Modeling and Simulation
Computer Lab 1 and 2

Per Jönsson
Faculty of Technology, Malmö University

Malmö 2019

Instructions for the computer lab

1. Computer lab 1 comprises linear algebra.
2. Computer lab 2 comprises calculus in one- and several variables.
3. The computer exercises should be done individually. You may help each other and discuss, but it is an individual assignment.
4. You should carefully document what you have done and make sure that you straighten out all question marks as the oral examination will be directly related to both the computer labs and the later assignments.
5. To pass you should show the documentation from the lab and the notes you have taken.

1 Linear algebra

In programing vectors and matrices (fields) are used to store data. Here we will instead treat matrices as mathematical objects on which different operations are defined. The mathematical theory for matrices, as treated in linear algebra, has important applications in numerical analysis and modeling.

1.1 Matrices and vectors

An $m \times n$ matrix is a rectangular array of real or complex numbers

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}.$$

The numbers a_{jk} are called matrix elements, and the first index j specifies in which row a_{jk} is found and the second index k specifies in which column. If the number of rows and the number of columns are the same the matrix is quadratic. Vectors are special cases of matrices. When $m = 1$ we have a row vector

$$(a_{11}, a_{12}, \dots, a_{1n}).$$

In this case only one index is needed and we write

$$(a_1, a_2, \dots, a_n).$$

When $n = 1$ we have a column vector

$$\begin{pmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{pmatrix}.$$

Also in this case one index is enough and we can write

$$\begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_m \end{pmatrix}.$$

Matrices and vectors are entered in MATLAB according to the examples below. For example the matrix

$$A = \begin{pmatrix} 1 & -2 & 3 \\ 0 & 5 & 4 \end{pmatrix}$$

is defined by the command

```
A = [1 -2 3 ; 0 5 4]
```

The row- and column vectors and

$$B = \begin{pmatrix} 3, & 2, & 5 \end{pmatrix} \quad \text{and} \quad C = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

are obtained by the commands

```
B = [3 2 5],     C = [1 ; 2]
```

1.2 Matrix operations

Matrices can be multiplied with numbers (scalars). Matrices with the same size can be added and subtracted. These operations are done element wise. For example, if

$$A = \begin{pmatrix} 1 & -2 & 3 \\ 0 & 5 & 4 \\ 1 & -1 & 0 \end{pmatrix}, \qquad B = \begin{pmatrix} 7 & 6 & 5 \\ -3 & 1 & 4 \\ -4 & 1 & 2 \end{pmatrix}$$

then

$$3 \cdot A = \begin{pmatrix} 3 \cdot 1 & 3 \cdot (-2) & 3 \cdot 3 \\ 3 \cdot 0 & 3 \cdot 5 & 3 \cdot 4 \\ 3 \cdot 1 & 3 \cdot (-1) & 3 \cdot 0 \end{pmatrix} = \begin{pmatrix} 3 & -6 & 9 \\ 0 & 15 & 12 \\ 3 & -3 & 0 \end{pmatrix}$$

and

$$A + B = \begin{pmatrix} 1+7 & -2+6 & 3+5 \\ 0-3 & 5+1 & 4+4 \\ 1-4 & -1+1 & 0+2 \end{pmatrix} = \begin{pmatrix} 8 & 4 & 8 \\ -3 & 6 & 8 \\ -3 & 0 & 2 \end{pmatrix}.$$

Under certain circumstances we can also define the product of two matrices. If

$$A = \underbrace{\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \dots & \dots & \dots & \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}}_{m \times n} \quad \text{and} \quad B = \underbrace{\begin{pmatrix} b_{11} & b_{12} & \dots & b_{1p} \\ b_{21} & b_{22} & \dots & b_{2p} \\ \dots & \dots & \dots & \\ b_{n1} & b_{n2} & \dots & b_{np} \end{pmatrix}}_{n \times p}$$

the product $C = AB$ is defined as the $m \times p$ matrix

$$C = \underbrace{\begin{pmatrix} c_{11} & c_{12} & \dots & c_{1p} \\ c_{21} & c_{22} & \dots & c_{2p} \\ \dots & \dots & \dots & \\ c_{m1} & c_{m2} & \dots & c_{mp} \end{pmatrix}}_{m \times p},$$

where the elements c_{jk} are obtained by pairwise multiplication of the elements in row j in A with the elements in column k in B followed by a summation (this can also be expressed as taking the scalar product of the vector in row j with the vector in column k)

$$c_{jk} = a_{j1}b_{1k} + a_{j2}b_{2k} + \ldots + a_{jn}b_{nk}.$$

Observe that the matrix product is only defined when the number of columns in A are equal to the number of rows in B. The order of the factors is important in matrix multiplication and normally AB is different from BA. If $AB = BA$ then the matrices A and B are said to commute. We take an example to illustrate matrix multiplication.

Example 1.

(a)

$$\begin{pmatrix} 1 & 2 \\ 4 & 3 \\ 0 & 2 \end{pmatrix} \begin{pmatrix} 5 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \cdot 5 + 2 \cdot 1 \\ 4 \cdot 5 + 3 \cdot 1 \\ 0 \cdot 5 + 2 \cdot 1 \end{pmatrix} = \begin{pmatrix} 7 \\ 23 \\ 2 \end{pmatrix}$$

(b)

$$\begin{aligned}(2,\ 7,\ 1)\begin{pmatrix}3 & 0\\ 2 & 4\\ 1 & 5\end{pmatrix} &= \big(2\cdot 3+7\cdot 2+1\cdot 1,\ 2\cdot 0+7\cdot 4+1\cdot 5\big)\\ &= \big(21,\ 33\big)\end{aligned}$$

(c)

$$\begin{pmatrix}1 & 2\\ 3 & 0\end{pmatrix}\begin{pmatrix}3 & 0\\ 2 & 4\end{pmatrix}=\begin{pmatrix}1\cdot 3+2\cdot 2 & 1\cdot 0+2\cdot 4\\ 3\cdot 3+0\cdot 2 & 3\cdot 0+0\cdot 4\end{pmatrix}=\begin{pmatrix}7 & 8\\ 9 & 0\end{pmatrix}$$

(d)

$$\begin{pmatrix}3 & 0\\ 2 & 4\end{pmatrix}\begin{pmatrix}1 & 2\\ 3 & 0\end{pmatrix}=\begin{pmatrix}3\cdot 1+0\cdot 3 & 3\cdot 2+0\cdot 0\\ 2\cdot 1+4\cdot 3 & 2\cdot 2+4\cdot 0\end{pmatrix}=\begin{pmatrix}3 & 6\\ 14 & 4\end{pmatrix}$$

□

In MATLAB we have the following commands for adding, subtracting and multiplying matrices. Please observe very carefully that matrix multiplication is denoted by `*` and *not* with `.*`, which gives element vise multiplication.

`+`	addition, matrices have the same dimension.
`-`	subtraction, matrices have the same dimension.
`*`	matrix multiplication, the number of columns for the left matrix must match the number of rows for the right matrix.

Example 2. We define four matrices

```
A = [1 0 -1 ; 4 6 2 ; -2 5 6];
B = [-1 1 0 ; 2 3 4 ; -1 0 1];
C = [2 3 4];
D = [1 ; 2 ; 3];
```

(a) Matrix A and B both have dimension 3×3. Addition

```
A + B
ans =
     0     1    -1
     6     9     6
    -3     5     7
```

Multiplication AB and BA

```
A*B
ans =
     0     1    -1
     6    22    26
     6    13    26

B*A
ans =
     3     6     3
     6    38    28
    -3     5     7
```

(b) The matrix C has dimension 1×3 and D the dimension 3×1. The matrix multiplication CD gives a 1×1 matrix whereas DC gives a 3×3 matrix.

```
C*D
```

```
ans =
    20

D*C
ans =
     2     3     4
     4     6     8
     6     9    12
```

□

1.3 Transpose

Sometimes there is a need to shift rows and columns in a matrix A. This operation is called transposition, and the resulting matrix is denoted A^T. If

$$A = \begin{pmatrix} 1 & 2 \\ 4 & 3 \\ 0 & 2 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 1 & 2 & 3 \end{pmatrix}$$

then

$$A^T = \begin{pmatrix} 1 & 4 & 0 \\ 2 & 3 & 2 \end{pmatrix} \quad \text{and} \quad B^T = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}.$$

Let $A, B, C, \ldots, G$ be matrices of matching size so that the multiplication is defined. One can then show that

$$(ABC \ldots G)^T = G^T \ldots C^T B^T A^T.$$

Transposition thus reverses the order between the factors in the multiplication. To transpose a matrix A in MATLAB you write `A.'`. For real matrices one can also simply write `A'`. We take an example to illustrate the transposition.

Example 3. We define two matrices

```
A = [1 0 -1 ; 4 6 2];
B = [1 ; 2 ; 3];
```

(a) Matrix A has the dimension 2×3 and the transposed matrix has the dimension 3×2

```
A'
ans =
     1     4
     0     6
    -1     2
```

(b) The matrix AB has the dimension 2×1 and the transpose $(AB)^T$ has the dimension 1×2. According to the rules for transposition this should equal $B^T A^T$

```
(A*B)'
ans =
    -2    22

B'*A'
ans =
    -2    22
```

□

1.4 Inverse matrix

The unit matrix I of order n is defined as a matrix $n \times n$ with ones on the diagonal and zeros in the remaining positions

$$I = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ & & & \dots & \\ 0 & 0 & 0 & \dots & 1 \end{pmatrix}.$$

The unit matrix is the matrix equivalent of the number 1. For a general $n \times p$ matrix C and a general $p \times n$ matrix D we have

$$IC = C$$

and

$$DI = D.$$

Let A be an $n \times n$ matrix. If there is an $n \times n$ matrix B such that

$$AB = I \quad \text{and} \quad BA = I$$

then A is said to be invertible with the inverse matrix B. For an invertible matrix A, the inverse matrix B is normally denoted A^{-1}. The matrices

$$A = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} -2 & 1 \\ 1.5 & -0.5 \end{pmatrix}$$

satisfy (check yourself)

$$AB = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \begin{pmatrix} -2 & 1 \\ 1.5 & -0.5 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

and

$$BA = \begin{pmatrix} -2 & 1 \\ 1.5 & -0.5 \end{pmatrix} \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

The matrix A is thus invertible with the inverse matrix $A^{-1} = B$. If $A, B, C, \dots, G$ are invertible, then also the product $ABC \dots G$ is invertible with

$$(ABC \dots G)^{-1} = G^{-1} \dots C^{-1} B^{-1} A^{-1}.$$

The inverse of the product is thus the product of the inverses in reversed order. Quadratic matrices that do not have an inverse are said to be singular. For a singular matrix at least on row (or column) can be obtained as a linear combination of the other rows (columns).

In MATLAB the inverse matrix of A is obtained by the command `inv(A)`. If A is a singular matrix MATLAB writes a warning message.

Example 4.

(a) We have the quadratic matrix

```
A = [1 0 -1 ; 4 6 2 ; 1 -1 0];
```

The inverse is obtained by

```
B = inv(A)
```

and MATLAB returns

```
B =
    0.1667    0.0833    0.5000
    0.1667    0.0833   -0.5000
   -0.8333    0.0833    0.5000
```

To check that B is an inverse we compute AB and BA to see if the product equals the unit matrix

```
A*B
ans =
    1.0000         0         0
   -0.0000    1.0000    0.0000
   -0.0000   -0.0000    1.0000

B*A
ans =
    1.0000   -0.0000         0
         0    1.0000   -0.0000
         0   -0.0000    1.0000
```

(b) Consider the matrix

```
C = [1 1 0 ; 0 0 1 ; 2 2 1];
```

The matrix is singular as row 3 can be obtained as $2 \times$ row $1 + 1 \times$ row 2. When giving the command

```
inv(C)
```

MATLAB returns

```
Warning: Matrix is singular to working precision.
(Type "warning off MATLAB:singularMatrix" to suppress
this warning.)
ans =
   Inf   Inf   Inf
   Inf   Inf   Inf
   Inf   Inf   Inf
```

□

1.5 Determinants

To every quadratic $n \times n$ matrix A there is a number, called the determinant of A, that is zero if the matrix is singular and non-zero if the matrix is non-singular and thus invertible.

The determinant of A is denoted $\det(A)$ or $|A|$. For $n = 2$ the determinant is computed as

$$\det(A) = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{21}a_{12}.$$

For $n = 3$ the determinant is defined with the help of Sarru's rule

$$\det(A) = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix} = a_{11}a_{22}a_{33} + a_{31}a_{12}a_{23} + a_{21}a_{32}a_{13} - a_{11}a_{32}a_{23} - a_{21}a_{12}a_{33} - a_{31}a_{22}a_{13}.$$

If

$$A = \begin{pmatrix} 1 & 2 \\ 4 & 3 \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 5 & 4 \\ 6 & 3 & 8 \end{pmatrix}$$

then

$$\det(A) = 1 \cdot 3 - 4 \cdot 2 = -5$$

and

$$\det(B) = 1 \cdot 5 \cdot 8 + 6 \cdot 2 \cdot 4 + 0 \cdot 3 \cdot 3 - 1 \cdot 3 \cdot 4 - 0 \cdot 2 \cdot 8 - 6 \cdot 5 \cdot 3 = -14.$$

Both matrices are non-singular and thus invertible. For matrices with order higher than $n = 3$ the determinant is computed from the general expression

$$\det(A) = \sum_{(i,j,k\ldots,r)} \pm a_{i1}a_{j2}a_{k3}\ldots a_{rn},$$

where the sum is over all possible permutations of the first index. The sign is positive if the permutation is even and negativ if the permutation is odd. Specially we have

$$\det(I) = 1.$$

Let $A, B, C, \ldots, G$ be $n \times n$ matrices. It is straight forward to show that

$$\det(ABC\ldots G) = \det(A)\det(B)\det(C)\ldots\det(G).$$

The determinant of the product is equal the product of the determinants. The product rule applied on $I = AA^{-1}$ gives

$$1 = \det(I) = \det(AA^{-1}) = \det(A)\det(A^{-1}) \Leftrightarrow \det(A^{-1}) = \frac{1}{\det(A)}.$$

In MATLAB the determinant of the quadratic matrix A is computed by the command `det(A)`. The example below illustrates the use of the command.

Example 5.

(a) We have the quadratic matrix

```
A = [1 0 -1 4; 4 6 2 1; 1 -1 0 9; 3 -1 -1 0];
```

The determinant is obtained by

```
det(A)
```

and MATLAB returns

```
ans =
  -219
```

(b) Consider the matrix

```
C = [1 1 0 ; 0 0 1 ; 2 2 1];
```

The matrix is singular since row 3 can be obtained as $2 \times$ row $1 + 1 \times$ row 2. Giving the command

```
det(C)
```

MATLAB returns

```
ans =
     0
```

□

1.6 Systems of linear equations

Systems of linear equations occure in many applications. A general system is given by

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= y_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &= y_2 \\ \qquad\qquad \ldots & \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n &= y_m \end{cases},$$

where the coefficients a_{ij} and the elements y_i to the right are known numbers. If all the elements y_i to the right are zero the systems is said to be homogenous. In other cases the system is said to be inhomogeneous. The system above is often written in matrix form

$$Ax = y,$$

where A is an $m \times n$ coefficient matrix, x an $n \times 1$ column vector and y an $m \times 1$ column vector

$$A = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ & \ldots & & \\ & \ldots & & \\ a_{m1} & a_{m2} & \ldots & a_{mn} \end{pmatrix}, \quad x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}, \quad y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ y_m \end{pmatrix}.$$

The equation system relates n unknowns x_j, $j = 1, 2, \ldots, n$ through m equations. We have three different cases:

case 1: $m = n$

case 2: $m > n$

case 3: $m < n$

In case 1 the number of equations are the same as the number of unknowns. In case 2 the number of equations are larger than the number of unknowns, and the system is said to be over determined. In case 3 the number of equations are smaller than the number of unknowns, and the system is said to be under determined.

In case 1 there is a unique solution precisely when the coefficient matrix A is non-singular and invertible. The solution $Ax = y$ can be determined by Gaussian elimination. Alternatively the solution can be obtained by first computing the inverse matrix A^{-1} and then multiply it from the left

$$Ax = y \Leftrightarrow \underbrace{A^{-1}A}_{I}x = A^{-1}y \Leftrightarrow x = A^{-1}y.$$

To determine the solution by Gaussian elimination we write

```
x = A\y
```

Observe `A\y` and *not* `A/y`. To obtain the solution based on the inverse give the command

```
x = inv(A)*y
```

The first method is computationally more efficient and is the preferred one. Cases where the coefficient matrix is singular can not be handled by MATLAB, and we get a warning message

In case 2, more equations than unknowns, there is normally no solution. These systems are instead solved in the least squares sense (we will return to this later on). Giving the command

```
x = A\y
```

MATLAB checks if the system is over determined. If so, MATLAB instead solves the so called normal equation

$$A^T Ax = A^T y.$$

The command to solve an equation system in the normal sense or in the least squares sense is thus the same. What matters is the dimension of the matrix A. If the matrix is quadratic the equation system is solved by Gaussian elimination, if the matrix has more rows than columns then the normal equation is solved instead.

In case 3, more unknowns than equation, there are normally an infinite number of solutions. These systems can not be handled by MATLAB.

Example 6.

(a) To solve the equation system

$$\begin{cases} 3x_1 - 2x_2 + 4x_3 &= 8 \\ 5x_1 + 8x_2 - 6x_3 &= -5 \\ 9x_1 - 2x_2 + 7x_3 &= -17 \end{cases}$$

we define the coefficient matrix and the right hand side

```
A = [3 -2 4 ; 5 8 -6 ; 9 -2 7]; y = [8 ; -5 ; -17];
```

and write

```
x = A\y
```

MATLAB returns

```
x =
  -36.7778
   71.2778
   65.2222
```

If we write `x = inv(A)*y` we get, just as expected, the same answer.

(b) We have the system

$$\begin{cases} -x_1 + 2x_2 &= 1 \\ -2x_1 + 4x_2 &= 0 \end{cases}.$$

The coefficient matrix is singular. In this case there is now solution to the system. To see how MATLAB handles this we define the coefficient matrix and the right hand side

```
A = [-1 2 ; -2 4]; y = [1 ; 0];
```

and write

```
x = A\y
```

giving the answer

```
Warning: Matrix is singular to working precision.
(Type "warning off MATLAB:singularMatrix" to
suppress this warning.)
x =
   Inf
   Inf
```

This is consistent with the fact that MATLAB does not handle singular matrices. □

Example 7. We consider an over determined system with more equations than unknown.

$$\begin{cases} 2x_1 - 2x_2 + 4x_3 &= 1 \\ 6x_1 + 8x_2 - 6x_3 &= -1 \\ 9x_1 - 5x_2 + 1x_3 &= 4 \\ 3x_1 + 2x_2 - 2x_3 &= -5 \end{cases}.$$

When writing

```
A = [2 -2 4 ; 6 8 -6 ; 9 -5 1 ; 3 2 -2];
y = [1 ; -1 ; 4 ; -5];
x = A\y
```

MATLAB answers

```
x =
    0.1533
   -0.3946
    0.0383
```

which is the solution to the normal equation $A^T Ax = A^T y$. □

1.7 Ill-conditioned systems

There are linear systems of equations that are uniquely solvable, but where the solution is inaccurate due to rounding errors during the computation. These systems are said to be ill-conditioned. Consider the system

$$\begin{cases} 1.12065x_1 + 0.98775x_2 &= 2.12\underline{\mathbf{3}}41 \\ 2.24135x_1 + 1.97553x_2 &= 4.24601 \end{cases}.$$

The solution to the system is

$$x_1 = -54.78056, \qquad x_2 = 64.30093.$$

To illustrate the sensitivity of the solution to the coefficient matrix we subtract the first number to the right by 0.001

$$\begin{cases} 1.12065x_1 + 0.98775x_2 &= 2.12\underline{\mathbf{2}}41 \\ 2.24135x_1 + 1.97553x_2 &= 4.24601 \end{cases}.$$

The new solution is

$$x_1 = 72.40932, \qquad x_2 = -81.01554.$$

The difference between the two solution is very large considering the very small change to the right hand side. Small changes to the coefficient matrix give similar effects. All computer systems work with a limited number of digits, and errors in the right hand side or in the coefficient matrix are always induced in the cause of the calculation of the solution.

To investigate the sensitivity to small changes in the coefficient matrix or the right hand side the concept of norm is introduced. The norm of a vector or a matrix is a number that can be used to measure the distance of the vector or matrix from the zero vector or the zero matrix. For vectors we have the following norms

$$\|x\|_1 = |x_1| + |x_2| + \ldots + |x_n| \qquad \text{1-norm}$$

$$\|x\|_2 = \sqrt{x_1^2 + x_2^2 + \ldots + x_n^2} \qquad \text{Euclidian-norm}$$

$$\|x\|_\infty = \max\{|x_1|, |x_2|, \ldots, |x_n|\} \qquad \text{maximum-norm.}$$

The norm for a vector can be seen as the generalization of the absolute value of an ordinary number. To each vector norm $\|x\|$ there is a corresponding matrix norm $\|A\|$ defined by

$$\|A\| = \max_{x \neq 0} \frac{\|Ax\|}{\|x\|}.$$

Suppose that the vector x is a solution to the equation system $Ax = y$. If we change the right hand side y with a quantity δy, then the solution is changed from x to $x + \delta x$. One can show that the relative changes satisfies

$$\frac{\|\delta x\|}{\|x\|} \leq \underbrace{\|A\|\|A^{-1}\|}_{\text{cond}(A)} \frac{\|\delta y\|}{\|y\|}.$$

A similar relation holds true for the changes in the coefficient matrix. The quantity

$$\|A\|\|A^{-1}\|$$

is known as the condition number for the matrix A and is denoted $\text{cond}(A)$. The condition number satisfies $\text{cond}(A) \geq 1$. If the condition number is very large, the matrix is ill-conditioned. The matrix in the example above has the condition number 7.1×10^5 and is relatively ill-conditioned. Rounding errors during the cause of the solution is thus magnified, and in this case we may loose up to five significant digits. For ill-conditioned matrices A, the product AA^{-1} differs, due to rounding errors, from the unit matrix. Also $\det(A)\det(A^{-1})$ differ from 1.

In MATLAB we have the following commands to compute norms and condition numbers.

`norm(x,n)`	gives vector norm of x. $n = 1$ gives $\|x\|_1$, $n = 2$ gives $\|x\|_2$ and $n = \inf$ gives $\|x\|_\infty$.
`norm(A,n)`	gives matrix norm of A.
`cond(A,n)`	gives the condition number for A.

Example 8. We have a vector v and a matrix C

```
v = [1 3 -4 1 2];    C = [1 -1 2 ; 3 4 1 ; -1 -4 -2];
```

The vector norm $\|v\|_2$ is obtained by

```
norm(v,2)
```

and MATLAB answers

```
ans =
    5.5678
```

The matrix norm $\|C\|_\infty$ is obtained by

```
norm(C,inf)
```

and MATLAB answers

```
ans =
     8
```

To compute the condition number $\mathrm{cond}(C)$ in 1-norm we write

```
cond(C,1)
```

which gives

```
ans =
    7.5600
```

□

Example 9. Hilbert matrices of order n have the form

$$A = \begin{pmatrix} 1 & 1/2 & 1/3 & \dots & 1/n \\ 1/2 & 1/3 & 1/4 & \dots & 1/(n+1) \\ 1/3 & 1/4 & 1/5 & \dots & 1/(n+2) \\ & & \vdots & & \\ 1/n & 1/(n+1) & 1/(n+2) & \dots & 1/(2n-1) \end{pmatrix}$$

and are known to be very ill-conditioned. The program **hilbtest.m** generates Hilbert matrices for n from 5 to 15 and computes the condition number $\mathrm{cond}(A)$ as well as $\det(A)\det(A^{-1})$. The difference from 1 for the the latter quantity can be seen as a manifestation of the matrix being ill-conditioned.

```
% hilbtest.m
clear all
for n = 5:2:15
   % generate Hilbert matrix
   for i = 1:n
      for j = 1:n
         A(i,j) = 1/(i+j-1);
      end
   end
   % compute condition number and determinant product
   c = cond(A);
   d = det(A)*det(inv(A));
   % print out
   fprintf('n=%3.0f cond(A)=%e, det*det=%9.6f\n',n,c,d)
end
```

Running the program MATLAB answers

```
n=  5 cond(A)=4.766073e+005, det*det= 1.000000
n=  7 cond(A)=4.753674e+008, det*det= 1.000000
n=  9 cond(A)=4.931538e+011, det*det= 1.000000
n= 11 cond(A)=5.223947e+014, det*det= 1.000118
n= 13 cond(A)=1.316877e+018, det*det= 1.211551
n= 15 cond(A)=3.154765e+017, det*det= 0.426277
```

Starting from the matrix of order $n = 11$ MATLAB issues a warning message that the result of the computation may be inaccurate. □

1.8 Eigenvalues and eigenvectors

Ler A be an $n \times n$ quadratic matrix. A vector $x \neq 0$ such that

$$Ax = \lambda x$$

for some number λ, is called called an eigenvector to A with the eigenvalue λ. Even for real matrices A the eigenvalues and eigenvectors may be complex. Counting with multiplicity an $n \times n$ matrix has always n eigenvalues.

To determine the eigenvalues and the eigenvectors we rearrange the equation above so that

$$(A - \lambda I)x = 0,$$

where I is the unit matrix. Since this is a homogenous quadratic system of equations there is a solution $x \neq 0$ if and only if the coefficient matrix is singular, i.e.

$$\det(A - \lambda I) = 0.$$

This equation is called the characteristic equation. Solving this equation gives the eigenvalues.

Example 10.

(a) The matrix

$$A = \begin{pmatrix} 1 & 3 \\ 1 & -1 \end{pmatrix}$$

has the characteristic equation

$$\begin{vmatrix} 1-\lambda & 3 \\ 1 & -1-\lambda \end{vmatrix} = (1-\lambda)(-1-\lambda) - 1 \cdot 3 = \lambda^2 - 4 = 0.$$

The eigenvalues are thus 2 and -2. To determine the eigenvectors belonging to the eigenvalue λ we should solve the equation system $(A - \lambda I)x = 0$. When $\lambda = 2$ we get

$$\begin{cases} -x_1 + 3x_2 &= 0 \\ x_1 - 3x_2 &= 0 \end{cases} \Leftrightarrow \begin{cases} x_1 = 3s \\ x_2 = s \end{cases}.$$

The eigenvectors belonging to the eigenvalue 2 are thus

$$x = s \begin{pmatrix} 3 \\ 1 \end{pmatrix},$$

where $s \neq 0$ is an arbitrary non-zero constant. In the same manner for $\lambda = -2$ we get the equation system

$$\begin{cases} 3x_1 + 3x_2 &= 0 \\ x_1 + x_2 &= 0 \end{cases} \Leftrightarrow \begin{cases} x_1 = -t \\ x_2 = t \end{cases}.$$

The eigenvectors belonging to the eigenvalue -2 are thus

$$x = t \begin{pmatrix} -1 \\ 1 \end{pmatrix},$$

where $t \neq 0$ is an arbitrary non-zero constant.

(b) Real matrices can have complex eigenvalues. The matrix

$$A = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$$

has the characteristic equation

$$\begin{vmatrix} -\lambda & -1 \\ 1 & -\lambda \end{vmatrix} = \lambda^2 + 1 = 0.$$

The eigenvalues are thus i and $-i$. Insertion of $\lambda = i$ in the equation system $(A - \lambda I)x = 0$ gives

$$\begin{cases} -ix_1 - x_2 &= 0 \\ x_1 - ix_2 &= 0 \end{cases} \Leftrightarrow \begin{cases} x_1 = si \\ x_2 = s \end{cases}.$$

The eigenvectors belonging to the eigenvalue i is thus

$$x = s \begin{pmatrix} i \\ 1 \end{pmatrix}$$

where $s \neq 0$ is an arbitrary constant. When $\lambda = -i$ we get

$$\begin{cases} ix_1 - x_2 &= 0 \\ x_1 + ix_2 &= 0 \end{cases} \Leftrightarrow \begin{cases} x_1 = t \\ x_2 = ti \end{cases}.$$

The eigenvectors belonging to the eigenvalues $-i$ is

$$x = t \begin{pmatrix} 1 \\ i \end{pmatrix},$$

where $t \neq 0$ is an arbitrary constant.

(c) Eigenvalues can be multiple. The matrix

$$C = \begin{pmatrix} 1 & 0 & 1 \\ -1 & 2 & 1 \\ -1 & 1 & 2 \end{pmatrix}$$

has the characteristic equation

$$\begin{vmatrix} 1-\lambda & 0 & 1 \\ -1 & 2-\lambda & 1 \\ -1 & 1 & 2-\lambda \end{vmatrix} = -\lambda^3 + 3\lambda^2 - 4\lambda + 2 = 0.$$

The equation has the solutions 1 and 2 (multiplicity 2). The eigenvalues are thus 1 and 2, where the last eigenvalues is said to that the algebraic multiplicity 2. Insertion of $\lambda = 1$ in the equation system $(A - \lambda I)x = 0$ gives the eigenvectors

$$x = s \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix},$$

where $s \neq 0$ is an arbitrary constant. Insertion of $\lambda = 2$ gives the eiegenvectors

$$x = t \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix},$$

where $t \neq 0$ is an arbitrary constant. □

In MATLAB we have the following commands to determine eigenvalues and eigenvectors.

`eig(A)`	gives a column vector with the eigenvalues to A.
`[X,D] = eig(A)`	gives a diagonal matrix D that contains the eigenvalues to A and a matrix X with corresponding normalized eigenvectors.

Example 11.

(a) We have the matrix

```
A = [1 -1 ; 3 4]
```

The eigenvalues are obtained by

```
lambda = eig(A)
```

and MATLAB answers

```
lambda =
   2.5000 + 0.8660i
   2.5000 - 0.8660i
```

(b) We have the matrix

```
A = [1 -1 4 ; 3 2 -1 ; 2 1 -1];
```

The eigenvalues and the normalized eigenvectors are obtained from

```
[X,D] = eig(A)
```

and MATLAB answers

```
X =
    0.4082    0.5774    0.2357
    0.8165   -0.5774   -0.9428
    0.4082   -0.5774   -0.2357
D =
    3.0000         0         0
         0   -2.0000         0
         0         0    1.0000
```

The eigenvalues are 3, -2 and 1 and the corresponding normalized eigenvectors are

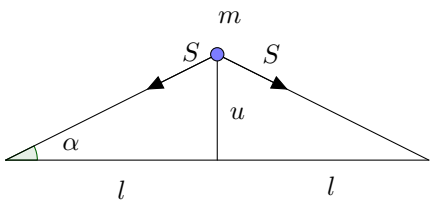
$$\begin{pmatrix} 0.4082 \\ 0.8165 \\ 0.4082 \end{pmatrix}, \quad \begin{pmatrix} 0.5774 \\ -0.5774 \\ -0.5774 \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} 0.2357 \\ -0.9428 \\ -0.2357 \end{pmatrix}.$$

□

1.9 Oscillations

String with one mass point

In the middle of a string with length $2l$ there is a mass point with mass m that oscillates around the equilibrium position



S is the tension in the string and the resulting force on the mass point is $-2S\sin\alpha$. For small oscillations we can assume that is S constant and replace $\sin\alpha$ with u/l. Newton's second law now gives

$$mu''(t) = -\frac{2S}{l}u(t).$$

Setting, for simplicity, $S/(ml) = 1$ we get

$$u''(t) = -2u(t).$$

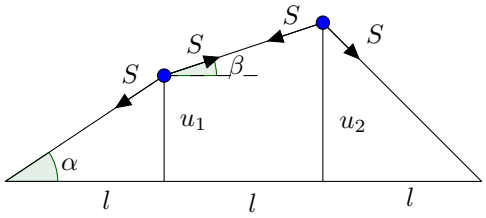
The solution to this equation is

$$u(t) = c \cdot \sin(kt + \delta), \quad \text{with} \quad k = \sqrt{2}.$$

Here c and δ are constants. The mass point oscillates with the frequency $k/2\pi$ (or angular frequency k).

String with two mass points

Now consider two mass points on a string with length $3l$



The mass point belonging to u_1 is affected by the force

$$-S\sin\alpha + S\sin\beta \approx -S\frac{u_1}{l} + S\frac{u_2 - u_1}{l} = -\frac{2S}{l}u_1 + \frac{S}{l}u_2.$$

Newton's secon law gives

$$mu_1''(t) = -\frac{2S}{l}u_1(t) + \frac{S}{l}u_2(t).$$

From the symmetry we have

$$mu_2''(t) = \frac{S}{l}u_1(t) - \frac{2S}{l}u_2(t)$$

which yields

$$\begin{pmatrix} u_1''(t) \\ u_2''(t) \end{pmatrix} = \begin{pmatrix} -2 & 1 \\ 1 & -2 \end{pmatrix} \begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix}.$$

We now let

$$\begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = \sin(kt + \delta) \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}.$$

Insertion in the equations and simplification gives the eigenvalue problem

$$\begin{pmatrix} -2 & 1 \\ 1 & -2 \end{pmatrix} \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \underbrace{-k^2}_{\lambda} \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}.$$

There are two eigenvalues $\lambda = -1$ and $\lambda = -3$. The normalized eigenvector belonging to the eigenvalue $\lambda = -1$ is

$$\begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}.$$

The normalized eigenvector belonging to the eigenvalue $\lambda = -3$ is

$$\begin{pmatrix} c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}.$$

We thus have two solutions

$$\begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = \sin(t + \delta) \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$$

and

$$\begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = \sin(\sqrt{3}t + \delta) \begin{pmatrix} 1/\sqrt{2} \\ -1/\sqrt{2} \end{pmatrix}.$$

These two solutions are known as eigen oscillations (or free oscillations). $k = 1$ is the fundamental frequency (grundtonen) and $k = \sqrt{3}$ is the over tone. The general solution to the equation systems is a linear combination of eigen oscillations

String with n mass points

It is easy to generalize. For n mass points we have the eigenvalue problem

$$\underbrace{\begin{pmatrix} -2 & 1 & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 & \dots & 0 & 0 & 0 \\ & \ddots & & & & \vdots & & \ddots & \\ 0 & 0 & 0 & 0 & 0 & \dots & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & 0 & \dots & 0 & 1 & -2 \end{pmatrix}}_{n \times n} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_{n-1} \\ c_n \end{pmatrix} = \underbrace{-k^2}_{\lambda} \begin{pmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_{n-1} \\ c_n \end{pmatrix}$$

The eigen oscillations are obtained from the eigenvalues and eigenvectors just as in the example above.

Example 12. The following M-file animates the eigen oscillations in the case of two mass points as t goes from $t = 0$ to $t = 10\pi$. We put $\delta = 0$.

```
% eigenoscillation.m
A = [-2 1 ; 1 -2];
[C,D] = eig(A);
l = [0 1 2 3]'; % Positions for the end points and
                % mass points as a column vector
figure(1)
% smallest k
k = sqrt(-D(2,2));
for t = 0:0.1:10*pi
   u = sin(k*t)*[0 ; C(:,2) ; 0]; % u as a column vectors,
                                  % u = 0 for the end points
   plot(l,u,l,u,'o')
   ylim([-1 1])
   pause (0.1)
end

figure(2)
% largest k
k = sqrt(-D(1,1));
for t = 0:0.1:10*pi
   u = sin(k*t)*[0 ; C(:,1) ; 0]; % u as a column vector
                                  % u = 0 for the end points
   plot(l,u,l,u,'o')
   ylim([-1 1])
   pause (0.1)
end
```

Study questions

1. How is matrix multiplication defined?
2. What do we mean when we say that two matrices A and B commute?
3. What is the difference between `A.*B` and `A*B`?
4. What is the transpose of a matrix?
5. How does the unit matrix look like?
6. What is meant by a singular matrix?
7. What value has the determinant of a singular matrix?
8. What is meant by a homogenous equation system?
9. What is meant by an equation system that is over- and under determined, respectively?
10. Explain how MATLAB solves a system that is over determined.
11. What is mean by an ill-conditioned system?
12. What is meant by the condition number?
13. How is eigenvalues and eigenvectors defined?

Exercises

1. Let

$$A = \begin{pmatrix} 1 & 0 & 2 \\ 0 & 3 & 1 \\ 2 & 2 & -1 \end{pmatrix}, \quad B = \begin{pmatrix} 0 & 1 & 1 \\ 2 & -2 & 0 \\ 1 & 2 & 3 \end{pmatrix}, \quad C = \begin{pmatrix} 2 & 1 \\ -1 & 1 \\ 1 & 2 \end{pmatrix}$$

   Compute

   (a) AB (b) BA (c) $A^T B^T$ (d) $(A + 3B)C$

   Do the computations by hand. Use MATLAB to check the answer.

2. Determine which of the following matrices that are invertible. If invertible, then compute the inverse. Do the computations by hand and then use MATLAB to check the answer.

   (a) $A = \begin{pmatrix} 1 & 2 & 3 \\ 0 & 1 & 2 \\ 0 & 0 & 1 \end{pmatrix}$ (b) $B = \begin{pmatrix} 1 & 1 & 2 \\ 2 & 1 & 1 \\ -1 & 1 & 4 \end{pmatrix}$

3. Compute the inverse to

$$A = \begin{pmatrix} 4 & 2 & 3 & 1 \\ 2 & 5 & 6 & 2 \\ 0 & 0 & 1 & 3 \\ -1 & -2 & 9 & 8 \end{pmatrix}$$

   Verify that $AA^{-1} = A^{-1}A = I$.

4. Let

$$A = \begin{pmatrix} 1 & 1 & 3 \\ -1 & 1 & 4 \\ 0 & 2 & 8 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & -2 & 9 \\ 3 & 4 & 1 \\ 0 & 1 & -3 \end{pmatrix}$$

   show by using MATLAB that $(A \cdot B)^{-1} = B^{-1} \cdot A^{-1}$.

5. Compute the determinant to the following matrices

   (a) $\begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \\ -1 & 0 & 1 \end{pmatrix}$ (b) $\begin{pmatrix} 1 & 4 & 2 \\ 2 & 2 & 1 \\ -1 & 1 & 4 \end{pmatrix}$

   Do the computation by hand. Check the answer by running MATLAB.

6. Consider the over determined system $Ax = y$ where

$$A = \begin{pmatrix} 2 & -2 & 4 \\ 6 & 8 & -6 \\ 9 & -5 & 1 \\ 3 & 2 & -2 \end{pmatrix} \quad \text{and} \quad y = \begin{pmatrix} 1 \\ -1 \\ 4 \\ -5 \end{pmatrix}$$

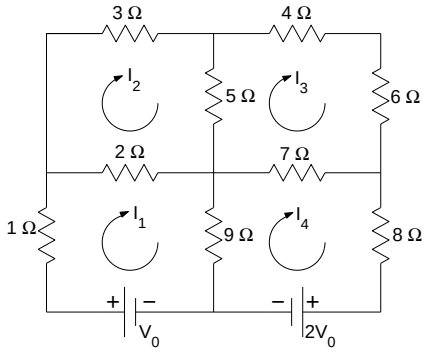
   Show, by insertion, that x determined by the command `x = A\y` is a solution to the normal equation $A^T Ax = A^T y$.

7. Based on a vector $x = (x_1, x_2, \ldots, x_n)$ one constructs a so called Vandermonde matrix.

$$V = \begin{pmatrix} x_1^{n-1} & x_1^{n-2} & \ldots & x_1 & 1 \\ x_2^{n-1} & x_2^{n-2} & \ldots & x_2 & 1 \\ & & \ldots & & \\ x_n^{n-1} & x_n^{n-2} & \ldots & x_n & 1 \end{pmatrix}$$

Vandermonde matrices are known to be ill-conditioned Generate the Vandermonde matrix for $x = (1, 2, 3 \ldots, 10)$ and compute and write the condition number $\mathrm{cond}(V)$ as well as the product $\det(V)\det(V^{-1})$. Use `format long` for the write out.

8. Consider the electrical grid below.

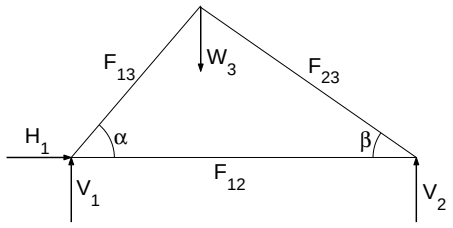


According to the laws of Kirchhoff the sum of the potential drops in a closed loop is equal to zero. The four loops yield the equations

$$\begin{cases} V_0 - I_1 - 2(I_1 - I_2) - 9(I_1 - I_4) & = & 0 \\ -3I_2 - 5(I_2 - I_3) - 2(I_2 - I_1) & = & 0 \\ -4I_3 - 6I_3 - 7(I_3 - I_4) - 5(I_3 - I_2) & = & 0 \\ -2V_0 - 7(I_4 - I_3) - 8I_4 - 9(I_4 - I_1) & = & 0 \end{cases}$$

Compute the current I_1, I_2, I_3, I_4 when $V_0 = 100$ V.

9. Structures constructed using triangular elements often hold heavy loads.



The unknown tensions are denoted F_{12}, F_{13} and F_{23}. V_1 and V_2 are the unknown vertical forces supporting the construction at nodes 1 and 2. H_1 is the unknown horizontal force at node 1. Finally, W_3 is the known force representing the weight of the structure. Equilibria in vertical and horizontal directions in each of the

three nodes lead to the following equation system

$$\begin{cases} V_1 + F_{13}\sin\alpha &= 0 \\ H_1 + F_{12} + F_{13}\cos\alpha &= 0 \\ V_2 + F_{23}\sin\beta &= 0 \\ -F_{12} - F_{23}\cos\beta &= 0 \\ -F_{13}\sin\alpha - F_{23}\sin\beta &= W_3 \\ -F_{13}\cos\alpha + F_{23}\cos\beta &= 0 \end{cases}.$$

Set $\alpha = \pi/6$, $\beta = \pi/3$ and $W_3 = 100$. Solve the equation system and determine the unknown forces.

10. Determine all the eigenvalues and eigenvectors of the following matrices. Do the calculations by hand and check the results using MATLAB.

(a) $A = \begin{pmatrix} 5 & 6 \\ -2 & -2 \end{pmatrix}$ (b) $B = \begin{pmatrix} 1 & -3 \\ 3 & 2 \end{pmatrix}$ (c) $C = \begin{pmatrix} 1 & 3 \\ 4 & 2 \end{pmatrix}$

11. Compute all eigenvalues and eigenvectors to the following matrices. Use MATLAB. Give the multiplicity for each of the eigenvalues.

(a) $A = \begin{pmatrix} 2 & -1 & 1 \\ -1 & 2 & 1 \\ -1 & 1 & 2 \end{pmatrix}$ (b) $B = \begin{pmatrix} 0 & 1 & 1 \\ -4 & 4 & 1 \\ -2 & 1 & 2 \end{pmatrix}$ (c) $C = \begin{pmatrix} 1 & -3 & 4 \\ 4 & -7 & 8 \\ 6 & -7 & 7 \end{pmatrix}$

12. Write a general program that shows an animation of the oscillating mass points. The number of mass points as well as which eigen oscillation to display should be read using the command `input`. Also the duration of the animation should be read using the command `input`.

2 One-dimensional calculus

In one-dimensional calculus we study functions $y = f(x)$. Important aspects are: plotting, solving equations, computing derivatives, determining maxima and minima (optimization), Taylor expansions, integrals and differential equations.

2.1 Anonymous functions

In MATLAB functions can be defined in M-files or as so called anonymous functions.

Example 13. We have a function $y = x\sin(x)$. The corresponding anonymous function in MATLAB is defined as

```
f = @(x) x.*sin(x)
```

Anonymous functions work exactly as ordinary functions, and we can call them with scalar arguments. In addition we may call them using vectors or matrices as arguments. For example

```
y = f(0)
```

gives

```
y =
    0
```

whereas

```
y = f([0 1 2])
```

gives

```
y =
    0    0.8415    1.8186
```

2.2 Plotting

We illustrate the basic plotting with two examples.

Example 14. To plot the function $y = \sin(x)$ in the interval $0 \leq x \leq 10$ with a solid line we write

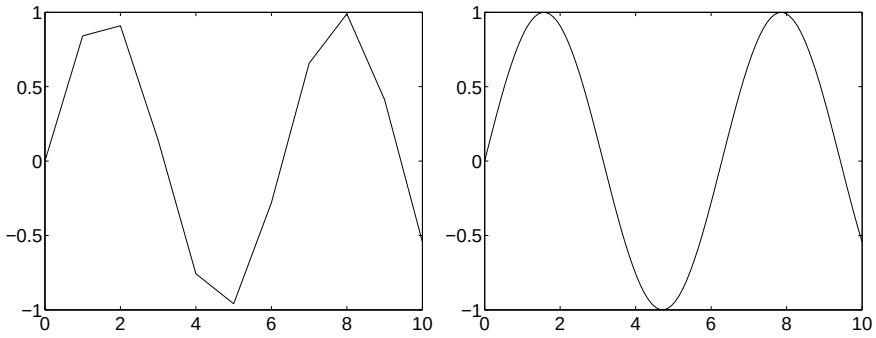
```
x = 0:1:10;
y = sin(x);
plot(x,y)
```

The generate plot is shown to the left in figure 1. The plot is ragged since we have too few values for x. If we instead define x by the command `x = linspace(0,10)`, i.e. hundred values evenly distributed between 0 and 10, then we have the smooth plot to the right in figure 1. □

Example 15.

(a) One can plot using different types of lines. The commands

```
x = 0:0.1:3;
y1 = exp(-x).*sin(x);
y2 = exp(-x).*cos(x);
plot(x,y1,'+',x,y2,'o')  % plot using plus + and rings o
```



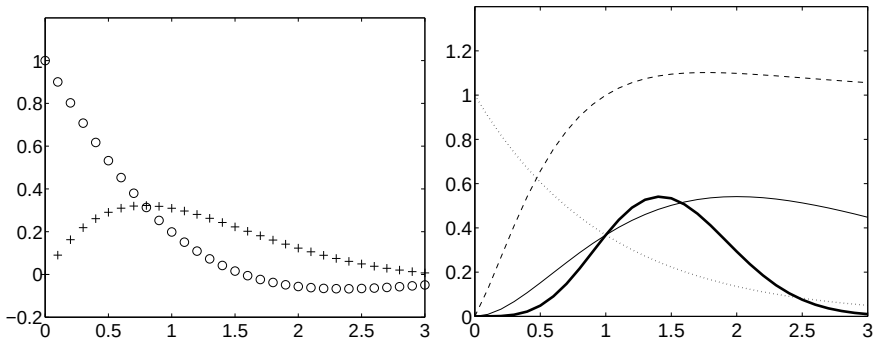
Figur 1: *The function* $y = \sin(x)$ *plotted using different number of points.*

gives the plot to the left in figure 2.

(b) It is possible to use different line types and line widths. The commands

```
x = 0:0.1:3;
y1 = exp(-x); y2 = x.^exp(-x);
y3 = x.^2.*exp(-x); y4 = x.^4.*exp(-x.^2);
plot(x,y1,':')                % dotted line
hold on                       % lock graphical window
plot(x,y2,'--')               % dashed line
plot(x,y3,'LineWidth',1)      % line width 1 (standard)
plot(x,y4,'LineWidth',3)      % line width 3 (thick)
```

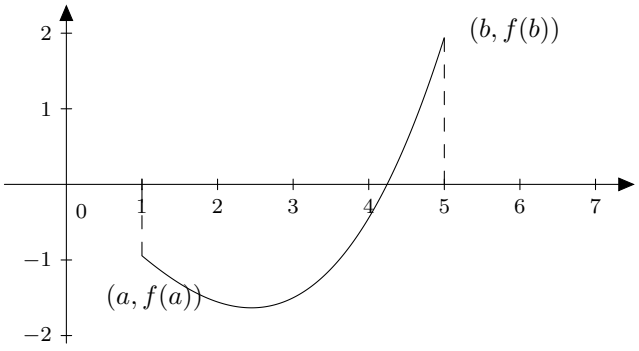
give the plot to the right in figure 2. Note that if we use separate plot commands then `hold on` must be used otherwise the previous plot is erased. □



Figur 2: *Functions with different line types and line widths.*

2.3 Zeros of functions and equations

To determine the zeros of the function $f(x)$ is the same as finding the roots to the equation $f(x) = 0$. In what is to follow we will only consider continuous functions in one variable. If one finds two points a and b such that $f(a)$ and $f(b)$ have different signs, the intermediate value theorem states that there is at least one zero in the interval $[a, b]$, see figure 3.



Figur 3: *When $f(a)$ and $f(b)$ have different signs there is at least one zero in the interval $[a, b]$.*

Analytical computation

In MATLAB it is possible to solve simple equations analytically using the symbolic toolbox. In addition to real valued solutions also complex valued solutions are obtained.

`solve(s,v)`	solves the equation $s = 0$, where s is a symbolic expression with respect to the variable v.

If exact analytical solutions can not be determined, the corresponding numerical solution is, in some cases, returned. Please note that in solving trigonometric equations only some solutions are returned, and the user has to deal with the periodicity himself.

Example 16.

(a) To solve the equation $x^2 = 2x - 2$ and save the solution in the variable x we first rearrange so we have zero at the right hand side $x^2 - 2x + 2 = 0$. We then write

```
syms x
x = solve(x^2-2*x+2,x)
```

and MATLAB answers

```
x =
 1+i
 1-i
```

We thus have two complex solutions $1 + i$ and $1 - i$ stored in $x(1)$ and $x(2)$, respectively.

(b) We have the expression

$$\frac{T_1}{T_2} = \left(\frac{P_1}{P_2}\right)^{\frac{\kappa-1}{\kappa}}$$

To solve for κ one first rearranges so that we have zero at the right hand side

$$\frac{T_1}{T_2} - \left(\frac{P_1}{P_2}\right)^{\frac{\kappa-1}{\kappa}} = 0$$

We then give the commands

```
syms T1 T2 P1 P2 k
k = solve(T1/T2-(P1/P2)^((k-1)/k),k);
pretty(k)
```

and MATLAB answers

```
                 P1
          log(----)
                 P2
- ----------------------
          P1           T1
  -log(----) + log(----)
          P2           T2
```

(c) The equation $\cos(x) = x$ has no analytical solution. When we write

```
syms x
solve(cos(x)-x,x)
```

we instead get the numerical value

```
ans =
.739085133215160641655312087673 87
```

□

Numerical computation

Numerical determination of zeros or roots is an iterative process. Given an interval that encapsulate the zero, or a starting value close to the zero, a sequence of numbers approaching the zero is constructed. The process is terminated when the constructed numbers are sufficiently close to the zero. In MATLAB numerical determination of zeros is done using the command `fzero`. Intervals that encapsulate the zero, or a starting value close to the zero, needed for the process are normally determined from a plot of the function.

`fzero(fun,x0,opt)`	determines a zero to the function *fun* close to the starting value x_0. If no zero is found the value *NaN* (Not a Number) is returned. x_0 can also be given as a vector of length 2 representing the interval $[x_0(1), x_0(2)]$ encapsulating the zero. In this case the zero is determined in the interval.
`[x,y,flag,inf]= fzero(fun,x0,opt)`	as above. Returns zero and corresponding function value in the variables x and y. The variable *flag* is larger than 0 if a zero has been found and less than 0 if the routine failed. *inf* stores information about the iterations.

Example 17.

(a) We want to determine the zeros to the function

$$f(x) = 3e^{-x/4} - \sin(x) - 2.$$

The function is defined as an anonymous function

```
f = @(x) 3*exp(-x/4) - sin(x) - 2;
```

We start by plotting the function between -4 and 4

```
x = linspace(-4,4);
y = f(x);
plot(x,y), grid on
```

From the plot to the left in figure 4 we see out that there is a zero close to $x = 0.5$. To determine the zero more accurately we give the command

```
x = fzero(@fun,0.5)
```

and MATLAB returns

```
x =
    0.6119
```

(b) In a chemical reaction the concentration of an ion is given by the expression

$$10e^{-3t} + 2e^{-5t}, \quad t \geq 0.$$

The concentration is decreasing from the maximal value 12 (see plot to the right in figure 4.) We want to determine the time t for which the concentration has gone down to 6.

The task is equivalent with solving the equation

$$10e^{-3t} + 2e^{-5t} - 6 = 0.$$

From the plot we see that there is a solution to the equation close to 0.2. To obtain a more accurate value we define $10e^{-3t} + 2e^{-5t} - 6$ as an anonymous function

```
f = @(t) 10*exp(-3*t) + 2*exp(-5*t) - 6;
```

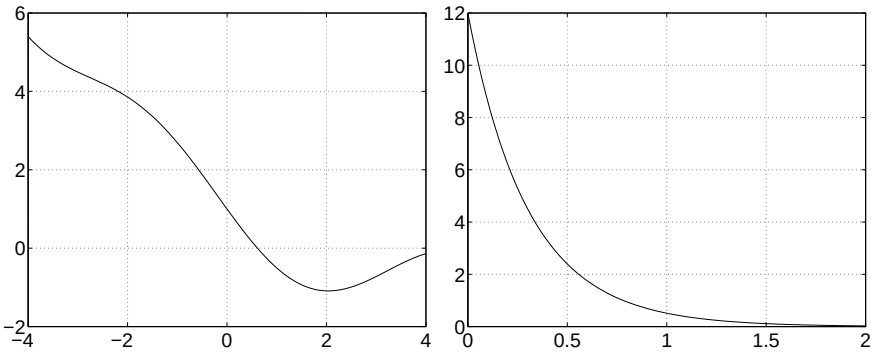
The command

```
t = fzero(@fun2,0.2)
```

gives

```
t =
    0.2113
```

Insertion confirms that this value corresponds to a concentration of 6. □



Figur 4: *Left: the function* $f(x) = 3e^{-x/4} - \sin(x) - 2$. *Right: the ion concentration as a function of time.*

2.4 Derivatives

Derivatives $f'(x)$, $f''(x)$ etc. are used in a number of situations; often to characterize functions, but also to solve optimization problems (finding maxima or minima). Determining the derivative is referred to as differentiation.

Analytical computation

MATLAB has the following commands to determine derivatives analytically.

`diff(f,x)`	differentiates the symbolic function f with respect to x.
`diff(f,x,n)`	differentiates the function f n times with respect to x

Example 18. Consider the function

$$f(x) = \frac{ax}{1 + \cos(bx)}.$$

(a) Differentiating with respect to x is done by

```
syms x a b
f = a*x/(1 + cos(b*x));
dfx = diff(f,x);
pretty(dfx)
```

MATLAB answers

```
     a          a x sin(b x) b
------------ + ---------------
1 + cos(b x)                 2
              (1 + cos(b x))
```

(b) To differentiate two times we give the commands

```
syms x a b
f = a*x/(1 + cos(b*x));
d2fx = diff(f,x,2);
pretty(d2fx)
```

which gives

```
                                      2  2                  2
   a sin(b x) b       a x sin(b x)  b    a x cos(b x) b
2 -------------- + 2 ---------------- + --------------
                2                   3                  2
  (1 + cos(b x))     (1 + cos(b x))     (1 + cos(b x))
```

□

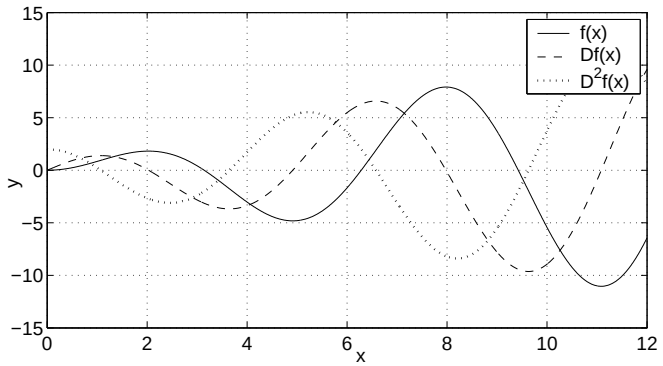
Example 19. We have the function

$$f(x) = x\sin(x)$$

The function, the first derivative and the second derivative is plotted by the commands

```
syms x                    % def. x as a symbolic variabel
f = x*sin(x);             % f(x)
df = diff(f,x);           % f'(x)
d2f = diff(f,x,2);        % f''(x)
xv = linspace(0,12,500);  % vector with values from 0 to 12
fv = subs(f,x,xv);        % vector with values for f(x)
dfv = subs(df,x,xv);      % vector with values for f'(x)
d2fv = subs(d2f,x,xv);    % vector with values for f''(x)
plot(xv,fv,'-',xv,dfv,'--',xv,d2fv,':')
xlabel('x'), ylabel('y')
legend('f(x)','Df(x)','D^2f(x)')
grid on                   % grid lines
```

The generated plot is shown in figure 5. Please note that $f(x)$ has inflexion points (points where the functions goes from being concave to convex) exactly where the second derivative is zero. □



Figur 5: *The function $f(x) = x\sin(x)$ together with the derivative and the second derivative. $f(x)$ has inflexion points where the second derivative is zero.*

Numerical computation

The MATLAB command `diff` performs works in two ways. When operating on a symbolic expression it computes the derivative. When operating in a numerical vector it computes the differences between the elements in the vector. By dividing with the step size of the grid on which f is computed we get the derivative.

`diff(y)`	calculates differences between adjacent elements of the vector y i.e. $[y(2)-y(1), y(3)-y(2), \ldots, y(n)-y(n-1)]$.

Example 20. To compute and plot the first and second derivative of the function in example 19 we give the commands

```
h = 0.001;          % step size
x = 0:h:12;         % interval
f = x.*sin(x);      % function values
df = diff(f)/h;     % first derivative approximated as (f(i+i)-f(i))/h
d2f = diff(df)/h;   % second derivative
plot(x,f,'-',x(1:length(df)),df,'--',x(1:length(d2f)),d2f,':')
xlabel('x'), ylabel('y')
legend('f(x)','Df(x)','D^2f(x)')
grid on             % grid lines
```

We will get a plot very similar to the one in example 19.

2.5 Taylor expansion

A function $f(x)$ that is sufficiently differentiable around a point $x = a$ can be approximated with a Taylor polynomial

$$f(x) \approx f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \ldots + \frac{f^{(n-1)}(a)}{(n-1)!}(x-a)^{(n-1)}.$$

The error in the approximation is a given by a reminder that can be written on the form

$$R_n(x) = \frac{f^{(n)}(a+\theta(x-a))}{n!}(x-a)^n,$$

where $0 \leq \theta \leq 1$. Taylor expansions and estimates of the reminder play a very important role both in mathematics and numerical analysis.

Analytical computation

We have the following command for computing Taylor polynomials.

`taylor(f,x,a,'order',n)`	gives the Taylor polynomial of degree $n-1$ to f, which is a functions of x, around a point a. When n is left out the default value 6 is used.

Example 21. We have the function

$$f(x) = e^{-x}.$$

(a) The Taylor polynomial of degree 3 around $x = 0$ is given by

```
syms x
f = exp(-x);
p = taylor(f,x,0,'order',4); pretty(p)
```

MATLAB answers

```
   3    2
  x    x
- -- + -- - x + 1
   6    2
```

(b) The Taylor polynomial of degree 2 around the point $x = 1$ is obtained by

```
syms x
f = exp(-x);
p = taylor(f,x,1,'order',3); pretty(p)
```

which gives

```
                                        2
                              exp(-1) (x - 1)
exp(-1) - exp(-1) (x - 1) + ---------------
                                    2
```

□

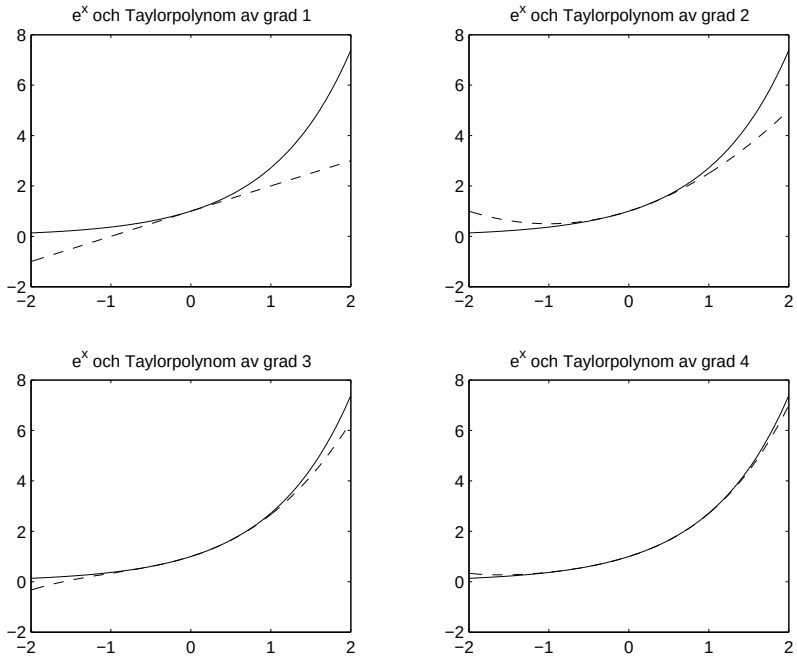
Example 22. The following commands plots the function

$$f(x) = e^x$$

together with the Taylor polynomial of degree 1 to 4 around the point $x = 0$

```
syms x
f = exp(x);
xv = linspace(-2,2);
fv = subs(f,x,xv);
for i = 2:5
  p = taylor(f,x,0,'Order',i);
  pv = subs(p,x,xv);
  subplot(2,2,i-1)
  plot(xv,fv,xv,pv,'--')
  ylim([-2 8])
  string = num2str(i-1,'%2.0f');
  title(['e^x and Taylor polynomial of degree ' string])
end
```

The plots are shown in figure 6. □



Figur 6: *The function e^x together with Taylor polynomials of degree 1 to 4. Indeed the approximation is very good and polynomials are so much easier to work with than many other classes of functions.*

2.6 Local maxima and minima

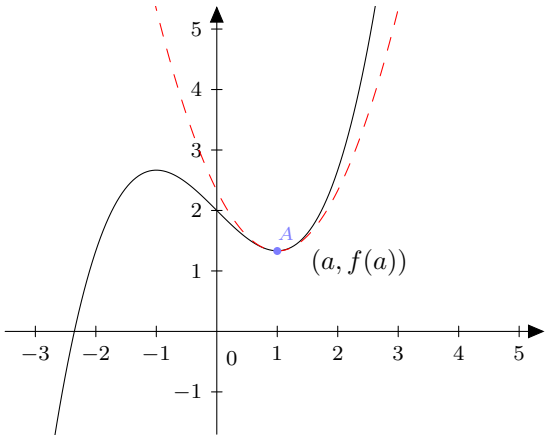
Determining a local or global maxima or minima is known as optimization. To find a local maxima or minima one computes the derivative and sets it to zero.

$$f'(x) = 0.$$

The solutions to this equations are known as stationary points. A stationary point $x = a$ can be a local maximum, a local minimum or a terrace point. By approximating the function around $x = a$ with the second degree Taylor polynomial

$$f(x) \approx f(a) + \underbrace{f'(a)(x-a)}_{0} + \underbrace{\frac{f''(a)}{2!}(x-a)^2}_{\text{parabola.}}$$

we infer that if $f''(a) > 0$ (parabola bending upwards) we have a local minimum and if $f''(a) < 0$ (parabola bending downwards) we have a local maximum. If the second derivative is zero we can not tell which case we have, but have to include more terms in our investigation.



Figur 7: *Let $x = a$ be a stationary point, i.e. $f'(a) = 0$. If $f''(a) > 0$ the function is approximated by an upward bending parabola, and we have a local minimum.*

Analytical computation

Example 23. We have the function $f(x) = \dfrac{x+1}{x^2+x+1}$. The following commands determine stationary points and compute the value of the second derivative from which we can infer if the point is a local maximum or a local minimum. The function is also plotted.

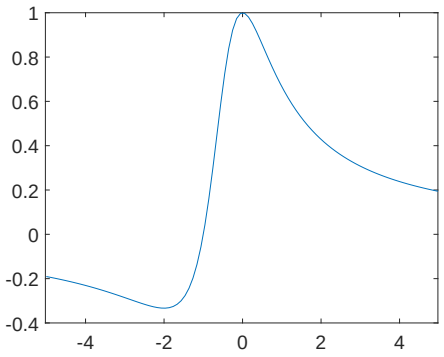
```
syms x
f = (x+1)./(x.^2+x+1);
df = diff(f,x);
a = solve(df,x);
d2f = diff(f,x,2);
for i = 1:2
   disp('point'), disp(a(i))
   disp('second derivative'), disp(subs(d2f,x,a(i)))
end
```

```
xv = linspace(-5,5);
fv = subs(f,x,xv);
plot(xv,fv)
```

MATLAB outputs.

```
point
-2
second derivative
2/9
point
0
second derivative
-2
```

Based on this we know that $x = -2$ is a local minimum and $x = 0$ is a local maximum. The plot is given in figure 8. □



Figur 8: *The function* $f(x) = \dfrac{x+1}{x^2+x+1}$. *The points* $x = -2$ *and* $x = 0$ *are stationary points. For* $x = -2$ *the second derivative is positive, and we have a local minimum. For* $x = 0$ *the second derivative is negative, and we have a local maximum.*

Numerical computation

In MATLAB there are two commands to numerically determine an x that minimizes a function. The first command determines x in given interval. The other command requires an approximate starting value x. There are no commands to determine x that maximizes a function $f(x)$. Instead, this is done by determining the x that minimizes the negative function $-f(x)$.

`fminbnd(fun,x1,x2,opt)`	gives the x-value for a local mimimum to the function *fun* in the interval $[x_1, x_2]$. The parameter *opt*, which is optional, guides print out and relative accuracy for the computation (see `optimset` for more information).
`[x,y,flag,inf] = fminbnd(fun,x1,x2,opt)`	as above. Returns minimum and corresponding function values in x and y, respectively.
`fminsearch(fun,x0,opt)`	gives the x-value for a local minimum to the function *fun* close to a given start value x_0. Works also for functions of several variables.
`[x,y,flag,inf] = fminsearch(fun,x0,opt)`	as above. Returns minimum and corresponding function values in x and y, respectively.

Example 24. Consider the function

$$y = g(x) = \frac{1}{(x-5)^2+1} - \frac{x\sin(x)}{x^2+2}, \quad 0 \le x \le 10.$$

(a) To determine an x that minimizes the function in the interval $[0, 10]$ start by defining the function

```
g = @(x)  1./((x-5).^2+1) - x.*sin(x)./(x.^2 + 2);
```

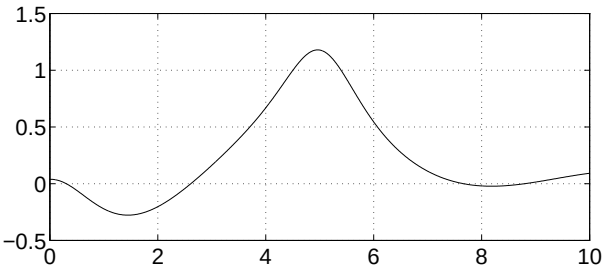
Then give the command

```
[x,y] = fminbnd(g,0,10)
```

and MATLAB answers

```
x =
    8.1838
y =
   -0.0225
```

When plotting the function we see that $x = 8.1838$ is not a global but a local mimimum. To get the global minimum we must, in this case, start from a smaller interval e.g. $[0, 2]$.



Figur 9: *The function* $y = g(x) = \dfrac{1}{(x-5)^2+1} - \dfrac{x\sin(x)}{x^2+2}$

(b) To get more information on the computation use the parameter *opt*. The command

```
x = fminbnd(@g,0,10,optimset('disp','iter'))
```

gives

```
Func-count     x          f(x)         Procedure
    1        3.81966     0.562278       initial
    2        6.18034     0.433635       golden
    3        7.63932   0.00187389       golden
    4        8.54102   -0.0142422       golden
    5        8.16604   -0.0224377       parabolic
    6        8.20865   -0.0224162       parabolic
    7        8.18342   -0.0224603       parabolic
    8        8.18382   -0.0224603       parabolic
    9        8.18375   -0.0224603       parabolic
   10        8.18372   -0.0224603       parabolic

Optimization terminated successfully:
 the current x satisfies the termination criteria using
 OPTIONS.TolX of 1.000000e-004
x =
    8.1838
```

After some initial step with the so called golden search, a faster parabolic method is used to localize the minima. There is a local minimum in $x = 8.18372$ and the corresponding function values is $g(x) = -0.0224603$.

(c) To determine an x that maximizes the function $g(x)$ begin by defining the negative function $-g(x)$

```
gneg = @(x) - 1./((x-5).^2+1) + x.*sin(x)./(x.^2 + 2) ;
```

and give the command

```
x = fminbnd(gneg,0,10)
```

MATLAB answers

```
x =
    4.9614
```

which is the x-value maximizing the function $g(x)$. □

2.7 Integrals

We begin with some geometrical interpretations of integrals.

Area with sign under the graph

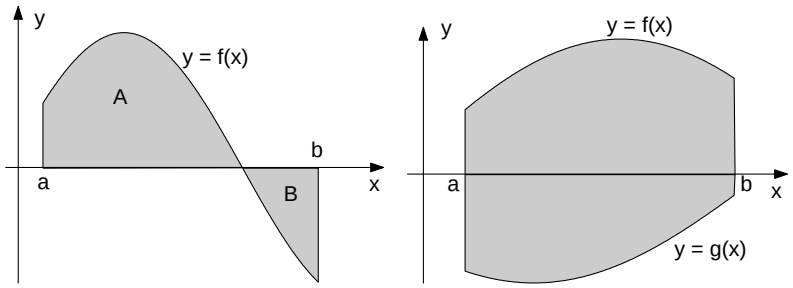
The integral

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

can be interpreted as the area under the graph in the interval $[a, b]$. The area is counted with sign, and parts above the x-axis give positive contributions whereas parts below the x-axis give negative contributions. The area between $f(x)$ and $g(x)$ is given by

$$I = \int_a^b (f(x) - g(x))\,dx.$$

The interpretation is shown in figure 10.



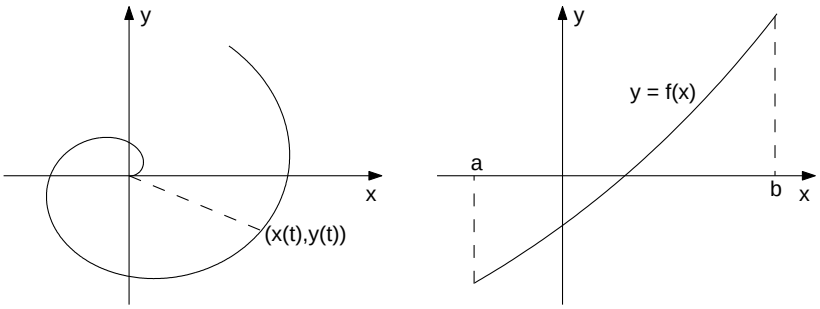
Figur 10: *Integrals can be interpreted in terms of area. Parts above the x-axis give positive contributions and parts below give negative contributions.*

Curve length

A parameter curve in the plane is given by $(x(t), y(t))$, where t is a parameter in some interval $[a, b]$. The length of the parameter curve is given by

$$L = \int_a^b \sqrt{x'(t)^2 + y'(t)^2}\, dt.$$

In mechanical applications $(x(t), y(t))$ is the position of a particle as a function of time t. The integral above is then interpreted as the distance traveled by the particle between time $t = a$ and $t = b$ (see plot to the left in figure 11.).



Figur 11: *Parameter curve $(x(t), y(t),\ a \le t \le b$ and function curve $y = f(x),\ a \le x \le b$.*

Ordinary function curves $(x, f(x))$, where x is in the interval $[a, b]$ (see plot to the right in figure 11), are special cases of the more general parameter curves. Now x itself is the parameter. The length of the function curve is given by the formula

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

We take an example to show the integrals that occur when determining curve length.

Example 25.

(a) A particle is moving along an ellipsoidal curve described by

$$(x(t), y(t)) = (2\cos(t), \sin(t)).$$

We want to determine the length of the curve when $0 \le t \le 2\pi$. Differentiating the coordinates

$$(x'(t), y'(t)) = (-2\sin(t), \cos(t))$$

and insertion into the formula for the curve length gives

$$L = \int_0^{2\pi} \sqrt{4\sin^2(t) + \cos^2(t)}\, dt.$$

(b) We have the function

$$f(x) = x\sin(x)$$

and want to determine the length of the function curve when $0 \leq x \leq 6\pi$. Derivation

$$f'(x) = \sin(x) + x\cos(x)$$

and insertion into the formula for the curve length gives

$$L = \int_0^{6\pi} \sqrt{1 + (\sin(x) + x\cos(x))^2}\, dx.$$

□

Rotation volumes

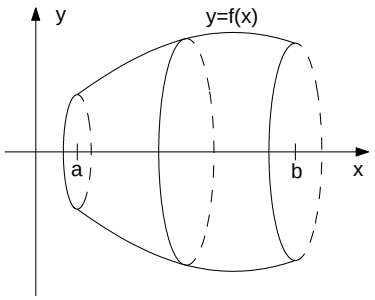
Rotation volumes occur in many applications. A rotation volume is created when a function curve $y = f(x)$, $f(x) > 0$ rotates around the x-axis. The rotation volume is given by

$$V = \pi \int_a^b f(x)^2\, dx.$$

The surface area (mantelytan) of the rotation volume is given by

$$Y = 2\pi \int_a^b f(x)\sqrt{1 + f'(x)^2}\, dx.$$

A typical rotation volume is shown in figure 12.



Figur 12: *Rotation volume created by the function curve* $y = f(x), a \leq x \leq b$.

Example 26. We want to determine the surface area of the rotation volume of the function curve $y = f(x) = \cos x$, $0 \leq x \leq \pi/4$. Differentiation

$$f'(x) = -\sin(x)$$

and insertion into the formula for the surface area gives

$$Y = 2\pi \int_0^{\pi/4} \cos(x)\sqrt{1 + \sin(x)^2}\, dx.$$

□

Analytical computation

Analytically, an integral is computed based on the primitive function $F(x) = \int f(x)\,dx$, and we have the insertion formula

$$\int_a^b f(x)\,dx = [F(x)]_a^b = F(b) - F(a).$$

In MATLAB we have the following commands.

`int(f,x)`	computes $\int f(x)\,dx$.
`int(f,x,a,b)`	computes $\int_a^b f(x)\,dx$.

Example 27.

(a) The primitive function (undetermined integral)

$$\int x^2 \sin(x)\,dx$$

is obtained by

```
syms x
int(x^2*sin(x),x)
```

MATLAB answers

```
ans =
-x^2*cos(x)+2*cos(x)+2*x*sin(x)
```

(b) To compute the integral

$$\int_0^\pi x^2 \sin(x)\,dx$$

we give the commands

```
syms x
int(x^2*sin(x),x,0,pi)
```

which results in

```
ans =
pi^2-4
```

□

2.7.1 Numerical computation

Numerical determination of integrals is called quadrature. In MATLAB we have the following commands

`quad(f,a,b,tol)`	computes the integral over the interval $[a, b]$ of the function *f*. The function *f* should be such that it accepts a vector as the input variable. The parameter *tol*, which can be left out, determines the relative accuracy. If *tol* is left out the relative accuracy is set to 10^{-6}.

Example 28.

(a) Let us compute the integral

$$\int_1^4 \sqrt{x}\, dx.$$

We define the integrand as an anonymous function

```
f = @(x) sqrt(x);
```

When we write

```
I1 = quad(f,1,4)
```

MATLAB returns (we now use format long)

```
I1 =
   4.66666664876325
```

We recompute the integral, now with the relative accuracy 10^{-12}

```
I2 = quad(f,1,4,1.e-12)
```

and the answer is

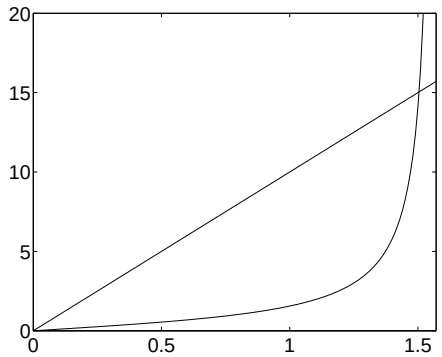
```
I2 =
   4.66666666666667
```

Both values can be compared with the exact value $14/3 \approx 4.66666666667$.

(b) We want to compute the area between the functions $f(x) = 10\,x$ and $g(x) = \tan(x)$ in the interval $[0, \pi/2]$. We start by plotting. Given the plot we try to determine the integration bounds. The commands

```
x = linspace(0,pi/2);
y1 = 10*x; y2 = tan(x);
plot(x,y1,x,y2)
xlim([0 pi/2]), ylim([0 20])
```

gives figure 13. The functions $f(x)$ and $g(x)$ cross at $x = 0$. The other crossing is close to $x = 1.5$ and is obtained by solving the equation $h(x) = f(x) - g(x) = 0$.



Figur 13: *The functions* $f(x) = 10\,x$ *and* $g(x) = \tan(x)$.

Defining $h(x) = f(x) - g(x)$ as an anonymous function

```
h = @(x) 10*x - tan(x);
```

the crossing is determined by the command

```
b = fzero(h,1.5)
```

which gives

```
b =
    1.5044
```

Now, to compute the area, we give the command

```
A = quad(h,0,b)
```

and MATLAB answers

```
A =
   8.6032
```

□

Study questions

1. Numerical determination of roots to equations is an iterative process. What is meant by this?
2. Let a and b be two points. Ge a condition such that a continuous function $f(x)$ is guaranteed to have a zero value in the interval $[a, b]$.
3. What is meant by an inflexion point?
4. How do we analytically compute the first and second derivative of a function in MATLAB?
5. Describe how the second derivative can be used to characterize a stationary point.
6. What is meant by optimization?
7. We have a function $f(x)$ in an interval $[a, b]$. What is the difference between a local and a global mimimum. Draw a graph and explain.
8. How can MATLAB be used to determine a local maxima or minima to a function $f(x)$. Account for both the analytical and numercal cases?
9. Give the geometrical interpretation of the integral $\int_a^b f(x)\,dx$.
10. What is a parameter curve and how do you compute the curve length?
11. How do you compute the length of a function curve?
12. How do you compute the volume of a rotation volume?
13. How do you compute the surface area of a rotation volume?
14. Explain how integrals can be computed a) analytically and b) numerically.

Exercises

1. Determine all roots to the equations

   (a) $\tan(x) - x + 1 = 0, \quad 0 \le x \le 3\pi$

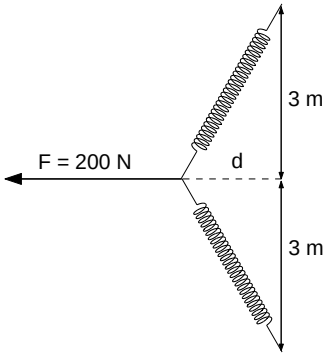
   (Warning: the function is not continuous for $x = n\pi$. Make sure you don't get any false roots.)

   (b) $\sin(x) - 0.3e^x = 0, \quad x \ge 0$

   (c) $0.1x^3 - 5x^2 - x + e^{-x} = -4, \quad x \ge 0$

   Hint: plot the functions to get starting values for the command `fzero`.

2. A force of 200 N acts on two identical spring.



   The springs have spring constants $k = 400$ N/m and unsuspended lengths 3 m. Determine d. Hint: the force of the upper spring is

   $$400(\sqrt{3^2 + d^2} - 3).$$

   The horizontal component is

   $$400(\sqrt{3^2 + d^2} - 3)\frac{d}{\sqrt{3^2 + d^2}}.$$

   At equilibrium this component is 100 N.

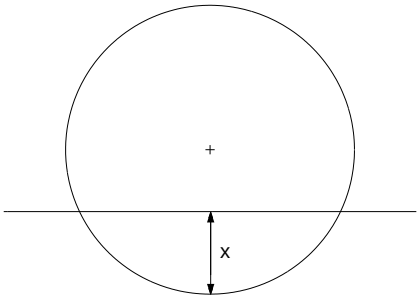
3. van der Waals equaion is an extension of the gas law and relates the number of molecules, pressure, volume and temperature of a gas

   $$P = \frac{nRT}{V - nb} - a\left(\frac{n}{V}\right)^2, \quad V > nb.$$

   Here n is the number of mol of the gas, P is the pressure in atmospheres, V volume in liters, T the temperature in Kelvin, $R = 8.20578 \times 10^{-2}$ L atm K^{-1} mol^{-1} is the gas constant. Estimate the volume in liters of 2 mol CO_2 at $T = 500$ K and $P = 100$ atm. Use the consonants a and b from the table below

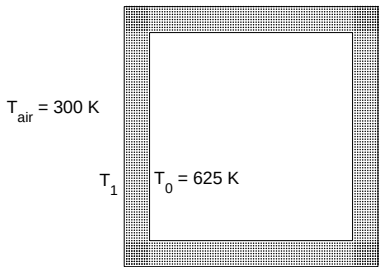
	a (atm L^2 mol^{-1})	b (L mol^{-1})
Ar	1.363	3.219×10^{-2}
CO_2	3.640	4.267×10^{-2}
He	0.057	2.370×10^{-2}
N_2	1.408	3.913×10^{-2}

4. Archimedes' principle states that the upward buoyant force that is exerted on a body immersed in a fluid, whether fully or partially, submerged, is equal to the weight of the fluid that the body displaces. Use the principle to determine how deep a ball of plastic with radius 1 m and density $\rho_p = 700$ kg/m^3 floats in water with density $\rho_v = 1000$ kg/m^3.



Hint: volume of the submerged part $V = \pi x\left(x - x^2/3\right)$ leading to an upward buoyant force of $\rho_v g \pi x\left(x - x^2/3\right)$ where $g \approx 9.81$ m/s^2. At equilibrium this force should balance weight of the ball.

5. We have a furnace made of refractory material with thickness 0.05 m. The inner wall of the furnace has a temperature of $T_0 = 625$ K. The temperature T_1 of the outer wall is unknown. The temperature of the surrounding air is $T_{air} = 300$ K.



Energy balance for the outer wall

$$\underbrace{\frac{k}{\Delta x}(T_0 - T_1)}_{\substack{\text{incoming heat from furnace} \\ \text{by conduction}}} = \underbrace{\epsilon\sigma(T_1^4 - T_{air}^4)}_{\substack{\text{outgoing heat to} \\ \text{air from radiation}}} + \underbrace{h(T_1 - T_{air})}_{\substack{\text{outgoing heat to air} \\ \text{from conduction}}}$$

where

k	heat conduction coefficient for the refractory material, 1.2 W/mK
ϵ	emissivity, 0.8
T_0	temperature of inner wall, 625 K
T_1	temperature of outer wall(unknown)
T_{air}	temperature of surrounding air, 300 K
h	heat transfer coefficient, 20 W/m^{2}K
σ	Stefan-Boltzmann's constant, 5.67×10^{-8}W/m^{2}K^4
Δx	wall thickness, 0.05 m

Compute the temperature T_1.

5. Compute the first and second derivative of $f(x) = \dfrac{x^2}{x-1}$ by hand. Check the result with MATLAB.

6. Consider the function $y = \ln(x)$. Compute the Taylor polynomial of degree 1 to 4 around the point $x = 1$. Do the computation by hand and using MATLAB. Plot the original function and the Taylor polynomial in the same figure.

6. Determine local and global minima to the functions

   (a) $f(x) = -e^{-x}\sin(4x) \quad 0 \le x \le 3$

   (b) $f(x) = \cos(2x) + \sin(x) + e^{-x^2}, \quad -2 \le x \le 2$

   (c) $f(x) = \dfrac{1}{(x-3)^2+1} - \dfrac{x}{(x-1)^2+0.1}, \quad 0 \le x \le 5$

   Hint: start to plot the functions.

7. Determine global maxima to the functions above.

8. Compute the integral $\int x^2 \sin(x)\,dx$ by hand. Check the result by running MATLAB.

9. Compute the integral $\int_0^{10} x\sin(x^2)\,dx$ numerically and analytically. Compare the values.

10. A particle is moving along the curve $(x(t), y(t)) = (t\cos(5t), \sin(3t))$, $0 \le t \le 3$.

    (a) Plot the curve of the particle.

    (b) Compute the distance the particle has moved.

11. We have an ellipsoid $(x(t), y(t)) = (8\cos(t), \sin(t)), \quad 0 \le t \le 2\pi$.

    (a) Plot the ellipsoid.

    (b) Plot the length of the ellipsoid.

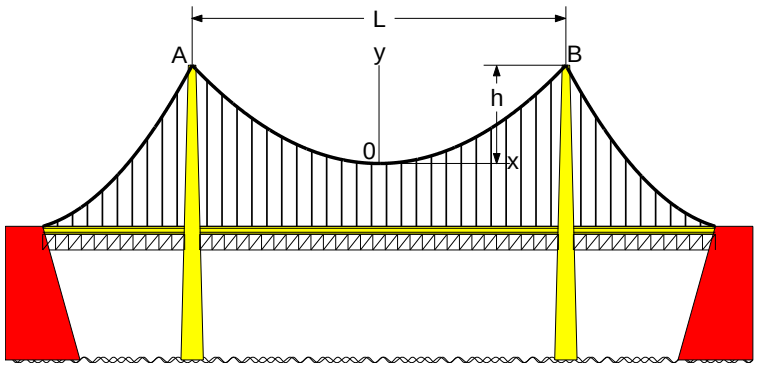
    (c) Compare the computed length with the value you get from the formula of Ramanujan (Srinivasa Ramanujan (1887-1920), autodidact Indian mathematician)

$$L \approx \pi\left(a + b + \frac{3(a-b)^2}{10(a+b) + \sqrt{a^2 + 14ab + b^2}}\right),$$

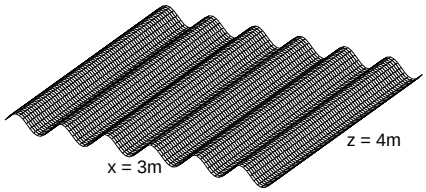
    where a and b are the ellipsoidal half axes. Give the relative error in percent..

12. Consider the bridge in figure below. The distance L between the pylons A and B is 120 m and h is 30 m. Determine the length of the parabolic shaped cable from from A to B.

    Hint: Introduce a coordinate system according to the figure. Determine the constant c_1 so that the parabola $y = f(x) = c_1x^2$ passes A and B. Insert $f(x)$ in the formula $L = \displaystyle\int_a^b \sqrt{1 + f'(x)^2}\,dx$.



13. A factory makes ribbed tin roofs of the dimension 3 m $\times$ 4 m, see figure below. The profile of the ribbed roof can be described by $y = f(x) = 0.02\sin(4\pi x)$, $0 \leq x \leq 3$. Compute how many square meters of tin plate that is needed to compute the roof. Hint: utilize the periodicity of the integrand.



14. The intensity of refracted light close to a sharp edge is given by the Fresnel integrals

$$C(x) = \int_0^x \cos\left(\frac{\pi t^2}{2}\right)\, dt$$

and

$$S(x) = \int_0^x \sin\left(\frac{\pi t^2}{2}\right)\, dt.$$

Use the command `quad` to compute theses integrals for enough x-values to make smooth plots of $C(x)$ and $S(x)$ in the interval $0 \leq x \leq 5$.

3 Calculus in several variables

In this section we will study functions of several variables. For simplicity we look at functions

$$z = f(x, y),$$

where (x, y) is in some domain. The discussions are easily generalized to functions of more variables.

3.1 3D-plots

We have the following commands to generate grids in the xy-plane and plot functions.

`[X,Y] = meshgrid(x,y)`	given x and y with points in the intervals $[a, b]$ and $[c, d]$, respectively, grid matrices X_{ij} and Y_{ij} are generated.
`mesh(X,Y,Z)`	plots the function given by X_{ij}, Y_{ij} and Z_{ij}.
`meshz(X,Y,Z)`	works as `mesh` but gives reference lines to the xy-plane.
`meshc(X,Y,Z)`	works as `mesh` but plots also level curves in the xy-plane.

Example 29.

(a) The following commands plots $z = f(x, y) = 1 + xe^{-x^2-y^2}$ in $-2 \leq x \leq 2$, $-2 \leq y \leq 2$.

```
x = -2:0.2:2; y = -2:0.2:2;
[X,Y] = meshgrid(x,y);          % generate grid matrices
Z = 1+X.*exp(-X.^2 - Y.^2);     % compute the function values
mesh(X,Y,Z)                     % plot
xlabel('x'), ylabel('y'), zlabel('z')
```

The generated plot is shown to the left in figure 14.

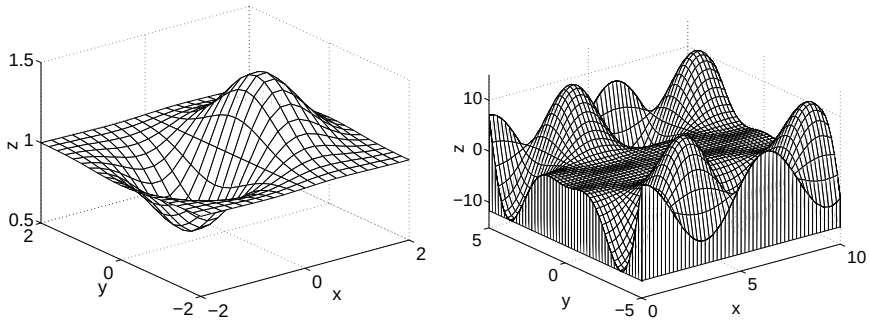
(b) The following commands plots$z = f(x, y) = y^2 \cos(x) \cos(y)$ in $0 \leq x \leq 10$, $-5 \leq y \leq 5$ using the command `meshz`

```
x = 0:0.2:10;
y = -5:0.2:5;
[X,Y] = meshgrid(x,y);
Z = Y.^2.*cos(X).*cos(Y);
meshz(X,Y,Z)
xlabel('x'), ylabel('y'), zlabel('z')
```

The plot is shown to the right in figure 14. □

Example 30. 3D-plots can be rotated in space. In the menu row, click on *Tools* and choose *Rotate 3D*. Place yourself on the plot and press the left button on the mouse and rotate. The commands

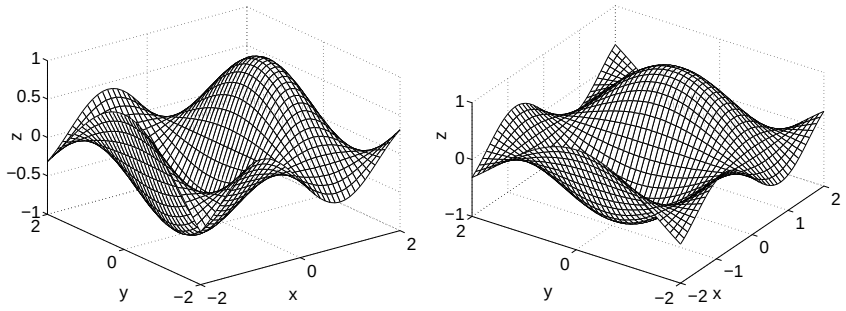
```
x = -2:0.1:2;
y = -2:0.1:2;
```



Figur 14: *Functions plotted with* `mesh` *and* `meshz`.

```
[X,Y] = meshgrid(x,y);
Z = sin(2*X).*cos(Y);
mesh(X,Y,Z)
xlabel('x'), ylabel('y'), zlabel('z')
```

give the left part of figure 15. The right part is obtained after rotating the plot using the mouse. □



Figur 15: *A function viewed from different angles.*

Example 31. To plot functions over more complicated domains we perform a suitable coordinate transformation.

(a) To plot $z = f(x, y) = x^2 + y^2$ over the circular disc $x^2 + y^2 \leq 4$ we use polar coordinates (r, θ) and write

```
r = 0:0.2:2;
theta = linspace(0,2*pi,40);
[R,THETA] = meshgrid(r,theta);  % generate gridmatrices
X = R.*cos(THETA);
Y = R.*sin(THETA);
Z = X.^2 + Y.^2;                % compute the function value
mesh(X,Y,Z)                     % plot
xlabel('x'), ylabel('y'), zlabel('z')
```

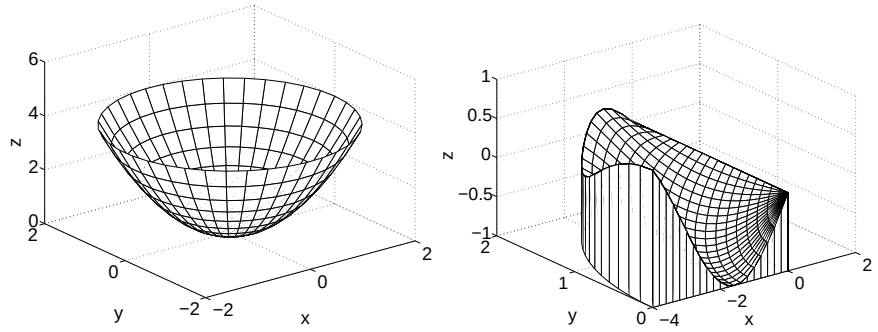
The generated plot is shown to the left in figure 16.

(b) The following commands plot $z = \sin(x)\cos(y)$ in $\{(x, y) \in R^2 \mid x^2/4 + y^2 \leq 4,\ x < 0,\ y > 0\}$ using `meshz`

```
r = 0:0.1:2;
```

```
theta = linspace(pi/2,pi,20);
[R,THETA] = meshgrid(r,theta);
X = 2*R.*cos(THETA);
Y = R.*sin(THETA);
Z = sin(X).*cos(Y);
meshz(X,Y,Z)
xlabel('x'), ylabel('y'), zlabel('z')
```

The plot is shown to the right in figure 16. □



Figur 16: *Functions in over general domains in the xy-plane can be obtained using a suitable variable transformation.*

3.2 Contour plots

As an alternative to the usual plots we can display a function $z = f(x, y)$ using contours or level curves. The level curves are curves in the xy-plane along which $z = f(x, y)$ is constant. Where the level curves are closely spaced the function values are changing rapidly. This works the same way as for a topographic map.

`contour(X,Y,Z)`	gives level curves for the function given by X_{ij}, Y_{ij} and Z_{ij}. The number of level curves automatically set.
`contour(X,Y,Z,n)`	as above but with n level curves.
`contour(X,Y,Z,v)`	gives levels curves for z defined in the vector $v = (v_1, v_2, ..., v_n)$.

Example 32.

(a) The following commands plots

$$z = \frac{1}{0.2x^2 + y^2 + 0.3} + \frac{1}{(x-2)^2 + (y+2)^2 + 0.5}$$

in $-2 \leq x \leq 4$, $-4 \leq y \leq 2$.

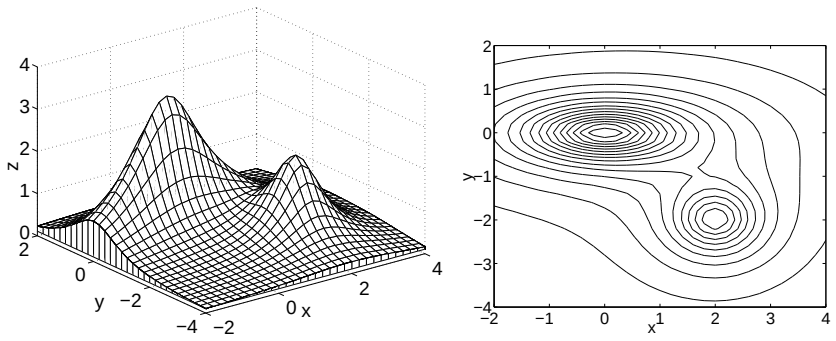
```
x = -2:0.2:4; y = -4:0.2:2;
[X,Y] = meshgrid(x,y);
Z = 1./(0.2*X.^2 + Y.^2 + 0.3) ...
  + 1./((X-2).^2 + (Y+2).^2 + 0.5);
meshz(X,Y,Z)
xlabel('x'), ylabel('y'), zlabel('z')
```

The generated plots is shown to the left in figure 17.

(b) To get the corresponding level curves we write

```
x = -2:0.2:4; y = -4:0.2:2;
[X,Y] = meshgrid(x,y);
Z = 1./(0.2*X.^2 + Y.^2 + 0.3) ...
  + 1./((X-2).^2 + (Y+2).^2 + 0.5);
contour(X,Y,Z,15)
xlabel('x'), ylabel('y')
```

The generated plots is shown to the right in figure 17. □



Figur 17: *Function plot and corresponding level curves*

Example 33.

(a) The following commands plots $z = 1 + xe^{-x^2-y^2}$ in the domain $-2 \leq x \leq 2$, $-2 \leq y \leq 2$.

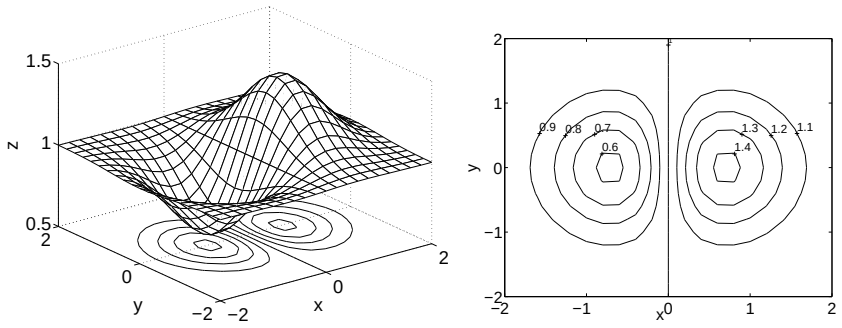
```
x = -2:0.2:2; y = -2:0.2:2;
[X,Y] = meshgrid(x,y);
Z = 1+X.*exp(-X.^2 - Y.^2);
meshc(X,Y,Z)
xlabel('x'), ylabel('y'), zlabel('z')
```

The generated plot is shown to the left in figure 18.

(b) To get the corresponding level curves with z labels we write

```
x = -2:0.2:2; y = -2:0.2:2;
[X,Y] = meshgrid(x,y);
Z = 1+X.*exp(-X.^2 - Y.^2);
C = contour(X,Y,Z); clabel(C)
xlabel('x'), ylabel('y')
```

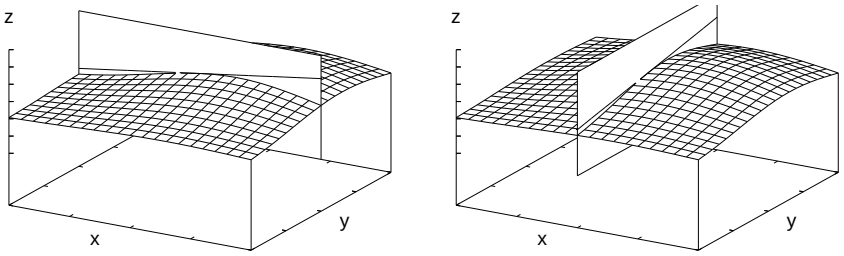
The corresponding plot is shown to the right in figure 18. □



Figur 18: *Function plot and level plots where z for each of the levels are given.*

3.3 Gradient and Hessian matrix

Let $f(x, y)$ be a function of two variables (three and more variables are treated in an analogue manner). The partial derivatives $f'_x(x, y)$ and $f'_y(x, y)$ determine the rate of growth in the x- and y-directions at the point (x, y). The interpretation of the partial derivatives is illustrated in figure 19.



Figur 19: *The partial derivatives $f'_x(x, y)$ and $f'_y(x, y)$ determine the rate of growth in the x- and y-directions, respectively.*

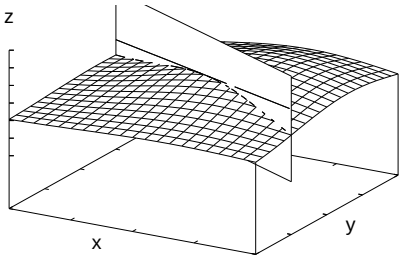
The partial derivatives är component of the so called gradient vector or Jacobian

$$\text{grad} f(x, y) = \left(f'_x(x, y), f'_y(x, y)\right).$$

The gradient points in the direction in the xy-plane in which $f(x, y)$ grows the fastest and is perpendicular to the level curve through the point (x, y). The length of the gradient is equal to the rate of growth in the gradient direction. The fact that the gradient is perpendicular to the level curves is often used in numerical analysis. The rate of growth in a general direction $\mathbf{v}$ in the xy-plane starting from (x, y) is denoted $f'_{\mathbf{v}}(x, y)$ and is given by the scalar product of the gradient and the normalized direction vector

$$f'_{\mathbf{v}}(x, y) = \text{grad} f(x, y) \cdot \frac{\mathbf{v}}{|\mathbf{v}|}.$$

The rate of growth in a general direction $\mathbf{v}$ is illustrated in figure 20.



Figur 20: *The rate of growth* $f'_{\mathbf{v}}(x,y)$ *in a general direction* $\mathbf{v}$ *in the* xy*-plane.*

The gradient is the many-dimensional equivalent of the first derivative. The many-dimensional equivalent of the second derivative is called the Hessian matrix

$$H(x,y) = \begin{pmatrix} f''_{xx}(x,y) & f''_{xy}(x,y) \\ f''_{yx}(x,y) & f''_{yy}(x,y) \end{pmatrix}.$$

When $f(x,y)$ is two time continuously differentiable $f''_{xy}(x,y) = f''_{yx}(x,y)$ and the Hessian matrix is symmetric.

Analytical computation

In MATLAB we have the following commands to determine gradients and Hessian matrices

`jacobian(f,v)`	computes the gradient with respect to the vector v.

Example 34.

(a) Consider the function

$$f(x,y) = x\sin(x/y).$$

To compute the gradient

$$\text{grad} f(x,y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

we give the command

```
syms x y
f = x*sin(x/y);
gradf = jacobian(f,[x y]);
pretty(gradf)
```

MATLAB answers

```
/                 / x \     2    / x \ \
|            x cos| - |    x  cos| - | |
|    / x \         \ y /          \ y / |
| sin| - | + ----------, - ----------- |
|    \ y /         y               2    |
\                                   y    /
```

(b) We have the function

$$f(x,y) = \frac{x^2}{\ln(xy)}.$$

The gradient in the point $(1, 8)$ is given by

```
syms x y
f = x^2/log(x*y);
gradf = jacobian(f,[x y])
subs(gradf,[x y],[1 8])
```

and we get the answer

```
/    2        1           1     \
| ------ - -------, - --------- |
| log(8)         2            2 |
\          log(8)     8 log(8)  /
```

To get the corresponding numerical value give the command `double`.

(d) We have a function $f(x, y)$. To compute the Hessian matrix to $f(x, y) = \sin(x\,y + 1)$ use the command `jacobian` two times

```
syms x y
H = jacobian(jacobian(sin(x*y+1),[x y]),[x y]);
pretty(H)
```

The matrix is

```
/           2                                                    \
|        - y  sin(x y + 1),        cos(x y + 1) - x y sin(x y + 1) |
|                                                                 |
|                                          2                      |
\ cos(x y + 1) - x y sin(x y + 1),        - x  sin(x y + 1)        /
```

□

Numerical computation

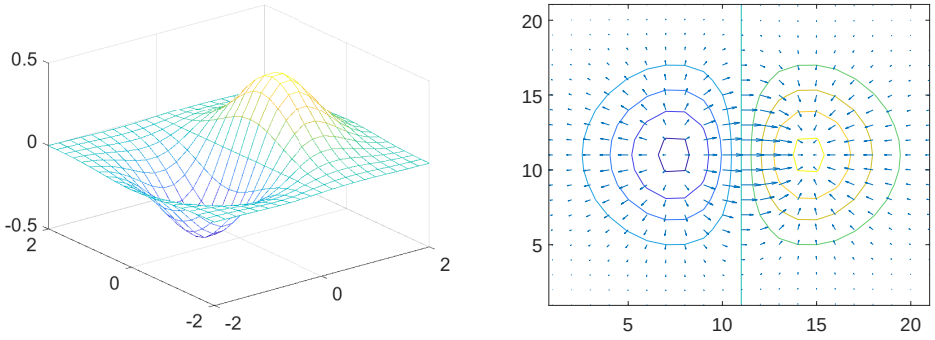
There are also commands to compute the gradient numerically and to plot the gradient vector in different points in the plane.

`[Fx,Fy]=gradient(F,h,k)`	returns the numerical gradient of the matrix F. FX corresponds to dF/dx, the differences in x (horizontal) direction. FY corresponds to dF/dy, the differences in y (vertical) direction. The spacing between points in each direction is h and k, respectively.
`quiver(Fx,Fy)`	plots gradient vectors at equally spaced points the xy-plane. The length of the vectors automatically scaled to give a nice plot.

Example 35. We have a function $f(x, y) = xe^{-x^2-y^2}$ with $-2 \leq x \leq 2$ and $-2 \leq y \leq 2$ which is displayed to the left in figure 21. The following commands compute the gradient vector at a number of points and plots it on top of the contour plot.

```
[X,Y] = meshgrid(-2:0.2:2,-2:0.2:2);
Z = X.*exp(-X.^2 - X.^2);
[Fx,Fy] = gradient(Z,0.2,0.2);
contour(Z)
hold on
quiver(Fx,Fy)
```

The gradient vectors are shown to the right in figure 21.



Figur 21: *Gradient vector computed at a number of points. Note that the gradient vectors are perpendicular to the level curves.*

3.4 Tangent plane and Taylor polynomials

Let $f(x, y)$ be a differentiable function in a domain around (a, b). Just as in one-dimensional calculus the function can be approximated by a Taylor polynomial. The Taylor polynomial of degree one is called the tangent plane and we have

$$z = f(a, b) + f'_x(a, b)h + f'_y(a, b)k,$$

where $(h, k) = (x - a, y - b)$. It is convenient to write the tangent plane in matrix form which gives

$$z = f(a, b) + \text{grad} f(a, b)(h, k)^T.$$

The tangent plane only gives a rough approximation of the function. A better approximation is given by the polynomial of degree two

$$\begin{aligned} z = f(a, b) &+ f'_x(a, b)h + f'_y(a, b)k \\ &+ \frac{1}{2}(f''_{xx}(a, b)h^2 + 2f''_{xy}(a, b)hk + f''_{yy}(a, b)k^2). \end{aligned}$$

The polynomial is normally written in terms of the gradient and the Hessian matrix

$$z = f(a, b) + \text{grad} f(a, b)(h, k)^T + \frac{1}{2}(h, k)H(a, b)(h, k)^T.$$

Analytical computation

We have the following command for computing Taylor polynomials.

`taylor(f,[x y],[a b],'order',n)`	gives the Taylor polynomial of degree $n-1$ to f, which is a functions of (x, y), around a point (a, b). When n is left out the default value 6 is used.

Example 36. We have a function

$$f(x,y) = e^{-x^2-y^2}.$$

(a) To compute the tangent plane (Taylor polynomial of degree 1) in $(x,y) = (-1/10,-1/10)$ we give the commands

```
syms x y
f = exp(-x.^2-y.^2);
t1 = taylor(f,[x y],[-1/10 -1/10],'order',2)
pretty(t1)
```

and MATLAB replies

```
                  /    1 \ /      1 \      /    1 \ /      1 \
                exp| - -- | | x + -- |   exp| - -- | | y + -- |
   /    1 \       \   50 / \     10 /      \   50 / \     10 /
exp| - -- | + ---------------------- + ----------------------
   \   50 /              5                        5
```

(b) To compute the Taylor polynomial of degree 2 in $(x,y) = (0,0)$ give the commands

```
syms x y
f = exp(-x.^2-y.^2);
t2 = taylor(f,[x y],[0 0],'order',3)
pretty(t2)
```

and MATLAB replies

```
   2    2
- x  - y  + 1
```

□

4 Local maximum and minimum

En extremely important task is to determine the largest or smallest value of a function in a given domain. This is in general very difficult and in many cases we should be happy if we can determine the values in a number of maxima or minima that we are able to localize. If a function has a local extrema in point (a,b) then

$$f'_x(a,b) = 0 \quad \text{and} \quad f'_y(a,b) = 0.$$

A point (a,b) for which the partial derivatives are zero is called a stationary point. The tangent plane in a stationary point is parallel to the xy-planet (see figure 22).

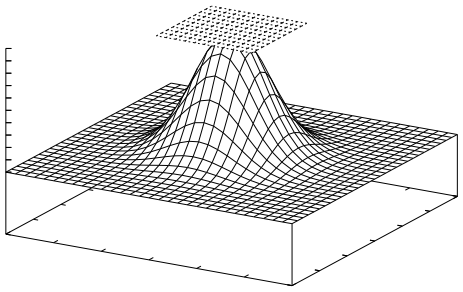
To determine if a stationary point is a local maxima- or minima we determine the eigenvalues of the Hessian matrix in this point. We have the following cases:

(1) Both eigenvalues are positive and we have a local minimum

(2) Both eigenvalues are negative and we have a local maximum

(3) If one eigenvalue is positive and one is negative then we don't have a local maximum or local minimum. Instead we have a saddle point.

(4) If one of the eigenvalues is zero we can not determine the character of the point.



Figur 22: *In a stationary point the partial derivatives are zero. The tangent plane in the point is parallel with the xy-plane.*

Please observe the analogue with the one-dimensional cases where the second derivative determines if we have a maximum or a minimum.

The equation systems that determines the stationary points are often difficult to solve and one normally use numerical methods for this.

Analytical computation

Here we give an example of analytical determination of maxima- and minima and characterization based on the eigenvalues of the Hessian matrix.

Example 37.

(a) We have the function

$$f(x, y) = x^2 - y^2$$

The stationary points are determined by the commands

```
syms x y
f= x^2-y^2;
gradf = jacobian(f,[x y]);
[xst,yst] = solve(gradf(1),gradf(2),x,y)
```

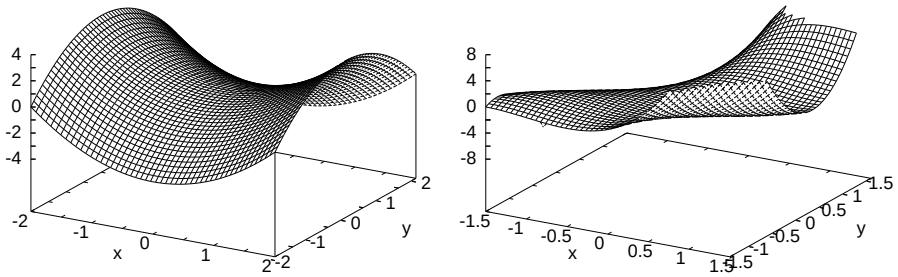
which gives the solution $(0, 0)$. To give the character of the point we determine the Hessian matrix in the point and compute the eigenvalues.

```
hessian = jacobian(gradf,[x y]);
hessian_st = double(subs(hessian,[x y],[0 0]))
eigs(hessian_st)
```

MATLAB returns

```
ans =
    -2
     2
```

There is a positive and a negative eigenvalue in $(0, 0)$ and we have a saddle point. The function is plotted to the left in 23. It is not hard to see why the one has chosen the name saddle point.



Figur 23: *Function surfaces $z = x^2 - y^2$ and $z = 2x^3 - 3x^2 - 6xy(x - y - 1)$. The latter functions has both max-, min- and saddle points, which are more clearly seen when rotating the plot. The points can also bee seen by plotting contours and adding gradient vectors.*

(b) We have the function

$$f(x, y) = 2x^3 - 3x^2 - 6xy(x - y - 1)$$

The stationary points are obtained by determining the values for which the partial derivatives are zero

```
syms x y
f =  2*x^3 - 3*x^2 -6*x*y*(x-y-1)
gradf = jacobian(f,[x,y])
[xst,yst] = solve(gradf(1),gradf(2),x,y)
```

MATLAB returns

```
xst =
  0
  1
  0
 -1
yst =
  0
  0
 -1
 -1
```

There are thus four stationary points. To determine the character of these points we evaluate the Hessian matrix and determine the eigenvalues

```
hessian = jacobian(gradf,[x y]);
for i = 1:4
   hessian_st = double(subs(hessian,[x y],[xst(i) yst(i)]));
   disp(eigs(hessian_st))
end
```

MATLAB returns

```
-9.7082
 3.7082

15.7082
 2.2918
```

```
 9.7082
-3.7082

-15.7082
 -2.2918
```

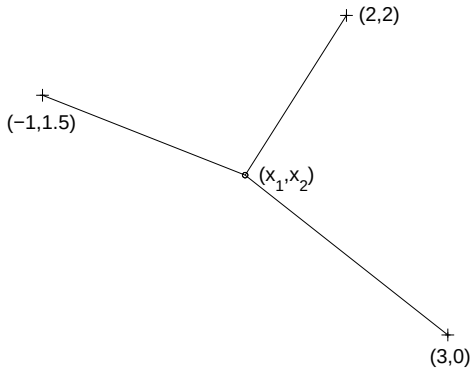
The points $(0,0)$ and $(0,-1)$ are saddle points as the eigenvalues have different signs. The point $(-1,-1)$ is a local maximum as both eigenvalues is negative. Finally, $(1,0)$ is a local minimum as both eigenvalues are negative. The function is plotted to the right in figure 23. □

Numerical computation

We have the following command for finding local mimima for functions of several variables.

`fminsearch(fun,x0,opt)`	gives the x-value for a local minimum to the function *fun* close to a given start vector x_0.
`[x,y,flag,inf] = fminsearch(fun,x0,opt)`	as above. Returns minimum and corresponding function values in x and y, respectively.

Example 38. We want to determine the point $P = (x_1, x_2)$ that minimizes the sum of the quadratic distance from the three points $(-1, 1.5)$, $(2,2)$ and $(3,0)$, see figure 24.



Figur 24: *Minimization of the distance to three given points.*

The sum of the quadratic distance between (x_1, x_2) and the three points is given by

$$d = \underbrace{(x_1+1)^2 + (x_2-1.5)^2}_{\substack{\text{dist. between } (x_1,x_2) \\ \text{and first point}}} + \underbrace{(x_1-2)^2 + (x_2-2)^2}_{\substack{\text{dist. between } (x_1,x_2) \\ \text{and second point}}} + \underbrace{(x_1-3)^2 + (x_2-0)^2}_{\substack{\text{dist. between } (x_1,x_2) \\ \text{and third point}}}$$

The function is defined as an anonymous function

```
d = @(x) (x(1)+1).^2 + (x(2)-1.5).^2 + (x(1)-2).^2 ...
  + (x(2)-2).^2 + (x(1)-3).^2 + x(2).^2;
```

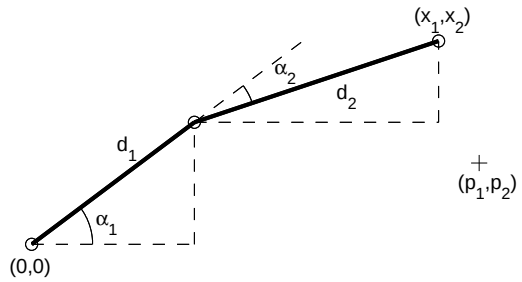
We take the point $(x_1, x_2) = (1,1)$ as a start guess and the point that minimizes the distance is given by

```
xmin = fminsearch(d,[1 1])
```

```
xmin =
    1.3333    1.1667
```

□

Example 39. The position (x_1, x_2) of a robot arm (see figure 25) can be specified by the angle α_1 between the first arm and the horizontal axis and the angle m α_2 between the first and the second arm



Figur 25: *We want to determine the angles α_1 and α_2 so that the arm comes to the desired position (p_1, p_2).*

Our problem is to determine the angles so that the second arm comes to the desired position (p_1, p_2). The position (x_1, x_2) of the second robot arm is given by

$$\begin{cases} x_1 = f_1(\alpha_1, \alpha_2) = d_1 \cos(\alpha_1) + d_2 \cos(\alpha_1 - \alpha_2) \\ x_2 = f_2(\alpha_1, \alpha_2) = d_1 \sin(\alpha_1) + d_2 \sin(\alpha_1 - \alpha_2) \end{cases},$$

Thus, we should determine α_1 and α_2 such that

$$\begin{cases} F_1(\alpha_1, \alpha_2) = f_1(\alpha_1, \alpha_2) - p_1 = 0 \\ F_2(\alpha_1, \alpha_2) = f_2(\alpha_1, \alpha_2) - p_2 = 0 \end{cases}.$$

The equation system above can be transformed to a minimization problem by defining

$$h(\alpha_1, \alpha_2) = F_1(\alpha_1, \alpha_2)^2 + F_2(\alpha_1, \alpha_2)^2.$$

The smallest value of $h(\alpha_1, \alpha_2)$ is 0 and is obtained exactly when $F_1(\alpha_1, \alpha_2)$ and $F_2(\alpha_1, \alpha_2)$ are zero.

We take a concrete example where$d_1 = 5$ and $d_2 = 6$. We want to determine α_1 and α_2 so that the robot arm moves to a position with coordinates $(10, 4)$. We start by defining the function $h(\alpha_1, \alpha_2)$ in a function file with name **h.m**.

```
function y = h(a)
global d1 d2 p1 p2
F1 = d1*cos(a(1)) + d2*cos(a(1) - a(2)) - p1;
F2 = d1*sin(a(1)) + d2*sin(a(1) - a(2)) - p2;
y = F1^2 + F2^2;
```

We take the angles $(\alpha_1, \alpha_2) = (0, 0)$ as a start guess and give the command

```
clear all
global d1 d2 p1 p2
d1 = 5; d2 = 6;
p1 = 10; p2 = 4;
[a,y] = fminsearch(@h,[0 0])
```

```
a =
    0.1560   -0.4112
y =
  2.5663e-009
```

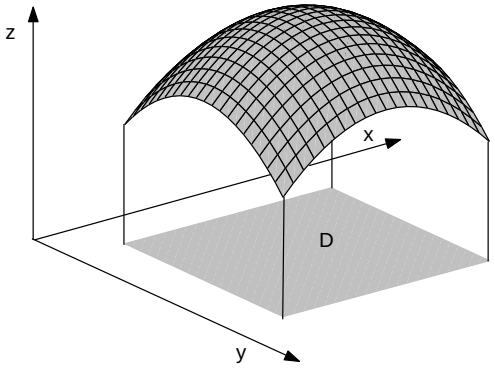
The arm thus moves into the correct position when $\alpha_1 = 0.1560$ and $\alpha_2 = -0.4112$.. Please note that if we define a function f as an anonymous function, as we have done in many examples, the function handle is already included in the definition and the call to a built-in command is `command(f)`. If we define a function f in a function file we need to include the handle in the call, i.e. `command(@f)`. □

5 Double integrals

We will study double integrals (triple and multiple integrals are treated in the same way). The double integral

$$\iint_D f(x,y)\,dxdy$$

can be interpreted as the volume (with sign) under the function surface in the domain D. In the simplest case the domain D is a rectangle, but we can also have more complicated domains (see figure 26).



Figur 26: *The double integral over a domain D is geometrically interpreted as the volume (with sign) under the function surface.*

Double integrals can be computed with iterated integration. Consider the function $f(x,y)$ over the domain D defined by $a \le x \le b$ and $c \le y \le d$. The integral is

$$\iint_D f(x,y)\,dxdy = \int_a^b \left(\int_c^d f(x,y)\,dy \right) dx.$$

The order of the integration does not matter and we also have

$$\iint_D f(x,y)\,dxdy = \int_c^d \left(\int_a^b f(x,y)\,dx \right) dy.$$

We can use the built-in integration commands for this task.

Analytical computation

Repeated use of the following command will facilitate the computation.

`int(f,x,a,b)` computes $\int_a^b f(x)\,dx$.

Example 40.

(a) To compute

$$I = \iint_D (x^2 + y^2 + 2xy + 1)e^x \, dxdy$$

over the domain D defined by $0 \leq x \leq 1$ and $-2 \leq y \leq 3$ we give the commands

```
syms x y
f = (x^2+y^2+2*x*y+1)*exp(x);
I = int(int(f,x,0,1),y,-2,3)
```

and gets

```
(65*exp(1))/3 - 65/3
```

Testing the reversed integration order

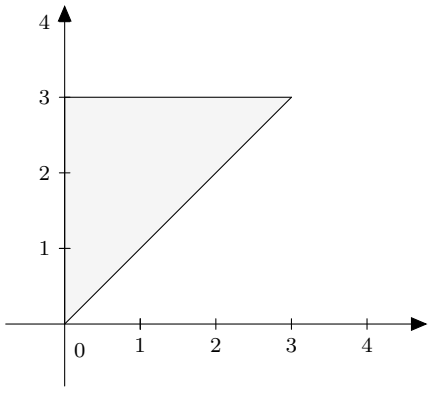
```
syms x y
f = (x^2+y^2+2*x*y+1)*exp(x);
I = int(int(f,y,-2,3),x,0,1)
```

gives the same result.

(b) Iterated integration works also if the domain is more general. To illustrate this we compute the integral

$$I = \iint_D (x + y)\sin(x + y)\, dxdy$$

where D is the triangle area defined by the y-axis, the line $y = x$ and $y = 3$.



Figur 27: *The integration domain D in terms of a triangular area.*

We can do th integration in two ways: first in x (inner) and then in y (outer)

$$I = \underbrace{\int_{y=0}^{y=3} \left(\int_{x=0}^{x=y} (x + y)\sin(x + y)\, dx \right) dy}_{\text{for each } y \text{ between 0 and 3, } x \text{ goes between 0 and } y}$$

or first in y (inner) and then in x (outer)

$$I = \underbrace{\int_{x=0}^{x=3} \left(\int_{y=x}^{y=3} (x+y)\sin(x+y)\,dy \right) dx}_{\text{for each } x \text{ between 0 and 3, } y \text{ goes between } x \text{ and 3}}$$

In MATLAB we have

```
syms x y
f = (x+y)*sin(x+y);
I = int(int(f,x,0,y),y,0,3)
```

or

```
syms x y
f = (x+y)*sin(x+y);
I = int(int(f,y,x,3),x,0,3)
```

Both computations give

```
2*cos(3) - cos(6) + 3*sin(3) - 3*sin(6) - 1
```

Numerical computation

For numerical computation of double- and triple integrals we have the following commands.

`dblquad(fun,a,b, c,d,tol)`	computes the double integral over the domain $[a,b]\times[c,d]$ of the function *fun*. The function *fun* must be such that it accepts a vector as input parameter for the first variable and a scalar for the second.
`triplequad(fun, a,b,c,d,e,f,tol)`	computes the triple integral over the domain $[a,b]\times[c,d]\times[e,f]$ of the function *fun*. *fun* must be such that it accepts a vector as input parameter for the first variable and a scalars for the other two.

Example 41.

(a) To compute the double integral

$$\iint_D x\sin(xy^2)\,dxdy,$$

where $D = [0,1]\times[-1,3]$ we start by defining the function

```
f = @(x,y) x.*sin(x*y^2);
```

The command `dblquad` will internally call the function with a vector as an input parameter for x and a scalar for y and the anonymous function must handle this. Thus we need `.*` between `x` and `sin(x*y^2)`. We now compute the integral by

```
I = dblquad(f,0,1,-1,3)
```

and MATLAB replies

```
I =
    0.5158
```

(b) To compute an integral

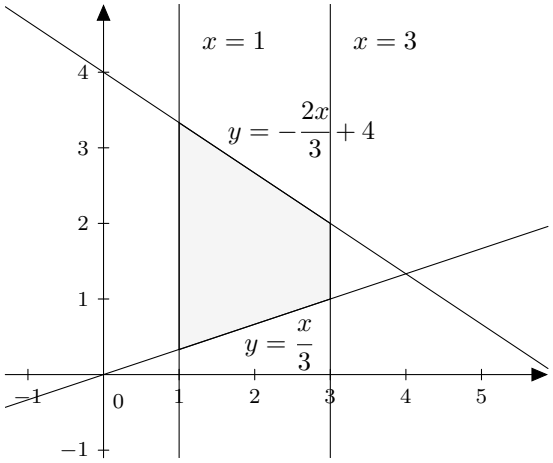
$$\iint_D f(x,y)\,dxdy$$

over domains D that are not rectangular, we put the integrand to zero outside the domain and integrate over the smallest rectangle that contains D.

Compute the integral

$$\iint_D (3x+18y)\,dxdy,$$

where D is the domain that is defined by the lines $x=1$, $x=3$, $y=-\frac{2}{3}x+4$, and $y=\frac{1}{3}x$ (see figure 28).



Figur 28: *Domain D for the double integral.*

We write a function file with name **integrand.m**, that returns function values $3x+18y$ when (x,y) is in the domain D and 0 when (x,y) is outside the domain.

```
function z = integrand(x,y)
n = length(x);
z = zeros(1,n);
for i = 1:n
  if (x(i) >= 1 & x(i) <= 3) & ...
              (x(i)/3 < y & y < -2*x(i)/3 + 4)
    z(i) = 3*x(i) + 18*y;
  end
end
```

The domain D is inscribed in the rectangle $[1,3]\times[\frac{1}{3},\frac{10}{3}]$ and the integral is computed by the commands

```
I = dblquad(@integrand,1,3,1/3,10/3)
```

We get the answer.

```
I =
  144.0000
```

□

Study questions

1. What is meant by contours (or level curves) and how are these plotted?
2. What is the geometrical interpretation of the gradient vector?
3. How do you compute $f'_{\mathbf{v}}(x, y)$ in a general direction $\mathbf{v}$?
4. How is the Hessian matrix defined?
5. What are the commands to compute the gradient and the Hessian matrix, respectively?
7. Make an account for Taylor expansions of the function $f(x, y)$ around a point punkt (a, b).
8. What is meant by a stationary point for a two-variable function? How do you determine a stationary point?
9. How can you use the Hessian matrix to characterize a stationary point?
10. What are the commands for optimization?
11. Explain how the solution of a set of equations

$$\begin{cases} F_1(\alpha_1, \alpha_2, \ldots, \alpha_n) = 0 \\ F_2(\alpha_1, \alpha_2, \ldots, \alpha_n) = 0 \\ \cdots \\ F_n(\alpha_1, \alpha_2, \ldots, \alpha_n) = 0 \end{cases}.$$

    can be transformed into a minimization problem.
12. What is the geometrical interpretation of a double integral over a domain?
13. How do you compute a double integral by iterated integration?

Exercises

1. We have a function $f(x, y) = \sin(x^2 + y^2)$.

    (a) Plot the function over the domain $-2 \leq x \leq 2$, $-2 \leq y \leq 2$.

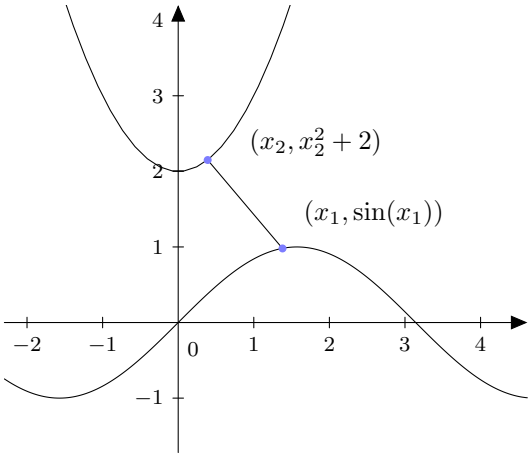
    (b) Plot the function over the domain $x^2 + y^2 < 2^2$.

    (c) Plot the corresponding contours.
2. Compute the gradient vectors to the function above by hand. Check the result with Malan.
3. Compute the gradient vectors numerically and overlay them on the contour plots.
4. Consider the half sphere $z = f(x, y) = \sqrt{49 - x^2 - y^2}$.

    (a) Determine the Tangent plane in the point $(6, 2, 3)$. Plot the function surface and the tangent plane in the same plot. Rotate the figure.

    (b) Determine the Taylor polynomial of degree two in $(6, 2, 3)$
5. Determine local maxima and minima to $f(x, y) = x - \ln(x - y^2)$.

6. We have two functions $f(x) = \sin(x)$ and $g(x) = x^2 + 2$. Determine the shortest distance between the two functions.



Hint: Consider the points $(x_1, \sin(x_1)$ and $(x_2, x_2^2 + 2)$ on the two curves. Compute the distance between the two points using the Pythagorean theorem and minimize this distance with respect to (x_1, x_2).

7. Compute the integral

$$\iint_D (x + y)e^x \, dxdy$$

over the domain D given by $0 \leq x \leq 2$ and $-1 \leq y \leq 2$. Perform the computation by hand. Check your results by computing the integral analytically with MATLAB. Finally, compute the integral numerically.

8. A surface has the temperature distribution

$$T(x, y) = 20 + \frac{30}{(x-2)^2 + (y-1)^2 + 1}, \quad 0 \leq x \leq 3, \ 0 \leq y \leq 5.$$

(a) Plot the distribution.

(b) Compute the mean temperature $\overline{T}$ of the surface.

Hint: $\overline{T} = \dfrac{\iint_D T(x, y) \, dxdy}{(b-a)(d-c)}$ where $D = [a, b] \times [c, d]$.

9. Determine the volume of the area between the function surface

$$z = f(x, y) = \frac{1}{1 + x + y}$$

and the area in the xy-plane defined by $0 < x < 2$ and $0 < y < 1$.

10. Compute numerically the integral

$$\iint_D x^3 y + x^2 y^2 \, dxdy$$

where D is given by $(x + 1)^2 + y^2 < 1$.