strictly positive definite if $0 \leq \nu \leq \frac{1}{2}$.
Hint: use one step of the Cholesky factorization to reduce the problem to a 2×2 matrix. Then multiply the matrix by an appropriate positive factor to simplify the further calculations.

• **Exercise 11–6:**
Consider the code segment below

```
int jMax;  double tmp;
int kj,k0,k0j;

k0=kj=k0j=0;  // initialize Rp[0][0] = R[0*band]
for(int k= 0; k < length-1; k++) {
  if ( fabs(Rp[k][0]) <= TOL ) { return message ;}
  };
  jMax = min(band,length-k);
  for (int j=1;j<jMax; j++){
    kj += band;
    k0j++;
    tmp = R[k0j]/R[k0];
    for(int i= jMax-j-1 ; i>=0; i--) R[kj+i] -= tmp*R[k0j+i];
    R[k0j] = tmp;
  }; //for j flops: band^2/2
  kj=k0j=(k0+=band);  // initialise Rp[k][0] = R[k*band]
}; //for k   length iterations
```

Verify that this code returns the same result as the codes in tables 11.3 and 11.4. This version does not require a temporary storage for one row of the banded matrix. On the Pentium III computer the performance was not as good as the code in 11.4. On the Alpha 21264 with the Compaq compiler the results of the three implementations were rather close together. With the gcc compiler we found 11.4 to be faster than this code and 11.3 slower than this code. This illustrates that hardware architecture and compiler are important when choosing the best implementation.

• Exercise 11–7:
Consider the 5×5 matrix $\mathbf{A}$ with 2 on the diagonal and -1 on the first upper and lower diagonal. As an initial vector $\vec{x}$ choose all components equal to $1/\sqrt{5}$, thus this vector is normalized.

(a) Compute 5 steps of the power iteration to approximate the largest eigenvalue λ_5.

(b) Find the maximal error of the above result by the estimate $\|\mathbf{A}\,\vec{x} - \beta\,\vec{x}\|$ in theorem 11–25.

(c) Compute the Rayleigh quotient and verify that its value is closer to the exact eigenvalue $\lambda_5 = 2\,\cos(\frac{5\,\pi}{6}) \approx 3.7321$.

The exact eigenvalues of this type of $n \times n$ matrix are $\lambda_i = 2\,(1 - \cos(\frac{i\,\pi}{n+1}))$ for $i = 1,\ 2, \ldots\ n$.
Use appropriate software to solve this problem.

• Exercise 11–8:
Reconsider the previous exercise with a 10×10 matrix.

(a) Can you explain why the result after 10 iterations (3.6519) is **not** close to the largest eigenvalue $\lambda_{10} \approx 3.9190$. The error is larger than permitted by the à posteriori error bound (0.13149).

(b) Rerun the above computations again, but set the first component of the initial vector to 0 and do 20 iterations. Why is the result correct for this run?

• Exercise 11–9:
Consider the same type of matrix as in the two previous problems. For an $n \times n$ matrix the exact eigenvalues are $\lambda_i = 2\,(1 - \cos(\frac{i\,\pi}{n+1})) = 4\ \sin^2\left(\frac{i\,\pi}{2\,(n+1)}\right)$ for $i = 1,\ 2, \ldots\ n$. If $i \ll n$ we find $\lambda_i = 4\ \sin^2\left(\frac{i\,\pi}{2\,(n+1)}\right) \approx i^2\ \left(\frac{\pi}{n+1}\right)^2$ and thus $\lambda_i \approx i^2\,\lambda_1$. Use the inverse power iteration method to determine the first m eigenvalues, where $m \ll n$.

(a) How many iterations are necessary (approximately) to determine the first eigenvalue with 4 significant digits?

(b) How many iterations are necessary (approximately) to determine the first five eigenvalues with 4 significant digits?

(c) Explain why the above result is an rough estimate at best and by no means a guarantee for 4 correct digits. Describe a simple stopping criterion for the algorithm.

• Exercise 11–10:
Describe an algorithm to determine the m largest eigenvalues of a positive definite, symmetric matrix.

• Exercise 11–11:
Let $\mathbf{A}$ be a symmetric matrix and $\mathbf{B}$ a symmetric, strictly positive definite matrix with classical Cholesky factorization $\mathbf{R}$, i.e. $\mathbf{B} = \mathbf{R}^T \cdot \mathbf{R}$. Examine the generalized eigenvalue problem

$$\mathbf{A}\,\vec{v} = \lambda\,\mathbf{B}\,\vec{v}$$

(a) Verify that the generalized eigenvalues above are classical eigenvalues of the matrix $\tilde{\mathbf{A}}$, where

$$\mathbf{R}^T \cdot \tilde{\mathbf{A}} \cdot \mathbf{R} = \mathbf{A}$$

If $\vec{v}$ is a generalized eigenvector, then $\vec{y} = \mathbf{R}\,\vec{v}$ is an eigenvector of $\tilde{\mathbf{A}}$ for the same eigenvalue.

(b) Let $\vec{r}$ and $\vec{s}$ be the residuals for approximations of the eigenvalues λ and eigenvectors $\vec{v}$, resp. $\vec{y}$. We have

$$\vec{r} = \mathbf{A}\,\vec{v} - \lambda\,\mathbf{B}\,\vec{v} \quad \text{and} \quad \vec{s} = \tilde{\mathbf{A}}\,\vec{y} - \lambda\,\vec{y}$$

Verify the identity

$$\langle \vec{r},\, \mathbf{B}^{-1}\,\vec{r} \rangle = \langle \vec{s},\, \vec{s} \rangle$$

(c) Use the above and result 11–25 to verify à posteriori estimates for the generalized eigenvalue problem. Let β be a scalar and $\vec{v}$ an approximate eigenvector with $\langle \vec{v},\, \mathbf{B}\,\vec{v} \rangle = 1$. With the residual $\vec{r} = \mathbf{A}\,\vec{v} - \beta\,\mathbf{B}\,\vec{v}$ verify the estimates

$$\min(|\lambda_i - \beta|) \leq \sqrt{\langle \vec{r},\, \mathbf{B}^{-1}\,\vec{r} \rangle} \quad \text{and} \quad \min(|\lambda_i - \beta|) \leq \frac{\langle \vec{r},\, \mathbf{B}^{-1}\,\vec{r} \rangle}{\text{gap}}$$

Appendix A

Some mathematical results and formulas

A.1 Vectors and matrices

A.1.1 Products of matrices and vectors

A.1.2 Scalar product of vectors

The scaler product of two vector in $\mathbb{R}^n$ is given by

$$\langle \vec{x}\,,\, \vec{y} \rangle = \vec{x}^T \cdot \vec{y} = (x_1, x_2, \ldots, x_n) \cdot \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix} = \sum_{k=1}^{n} x_k\, y_k$$

If $\mathbf{A}$ is a $n \times n$ matrix we have

$$\langle \vec{x}\,,\, \mathbf{A} \cdot \vec{y} \rangle = \vec{x}^T \cdot (\mathbf{A} \cdot \vec{y}) = \left(\vec{x}^T \cdot \mathbf{A}\right) \cdot \vec{y} = (\mathbf{A} \cdot \vec{x})^T \cdot \vec{y} = \langle \mathbf{A}^T \cdot \vec{x}\,,\, \vec{y} \rangle$$

Thus for symmetric, real matrices $\mathbf{A} = \mathbf{A}^T$ we have the handy property

$$\langle \vec{x}\,,\, \mathbf{A} \cdot \vec{y} \rangle = \langle \mathbf{A} \cdot \vec{x}\,,\, \vec{y} \rangle$$

A.1.3 Diagonalisation of a symmetric matrix, orthogonal matrices

A $n \times n$ matrix $\mathbf{Q}$ with $\mathbf{Q}^{-1} = \mathbf{Q}^T$ is called an **orthogonal matrix**. Thus the important property is

$$\mathbf{Q} \cdot \mathbf{Q}^T = \mathbf{Q}^T \cdot \mathbf{Q} = \mathbb{I}_n$$

If we consider the columns of $\mathbf{Q}$ as vectors, i.e.

$$\mathbf{Q} = [\vec{r}_1, \vec{r}_2, \ldots, \vec{r}_n]$$

then the vectors $\vec{r}_k$ are of length 1 and are pairwise orthogonal.

For a **symmetric**, real $n \times n$ matrix $\mathbf{A}$

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{12} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \ldots & a_{nn} \end{bmatrix}$$

there is an orthogonal matrix **Q** and a diagonal matrix **D**, such that

$$\mathbf{Q}^T\mathbf{A}\,\mathbf{Q} = \mathbf{D} = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda_n \end{bmatrix}$$

or

$$\mathbf{A} = \mathbf{Q}\,\mathbf{D}\,\mathbf{Q}^T$$

The elements λ_i on the diagonal of **D** are **eigenvalues** of the symmetric matrix **A** and the column vectors of **Q** are normalized **eigenvectors** of **A**. Since finding all eigenvectors and eigenvalues of a symmetric matrix can be a computationally expensive task one should try to avoid numerical methods requiring all those values. The decomposition is a very valuable tool for theoretical considerations.

A.2 Gradient, divergence and the Laplace operator

A.2.1 Vectors in different coordinate systems

Most often a vector in space $\mathbb{R}^3$ is represented by cartesian coordinates, a triple of numbers

$$\vec{v} = \begin{pmatrix} v_1 \\ v_2 \\ v_3 \end{pmatrix} = v_1\,\vec{e}_x + v_2\,\vec{e}_y + v_3\,\vec{e}_z$$

where $\vec{e}_i$ are the three orthonormal basis vectors.

If cylindrical coordinates are used then another set of basis vectors can be useful, namely

$$\vec{e}_\rho = \frac{1}{\sqrt{x^2+y^2}} \begin{pmatrix} x \\ y \\ 0 \end{pmatrix} = \begin{pmatrix} \cos\phi \\ \sin\phi \\ 0 \end{pmatrix} \quad , \quad \vec{e}_\phi = \begin{pmatrix} -\sin\phi \\ \cos\phi \\ 0 \end{pmatrix} \quad \text{and} \quad \vec{e}_z = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

All the vectors have lenght 1. If the radius ρ is increased then the vector „grows“ in the direction of $\vec{e}_\rho$. If the angle ϕ is increased then the vector „grows“ in the direction of $\vec{e}_\phi$. An arbitrary vector can be written as linear combination of these basis vectors

$$\vec{v} = v_\rho\,\vec{e}_\rho + v_\phi\,\vec{e}_\phi + v_z\,\vec{e}_z$$

For spherical coordinates

$$\begin{aligned} x &= r\,\cos\phi\,\sin\theta \\ y &= r\,\sin\phi\,\sin\theta \\ z &= r\,\cos\theta \end{aligned}$$

we have similarly

$$\vec{e}_r = \begin{pmatrix} \cos\phi\,\sin\theta \\ \sin\phi\,\sin\theta \\ r\,\cos\theta \end{pmatrix} \quad , \quad \vec{e}_\phi = \begin{pmatrix} -\sin\phi \\ \cos\phi \\ 0 \end{pmatrix} \quad \text{and} \quad \vec{e}_\theta = \begin{pmatrix} \cos\phi\,\cos\theta \\ \cos\phi\,\cos\theta \\ -\cos\theta \end{pmatrix}$$

and an arbitrary vector can be written as

$$\vec{v} = v_r\,\vec{e}_r + v_\phi\,\vec{e}_\phi + v_\theta\,\vec{e}_\theta$$

A.2.2 Gradient

The **gradient** of a scalar function $f(x, y, z)$ is a vector given by

$$\operatorname{grad} f = \nabla f = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial x} \right) = \frac{\partial f}{\partial x} \vec{e}_x + \frac{\partial f}{\partial y} \vec{e}_y + \frac{\partial f}{\partial z} \vec{e}_z$$

If the function is easier to describe in cylindrical coordinates, then we can use

$$\operatorname{grad} f = \nabla f = \frac{\partial f}{\partial \rho} \vec{e}_\rho + \frac{1}{\rho} \frac{\partial f}{\partial \phi} \vec{e}_\phi + \frac{\partial f}{\partial z} \vec{e}_z$$

or in spherical coordinates

$$\operatorname{grad} f = \nabla f = \frac{\partial f}{\partial \rho} \vec{e}_r + \frac{1}{r \sin\theta} \frac{\partial f}{\partial \phi} \vec{e}_\phi + \frac{1}{r} \frac{\partial f}{\partial \theta} \vec{e}_\theta$$

A.2.3 Divergence

For a vector valued function

$$\vec{v} = v_1 \vec{e}_x + v_2 \vec{e}_y + v_3 \vec{e}_z$$

the **divergence** is a scalar given by

$$\operatorname{div} v = \nabla \cdot \vec{v} = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z}$$

If the vector function $v(\rho, \phi, z)$ is given in cylindrical coordinates

$$\vec{v} = v_\rho \vec{e}_\rho + v_\phi \vec{e}_\phi + v_z \vec{e}_z$$

then

$$\operatorname{div} v = \nabla \cdot \vec{v} = \frac{1}{\rho} \frac{\partial (\rho v_\rho)}{\partial \rho} + \frac{1}{\rho} \frac{\partial v_\phi}{\partial \phi} + \frac{\partial v_z}{\partial z}$$

If the vector function $v(r, \phi, \theta)$ is given in spherical coordinates

$$\vec{v} = v_r \vec{e}_r + v_\phi \vec{e}_\phi + v_\theta \vec{e}_\theta$$

then

$$\operatorname{div} v = \nabla \cdot \vec{v} = \frac{1}{r^2} \frac{\partial (r^2 v_r)}{\partial r} + \frac{1}{r \sin\theta} \frac{\partial v_\phi}{\partial \phi} + \frac{1}{r \sin\theta} \frac{\partial (\sin\theta \, v_\theta)}{\partial \theta}$$

A.2.4 The Laplace operator

The differential operator $\Delta = \operatorname{div} \operatorname{grad}$ is called **Laplace operator**. It can be applied to functions of two or three variables. It is defined by

$$\Delta u = \operatorname{div} \operatorname{grad} u = \nabla \cdot (\nabla u) = \nabla^2 u$$

The exact form depends on the coordinate system to be used. For cartesian coordinates we find

$$\begin{aligned} \Delta u(x, y, z) &= u_{xx} + u_{yy} + u_{zz} \\ &= \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \frac{\partial^2 u}{\partial z^2} \end{aligned}$$

In cylindrical coordinates this leads to

$$\begin{aligned}\Delta u(\rho,\phi,z) &= \frac{1}{\rho}\,(\rho\,u_\rho)_\rho + \frac{1}{\rho^2}\,u_{\phi\phi} + u_{zz}\\ &= \frac{1}{\rho}\,\frac{\partial}{\partial\rho}(\rho\,\frac{\partial\,u}{\partial\rho}) + \frac{1}{\rho^2}\frac{\partial^2\,u}{\partial\phi^2} + \frac{\partial^2\,u}{\partial z^2}\end{aligned}$$

and for spherical coordinates

$$\begin{aligned}\Delta u(r,\phi,\theta) &= \frac{1}{r^2}\,(r^2u_r)_r + \frac{1}{r^2\sin^2\theta}\,u_{\phi\phi} + \frac{1}{r^2\sin\theta}\,(\sin\theta\,u_\theta)_\theta\\ &= \frac{1}{r^2}\,\frac{\partial}{\partial r}(r^2\frac{\partial\,u}{\partial r}) + \frac{1}{r^2\sin^2\theta}\frac{\partial^2\,u}{\partial\phi^2} + \frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}(\sin\theta\,\frac{\partial u}{\partial\theta})\end{aligned}$$

A.3 Divergence theorems

The well know fundamental theorem of calculus for functions of one variable

$$\int_a^b f'(x)\,dx = -f(a) + f(b)$$

can be extended to functions of multiple variables. If $G \subset \mathbb{R}^n$ is a „nice“ domain with boundary ∂G and outer unit normal vector $\vec{n}$ the we have the **divergence theorem**. For domains $G \subset \mathbb{R}^2$ we have

$$\iint\limits_G \operatorname{div}\vec{v}\,dA = \oint_{\partial G} \vec{v}\cdot\vec{n}\,ds$$

and if $G \subset \mathbb{R}^3$ then the notation is

$$\iint\limits_G \operatorname{div}\vec{v}\,dV = \oint_{\partial G} \vec{v}\cdot\vec{n}\,dA$$

where dV is the standard volume element and dA a surface element. The usual rule to differentiate products of two functions leads to

$$\begin{aligned}\nabla\cdot(f\,\vec{v}) &= (\nabla f)\cdot\vec{v} + f\,(\nabla\cdot\vec{v})\\ \operatorname{div}(f\,\vec{v}) &= (\operatorname{grad} f)\cdot\vec{v} + f\,(\operatorname{div}\vec{v})\end{aligned}$$

Using this and the divergence theorem we find

$$\begin{aligned}\iint\limits_G f\,(\operatorname{div}\vec{v})\,dA &= \iint\limits_G \operatorname{div}(f\,\vec{v}) - (\operatorname{grad} f)\cdot\vec{v}\,dA\\ &= \oint_{\partial G} f\,\vec{v}\cdot\vec{n}\,ds - \iint\limits_G (\operatorname{grad} f)\cdot\vec{v}\,dA\end{aligned}$$

This formula is referred to as **Green–Gauss theorem** and is similar to integration by parts for function of one variable

$$\int_a^b f\cdot g'\,dx = -f(a)\cdot g(a) + f(b)\cdot g(b) - \int_a^b f'\cdot g\,dx$$

For finite elements and calculus of variation the divergence theorem is most often used in the form below.

$$\begin{aligned}\iint\limits_G f\,(\operatorname{div}\operatorname{grad} g)\,dA &= \oint_{\partial G} f\,(\operatorname{grad} g)\cdot\vec{n}\,ds - \iint\limits_G (\operatorname{grad} f)\cdot(\operatorname{grad} g)\,dA\\ \iint\limits_G f\,\Delta g\,dA = \iint\limits_G f\,\nabla^2 g\,dA &= \oint_{\partial G} f\,\nabla g\cdot\vec{n}\,ds - \iint\limits_G \nabla f\cdot\nabla g\,dA\end{aligned}$$

A.4 Scalar product on function spaces

Let f and g be piecewise continuous, real valued functions on the interval $[a, b]$ and $\vec{u}, \vec{v} \in \mathbb{R}^n$. Then we can compare the **scalar product** for functions and vectors. For vectors we have

$$\langle \vec{u}, \vec{v} \rangle = \sum_{i=1}^{n} u_i \overline{v}_i$$

and for functions $f\ g$

$$\langle f, g \rangle = \int_a^b f(x)\, \overline{g(x)}\, dx$$

The symbol $f \in L_2$ denotes all functions f defined on $[a, b]$ for which the integral

$$\int_a^b |f(x)|^2\, dx$$

is finite. Now compare the behaviour of vectors and functions.

	R^n	$\mathcal{L}_2$
Objects	vectors $\vec{u}, \vec{v}, \vec{w}$	functions f, g, h
Scalar product	$\langle \vec{u}, \vec{v} \rangle = \sum_{i=1}^n u_i\, v_i$	$\langle f, g \rangle = \int_a^b f(x)\, g(x)\, dx$
Norm	$\|\vec{u}\|^2 = \sum_{i=1}^n \|u_i\|^2$	$\|f\|_2^2 = \int_a^b \|f(x)\|^2\, dx$
Orthogonality	$\langle \vec{u}, \vec{v} \rangle = 0$	$\langle f, g \rangle = 0$
Linearity	$\langle \vec{u} + \vec{w}, \vec{v} \rangle = \langle \vec{u}, \vec{v} \rangle + \langle \vec{w}, \vec{v} \rangle$ $\langle \lambda \vec{u}, \vec{v} \rangle = \lambda \langle \vec{u}, \vec{v} \rangle$	$\langle f + h, g \rangle = \langle f, g \rangle + \langle h, g \rangle$ $\langle \lambda f, g \rangle = \lambda \langle f, g \rangle$

For vectors we have the basic inequality of **Cauchy–Schwarz**[1]

$$|\ \langle \vec{u}\,,\ \vec{v} \rangle\ | \leq \|\vec{u}\|\ \|\vec{v}\|$$

The same result is correct for functions and reads as

$$\left| \int_a^b f(x)\, g(x)\, dx \right| \leq \left(\int_a^b |f(x)|^2\, dx \right)^{1/2} \cdot \left(\int_a^b |g(x)|^2\, dx \right)^{1/2}$$

The above similarities of vectors and functions can be used to motivate and construct the **Fourier series** of a function.

A.5 Fundamental lemma of calculus of variations

- One variable
- multiple variables

A–1 Lemma : ***Fundamental Lemma of the calculus of variations in one variable***
If $u(x)$ is a continuous function for $a \leq x \leq b$ and

$$\int_a^b u(x) \cdot \phi(x)\, dx = 0$$

for all infinitely often differentiable functions $\phi(x)$ with $\phi(a) = \phi(b) = 0$ then

$$u(x) = 0 \quad \textit{for all} \quad a \leq x \leq b$$

$\diamond$

[1] To verify this inequality use $|\langle \vec{u}, \vec{v} \rangle| = \|\vec{u}\|\ \|\vec{v}\|\ |\cos\alpha| \leq \|\vec{u}\|\ \|\vec{v}\|$

A–2 Lemma : ***Fundamental Lemma of the calculus of variations for multiple variables***
If $u(\vec{x})$ is a continuous function of $\vec{x} \in \Omega \subset \mathbb{R}^n$ for a subset Ω of $\mathbb{R}^n$ with piecewise smooth boundary and

$$\int_\Omega u(\vec{x}) \cdot \phi(\vec{x}) \, dV = 0$$

for all infinitely often differentiable functions ϕ with $\phi(\vec{x}) = 0$ for alle $\vec{x} \in \partial\Omega$ then

$$u(\vec{x}) = 0 \quad \textit{for all} \quad \vec{x} \in \Omega$$

$\diamond$

A.6 Maxwell's equation

To describe electric and magnetic fields the following physical quantities are relevant.

Symbol	Description	Units
$\vec{E}$	electric field	$\frac{V}{m}$
$\vec{D}$	electric flux density	$\frac{C}{m^2}$
$\vec{H}$	magnetic field	$\frac{A}{m}$
$\vec{B}$	magnetic flux density	$\frac{V\,s}{m^2}$
$\vec{J}$	electric current density	$\frac{A}{m^2}$
ρ	electric charge density	$\frac{C}{m^3}$

Use the **equation of continuity** (conservation of charge)

$$\operatorname{div} \vec{J} + \frac{d}{dt}\,\rho = 0$$

and the **constitutive relation**

$$\vec{D} = \varepsilon\,\varepsilon_0\,\vec{E} \quad , \quad \vec{B} = \nu\,\nu_0\,\vec{H} \quad , \quad \vec{J} = \sigma\,\vec{E} \tag{A.1}$$

where

ε_0	permittivity of vacuum	$8.854 \cdot 10^{-12}\,\frac{C}{V\,m}$
ν_0	permeability of vacuum	$4\,\pi \cdot 10^{-7}\,\frac{V\,s}{A\,m}$
σ	conductivity	$\frac{1}{\Omega\,m} = \frac{A}{V\,m}$

The values of ε, ν and σ depend on the material. In vacuum we have $\varepsilon = \nu = 1$.

A.6.1 Dynamic equations of Maxwell

The dynamic version of the fundamental **Maxwell's equations** is given by

$$\begin{array}{rcll} \operatorname{div} \vec{D} & = & \rho & \text{Gauss's law} \\ \operatorname{div} \vec{B} & = & 0 & \text{Gauss's magnetic law} \\ \operatorname{rot} \vec{E} & = & -\frac{\partial}{\partial t}\,\vec{B} & \text{Farraday's law} \\ \operatorname{rot} \vec{H} & = & \frac{\partial}{\partial t}\,\vec{D} & \text{Ampère's law} \end{array} \tag{A.2}$$

A.6.2 Static equations

If all expressions to be examined are independent on time, then the equations (A.2) simplify and in particular the electric and magnetic fields are decoupled.

Electrostatic equations

Since $\mathrm{rot}\,\vec{E} = \vec{0}$ we know that the electric field has a scalar potential function U, such that

$$\vec{E} = -\nabla U$$

This can be combined with (A.1) and we find the basic electrostatic equation

$$-\,\mathrm{div}\,(\varepsilon\,\varepsilon_0\,\nabla\,U) = \rho \tag{A.3}$$

This is a second order differential equation for the electric potential U

Magnetostatic equations

Similarly we use $\mathrm{rot}\,\vec{H} = \vec{0}$ we know that the magnetic field has a scalar potential function U_m, such that

$$\vec{H} = -\nabla U_m$$

This can be combined with (A.1) and we find the basic electrostatic equation

$$-\,\mathrm{div}\,(\nu\,\nu_0\,\nabla\,U_m) = 0 \tag{A.4}$$

This is a second order differential equation for the magnetic potential U_m

A.6.3 Time-harmonic fields

Assume that all fields depend on time with a factor $e^{i\omega t}$, using complex notation.

$$\begin{aligned} \mathrm{div}\,\varepsilon\,\varepsilon_0\,\vec{E} &= i\,\omega\,\rho \\ \mathrm{div}\,\nu\,\nu_0\,\vec{H} &= 0 \\ \mathrm{rot}\,\vec{E} &= -i\,\omega\,\nu\,\nu_0\,\vec{H} \\ \mathrm{rot}\,\vec{H} &= i\,\omega\,\vec{\varepsilon}\,\varepsilon_0\,\vec{E} \end{aligned} \tag{A.5}$$

Appendix B

Solutions to some exercises

Solution to Exercise 1–1 :

(a)

$$f\,(x_1, x_2) = \frac{1}{2}\,\langle \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\,,\, \begin{bmatrix} 2 & 4 \\ 4 & -4 \end{bmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \rangle + \langle \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\,,\, \begin{pmatrix} 3 \\ 6 \end{pmatrix} \rangle$$

(b) Solve the equation

$$\begin{bmatrix} 2 & 4 \\ 4 & -4 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = - \begin{pmatrix} 3 \\ 6 \end{pmatrix}$$

with the unique solution $x_1 = \frac{-3}{2}$ and $x_2 = 0$.

(c) The eigenvalues and eigenvectors are given by

$$\vec{e}_1 = \begin{pmatrix} -1 \\ 2 \end{pmatrix} \quad , \quad \lambda_1 = -6 \quad \text{and} \quad \vec{e}_2 = \begin{pmatrix} 2 \\ 1 \end{pmatrix} \quad , \quad \lambda_1 = 4$$

Use `eigensystem[]` in *Mathematica*. Since one of the eigenvalues is positive and the other is negative we have a saddle point.

(d) With *Mathematica* one may use the following code.

Mathematica

```
a={{2,4},{4,-4}};
f[x_] := x.a.x /2 + x.{3,6}
f[{x1,x2}]//Expand
Plot3D[f[{x1,x2}],{x1,-4,1},{x2,-2,2}];
```

Solution to Exercise 1–2 : The location (x_m, y_m) of the minimum is given by the solution of

$$\begin{bmatrix} 8 & -2 \\ -2 & 6 \end{bmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} = - \begin{pmatrix} b \\ -2\,b \end{pmatrix} = b \begin{pmatrix} -1 \\ 2 \end{pmatrix}$$

and thus

$$\begin{pmatrix} x_m \\ y_m \end{pmatrix} = b \begin{pmatrix} \frac{-1}{2} \\ \frac{7}{22} \end{pmatrix}$$

Solution to Exercise 2–1 :

(a) $F = A(x)\,\sigma(x)$ and thus $\sigma(x) = F/A(x)$.

(b) Hooke's law implies

$$\varepsilon(x) = \frac{\sigma(x)}{E} = \frac{F}{E\,A(x)}$$

An integration of the energy density $e(x)$ yields

$$\begin{aligned} U &= \int_0^L e(x)\,A(x)\,dx = \int_0^L \frac{1}{2}\,\sigma(x)\cdot\varepsilon(x)\;A(x)\,dx \\ &= \frac{1}{2}\int_0^L \frac{\varepsilon(x)^2}{E}\;A(x)\,dx = \frac{1}{2}\int_0^L \frac{F^2}{E\,A(x)^2}\;A(x)\,dx \\ &= \frac{F^2}{2\,E}\int_0^L \frac{1}{A(x)}\,dx \end{aligned}$$

Another solution could be based on

$$U = \int_0^L \varepsilon(x)\,A(x)\,dx = \int_0^L \frac{1}{2}\,\sigma(x)\cdot\varepsilon(x)\;A(x)\,dx = \frac{E}{2}\int_0^L \varepsilon(x)^2\;A(x)\,dx$$

The final result will not change.

(c) For the displacement $u(x)$ we know that $\varepsilon(x) = \frac{d}{dx}\,u(x)$ and thus

$$u(x) = u(0) + \int_0^x u'(s)\,ds = 0 + \int_0^x \varepsilon(s)\,ds$$

This leads to

$$\Delta L = u(L) = \int_0^L \varepsilon(s)\,ds = \int_0^L \frac{F}{E\,A(s)}\,ds$$

(d)

$$\frac{1}{k} = \frac{\Delta L}{F} = \int_0^L \frac{1}{E\,A(s)}\,ds$$

Solution to Exercise 3–1 :

(a) This is a standard calculus problem.

(b) As the integrand does not depend explicitly on x we have a first integral. This is simpler than starting with the Euler Lagrange equation.

$$u'\,f_{u'} - f = u'\,\frac{u\,u'}{\sqrt{1+(u')^2}} - u\,\sqrt{1+(u')^2} = C$$

(c) Now we can solve for u' and then do a separation of variables.

$$\begin{aligned} u\,(u')^2 - u\,(1+(u')^2) &= C\,\sqrt{1+(u')^2} \\ -u &= C\,\sqrt{1+(u')^2} \\ u' &= \sqrt{\frac{u^2-C^2}{C^2}} \\ dx &= \frac{C}{\sqrt{u^2-C^2}}\,du \\ \int 1\,dx &= \int \frac{C}{\sqrt{u^2-C^2}}\,du = C\,\cosh^{-1}\frac{u}{C} + k \end{aligned}$$

Thus we have

$$y = u(x) = C\,\cosh\frac{x-k}{C}$$

Solution to Exercise 3–2 :

(a)

$$E(u) = \int_a^b e(x)\, A(x)\, dx = \frac{E}{2} \int_a^b A(x)\, u'(x)^2\, dx$$

As we have

$$f(x, u, u') = \frac{E}{2}\, A(x)\, (u')^2$$

the Euler Lagrange equation simplifies to

$$\frac{d}{dx}\, f_{u'} = E\, \frac{d}{dx}\, \big(A(x)\, u'(x)\big) = 0$$

Thus the expression $A(x)\, u'(x)$ is a first integral. The second order differential equation is

$$A(x)\, u'' + A'(x)\, u'(x) = 0$$

(b) In this example we have $a = 0$, $b = 100$ and $A(x) = 10 - 0.09\, x$. We use the first integral

$$\begin{aligned} A(x)\, u'(x) &= (10 - 0.09\, x)\, u'(x) = C_1 \\ u'(x) &= \frac{C_1}{10 - 0.09\, x} \\ u(x) &= u(0) + \int_0^x \frac{C_1}{10 - 0.09\, z}\, dz = \frac{C_1}{0.09}\, (\ln 10 - \ln(10 - 0.09\, x)) \\ &= \frac{C_1}{0.09}\, \ln \frac{10}{10 - 0.09\, x} \end{aligned}$$

With the help of the condition $u(100) = B$ we determine the constant C_1.

$$B = \frac{C_1}{0.09}\, \ln 10 \qquad \Longrightarrow \qquad C_1 = \frac{0.09\, B}{\ln 10}$$

Now we have the solution

$$u(x) = \frac{B}{\ln 10}\, \ln \frac{10}{10 - 0.09\, x} = \frac{-B}{\ln 10}\, \ln \frac{10 - 0.09\, x}{10}$$

(c) To find the physical interpretation of the first integral use Hooke'ss law

$$\sigma = E\, \varepsilon = E\, u'$$

and thus the first integral

$$C_1 = A(x) \cdot u'(x) \qquad \Longrightarrow \qquad E\, C_1 = E\, A(x) \cdot \varepsilon(x) = A(x) \cdot \sigma(x) = F(x)$$

implies that the total force applied to each cross section has to be independent on x.

(d) We have

$$\varepsilon(100) = \frac{d\, u(x)}{dx}\Big|_{x=100} = \frac{B}{\ln 10}\, \frac{0.09}{10 - 0.09\, x}\Big|_{x=100} = \frac{0.09}{\ln 10}\, B$$

Thus the stress is given by

$$\sigma(100) = E\, \varepsilon(100) = E\, \frac{0.09}{\ln 10}\, B$$

and

$$F = A(100) \cdot \sigma(100) = 3 \cdot 10^6\, \frac{0.09}{\ln 10}\, B = 2 \cdot 10^4$$

This leads to $B \approx 0.17056$ and to the exact solution shown in section 2.2.2.

Solution to Exercise 3–3 : The functional to be considered is

$$L(y,z) = \int_a^b \sqrt{1+(y')^2+(z')^2}\, dz$$

The Euler Lagrange equation for the dependent variables $y(x)$ and $z(x)$ are

$$\begin{aligned} \frac{d}{dx}\, f_y &= f_y \\ \frac{d}{dx}\, \frac{y'}{\sqrt{1+(y')^2+(z')^2}} &= 0 \\ \frac{d}{dx}\, \frac{z'}{\sqrt{1+(y')^2+(z')^2}} &= 0 \end{aligned}$$

If y' and z' are constant, then those two equations are certainly solved. We have to verify that there are no other solutions. The equations implies

$$y' = C_1\, \sqrt{1+(y')^2+(z')^2} \quad \text{and} \quad z' = C_2\, \sqrt{1+(y')^2+(z')^2}$$

Thus z' can be written as a multiple of y' ($z' = \lambda y'$) and we conclude

$$y' = C_1\, \sqrt{1+(1+\lambda^2)\,(y')^2}$$

The only way to solve this equation is by a y' being constant.

Solution to Exercise 3–4 : This is an extremal problem with constraint, thus we consider the functional

$$F(u) = A + \lambda\, L = \int_0^b u(x) + \lambda\, \sqrt{1+(u'(x))^2}\, dx$$

with the unknown Lagrange multiplier $\lambda \in \mathbb{R}$. The Euler Lagrange equation is given by

$$\begin{aligned} \frac{d}{dx}\, f_{u'} &= f_u \\ \frac{d}{dx}\, \frac{\lambda\, u'(x)}{\sqrt{1+(u'(x))^2}} &= 1 \\ \frac{u''(x)}{\sqrt{1+(u'(x))^2}} - \frac{(u'(x))^2\, u''(x)}{\sqrt{1+(u'(x))^2{}^3}} &= \frac{1}{\lambda} \\ \frac{u''(x)}{\sqrt{1+(u'(x))^2{}^3}} &= \frac{1}{\lambda} \end{aligned}$$

Thus the curvature κ is constant along the optimal solution. The natural boundary condition at $x = b$ is

$$f_{u'} = \left. \frac{\lambda\, u'(x)}{\sqrt{1+(u'(x))^2}} \right|_{x=b} = 0$$

Thus we conclude that $u'(b) = 0$, i.e. we have a horizontal tangent line.

Solutions of the above form will only exist if $b < L < \frac{b\,\pi}{2}$.

Solution to Exercise 3–5 :

(a) Let ρ be the specific mass of water. Use cylindrical coordinates for the integrations.

$$\begin{aligned} V(u) &= \int_0^R 2\,\pi\,r\,u(r)\;dr \\ U(u) &= \int_0^R \rho\,g\,2\,\pi\,r\,u(r)\,\frac{u(r)}{2}\;dr \\ T(u) &= \int_0^R \rho\,2\,\pi\,r\,u(r)\,\frac{\omega^2\,r^2}{2}\;dr \\ L(u) &= T(u)-U(u) = \rho\,\pi\int_0^R r^3\,\omega^2\,u(r) - g\,r\,u^2(r)\;dr \end{aligned}$$

(b) Consider the functional

$$L(u)-\lambda V(u) = \pi\int_0^R \rho\,(r^3\,\omega^2\,u(r) - g\,r\,u^2(r)) + \lambda\,2\,u(r)\;dr$$

If $u(r)$ is a critical point then

$$\begin{aligned} \frac{d}{dr}\,f_{u'} &= f_u \\ 0 &= \rho\,\pi(r^3\,\omega^2 - g\,r\,u(r)) - \lambda\,2\,\pi\,r \\ -r^2\,\omega^2 + g\,u(r) &= \frac{2\,\lambda}{\rho} \\ u(r) &= \frac{2\,\lambda}{\rho\,g} + \frac{1}{\omega^2\,g}\,r^2 = c + \frac{1}{\omega^2\,g}\,r^2 \end{aligned}$$

Thus the surface will have the shape of a rotated parabola. The unknown constant c can be determined using the known volume V_0 of water.

Solution to Exercise 3–6 :

$$\begin{aligned} F(u+\phi)-F(u) &\approx \int_a^b a(x)\,u'(x)\cdot\phi'(x) + b(x)\,u(x)\cdot\phi(x) + g(x)\cdot\phi(x)\;dx \\ &\quad + r(a)\,\phi(a) - r(b)\,\phi(b) \\ &= \int_a^b -\frac{d}{dx}\big(a(x)\,u'(x)\big)\cdot\phi(x) + b(x)\,u(x)\cdot\phi(x) + g(x)\cdot\phi(x)\;dx \\ &\quad + a(x)\,u'(x)\cdot\phi(x)\big|_{x=a}^b + r(a)\,\phi(a) - r(b)\,\phi(b) \\ &= \int_a^b \left(-\frac{d}{dx}\big(a(x)\,u'(x)\big) + b(x)\,u(x) + g(x)\right)\cdot\phi(x)\;dx \\ &\quad + \big(a(x)\,u'(x) - r(x)\big)\cdot\phi(x)\big|_{x=a}^b \end{aligned}$$

If the function $u(x)$ is a minimiser then the above expression has to vanish for „arbitrary“ function $\phi(x)$. This implies the claimed result.

Solution to Exercise 3–7 : Consider the perturbed function $u(x)+\varepsilon\,\eta(x)$ and compute the functional. As a necessary condition the derivative with respect to ε has to vanish for $\varepsilon = 0$. Two subsequent integration by parts lead to the following.

$$F(u+\varepsilon\,\eta) = \frac{1}{2}\int_a^b A(x)\,\big(u''(x)+\varepsilon\,\eta''(x)\big)^2\;dx$$

$$\begin{aligned}
\frac{d}{d\varepsilon}\, F(u+\varepsilon\,\eta) &= \frac{1}{2}\int_a^b A(x)\,2\,\left(u''(x)+\varepsilon\,\eta''(x)\right)\,\eta''(x)\,dx \\
\frac{d}{d\varepsilon}\, F(u+\varepsilon\,\eta)\Big|_{\varepsilon=0} &= \int_a^b \eta''(x)\,A(x)\,u''(x)\,dx \\
&= \eta'(x)\,A(x)\,u''(x)\Big|_{x=a}^{b} - \int_a^b \eta'(x)\,\frac{d}{dx}\,\left(A(x)\,u''(x)\right)dx \\
&= \left(\eta'(x)\,A(x)\,u''(x) - \eta(x)\,\frac{d}{dx}\left(A(x)\,u''(x)\right)\right)\Big|_{x=a}^{b} + \\
&\quad + \int_a^b \eta(x)\,\frac{d^2}{dx^2}\,\left(A(x)\,u''(x)\right)dx
\end{aligned}$$

The fundamental lemma implies now

$$\frac{d^2}{dx^2}\,\left(A(x)\,u''(x)\right) = 0 \quad \text{for} \quad a < x < b$$

and since the values of η and η' at the endpoints are independent we also have the four boundary conditions

$$\begin{aligned}
\eta'(a)\,A(a)\,u''(a) &= 0 \\
\eta(a)\,\frac{d}{dx}\left(A(x)\,u''(x)\right)\Big|\,x=a &= 0 \\
\eta'(b)\,A(b)\,u''(b) &= 0 \\
\eta(b)\,\frac{d}{dx}\left(A(x)\,u''(x)\right)\Big|\,x=b &= 0
\end{aligned}$$

Depending on the constraints imposed on the variable we find boundary conditions on the solution $u(x)$.

- If the bar is fixed and clamped at both ends then the test function η at is derivative have to vanish at the endpoints, $\eta(a) = \eta'(a) = \eta(b) = \eta'(b) = 0$. The known values of u and u' serve as boundary conditions.
- If the bar is fixed and clamped at the left end and free at the right end then $\eta(a) = \eta'(a) = 0$. Since the values of η are free at $x = b$ the conditions above imply

$$A(b)\,u''(b) = 0 \quad \text{and} \quad \frac{d}{dx}\left(A(b)\,u''(b)\right) = 0$$

In both situations four boundary conditions for the ordinary differential equation of order four are given.

Solution to Exercise 4–1 :

Mathematica

```
<<BVP.m
Clear[a,b,c,f,x,n,yn,y,t]
a=Function[x,x];
b=Function[x,-7];
f=Function[x,-Sin[x]];
n=10;
x=Table[t,{t,0,3,3/n}];
data=BVP[a,b,f,x,{"D","N"},{3,0}];
g1=ListPlot[data,PlotStyle -> PointSize[0.02],
            DisplayFunction ->Identity] ;
yn[t_]=y[t]/.NDSolve[ {D[a[t]*y'[t],t]-b[t]*y[t]==f[t],
                       y[data[[2,1]]]==data[[2,2]],y'[3]==0},
                     y[t],{t,0.01,3}];
g2=Plot[yn[t],{t,data[[2,1]],3},DisplayFunction -> Identity];
Show[{g1,g2}, PlotRange -> All,
     DisplayFunction ->$DisplayFunction];
```

Solution to Exercise 7–1 : The first two points are answered by the explicit computations

$$\begin{aligned}
\mathbf{A}_\Delta &= \frac{1}{4\,A^2}\begin{bmatrix} (y_3-y_2) & (x_2-x_3) \\ (y_1-y_3) & (x_3-x_1) \\ (y_2-y_1) & (x_1-x_2) \end{bmatrix} \cdot \begin{bmatrix} (y_3-y_2) & (y_1-y_3) & (y_2-y_1) \\ (x_2-x_3) & (x_3-x_1) & (x_1-x_2) \end{bmatrix} \\
&= \frac{1}{4\,A^2}\begin{bmatrix} (y_3-y_2) & (x_2-x_3) \\ -y_3 & x_3 \\ y_2 & -x_2 \end{bmatrix} \cdot \begin{bmatrix} (y_3-y_2) & -y_3 & y_2 \\ (x_2-x_3) & x_3 & -x_2 \end{bmatrix} \\
&= \frac{1}{4\,A^2}\begin{bmatrix} (y_3-y_2)^2+(x_2-x_3)^2 & (x_2-x_3)\,x_3-(y_3-y_2)\,y_3 & (y_3-y_2)\,y_2-(x_2-x_3)\,x_2 \\ (x_2-x_3)\,x_3-(y_3-y_2)\,y_3 & y_3^2+x_3^2 & -y_3\,y_2-x_3\,x_2 \\ (y_3-y_2)\,y_2-(x_2-x_3)\,x_2 & -y_2\,y_3-x_2\,x_3 & y_2^2+x_2^2 \end{bmatrix}
\end{aligned}$$

(a) The matrix is symmetric and all diagonal entries are positive as sum of squares.

(b) The matrix $\mathbf{A}_\Delta$ is positive semidefinite since

$$\langle \vec{u}\,,\, \mathbf{A}_\Delta \cdot \vec{u}\rangle = \frac{1}{4\,A^2}\,\langle \vec{u}\,,\, \mathbf{M}^T\cdot\mathbf{M}\cdot\vec{u}\rangle = \frac{1}{4\,A^2}\,\langle \mathbf{M}\cdot\vec{u}\,,\, \mathbf{M}\cdot\vec{u}\rangle = \frac{1}{4\,A^2}\,\|\mathbf{M}\cdot\vec{u}\|^2 \geq 0$$

(c) The above expression equals zero if and only if $\mathbf{M}\cdot\vec{u}=\vec{0}$. This is equivalent to the system

$$\begin{array}{rrrcl} (y_3-y_2)\,u_1 & -y_3\,u_2 & +y_2\,u_3 & = & 0 \\ (x_2-x_3)\,u_1 & +x_3\,u_2 & -x_2\,u_3 & = & 0 \end{array} \quad \text{or} \quad \begin{array}{rrrcl} & -y_3\,u_2 & +y_2\,u_3 & = & -(y_3-y_2)\,u_1 \\ & +x_3\,u_2 & -x_2\,u_3 & = & -(x_2-x_3)\,u_1 \end{array}$$

Multiply the first equation by x_2, the second by y_2, then add the two equations to arrive at

$$u_2\,(-y_3\,x_2+x_3\,y_2) = -x_2\,(y_3-y_2)\,u_1 - y_2\,(x_2-x_3)\,u_1 = (-x_2\,y_3+y_2\,x_3)\,u_1$$

Since the triangle with the corners $(0,0)$, $(x_1,y_1)^t$ and $(x_2,y_2)^T$ has nonzero area we conclude that $-x_2\,y_3+y_2\,x_3\neq 0$ and thus $u_1=u_2$. Similarly we find that $u_3=u_1$.

There is an geometric interpretation for this result. Equation (7.2) implies

$$\nabla u = \begin{pmatrix} \frac{\partial\,u}{\partial x} \\ \frac{\partial\,u}{\partial y} \end{pmatrix} = \frac{-1}{2\,A}\,\mathbf{M}\cdot\begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix}$$

The gradient for a linear (resp. affine) function vanishes if and only if the function is constant, i.e. $u_1 = u_2 = u_3$.

Solution to Exercise 11–2 :

(a) The result is implied by

$$\max_{\vec{x}\neq\vec{0}}\kappa(\vec{x}) = \max_{\vec{x}\neq\vec{0}}\|\mathbf{A}^{-1}\|\,\frac{\|\mathbf{A}\,\vec{x}\|}{\|\vec{x}\|} = \|\mathbf{A}^{-1}\|\,\max_{\vec{x}\neq\vec{0}}\frac{\|\mathbf{A}\,\vec{x}\|}{\|\vec{x}\|} = \|\mathbf{A}^{-1}\|\,\|\mathbf{A}\| = \kappa$$

(b) The proof is similar to the proof of 11–4 or by the calculation below

$$\frac{\|\Delta\vec{x}\|}{\|\vec{x}\|} = \frac{\|\mathbf{A}^{-1}\,\Delta\vec{b}\|}{\|\vec{x}\|} = \frac{\|\mathbf{A}^{-1}\,\Delta\vec{b}\|}{\|\Delta\vec{b}\|}\,\frac{\|\Delta\vec{b}\|}{\|\mathbf{A}\,\vec{x}\|}\,\frac{\|\mathbf{A}\,\vec{x}\|}{\|\vec{x}\|} \leq \|\mathbf{A}^{-1}\|\,\frac{\|\mathbf{A}\,\vec{x}\|}{\|\vec{x}\|}\,\frac{\|\Delta\vec{b}\|}{\|\vec{b}\|} = \kappa(\vec{x})\,\frac{\|\Delta\vec{b}\|}{\|\vec{b}\|}$$

(c) Let $\vec{e}_k$ be the ortho-normalized eigenvectors to the eigenvalues λ_k. If $\vec{b} = \sum_k c_k \, \vec{e}_k$ and $\mathbf{A}\,\vec{x} = \vec{b}$ then $\vec{x} = \sum_k \frac{1}{\lambda_k} \, c_k \, \vec{e}_k$. For the norms we find

$$\begin{aligned} \|\mathbf{A}\,\vec{x}\|^2 &= \|\vec{b}\|^2 = \sum_k |c_k|^2 \\ \|\vec{x}\|^2 &= \sum_k \frac{1}{\lambda_k^2} \, |c_k|^2 \geq \sum_k \frac{1}{\Lambda^2} \, |c_k|^2 = \frac{1}{\Lambda^2} \, \|\vec{b}\|^2 = \frac{1}{\Lambda^2} \, \|\mathbf{A}\,\vec{b}\|^2 \end{aligned}$$

and thus $\|\mathbf{A}\,\vec{b}\| \leq \Lambda \, \|\vec{x}\|$. Since $\|\mathbf{A}^{-1}\| = 1/\lambda_1$ this implies

$$\kappa(\vec{x}) = \|\mathbf{A}^{-1}\| \frac{\|\mathbf{A}\,\vec{x}\|}{\|\vec{x}\|} \leq \frac{1}{\lambda_1} \, \Lambda$$

Solution to Exercise 11–3 : Since $\mathbf{Q}^{-1} = \mathbf{Q}^T$ we know that $\mathbf{Q} \cdot \mathbf{Q}^T = \mathbf{Q}^T \cdot \mathbf{Q} = \mathbb{I}$ and thus

$$\|\mathbf{Q}\,\vec{x}\|^2 = \langle \mathbf{Q}\,\vec{x}\,,\, \mathbf{Q}\,\vec{x} \rangle = \langle \vec{x}\,,\, \mathbf{Q}^T \cdot \mathbf{Q}\,\vec{x} \rangle = \langle \vec{x}\,,\, \vec{x} \rangle = \|\vec{x}\|^2$$

This implies $\|\mathbf{Q}\| = 1$ and by similar calculations we obtain $\|\mathbf{Q}^{-1}\| = \|\mathbf{Q}^T\| = 1$. Now $\kappa = 1$ follows from the definition of the condition number.

Solution to Exercise 11–4 : The factorization is given by

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{-1}{2} & 1 & 0 & 0 \\ 0 & \frac{-2}{3} & 1 & 0 \\ 0 & 0 & \frac{-3}{4} & 1 \end{bmatrix} \cdot \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & \frac{3}{2} & 0 & 0 \\ 0 & 0 & \frac{4}{3} & 0 \\ 0 & 0 & 0 & \frac{5}{4} \end{bmatrix} \cdot \begin{bmatrix} 1 & \frac{-1}{2} & 0 & 0 \\ 0 & 1 & \frac{-2}{3} & 0 \\ 0 & 0 & 1 & \frac{-3}{4} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Solution to Exercise 11–5 : We only have to verify that the diagonal entries remain positive as we reduce the matrix to diagonal form. The first step of the Cholesky algorithm leads to

$$\begin{bmatrix} 1-\nu & \nu & \nu \\ \nu & 1-\nu & \nu \\ \nu & \nu & 1-\nu \end{bmatrix} \longrightarrow \begin{bmatrix} 1-\nu & 0 & 0 \\ 0 & 1-\nu-\frac{\nu^2}{1-\nu} & \nu-\frac{\nu^2}{1-\nu} \\ 0 & \nu-\frac{\nu^2}{1-\nu} & 1-\nu-\frac{\nu}{1-\nu} \end{bmatrix} = \begin{bmatrix} 1-\nu & 0 & 0 \\ 0 & \frac{1-2\,\nu}{1-\nu} & \frac{\nu-2\,\nu^2}{1-\nu} \\ 0 & \frac{\nu-2\,\nu^2}{1-\nu} & \frac{1-2\,\nu}{1-\nu} \end{bmatrix}$$

Since the next step modifies the lower right block only we restrict our attention to that block. If $\nu > \frac{1}{2}$ then the diagonal numbers are negative and the matrix is not positive definite. Multiplying the matrix by the positive factor $\frac{1-\nu}{1-2\,\nu}$ (use $0 \leq \nu \leq \frac{1}{2}$) does not influence positive definiteness. Thus we now have to consider only the smaller matrix and apply another step of the Cholesky algorithm.

$$\begin{bmatrix} 1 & \nu \\ \nu & 1 \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 0 \\ 0 & 1-\nu^2 \end{bmatrix}$$

This matrix is positive definite if $\nu^2 \leq \frac{1}{4}$.

Solution to Exercise 11–7 :

(a) The first five steps of the power iteration lead to the values

$$0.63246 \quad , \quad 2.2361 \quad , \quad 2.9326 \quad , \quad 3.3860 \quad \text{and} \quad 3.6151$$

(b) First make sure the vector $\vec{x}$ is normalized. Then the error has to be smaller that $\|\mathbf{A}\,\vec{x} - \beta\,\vec{x}\| \approx 0.28707$. Since $\lambda_5 = 3.7321$ this is the case.

(c) The Rayleigh quotient is given by

$$\rho(\vec{x}) = \langle \vec{x}\,,\, \mathbf{A}\,\vec{x} \rangle = \vec{x}^T \cdot \mathbf{A} \cdot \vec{x} \approx 3.6862$$

and is in fact closer to the exact eigenvalue.

The result can be generated by code similar to the one below.

Octave

```
n = 5;
A = diag(2*ones(1,n))-diag(ones(1,n-1),1)-diag(ones(1,n-1),-1);
x = ones(n,1);
for k=1:5
  x = x/norm(x);
  x = A*x;
  eigval = norm(x)
endfor

error = norm(A*x-eigval*x)/norm(x)

rayleigh = x'*A*x/norm(x)**2

eigvalexact= 2*(1-cos(5*pi/6))
```

Solution to Exercise 11–8 :

(a) The ratio $\frac{\lambda_{10}}{\lambda_9} \approx 1.06$ of the two largest eigenvalues is rather close to 1 and thus the convergence is slow. In addition is the constant initial vector orthogonal to the eigenvector belonging to the largest eigenvalue and thus the iteration will converge to the second largest eigenvalue. The à posteriori bound estimates the distance to one of the eigenvalues, not necessarily the largest. In this case it is the second largest eigenvalue.

(b) Both critical points of the first attempt are eliminated by the modification.

Solution to Exercise 11–9 : The basis for the estimates is the à priori bound

$$|\beta - \lambda| \leq c \left(\frac{\lambda_m}{\lambda_{m+1}} \right)^{2k}$$

for the error of the first m eigenvalues eigenvalues.

(a) The convergence is determined by $\frac{\lambda_1}{\lambda_2} \approx \frac{1}{4}$. Assuming that we start the iteration with zero correct digits we find the relative error to be $\left(\frac{1}{4}\right)^{2k}$. This leads to the condition $4^{2k} \geq 10^4$ and thus $k \geq \frac{\ln 10^4}{2\,\ln 4} \approx 3.3$. We need at least 4 iterations.

(b) The convergence is determined by $\frac{\lambda_5}{\lambda_6} \approx \frac{25}{36}$. Assuming that we start the iteration with zero correct digits we find the relative error to be $\left(\frac{25}{36}\right)^{2k}$. This leads to $k \geq \frac{\ln 10^4}{2\,(\ln 36 - \ln 25)} \approx 12.6$. We need at least 13 iterations.

(c) The error bound contains a constant c which we do not know. The above computations assumed $c = 1$. It is more reliable to run a few iterations and then keep track of which digits of the approximated λ_i keep changing. If 4 digits remain unchanged for a few iterations one may stop. One could also use the à posteriori bounds of theorem 11–25, but the computational effort is considerably larger. It is a good idea though to verify the final results using theorem 11–25.

Solution to Exercise 11–11 :

(a) Since $\mathbf{B}$ is strictly positive definite its Cholesky factorization $\mathbf{B} = \mathbf{R}^T \cdot \mathbf{R} = \mathbf{L} \cdot \mathbf{R}$ is invertible and we find $\tilde{\mathbf{A}} = \mathbf{L}^{-1} \cdot \mathbf{A} \cdot \mathbf{R}^{-1}$. Thus we conclude

$$\begin{aligned} \mathbf{A}\,\vec{v} = \lambda\,\mathbf{B}\,\vec{v} &\iff \mathbf{L}^{-1} \cdot \mathbf{A}\,\vec{v} = \lambda\,\mathbf{L}^{-1} \cdot \mathbf{L} \cdot \mathbf{R}\,\vec{v} \\ &\iff \mathbf{L}^{-1} \cdot \mathbf{A} \cdot \mathbf{R}^{-1}\,\vec{y} = \lambda\,\mathbf{R} \cdot \mathbf{R}^{-1}\,\vec{y} \quad \text{where} \quad \vec{y} = \mathbf{R}\,\vec{v} \\ &\iff \tilde{\mathbf{A}}\,\vec{y} = \lambda\,\vec{y} \end{aligned}$$

(b) Since $\mathbf{B} = \mathbf{L}\,\mathbf{R}$ we use $\mathbf{B}^{-1} = \mathbf{R}^{-1}\,\mathbf{L}^{-1}$ to conclude

$$\langle \vec{r},\, \mathbf{B}^{-1}\,\vec{r} \rangle = \langle \vec{r},\, \mathbf{R}^{-1} \cdot \mathbf{L}^{-1}\,\vec{r} \rangle = \langle \mathbf{L}^{-1}\vec{r},\, \mathbf{L}^{-1}\,\vec{r} \rangle$$

Using the definitions of the residual vectors $\vec{r}$ and $\vec{s}$ we find

$$\begin{aligned} \mathbf{L}^{-1}\vec{r} &= \mathbf{L}^{-1}\,\left(\mathbf{A}\,\vec{v} - \lambda\,\mathbf{B}\,\vec{v}\right) \\ &= \mathbf{L}^{-1} \cdot \mathbf{A}\,\vec{v} - \lambda\,\mathbf{L}^{-1} \cdot \mathbf{B}\,\vec{v} \\ &= \mathbf{L}^{-1} \cdot \mathbf{A} \cdot \mathbf{R}^{-1}\,\vec{y} - \lambda\,\mathbf{L}^{-1} \cdot \mathbf{L} \cdot \mathbf{R} \cdot \mathbf{R}^{-1}\,\vec{y} \\ &= \tilde{\mathbf{A}}\,\vec{y} - \lambda\,\vec{y} = \vec{s} \end{aligned}$$

Thus we have the desired identity

$$\langle \vec{r},\, \mathbf{B}^{-1}\,\vec{r} \rangle = \langle \vec{s},\, \vec{s} \rangle$$

(c) Let $\vec{y} = \mathbf{R}\,\vec{v}$. Since $\vec{v}$ is normalized, we find

$$1 = \langle \vec{v},\, \mathbf{B}\,\vec{v} \rangle = \langle \mathbf{R}^{-1}\vec{y},\, \mathbf{R}^T\,\mathbf{R}\,\mathbf{R}^{-1}\vec{y} \rangle = \langle \vec{y},\, \vec{y} \rangle$$

Now use the first estimate in result 11–25 for the standard eigenvalue problem with matrix $\tilde{\mathbf{A}}$ to conlude

$$\min(|\lambda_i - \beta|) \leq \|\vec{s}\| = \sqrt{\langle \vec{s},\, \vec{s} \rangle} = \sqrt{\langle \vec{r},\, \mathbf{B}^{-1}\,\vec{r} \rangle}$$

The second estimate follows similarly.

Bibliography

[Axel94] O. Axelsson. *Iterative Solution Methods*. Cambridge University Press, 1994.

[AxelBark84] O. Axelsson and V. A. Barker. *Finite Element Solution of Boundaru Values Problems*. Academic Press, 1984.

[Bent00] J. Bentley. *Programming Pearls*. Addison Wesley, second edition, 2000.

[Ciar02] P. G. Ciarlet. *The Finite Element Method for Elliptic Problems*. SIAM, 2002.

[Davi80] A. J. Davies. *The Finite Element Method: a First Approach*. Oxford University Press, 1980.

[Demm97] J. W. Demmel. *Applied Numerical Linear Algebra*. SIAM, Philadelphia, 1997.

[Demm75] G. Demmig. *Matrizen und Determinanten*. Demmig Verlag, 1975.

[Hart52] J. P. den Hartog. *Advanced Strength of Materials*. McGraw–Hill, 1952. republished by Dover 1987.

[DowdSeve98] K. Dowd and C. Severance. *High Performance Computing*. O'Reilly, 2nd edition, 1998.

[Gawe93] W. Gawehn. *Vektor– und Matrizenalgebra für Maschinenbauer*. Bibliographisches Institut, Wissenschaftsverlag, 1993.

[GoluVanLoan96] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, third edition, 1996.

[Gree77] D. T. Greenwood. *Classical Dynamics*. Prentice Hall, 1977. Dover edition 1997.

[GuenLee96] R. B. Guenther and J. W. Lee. *Partial Differntial Equations of Mathematical Physics and Integral Equations*. Dover, 1996.

[High96] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM Publications, Philadelphia, 1996.

[HildTrom86] S. Hildebrand and A. Tromba. *Panoptimum, Mathematische Grundmuster des Vollkommenen*. Spektrum der Wissenschaften, 1986.

[IsaaKell66] E. Isaacson and H. B. Keller. *Analysis of Numerical Methods*. John Wiley & Sons, 1966. republished by Dover in 1994.

[Isen78] C. Isenberg. *The Science of Soap Films and Soap Bubbles*. Tieto Ldt., 1978. republished by Dover in 1992.

[John87] C. Johnson. *Numerical Solution of Partial Differential Equations by the Finite Element Method*. Cambridge University Press, 1987.

[Kell75] H. B. Keller. Approximation methods for nonlinear problems with applications to two–point boundary value problems. In *Mathematics of Computations* [Kell92], pages 464–474.

[Kell92] H. B. Keller. *Numerical Methods for Two–Point Boundary Value Problems*. Dover, 1992.

[KnabAnge00] P. Knabner and L. Angermann. *Numerik partieller Differentialgleichungen*. Springer Verlag, Berlin, 2000.

[LandLifs75] L. D. Landau and E. M. Lifshitz. *Lehrbuch der Theoretischen Physik, Band VII, Elastizitätsthreorie*. Akademie Verlag, Berlin, 1975.

[LascTheo87] P. Lascaux and R. Théodor. *Analyse numérique matricielle appliquée a l'art de l'ingénieur, Tome 2*. Masson, Paris, 1987.

[CRC95] D. R. Linde. *CRC Handbook of Chemistry and Physics*. CRC Press, 1995.

[PDEToolbox95] Mathworks Staff. *Partial Differential Equation Toolbox*. The Mathworks, 1995.

[MullGrot97] G. Müller and C. Groth. *FEM für Praktiker*. expert verlag, 1997.

[OttoPete92] N. S. Ottosen and H. Petersson. *Introduction to the Finite Element Method*. Prentice Hall, 1992.

[Prze68] J. Przemieniecki. *Theory of Matrix Structural Analysis*. McGraw–Hill, 1968.

[Redd84] J. N. Reddy. *An Introduction to the Finite Element Analysis*. McGraw–Hill, 1984.

[www:triangle] J. R. Shewchuk. http://www-2.cs.cmu.edu/~quake/triangle.html.

[Smit84] G. D. Smith. *Numerical Solution of Partial Differential Equations: Finite Difference Methods*. Oxford Univerity Press, Oxford, third edition, 1986.

[Sout73] R. W. Soutas-Little. *Elasticity*. Prentice–Hall, 1973.

[www:sha] A. Stahel. Web page. www.hta-bi.bfh.ch/~sha.

[StraFix73] G. Strang and G. J. Fix. *An Analysis of the Finite Element Method*. Prentice–Hall, 1973.

[Wein74] R. Weinstock. *Calculus of Variations*. Dover, New York, 1974.

[Wilk63] J. H. Wilkinson. *Rounding Errors in Algebraic Processes*. Prentice-Hall, 1963. republished by Dover in 1994.

[Wlok82] J. Wloka. *Partielle Differentialgleichungen*. Teubner, Stuttgart, 1982.

List of Figures

List of Tables

Index