Numerical Analysis 23

Lecture Notes 2019

School of Mathematics, University of Bristol

Contents

0 Introduction

This unit is about numerical approximation methods. Most problems that occur in mathematical applications cannot be solved exactly and one has to apply numerical methods to get approximate solutions. The methods that we discuss in this unit apply to the types of problems that occur in most applications:

- Finding roots of an equation, or a system of equations.
- Numerical differentiation and integration.
- Numerical solutions to ordinary differential equations.

In the case of ordinary differential equations one distinguishes further between initial value problems (IVPs) and boundary value problems (BVPs). Partial differential equations are not part of this unit. They are covered in the level M unit Numerical Methods for PDEs (this unit is not taught presently).

The type of questions that we will investigate are

- What kind of methods do exist?
- When does a method converge?
- How rapidly does it converge?
- Is a method stable?
- How big is the error of an approximation?
- What is the effect of round-off errors when doing calculations on a computer?

This unit is not about writing computer programs. The implementation of a numerical method in a computer program is an additional step. For this reason, the knowledge of a computer language is not a prerequisite for this unit.

If you want to read about a topic in more detail then a recommended text is "Numerical Analysis" by Burden and Faires. The document BurdenFaires.pdf on Blackboard lists the different topics of this unit and where you can find them in the book. There are, however, many other good books on Numerical Analysis. Some of them are listed in the unit description. Others can be found in the Numerical Analysis section in the Queen's Library (books QA 297 ***).

These lecture notes are slightly less detailed than the lectures. They contain all the theory, but not all of the examples.

1 Root finding

1.1 Introduction

Finding solutions of an equation $f(x) = 0$, or a set of equations $\mathbf{f}(\mathbf{x}) = \mathbf{0}$, is one of the most basic problems of Numerical Analysis. Equations can either be linear or nonlinear in the unknown variables. Possibilities are

(A) Linear problem in one variable

$$ax + b = 0 \qquad \Longrightarrow \qquad \text{solution is} \quad x = -\frac{b}{a}\,.$$

(B) Linear problem in n variables

A system of linear equations has the form

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &= b_2 \\ \vdots \qquad\qquad & \qquad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n &= b_n \end{aligned}$$

or in matrix-vector form

$$A\mathbf{x} = \mathbf{b}\,,$$

where

$$A = \begin{bmatrix} a_{11} & \ldots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \ldots & a_{nn} \end{bmatrix}, \qquad \mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \qquad \mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}.$$

(C) Nonlinear problem in one variable

Any equation $f(x) = 0$ where $f(x)$ is not in the form $f(x) = ax + b$, e.g. $f(x) = \mathrm{e}^x - 2 - x$.

(D) Nonlinear problem in n variables

A system in the form

$$\begin{aligned} f_1(x_1, x_2, \ldots, x_n) &= 0 \\ f_2(x_1, x_2, \ldots, x_n) &= 0 \\ \vdots \qquad\qquad & \qquad \vdots \\ f_n(x_1, x_2, \ldots, x_n) &= 0 \end{aligned}$$

or in vector notation $\mathbf{f}(\mathbf{x}) = \mathbf{0}$.

Example:

$$\begin{aligned} f_1(x_1, x_2, x_3) &= \sinh(x_1 x_3) - x_1 &= 0 \\ f_2(x_1, x_2, x_3) &= \mathrm{e}^{x_1} - x_2 x_3 + 2 &= 0 \\ f_3(x_1, x_2, x_3) &= x_3^2 - x_2 + x_1^3 &= 0\,. \end{aligned}$$

1.2 Linear system of equations

Consider a system of the form

$$A\mathbf{x} = \mathbf{b}\,. \qquad (*)$$

If the inverse A^{-1} exists then there is a **unique solution**

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

Condition: The inverse A^{-1} exists if and only if $\det A \neq 0$.

On the other hand, if $\det A = 0$, A is said to be **singular** and $(*)$ does not have a unique solution. In this case $(*)$ may have no solution or an infinite number of solutions. To understand these two alternatives we need some results from Linear Algebra. Let A be an $n \times n$ matrix. The kernel and image of A are defined by

$$\begin{aligned} \ker(A) &= \{\mathbf{v} \,|\, A\mathbf{v} = \mathbf{0}\}\,, \\ \mathrm{im}(A) &= \{\mathbf{b} \,|\, \exists\, \mathbf{x} \text{ such that } A\mathbf{x} = \mathbf{b}\}\,. \end{aligned}$$

The dimensions of kernel and image of A are related by the rank-nullity theorem

$$\dim(\ker A) + \dim(\mathrm{im} A) = n\,.$$

If $\det(A) = 0$ then there exist vectors $\mathbf{v} \neq \mathbf{0}$ such that $A\mathbf{v} = \mathbf{0}$. (These are eigenvectors with eigenvalue 0.) This implies that $\dim(\ker A) > 0$ and $\dim(\mathrm{im} A) < n$.

Conclusion: If $\det(A) = 0$ and $\mathbf{b} \notin \mathrm{im} A$ then $A\mathbf{x} = \mathbf{b}$ has no solution. On the other hand, if $\det(A) = 0$ and $\mathbf{b} \in \mathrm{im} A$ then $A\mathbf{x} = \mathbf{b}$ has infinitely many solutions, because if $\mathbf{x}_0$ is a solution then so is $\mathbf{x}_0 + \mathbf{v}$ for any $\mathbf{v} \in \ker A$.

We found the solution $\mathbf{x} = A^{-1}\mathbf{b}$ if $\det(A) \neq 0$. However, numerically it is very expensive to calculate A^{-1} (number of operations). We are going to use a different method to find $\mathbf{x}$.

1.2.1 Triangular systems

If the matrix U is **upper triangular** ($u_{ij} = 0$ if $i > j$) then the system $U\mathbf{x} = \mathbf{b}$ is easy to solve

$$\begin{aligned} u_{11}x_1 + u_{12}x_2 + u_{13}x_3 + \quad \ldots \quad + u_{1n}x_n &= b_1 \\ u_{22}x_2 + u_{23}x_3 + \quad \ldots \quad + u_{2n}x_n &= b_2 \\ \vdots \qquad\qquad\qquad\qquad \vdots & \\ u_{n-1\,n-1}x_{n-1} + u_{n-1\,n}x_n &= b_{n-1} \\ u_{nn}x_n &= b_n \end{aligned}$$

If $u_{ii} \neq 0$, $i = 1, 2, \ldots, n$, (this is equivalent to $\det U = \prod_{i=1}^{n} u_{ii} \neq 0$) then the unknowns x_j can be computed by **backward substitution**

$$\begin{aligned} x_n &= \frac{b_n}{u_{nn}} \\ x_{n-1} &= \frac{b_{n-1} - u_{n-1\,n} x_n}{u_{n-1\,n-1}} \\ &\vdots \\ x_j &= \frac{b_j - \sum_{k=j+1}^{n} u_{jk} x_k}{u_{jj}} \\ &\vdots \\ x_1 &= \frac{b_1 - \sum_{k=2}^{n} u_{1k} x_k}{u_{11}} \end{aligned}$$

If the matrix L is **lower triangular** ($l_{ij} = 0$ if $i < j$) then the system $L\mathbf{x} = \mathbf{b}$ can be solved by **forward substitution**. The x_j are then computed in order $x_1, x_2, \ldots, x_n$.

If the system is not in triangular form we will try to bring it into triangular form.

1.2.2 Gaussian elimination

Consider the linear system

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1 \\ \vdots \qquad\qquad\qquad\qquad &\quad \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n &= b_n \end{aligned}$$

We will apply row operations to bring this system in triangular form. There are three row operations that do not change the solution vector $\mathbf{x}$:

- multiply row R_i by a constant λ: $\quad \lambda R_i \to R_i$
- add λ times row R_j to row R_i: $\quad R_i + \lambda R_j \to R_i$
- interchange rows R_i and R_j: $\quad R_i \leftrightarrow R_j$

We'll use (for now) the last two operations to bring the system into triangular form. This is the **Gaussian elimination** process.

Example: Solve the following problem by Gaussian elimination

$$\begin{bmatrix} 3 & 6 & 9 \\ 2 & 5 & 2 \\ -3 & -4 & -11 \end{bmatrix} \mathbf{x} = \begin{bmatrix} 3 \\ 4 \\ -5 \end{bmatrix}$$

We perform the row operations

$$R_2 - \frac{2}{3}R_1 \to R_2$$
$$R_3 + \ R_1 \to R_3$$

to obtain zeros below the diagonal in the first column:

$$\begin{bmatrix} 3 & 6 & 9 \\ 0 & 1 & -4 \\ 0 & 2 & -2 \end{bmatrix} \mathbf{x} = \begin{bmatrix} 3 \\ 2 \\ -2 \end{bmatrix} .$$

Then the row operation

$$R_3 - 2R_2 \to R_3$$

produces a zero below the diagonal in the second column:

$$\begin{bmatrix} 3 & 6 & 9 \\ 0 & 1 & -4 \\ 0 & 0 & 6 \end{bmatrix} \mathbf{x} = \begin{bmatrix} 3 \\ 2 \\ -6 \end{bmatrix} .$$

The resulting system is solved by **backward substitution**

$$\begin{aligned} 6x_3 = -6 \qquad &\Longrightarrow \qquad x_3 = -1 \\ x_2 - 4x_3 = 2 \qquad &\Longrightarrow \qquad x_2 = -2 \\ 3x_1 + 6x_2 + 9x_3 = 3 \qquad &\Longrightarrow \qquad x_1 = 8 \end{aligned}$$

Remark: In each step of the Gaussian elimination the matrix A and the vector $\mathbf{b}$ change, but not the solution vector $\mathbf{x}$. In order to save writing one can perform the row operations on the $n \times (n+1)$ **augmented matrix** $\tilde{A}$:

$$\tilde{A} = [A, \mathbf{b}] = \begin{bmatrix} a_{11} & \dots & a_{1n} & b_1 \\ \vdots & & \vdots & \vdots \\ a_{n1} & \dots & a_{nn} & b_n \end{bmatrix} .$$

It is easy to see how the calculations in the example generalise to a general matrix A. In **step 1** of the Gaussian elimination process one applies row operations to obtain zeros below the diagonal in the first column. This is possible if $a_{11} \neq 0$:

$$R_2 - \frac{a_{21}}{a_{11}}R_1 \to R_2$$
$$R_3 - \frac{a_{31}}{a_{11}}R_1 \to R_3$$
$$\vdots$$
$$R_i - \frac{a_{i1}}{a_{11}}R_1 \to R_i$$
$$\vdots$$

This results in an augmented matrix of the form

$$\begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ 0 & a'_{22} & \dots & a'_{2n} & b'_2 \\ \vdots & \vdots & & \vdots & \vdots \\ 0 & a'_{n2} & \dots & a'_{nn} & b'_n \end{bmatrix},$$

where

$$a'_{ij} = a_{ij} - \frac{a_{i1}}{a_{11}} a_{1j}\,, \qquad \text{and} \qquad b'_i = b_i - \frac{a_{i1}}{a_{11}} b_1\,.$$

The matrix element a_{11} is called the **pivot element** of the operations in step 1. It is the element that is used to put zeros in the first column below the diagonal.

The row operations of **step 2** are

$$R_i - \frac{a'_{i2}}{a'_{22}} R_2 \to R_i\,, \qquad i = 3, \dots, n.$$

They require that the pivot element of step 2, a'_{22}, is different from zero, and they result in the augmented matrix

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} & b_1 \\ 0 & a'_{22} & a'_{23} & \dots & a'_{2n} & b'_2 \\ 0 & 0 & a''_{33} & \dots & a''_{3n} & b''_3 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ 0 & 0 & a''_{n3} & \dots & a''_{nn} & b''_n \end{bmatrix}.$$

This process continues until the matrix is in upper triangular form.

What can go wrong? A pivot element can be zero!

Example: Consider the augmented matrix

$$\begin{bmatrix} 3 & 6 & 9 & 3 \\ 2 & 4 & 2 & 4 \\ -3 & -4 & -11 & -5 \end{bmatrix}.$$

We perform the row operations

$$\begin{aligned} R_2 - \frac{2}{3} R_1 &\to R_2 \\ R_3 + \; R_1 &\to R_3 \end{aligned}$$

This leads to

$$\begin{bmatrix} 3 & 6 & 9 & 3 \\ 0 & 0 & -4 & 2 \\ 0 & 2 & -2 & -2 \end{bmatrix}.$$

The next pivot element is zero and we cannot continue with the usual Gaussian elimination process. The solution is to swap rows to obtain a non-vanishing pivot element and then continue. If we swap the second and third row we obtain

$$\begin{bmatrix} 3 & 6 & 9 & 3 \\ 0 & 2 & -2 & -2 \\ 0 & 0 & -4 & 2 \end{bmatrix}.$$

In this example we obtain a matrix which is already in triangular form.

The process of swapping rows in order to obtain a non-vanishing pivot element is called **pivoting**.

In the case that the pivot element and all elements below it are zero then one can show that $\det A = 0$. Then $A\mathbf{x} = \mathbf{b}$ has no unique solution.

Before we continue with linear systems, we make a short excursion and discuss round-off errors.

Round-off errors

Calculations on the computer are done with finite digit accuracy. The standard for floating point arithmetic is IEEE 754-2008. There are standards for binary numbers and for decimal numbers.

For example, "decimal64" represents real numbers by 64 bits (double precision).

Example for floating point representation in decimal64:

$$-368.123 = -3.68123 \times 10^2 \,.$$

The prefactor can contain up to 16 digits, and the exponent of base 10 can be from -383 to 384.

Finite digit calculations can lead to loss of accuracy, in particular in large scale operations.

For example, assume that we calculate with 5 digits precision. We add 3 and subtract 3 from the number 0.0098765

$$\begin{aligned} 0.0098765 + 3 \quad &\rightarrow \quad 3.0099 \qquad \text{(five digits precision)} \\ 3.0099 - 3 \quad &\rightarrow \quad 0.0099 \end{aligned}$$

Because of round-off errors we lost 3 digits precision.

Partial pivoting In numerical computations row exchanges may be necessary even if the pivot element is not zero to avoid round-off errors.

Example: Consider the following system of equations with $\epsilon << 1$

$$\begin{aligned} \epsilon x_1 + x_2 &= 1 \\ x_1 + x_2 &= 2 \end{aligned}$$

The exact solution is

$$x_1 = \frac{1}{1-\epsilon} \approx 1 \,, \qquad x_2 = \frac{1-2\epsilon}{1-\epsilon} \approx 1 \,.$$

Take $\epsilon = 10^{-4}$ and calculate with 3 digits precision. The augmented matrix is

$$\begin{bmatrix} 10^{-4} & 1 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

The row operation $R_2 - 10^4 R_1 \rightarrow R_2$ leads to

$$\begin{bmatrix} 10^{-4} & 1 & 1 \\ 0 & -9999 & -9998 \end{bmatrix}$$

If we calculate with 3 significant digits this is rounded off to

$$\begin{bmatrix} 10^{-4} & 1 & 1 \\ 0 & -10000 & -10000 \end{bmatrix}$$

We obtain the system

$$\begin{aligned} 10^{-4}x_1 + x_2 &= 1 \\ -10000\, x_2 &= -10000\,, \end{aligned}$$

with the solutions $x_2 = 1$ and $x_1 = 0$ instead of $x_2 \approx 1$ and $x_1 \approx 1$.

Reason: The pivot element is much smaller than the element below it. In this case one can show that the Gaussian elimination process is numerically unstable.

Solution: interchange rows before Gaussian elimination

$$\begin{bmatrix} 1 & 1 & 2 \\ 10^{-4} & 1 & 1 \end{bmatrix}$$

The row operation $R_2 - 10^{-4}R_1 \to R_2$ with 3 digits accuracy now leads to

$$\begin{bmatrix} 1 & 1 & 2 \\ 0 & 1 & 1 \end{bmatrix}$$

with the solution $x_2 = 1$ and $x_1 = 1$. This is a good approximation as we have seen.

Partial pivoting: Assume that we are before step k of the Gaussian elimination process. The augmented matrix then has the form

$$\tilde{A} = \begin{bmatrix} a_{11} & \dots & \dots & \dots & \dots & a_{1n} & b_1 \\ 0 & a_{22} & \dots & \dots & \dots & a_{2n} & b_2 \\ 0 & \ddots & \ddots & & & \vdots & \vdots \\ 0 & & 0 & a_{kk} & \dots & a_{kn} & b_k \\ \vdots & & \vdots & \vdots & & \vdots & \vdots \\ 0 & \dots & 0 & a_{nk} & \dots & a_{nn} & b_n \end{bmatrix}$$

Before we perform step k of the Gaussian elimination process we look for the element with the largest magnitude amoung $a_{kk}, \dots, a_{nk}$, and we then interchange the corresponding row with the row of the pivot element.

Partial pivoting works in many cases, but not always. Look, for example, at our previous example and multiply the first row by 2×10^4. We obtain

$$\begin{aligned} 2x_1 + 2 \times 10^4 x_2 &= 2 \times 10^4 \\ x_1 + x_2 &= 2 \end{aligned}$$

Gaussian elimination leads to the same problem as before. We obtain $x_2 \approx 1$ and $x_1 \approx 0$ if we perform the calculations with 3 digits accuracy.

Reason: the largest elements in different rows have vastly different magnitude.

Scaled partial pivoting: scale each row such that the largest element has magnitude 1 before partial pivoting.

Other pivoting strategies are possible but are not discussed here.

1.2.3 LU decomposition

We are going to discuss a very useful representation of a matrix. If possible we want to factorize a matrix in the form

$$A = LU\,,$$

where L is a lower triangular matrix and U an upper triangular matrix. Once we have this factorization we can easily solve the system $A\mathbf{x} = \mathbf{b}$ as follows. Consider

$$A\mathbf{x} = LU\mathbf{x} = \mathbf{b}\,.$$

We can split this equation into two equations by defining $\mathbf{y} = U\mathbf{x}$. Then we have

$$L\mathbf{y} = \mathbf{b}\,, \qquad \text{and} \qquad U\mathbf{x} = \mathbf{y}\,.$$

The first equation $L\mathbf{y} = \mathbf{b}$ can be solved for $\mathbf{y}$ by forward substitution, and then the second equation $U\mathbf{x} = \mathbf{y}$ can be solved for $\mathbf{x}$ by backward substitution.

How do we obtain the LU decomposition? The good news is that the LU decomposition can be obtained directly from the Gaussian elimination process. There is no additional work required, one just has to keep track of the steps.

We start by claiming that the row operations of step 1 of the Gaussian elimination process can be performed by multiplying the matrix A from the left by a matrix M that is specified below. The row operations of step 1 are of the form

$$R_j - m_{j1}\,R_1 \to R_j\,, \qquad j = 2, \ldots, n, \qquad \text{where} \qquad m_{j1} = \frac{a_{j1}}{a_{11}}\,.$$

These row operations transform the matrix A into a new matrix A'. The claim is that $A' = M\,A$ with

$$M = \begin{bmatrix} 1 & 0 & \cdots & \cdots & 0 \\ -m_{21} & 1 & \ddots & & \vdots \\ -m_{31} & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ -m_{n1} & 0 & \cdots & 0 & 1 \end{bmatrix}\,, \quad \text{or} \quad M_{ij} = \delta_{ij} - m_{i1}\delta_{j1}\,, \quad \text{where} \quad \delta_{ij} = \begin{cases} 1 & \text{if} \quad i = j, \\ 0 & \text{if} \quad i \neq j. \end{cases}$$

Proof: $A'_{il} = (M\,A)_{il} = \sum_{j=1}^{n} M_{ij}A_{jl} = \sum_{j=1}^{n}(\delta_{ij} - m_{i1}\delta_{j1})\,A_{jl} = A_{il} - m_{i1}A_{1l}$ (where $m_{11} = 0$). This is the correct form of the matrix A'. It follows further that $A = M^{-1}A'$. The inverse matrix M^{-1} has almost the same form as M, the only difference is that the elements $-m_{j1}$ in the first column are replaced by $+m_{j1}$ for $j = 2, \ldots, n$ (you can check that then $M^{-1}M$ is the identity.)

Note: The symbol δ_{ij} is called Kronecker delta.

Example in lecture: Decomposition of the matrix

$$A = \begin{bmatrix} 3 & 6 & 9 \\ 2 & 5 & 2 \\ -3 & -4 & -11 \end{bmatrix}\,.$$

1.2.4 LU decomposition (continued)

We started to discuss a decomposition of an $n \times n$-matrix in the form

$$A = LU\,,$$

where L is a lower triangular matrix and U an upper triangular matrix. Let us do this now systematically.

During the Gaussian elimination process we use row operations to eliminate the matrix elements below the diagonal of a matrix. The LU-decomposition is obtained by noting that there is a simple relation between the original matrix A and the matrix that is obtained after performing the row operations. We discuss this step by step for a matrix A of the form

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix}$$

Step 1: In the first step of the Gaussian elimination process the pivot element a_{11} is used to eliminate the elements of the matrix A in the first column below the diagonal. The row operations are

$$\begin{aligned} R_2 - m_{21}R_1 \to R_2 \qquad & \text{where} \qquad m_{21} = \frac{a_{21}}{a_{11}}\,, \\ R_3 - m_{31}R_1 \to R_3 \qquad & \text{where} \qquad m_{31} = \frac{a_{31}}{a_{11}}\,, \\ & \cdots \\ R_n - m_{n1}R_1 \to R_n \qquad & \text{where} \qquad m_{n1} = \frac{a_{n1}}{a_{11}}\,, \end{aligned}$$

In the previous section it was shown that there exists the following relation between the matrix A and the matrix that is obtained after performing the row operations:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & \cdots & a_{2n} \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ a_{n1} & a_{n2} & \cdots & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ m_{21} & 1 & 0 & 0 & \cdots & 0 \\ m_{31} & 0 & 1 & 0 & \cdots & 0 \\ m_{41} & 0 & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ m_{n1} & 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & \cdots & \cdots & a_{1n} \\ 0 & a'_{22} & \cdots & \cdots & a'_{2n} \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ 0 & a'_{n2} & \cdots & \cdots & a'_{nn} \end{bmatrix}$$

Here $a'_{ij} = a_{ij} - a_{i1}\,a_{1j}/a_{11}$. We see that the matrix in the middle contains the coefficients m_{i1} that where used in the row operations.

Step 2: This process can be continued. In the second step of the Gaussian elimination process the new pivot element a'_{22} is used to eliminate the matrix elements in the second column below

the diagonal. The row operations are

$$R_3 - m_{32} R_2 \to R_3 \qquad \text{where} \qquad m_{32} = \frac{a'_{32}}{a'_{22}}\,,$$

$$\cdots$$

$$R_n - m_{n2} R_2 \to R_n \qquad \text{where} \qquad m_{n2} = \frac{a'_{n2}}{a'_{22}}\,,$$

One can show that there is again a simple relation between the original matrix A and the new matrix that is obtained after performing the row operations:

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & \cdots & a_{2n} \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ a_{n1} & a_{n2} & \cdots & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ m_{21} & 1 & 0 & 0 & \cdots & 0 \\ m_{31} & m_{32} & 1 & 0 & \cdots & 0 \\ m_{41} & m_{42} & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ m_{n1} & m_{n2} & 0 & \cdots & 0 & 1 \end{bmatrix} \begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & \cdots & a_{1n} \\ 0 & a'_{22} & a'_{23} & \cdots & \cdots & a'_{2n} \\ 0 & 0 & a''_{33} & \cdots & \cdots & a''_{3n} \\ \vdots & \vdots & \vdots & & & \vdots \\ \vdots & \vdots & \vdots & & & \vdots \\ 0 & 0 & a''_{n3} & \cdots & \cdots & a''_{nn} \end{bmatrix}$$

One can continue this process until the Gaussian elimination is completed. The final result is

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & \cdots & a_{2n} \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ \vdots & \vdots & & & \vdots \\ a_{n1} & a_{n2} & \cdots & \cdots & a_{nn} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ m_{21} & 1 & 0 & 0 & \cdots & 0 \\ m_{31} & m_{32} & 1 & 0 & \cdots & 0 \\ m_{41} & m_{42} & m_{43} & 1 & \ddots & \vdots \\ \vdots & \vdots & \vdots & \ddots & \ddots & 0 \\ m_{n1} & m_{n2} & m_{n3} & \cdots & m_{n\,n-1} & 1 \end{bmatrix} \begin{bmatrix} \tilde{a}_{11} & \tilde{a}_{12} & \tilde{a}_{13} & \cdots & \cdots & \tilde{a}_{1n} \\ 0 & \tilde{a}_{22} & \tilde{a}_{23} & \cdots & \cdots & \tilde{a}_{2n} \\ 0 & 0 & \tilde{a}_{33} & \cdots & \cdots & \tilde{a}_{3n} \\ \vdots & & \ddots & \ddots & & \vdots \\ \vdots & & & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & 0 & \tilde{a}_{nn} \end{bmatrix}$$

where we denote the elements of the matrix that is obtained from the Gaussian elimination process by $\tilde{a}_{ij}$. The final result has the form of a LU-decomposition. The upper triangular matrix U is the matrix that is obtained from the Gaussian elimination, and the matrix L contains the coefficients m_{ij} that were used in the row operations. In the following we discuss some applications of this decomposition.

Advantages of the LU-decomposition

- The LU-decomposition is particularly useful if one wants to solve the system $A\mathbf{x} = \mathbf{b}$ for several different vectors $\mathbf{b}$. This happens, for example, if one wants to calculate the inverse of a matrix, see next point.

- The inverse of a matrix can be calculated as follows. Consider the equation $A\,A^{-1} = I$ where I is the identity matrix. If we express the matrices A^{-1} and I in the following way

  $$A^{-1} = (\mathbf{v}_1, \mathbf{v}_2 \ldots, \mathbf{v}_n)\,, \qquad I = (\mathbf{e}_1, \mathbf{e}_2 \ldots, \mathbf{e}_n)\,,$$

  where $\mathbf{v}_j$ are the column vectors of A^{-1} and $\mathbf{e}_j$ are the column vectors of I, then we obtain from $A\,A^{-1} = I$ the following n equations

  $$A\,\mathbf{v}_1 = \mathbf{e}_1\,, \qquad A\,\mathbf{v}_2 = \mathbf{e}_2\,, \qquad \ldots \qquad A\,\mathbf{v}_n = \mathbf{e}_n\,.$$

  They are conveniently solved by using the LU-decomposition.

- The LU-decomposition is also useful for calculating the determinant of a matrix A. One obtains immediately

$$\det(A) = \det(L)\ \det(U) = \prod_{i=1}^{n} u_{ii}\,.$$

  This is much quicker than the Laplace expansion along rows or columns if n is large.

Note: The LU-decomposition is possible if the Gaussian elimination can be performed without row interchanges. What happens if one of the pivot elements is zero? Let us look at the example from the first week

$$A = \begin{bmatrix} 3 & 6 & 9 \\ 2 & 4 & 2 \\ -3 & -4 & -11 \end{bmatrix}$$

The row operations $R_2 - 2R_1/3 \to R_2$ and $R_3 + R_1 \to R_3$ lead to

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 2/3 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 & 6 & 9 \\ 0 & 0 & -4 \\ 0 & 2 & -2 \end{bmatrix}\,.$$

In order to continue we need to interchange R_2 and R_3. This can be achieved by multiplying the matrix from the left with the following permutation matrix

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

P swaps the second and third row of a matrix if multiplied from the left. We can do this also at the start and swap the second and third row of the original matrix A, and perform the Gaussian elimination afterwards

$$P\,A = \begin{bmatrix} 3 & 6 & 9 \\ -3 & -4 & -11 \\ 2 & 4 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 2/3 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 & 6 & 9 \\ 0 & 2 & -2 \\ 0 & 0 & -4 \end{bmatrix}$$

This is the most general form. We can always express a matrix in the form

$$P\,A = L\,U\,.$$

Here P is a permutation matrix that rearranges the rows of the matrix A when multiplied from the left. It has exactly one '1' in every row and every column. The other elements are zero. Permutation matrices satisfy $\det P = \pm 1$ and $P^{-1} = P^t$ where P^t is the transpose of P.

1.3 Root finding for a nonlinear equation

The technique for solving $A\mathbf{x} = \mathbf{b}$ was an example of a **direct method**. Theoretically it gives the exact solution in a finite number of steps. The roots of a nonlinear equation $f(x) = 0$ cannot in general be expressed in closed form. In this case one often uses iterative methods. They start from an initial approximation x_0 and produce a sequence of successive approximations x_0, x_1, x_2, ... which hopefully converges to the desired root.

With certain methods, like the bisection method, it is sufficient to know an interval $[a, b]$ which contains a root, and this guarantees the convergence of the method. Other methods, like the Newton-Raphson method, require a sufficiently good initial approximation. These methods typically converge more quickly to the solution, but they may fail if the initial guess is not good enough.

1.3.1 The bisection method

The following theorem from Calculus states the fact that a continuous function $f(x)$ takes on all values between $f(a)$ and $f(b)$ on an interval $[a, b]$.

Intermediate value theorem:
If $f \in C[a, b]$ and $f(a) < k < f(b)$ then $\exists\ c \in (a, b)$ such that $f(c) = k$. Similarly for $f(a) > k > f(b)$.

If we can find a and b such that $f(a)$ and $f(b)$ have opposite signs, $f(a) \cdot f(b) < 0$, then it follows that the interval (a, b) contains at least one root of $f(x) = 0$ if $f(x)$ is continuous. In fact there can be any odd number of roots.

The bisection method consists simply of repeatedly halving (or bisecting) subintervals of $[a, b]$ and choosing at each step the half in which the function f changes sign.

Example: Find one root of $f(x) = \exp(x) - 3x = 0$.

Evaluating the function $f(x)$ at a few points shows that $f(0) > 0$, $f(1) < 0$ and $f(2) > 0$. Hence there must be at least one root in the interval $[0, 1]$ and at least one other root in the interval $[1, 2]$. In fact there is exactly one root in each interval. In the following we concentrate on the second interval and use the bisection method to find a numerical approximation for the root. The following table shows the results of the first 9 iteration steps. The exact value of the root is $x^* = 1.51213\ldots$

n	a_n	x_n	b_n	$f(a_n)$	$f(x_n)$	$f(b_n)$	max. err.
1	1.0	1.50000	2.0	-0.28172	-0.01831	1.38906	2^{-1}
2	1.50000	1.75000	2.0	-0.01831	0.50460	1.38906	2^{-2}
3	1.50000	1.62500	1.75000	-0.01831	0.20342	0.50460	2^{-3}
4	1.50000	1.56250	1.62500	-0.01831	0.08323	0.20342	2^{-4}
5	1.50000	1.53125	1.56250	-0.01831	0.03020	0.08323	2^{-5}
6	1.50000	1.51562	1.53125	-0.01831	0.00538	0.03020	2^{-6}
7	1.50000	1.50781	1.51562	-0.01831	-0.00660	0.00538	2^{-7}
8	1.50781	1.51172	1.51562	-0.00660	-0.00064	0.00538	2^{-8}
9	1.51172	1.51367	1.51562	-0.00064	0.00236	0.00538	2^{-9}

Notes:

- The maximum error after n iterations satisfies

$$|x^* - x_n| \leq \frac{|b-a|}{2^n}$$

  where x^* denotes the root and x_n the n-th approximation. The error is only halved by an iteration. This convergence is considered to be slow.

- The algorithm is very safe. It will always converge to a root in the interval.

- The bisection method is often used as a 'starter' for more efficient methods (see later).

1.3.2 Fixed-point iteration

For the fixed-point iteration method the equation $f(x) = 0$ is rewritten in the form $x = g(x)$ such that

$$f(x) = 0 \Longleftrightarrow g(x) = x\,.$$

This can be achieved, for example, by setting $g(x) = x - f(x)$. Fixed-point problems are often easier to analyse than root-finding problems. It is also often possible to find a fixed-point by using the iterative procedure

$$x_{n+1} = g(x_n)\,, \qquad n = 0, 1, 2, 3, \ldots$$

that starts from an initial guess x_0.

Let us consider a numerical example. The fixed-point iteration method is applied to find the fixed point of $g(x) = \cos(x)$, starting with $x_0 = 1$. The position of the fixed point is $x^* = 0.739085$, e_n denotes the error $x_n - x^*$, and e_n/e_{n-1} is the quotient of subsequent errors. The following table lists the numerical results

n	x_n	e_n	e_n/e_{n-1}
0	1.000000	0.260915	
1	0.540302	-0.198783	-.761869
2	0.857553	0.118468	-.595967
3	0.654290	-0.084795	-.715765
4	0.793480	0.054395	-.641488
5	0.701369	-0.037716	-.693376
6	0.763960	0.024875	-.659516
7	0.722102	-0.016983	-.682734
8	0.750418	0.011333	-.667304
9	0.731404	-0.007681	-.677785
10	0.744237	0.005152	-.670767
20	0.739184	0.000099	-.673558
30	0.739087	0.000002	-.673620

We see that the numerical iteration converges to the fixed point, and that the errors decrease by a factor that is approximately constant for large n.

Questions:

- Is the reformulation of $f(x) = 0$ to $g(x) = x$ unique?
- When will the iterative procedure $x_{n+1} = g(x_n)$ converge?
- How fast is the convergence?

Let us look at some further examples. We now compare four different iteration schemes to find the positive root of $f(x) = x^2 - x - 1 = 0$. For this purpose the problem is reformulated as a fixed point problem, $x = g(x)$, with four different choices of the function $g(x)$. In all four cases the positive root of $x^2 - x - 1$ is a fixed point of $g(x)$. The four functions are

$$x = g_1(x) = x^2 - 1$$

$$x = g_2(x) = 1 + \frac{1}{x}$$

$$x = g_3(x) = \sqrt{1 + x}$$

$$x = g_4(x) = x - \frac{x^2 - x - 1}{2x - 1}$$

The following table shows that these iteration schemes have different convergence properties. The value of the root we are looking for is $x^* = (1 + \sqrt{5})/2 = 1.618034$. The starting value is $x_0 = 2$ in all four cases.

n	x_n from $g_1(x)$	x_n from $g_2(x)$	x_n from $g_3(x)$	x_n from $g_4(x)$
0	2	2	2	2
1	3	1.500000	1.732051	1.666667
2	8	1.666667	1.652892	1.619048
3	63	1.600000	1.628770	1.618034
4	3968	1.625000	1.621348	1.618034
5	15745023	1.615385	1.619058	
6		1.619048	1.618350	
7		1.617647	1.618132	
8		1.618182	1.618064	
9		1.617978	1.618043	
10		1.618056	1.618037	
11		1.618026	1.618035	
12		1.618037	1.618034	
13		1.618033	1.618034	
14		1.618034		
15		1.618034		

Clearly the choice of $g(x)$ is not unique and the method may not converge. If the method converges then the speed of convergence may be very different for different choices of $g(x)$. In the following, we first investigate when the method converges. Sufficient conditions for convergence are given by the following theorem.

Fixed-point theorem (FPT): Let $g(x)$ be continuous on $[a,b]$ and suppose that $g(x) \in [a,b]$ for all $x \in [a,b]$. Suppose further that $g(x)$ is differentiable on (a,b) and

$$|g'(x)| \leq k < 1 \qquad \forall x \in (a,b)\,.$$

Then there exists a unique fixed point x^* in $[a,b]$, where $g(x^*) = x^*$, and if $x_0 \in [a,b]$ then the sequence

$$x_{n+1} = g(x_n)\,, \qquad n = 0, 1, 2, \ldots$$

converges to this fixed point.

Before we prove this theorem we state a theorem from Calculus that we need for the proof.

Mean value theorem (MVT): If $f \in C[a,b]$ and f is differentiable on (a,b) then a number $c \in (a,b)$ exists such that

$$f'(c) = \frac{f(b) - f(a)}{b - a}\,.$$

One can demonstrate this theorem graphically.

Proof of FTP: The proof consists of three parts.

Existence of fixed point: If $g(a) = a$ or $g(b) = b$ then we are done. Now assume that $g(a) \neq a$ and $g(b) \neq b$. Then

$$g(a) - a > 0 \qquad \text{and} \qquad g(b) - b < 0$$

because $g(x) \in [a,b]$ if $x \in [a,b]$. It then follows from the intermediate value theorem that there exists $x^* \in (a,b)$ such that $g(x^*) - x^* = 0$. This proves the existence of the fixed point.

Uniqueness of fixed point: Assume that there exist two fixed points $x_a, x_b \in [a,b]$ such that $g(x_a) = x_a$, $g(x_b) = x_b$ and $x_b > x_a$. Then

$$\frac{g(x_b) - g(x_a)}{x_b - x_a} = \frac{x_b - x_a}{x_b - x_a} = 1\,.$$

However, by using the mean value theorem and one of the assumptions of the fixed-point theorem we obtain

$$\left|\frac{g(x_b) - g(x_a)}{x_b - x_a}\right| = |g'(c)| \leq k < 1\,,$$

for some $c \in (x_a, x_b)$. This is a contradiction and we conclude that the fixed point is unique.

Convergence of the iteration: We use the mean value theorem to obtain

$$|x_n - x^*| = |g(x_{n-1}) - g(x^*)| = |g'(\xi_n)|\,|x_{n-1} - x^*| \leq k\,|x_{n-1} - x^*|\,,$$

where ξ_n is between x_{n-1} and x^*. Using this relation iteratively we obtain

$$|x_n - x^*| \leq k\,|x_{n-1} - x^*| \leq k^2\,|x_{n-2} - x^*| \leq \ldots \leq k^n\,|x_0 - x^*|\,.$$

Taking the limit $n \to \infty$ then yields

$$\lim_{n\to\infty} |x_n - x^*| \leq \lim_{n\to\infty} k^n\,|x_0 - x^*| = 0\,.$$

This concludes the proof of the fixed-point theorem.

In fact, if x_n is sufficiently close to x^* (n sufficiently big) then one can apply a similar calculation to relate the error $\epsilon_n = x_n - x^*$ of the n-th iteration to the error of the previous step $\epsilon_{n-1} = x_{n-1} - x^*$.

$$x_n - x^* = g(x_{n-1}) - g(x^*) = g'(\xi_n)\,(x_{n-1} - x^*) \approx g'(x^*)\,(x_{n-1} - x^*)\,,$$

where ξ_n is between x_{n-1} and x^*. We find that $\epsilon_n \approx g'(x^*)\,\epsilon_{n-1}$. This shows that $g'(x^*)$ controls the speed of convergence. We can conclude further that the iteration diverges if $|g'(x^*)| > 1$.

Let us use this new information to interpret our previous numerical results. We looked for the positive root $x^* = 1.618034$ of the equation $x^2 - x - 1 = 0$, and we applied four different fixed-point iterations for this purpose. Let us calculate the value of $g'(x^*)$ in these four cases

$$g_1(x) = x^2 - 1\,, \qquad g_1'(x) = 2x\,, \qquad g_1'(x^*) \approx 3.24 > 1\,,$$

$$g_2(x) = 1 + \frac{1}{x}\,, \qquad g_2'(x) = -\frac{1}{x^2}\,, \qquad g_2'(x^*) \approx -0.382\,,$$

$$g_3(x) = \sqrt{1+x}\,, \qquad g_3'(x) = \frac{1}{2\sqrt{1+x}}\,, \qquad g_3'(x^*) \approx 0.309\,,$$

$$g_4(x) = x - \frac{x^2 - x - 1}{2x - 1}\,, \qquad g_4'(x) = \frac{2(x^2 - x - 1)}{(2x-1)^2}\,, \qquad g_4'(x^*) = 0\,.$$

We can now understand our numerical results. The iteration with $g_1(x)$ is divergent because $|g_1'(x^*)| > 1$. In the case of $g_2(x)$ and $g_3(x)$ successive errors are reduced by a factor of approximately $g_2'(x^*)$ and $g_3'(x^*)$, respectively, if n is large. This is called linear convergence since the new error is proportional to the error of the previous step. The convergence for $g_3(x)$ is slightly faster than the convergence for $g_2(x)$ because $|g_3'(x^*)| < |g_2'(x^*)|$. In the case of $g_4(x)$ we have $g_4'(x^*) = 0$ and the convergence is faster than linear. In this case we need to introduce a new definition in order to characterise how quickly the iteration converges.

The following definition characterises how rapidly a sequence converges:

Order of convergence

Let $\{x_n,\, n = 0, 1, 2, \ldots\}$ be a sequence that converges to x^*, $\lim_{n\to\infty} x_n = x^*$. If positive constants α and λ exist such that

$$\lim_{n\to\infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|^\alpha} = \lambda\,,$$

then the sequence converges to x^* with **order of convergence** α and **asymptotic error constant** λ. In particular, we distinguish the following important cases (with $\epsilon_n = x_n - x^*$)

- If $\alpha = 1$ (and $\lambda < 1$) the sequence is **linearly convergent** ($|\epsilon_{n+1}| \approx \lambda\,|\epsilon_n|$ if n is large).
- If $\alpha = 2$ the sequence is **quadratically convergent** ($|\epsilon_{n+1}| \approx \lambda\,|\epsilon_n|^2$ if n is large).
- If $\alpha = 3$ the sequence is **cubically convergent** ($|\epsilon_{n+1}| \approx \lambda\,|\epsilon_n|^3$ if n is large).

In the following we want to prove a theorem that allows us to determine the order of convergence of a fixed-point iteration scheme. For this purpose we need an important theorem from Calculus that we will use again and again in Numerical Analysis. This is

Taylor's theorem: Suppose $f \in C^{n+1}[a, b]$ and $x_0 \in [a, b]$. For every $x \in [a, b]$ there exists a number $\xi(x)$ between x and x_0 such that

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k + \frac{f^{n+1}(\xi(x))}{(n+1)!}(x - x_0)^{n+1}\,.$$

For later applications it will be important that this theorem provides us with an error estimate (second term on right-hand side) if we approximate a function by a finite number of terms (first term on right-hand side). Note that the mean value theorem follows from Taylor's theorem with $n = 0$, because then

$$f(x) = f(x_0) + f'(\xi(x))(x - x_0)\,.$$

Theorem (on the order of convergence): If the fixed-point iteration $x_{n+1} = g(x_n)$ converges to x^* in $[a, b]$ and g satisfies $g \in C^p[a, b]$ and

$$0 = g'(x^*) = g''(x^*) = \ldots = g^{(p-1)}(x^*)\,, \qquad g^{(p)}(x^*) \neq 0\,,$$

i. e. the first $p - 1$ derivatives vanish at $x = x^*$, then the order of convergence is p.

Proof: We use Taylor's theorem, together with $x_{n+1} = g(x_n)$ and $x^* = g(x^*)$, to obtain

$$\begin{aligned} x_{n+1} - x^* &= g(x_n) - g(x^*) \\ &= \sum_{k=0}^{p-1} \frac{g^{(k)}(x^*)}{k!}(x_n - x^*)^k + \frac{g^p(\xi(x_n))}{p!}(x_n - x^*)^p - g(x^*) \\ &= \sum_{k=1}^{p-1} \frac{g^{(k)}(x^*)}{k!}(x_n - x^*)^k + \frac{g^p(\xi(x_n))}{p!}(x_n - x^*)^p \end{aligned}$$

where $\xi(x_n)$ is between x_n and x^*. The first term on the right-hand side of the last line vanishes due to the assumption of the theorem and we obtain

$$\lim_{n\to\infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|^p} = \lim_{n\to\infty} \frac{|g^{(p)}(\xi(x_n))|}{p!} = \frac{|g^{(p)}(x^*)|}{p!}\,.$$

This proves that the order of convergence is p, and it further shows that the asymptotic error constant is $\lambda = |g^{(p)}(x^*)|/p!$.

Example: In our previous example $f(x) = x^2 - x - 1 = 0$ we applied several fixed-point iteration schemes to search for the root $x^* = (1 + \sqrt{5})/2$. One choice of the function $g(x)$ was

$$g_4(x) = x - \frac{x^2 - x - 1}{2x - 1}\,.$$

We already found that $g_4'(x^*) = 0$. One can further check that $g_4''(x^*) \neq 0$, and hence we conclude from the last theorem that the convergence is quadratic.

Further example in lecture: Consider the fixed-point iteration for $g(x) = \sin(\pi x/2)$.

1.3.3 Newton-Raphson method

The Newton-Raphson method, or simply Newton's method, is one of the most powerful and well-known methods for solving a root-finding problem of the form $f(x) = 0$. It was developed in the 17th century by Isaac Newton (1641-1727) and Joseph Raphson (1648-1715).

Derivation: Let x^* be a solution of $f(x) = 0$ and let x_n be a close approximation such that $|x_n - x^*|$ is small. We assume $f \in C^2[a, b]$ and x^*, $x_n \in [a, b]$ and use Taylor's theorem to obtain

$$f(x^*) = f(x_n) + f'(x_n)\,(x^* - x_n) + \frac{1}{2} f''(\xi(x_n))\,(x^* - x_n)^2,$$

where $\xi(x_n)$ is between x^* and x_n.

Now if $|x_n - x^*|$ is small then $|x_n - x^*|^2$ is smaller. We introduce an approximation by neglecting the term containing $|x_n - x^*|^2$. By definition $f(x^*) = 0$, and we obtain

$$0 \approx f(x_n) + f'(x_n)\,(x^* - x_n)\,.$$

Solving this equation for x^* results in

$$x^* \approx x_n - \frac{f(x_n)}{f'(x_n)}\,.$$

This approximation suggests to take the right-hand side as the next approximation for x^*

$$\boxed{x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}}\,.$$

This is the iteration scheme of Newton's method. It corresponds to a fixed-point iteration of the form $x_{n+1} = g(x_n)$ with the function

$$g(x) = x - \frac{f(x)}{f'(x)}\,.$$

Order of convergence of Newton's method

We know from the theorem of the last section that the order of convergence of a fixed-point iteration scheme is determined by how many of the first derivatives of the function $g(x)$ vanish at the fixed point x^*. For Newton's method

$$g = x - \frac{f}{f'}\,, \qquad g' = 1 - \frac{f'^2 - ff''}{f'^2} = \frac{ff''}{f'^2}\,, \qquad g'' = \frac{f'^2(f'f'' + ff''') - ff''2f'f''}{f'^4}\,.$$

We assume in the following that x^* is a **simple zero** of $f(x)$, meaning that $f(x^*) = 0$ and $f'(x^*) \neq 0$. Then

$$g'(x^*) = \frac{f(x^*)\,f''(x^*)}{[f'(x^*)]^2} = 0\,,$$

and

$$g''(x^*) = \frac{f'(x^*)^3 f''(x^*)}{f'(x^*)^4} = \frac{f''(x^*)}{f'(x^*)}\,.$$

The second derivative $g''(x^*)$ is in general different from zero and we find that the convergence of Newton's method is typically quadratic for simple zeros.

Graphical interpretation of Newton's method

Newton's method has a simple graphical interpretation. Let x_n be the n-th approximation to x^*. We can obtain the next approximation by drawing the tangent to the graph of the function $f(x)$ at the point $(x_n, f(x_n))$. The point were this tangent intersects the x-axis gives the value of x_{n+1}. This can be easily seen by starting from the equation of the tangent

$$\frac{y - f(x_n)}{x - x_n} = f'(x_n)\,.$$

The intersection with the x-axis is obtained by setting $y = 0$ and $x = x_{n+1}$.

$$\frac{-f(x_n)}{x_{n+1} - x_n} = f'(x_n)\,.$$

Solving this equation for x_{n+1} yields

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}\,,$$

which is indeed Newton's iteration.

Previous example: Let us revisit our previous example where we searched for the positive root of the equation $f(x) = x^2 - x - 1 = 0$. In this case $f'(x) = 2x - 1$ and Newton's method yields

$$x_{n+1} = x_n - \frac{x_n^2 - x_n - 1}{2x_n - 1}\,.$$

This agrees with our previous iteration scheme with the function $g_4(x) = x-(x^2-x-1)/(2x-1)$. It converges much more rapidly than the other fixed-point iteration schemes for this example.

Further example in lecture: Apply the method to calculate $\sqrt{5}$.

Convergence of Newton's method

We now investigate the region of convergence of Newton's method. We first show that there exists an interval around x^* in which the method converges. When the method is considered as a fixed-point iteration, we found that $g'(x^*) = 0$. It is not difficult to show that there exists an interval $I_\delta := [x^* + \delta, x^* - \delta]$ around x^* in which the fixed-point theorem applies and the method converges. We assume that $g(x)$ is continuous and differentiable in some neighbourhood of x^*. Because of the continuity of $g'(x)$ there exists and interval $I_\delta := [x^* + \delta, x^* - \delta]$, $\delta > 0$, in which $|g'(x)| \leq k < 1$. We can then use the Mean Value Theorem to show that $g(x) \in I_\delta$ if $x \in I_\delta$. This follows from

$$|g(x) - x^*| = |g(x) - g(x^*)| = |g'(\xi)|\,|x - x^*| < |x - x^*|$$

where ξ is between x and x^*, and we assumed that $x \in I_\delta$. It follows that $|g(x) - x^*| < \delta$ if $|x - x^*| < \delta$, in other words $g(x) \in I_\delta$ if $x \in I_\delta$. This means that the conditions of the Fixed-Point Theorem are satisfied and the method converges in the interval I_δ.

Can we estimate the size of the interval of convergence? Let us consider some interval I around x^* and assume that the second derivative of $g(x)$ is bounded in this interval, $|g''(x)| < M$ if $x \in I$. We found previously using Taylor's theorem that for quadratically convergent fixed-point iterations

$$|x_{n+1} - x^*| = \frac{1}{2}|g''(\xi_n)|\,|x_n - x^*|^2\,,$$

(see proof of the theorem on the order of convergence). Here ξ_n is between x_n and x^*. Using the bound for the second derivative we obtain

$$|\epsilon_{n+1}| = \frac{1}{2}|g''(\xi_n)|\,|\epsilon_n|^2 < \frac{M}{2}|\epsilon_n|^2\,,$$

where $\epsilon_n = x_n - x^*$. For convergence we need that the error decreases and hence we require that $M|\epsilon_n|/2 < 1$. When applied to $n = 0$ this gives a condition for the smallness of $|\epsilon_0|$. It gives us an estimate of how good the initial approximation x_0 has to be for the method to converge. If the initial value x_0 is not good enough then the method diverges. This can also be seen from the graphical method. For this reason, Newton's method is often used in combination with a secure method such as bisection to obtain a good starting point for Newton.

Multiple zeros

Definition: A solution x^* of $f(x) = 0$ is said to be a **zero of multiplicity** m if

$$0 = f(x^*) = f'(x^*) = \ldots = f^{(m-1)}(x^*)\,, \qquad f^{(m)}(x^*) \neq 0\,.$$

Then by Taylor's theorem

$$f(x) = \sum_{k=0}^{m-1} \frac{f^{(k)}(x^*)}{k!}(x-x^*)^k + \frac{f^{(m)}(\xi(x))}{m!}(x-x^*)^m =: q(x)\,(x-x^*)^m\,.$$

where $\xi(x)$ is between x and x^* and $q(x) = f^{(m)}(\xi(x))/m!$. Here we used the fact that the function $f(x)$ and its first $m-1$ derivatives vanish at x^*. From $f^{(m)}(x^*) \neq 0$ follows further that $q(x^*) \neq 0$. The first two derivatives of $f(x)$ are given by

$$\begin{aligned} f'(x) &= m\,q(x)\,(x-x^*)^{m-1} + q'(x)\,(x-x^*)^m\,,\\ f''(x) &= m\,(m-1)\,q(x)\,(x-x^*)^{m-2} + 2\,m\,q'(x)\,(x-x^*)^{m-1} + q''(x)\,(x-x^*)^m\,. \end{aligned}$$

Zeros with $m = 1$ are called **simple zeros**. If the multiplicity m of a zero is larger than one then Newton's method converges only linearly. This follows from

$$g'(x^*) = \lim_{x\to x^*} \frac{f(x)\,f''(x)}{[f'(x)]^2} = \frac{m(m-1)}{m^2} = 1 - \frac{1}{m} \neq 0 \qquad \text{if} \qquad m \neq 1\,.$$

Example in lecture: $f(x) = \exp(x) - x - 1$ has a zero of multiplicity $m = 2$ at $x^* = 0$.

The quadratic convergence can be restored by considering the function

$$F(x) = \frac{f(x)}{f'(x)}\,, \qquad \Longrightarrow \qquad F(x) = \frac{q(x)\,(x-x^*)^m}{m\,q(x)\,(x-x^*)^{m-1} + q'(x)\,(x-x^*)^m}\,.$$

We can write it in the following form

$$F(x) = (x-x^*)\,Q(x) \qquad \text{where} \qquad Q(x) = \frac{q(x)}{m\,q(x) + q'(x)\,(x-x^*)}\,.$$

The function $F(x)$ has a simple zero at $x = x^*$ because $Q(x^*) = 1/m \neq 0$. Hence we can apply Newton's method to $F(x)$ and obtain again quadratic convergence. This leads to the following iteration scheme

$$x_{n+1} = x_n - \frac{F(x_n)}{F'(x_n)}\,, \qquad \text{where} \qquad F = \frac{f}{f'}\,, \qquad F' = \frac{f'^2 - ff''}{f'^2}\,.$$

which results in

$$x_{n+1} = x_n - \frac{f(x_n)f'(x_n)}{f'(x_n)f'(x_n) - f(x_n)f''(x_n)}\,.$$

Although we have restored quadratic convergence there is a different problem with this iteration scheme. The denominator of the fraction consists of the difference of two very small nearly equal numbers. This can lead to large round-off errors.

The secant method

Newton's method requires the calculation of $f(x_n)$ and $f'(x_n)$ at each iteration step. This can be numerically expensive if the function $f(x)$ and its derivative are complicated to evaluate. Consider for example

$$f(x) = \sum_{n=0}^{\infty} \frac{(n+1)\,\cos(nx)}{1+n+n^3}\,.$$

The **secant method** is a method that converges faster than linearly and needs only the evaluation of the functional values $f(x_n)$. The idea is the following: if x_n and x_{n-1} are close then one can approximate the derivative $f'(x_n)$ by a discrete approximation

$$f'(x_n) \approx \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}\,.$$

Inserting this into Newton's iteration formula results in

$$x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} = \frac{f(x_n)x_{n-1} - f(x_{n-1})x_n}{f(x_n) - f(x_{n-1})}\,.$$

One sees that this method requires two starting points, x_0 and x_1.

There is a simple graphical interpretation of the secant method. Draw a straight line through the points $(x_n, f(x_n))$ and $(x_{n-1}, f(x_{n-1}))$. This line is a secant, a straight line joining two points on a curve. The value of x_{n+1} is then given by the intersection of the secant with the x-axis. Let us show that this graphical interpretation agrees with the iteration scheme. The secant goes through the points $(x_{n-1}, f(x_{n-1}))$, $(x_n, f(x_n))$ and $(x_{n+1}, 0)$, and its slope can be obtained in different ways

$$\frac{x_n - x_{n+1}}{f(x_n) - 0} = \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}\,.$$

Solving this equation for x_{n+1} yields

$$x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}\,,$$

as before. One can show (not easy) that the order of convergence of the secant method is not an integer. It is given by the "golden ratio" $\alpha = (1+\sqrt{5})/2 \approx 1.618$. This is faster than linear ($\alpha = 1$), but slower than quadratic ($\alpha = 2$).

1.3.4 Accelerating convergence

The final method that we discuss in section 1.3 is Aitken's Δ^2-method. This method can be used to accelerate the convergence of a **linearly convergent sequence**, regardless of its origin. Suppose that $\{x_n,\, n = 0, 1, 2, \ldots\}$ is a linearly convergent sequence with limit x^* and asymptotic error constant $\lambda < 1$.

$$\lim_{n\to\infty} \frac{|x_{n+1} - x^*|}{|x_n - x^*|} = \lambda < 1\,.$$

If n is large then we expect that the left-hand side is a good approximation for λ without taking the limit. Taking two consecutive values of n and dropping the absolute value signs we obtain

$$\frac{x_{n+1} - x^*}{x_n - x^*} \approx \pm\lambda \approx \frac{x_{n+2} - x^*}{x_{n+1} - x^*}\,.$$

One can use the connection between the left-hand side and the right-hand side to obtain another approximation for x^*. Multiplying with the denominators we obtain

$$\begin{aligned}(x_{n+1} - x^*)^2 &\approx (x_{n+2} - x^*)\,(x_n - x^*)\\ x_{n+1}^2 - 2x_{n+1}x^* + (x^*)^2 &\approx x_{n+2}x_n - x^*(x_{n+2} + x_n) + (x^*)^2\,.\end{aligned}$$

Solving this for x^* results in

$$x^* \approx \frac{x_{n+2}x_n - x_{n+1}^2}{x_{n+2} - 2x_{n+1} + x_n} = x_n - \frac{x_{n+1}^2 - 2x_{n+1}x_n + x_n^2}{x_{n+2} - 2x_{n+1} + x_n} = x_n - \frac{(x_{n+1} - x_n)^2}{x_{n+2} - 2x_{n+1} + x_n}\,.$$

This can be written more compactly with the definition of the **forward difference**

$$\Delta x_n = x_{n+1} - x_n\,.$$

We then have

$$\Delta^2 x_n = \Delta(\Delta x_n) = \Delta(x_{n+1} - x_n) = \Delta x_{n+1} - \Delta x_n = x_{n+2} - 2x_{n+1} + x_n\,.$$

With this notation we can write

$$x^* \approx x_n - \frac{(\Delta x_n)^2}{\Delta^2 x_n}\,.$$

This is used in the following way. In Aitken's method one starts from the sequence $\{x_n\}$ and constructs a new sequence $\{\hat{x}_n\}$ by defining

$$\hat{x}_n \approx x_n - \frac{(\Delta x_n)^2}{\Delta^2 x_n}\,.$$

This sequence should converge faster than the original sequence. Indeed one can show that the convergence is faster than linear (see for example Burden/Faires).

Example in lecture: Apply Aitken's Δ^2-method to the fixed-point iteration with $g(x) = \cos(x)$.

1.4 Solving systems of nonlinear equations

We consider now methods for solving systems of n nonlinear equations for n variables

$$f_i(x_1, x_2, \ldots, x_n) = 0 \qquad \text{for} \qquad i = 1, \ldots, n\,,$$

or in vector notation $\mathbf{f}(\mathbf{x}) = \mathbf{0}$. This is a difficult problem, and there are no good general methods for solving systems of more than one nonlinear equation. Consider the case in two dimensions. We want to solve simultaneously

$$f(x, y) = 0 \qquad \text{and} \qquad g(x, y) = 0\,.$$

Both equations define contours in the xy-plane, so we have to locate the intersections of contours that can be quite complicated. The picture quickly increases in complexity as the number of equations grows.

However, once a neighbourhood of a root is identified one has methods to continue. Newton's method is then generally the method of choice. One can also generalise fixed-point methods to higher dimensions, but we won't discuss them here.

1.4.1 Newton's method

This is the generalisation of the Newton-Raphson method to higher dimensions.

We consider the problem of finding the root of $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ where $\mathbf{f}$ and $\mathbf{x}$ are vectors of the form

$$\mathbf{f}(\mathbf{x}) = \begin{pmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_n(\mathbf{x}) \end{pmatrix}\,, \qquad \text{and} \qquad \mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}\,.$$

Let us denote the solution by $\mathbf{x}^*$, so $\mathbf{f}(\mathbf{x}^*) = \mathbf{0}$. Assume that $\mathbf{x}^{(m)}$ is a good approximation to the solution such that $|\mathbf{x}^{(m)} - \mathbf{x}^*|$ is small. We use Taylor's expansion in higher dimensions

$$f_i(\mathbf{x}^*) = f_i(\mathbf{x}^{(m)}) + \sum_{j=1}^{n} \frac{\partial f_i}{\partial x_j}(\mathbf{x}^{(m)})\,(x_j^* - x_j^{(m)}) + \text{higher order terms}\,, \qquad i = 1, 2, \ldots, n\,.$$

We can write this in vector notation

$$\mathbf{f}(\mathbf{x}^*) \approx \mathbf{f}(\mathbf{x}^{(m)}) + J(\mathbf{x}^{(m)})\,(\mathbf{x}^* - \mathbf{x}^{(m)})\,,$$

where $J(\mathbf{x})$ is the Jacobian matrix

$$J(\mathbf{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{x}) & \ldots & \frac{\partial f_1}{\partial x_n}(\mathbf{x}) \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1}(\mathbf{x}) & \ldots & \frac{\partial f_n}{\partial x_n}(\mathbf{x}) \end{pmatrix}\,.$$

Now we use $\mathbf{f}(\mathbf{x}^*) = \mathbf{0}$ and solve for $\mathbf{x}^*$.

$$\mathbf{x}^* \approx \mathbf{x}^{(m)} - J^{-1}(\mathbf{x}^{(m)})\,\mathbf{f}(\mathbf{x}^{(m)})\,.$$

This suggests the following iteration scheme

$$\mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} - J^{-1}(\mathbf{x}^{(m)})\,\mathbf{f}(\mathbf{x}^{(m)})\,.$$

Note that we can avoid to evaluate the inverse matrix J^{-1} by setting

$$\mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} - \mathbf{y}^{(m)} .$$

where

$$\mathbf{y}^{(m)} = J^{-1}(\mathbf{x}^{(m)})\ \mathbf{f}(\mathbf{x}^{(m)}) \qquad \text{or} \qquad J(\mathbf{x}^{(m)})\ \mathbf{y}^{(m)} = \mathbf{f}(\mathbf{x}^{(m)}) .$$

The last equation shows that we can determine $\mathbf{y}^{(m)}$ by Gaussian elimination.

As in one dimension, Newton's method has in general quadratic convergence, but there are two disadvantages of the method:

- The method may not converge if the initial approximation is not good enough.
- It requires to evaluate the matrix of derivatives, the Jacobian J, at each step.

Example in lecture: Apply Newton's method to solve the system of two equations

$$x^2 + y^2 = 4 , \qquad xy = 1 .$$

Further example in lecture: Apply Newton's method to solve the system of three equations

$$x_1^3 - 2x_2 - 2 = 0 , \qquad x_1^3 - 5x_3^2 + 7 = 0 , \qquad x_2 x_3^2 - 1 = 0 .$$

Quasi-Newton methods

Quasi-Newton methods replace the Jacobian matrix in Newton's method with an approximation matrix that is updated at each iteration. They have the advantage that they do not require to evaluate the derivatives of the function $\mathbf{f}(\mathbf{x})$. Broyden's method is the generalisation of the secant method to multi-dimensional systems. The details are somewhat involved and we will not pursue this further.

1.4.2 Steepest decent

This is a global method that does not necessarily require a good initial approximation, but the convergence can be quite slow. The idea is to transform the root finding problem into a minimum finding problem. Consider

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}\,, \qquad \text{where} \qquad \mathbf{f}(\mathbf{x}) = \begin{pmatrix} f_1(\mathbf{x}) \\ \vdots \\ f_n(\mathbf{x}) \end{pmatrix}\,.$$

Introduce the function

$$g(\mathbf{x}) = \sum_{i=1}^{n} f_i^2(\mathbf{x}) \geq 0\,.$$

The function $g(\mathbf{x})$ has a minimum at $\mathbf{x}^*$ with value $g(\mathbf{x}^*) = 0$ if and only if $\mathbf{f}(\mathbf{x}^*) = \mathbf{0}$. The method proceeds as follows

- Evaluate $g(\mathbf{x})$ at an approximation $\mathbf{x}^{(m)}$.
- Determine the direction from $\mathbf{x}^{(m)}$ in which $g(\mathbf{x})$ decreases most rapidly. This direction is given by minus the gradient of g

  $$-\nabla g = \begin{pmatrix} -\frac{\partial g}{\partial x_1} \\ \vdots \\ -\frac{\partial g}{\partial x_n} \end{pmatrix}\,.$$

  The next iteration point is chosen in this direction

  $$\mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} - \alpha \nabla g(\mathbf{x}^{(m)})\,.$$

  The value of α is determined by requiring that the value of g at the next point is reduced most, as will be explained in the following.
- To determine α define

  $$h(\alpha) = g[\mathbf{x}^{(m)} - \alpha \nabla g(\mathbf{x}^{(m)})]\,,$$

  and search for a value $\alpha = \hat{\alpha}$ that minimises $h(\alpha)$. Crude methods are often sufficient: Calculate $h(\alpha)$ for a few values of α and stop when $h(\alpha)$ starts to increase. This gives the value $\alpha = \hat{\alpha}$.
- The next iteration point is then given by $\mathbf{x}^{(m+1)} = \mathbf{x}^{(m)} - \hat{\alpha} \nabla g(\mathbf{x}^{(m)})\,.$

Note that the method always converges to a minimum. This may be the global minimum where $g(\mathbf{x}) = 0$, or it may be a local minimum where $g(\mathbf{x}) \neq 0$.

Example in lecture: Apply the method to the system of two equations

$$x_1^2 - 10x_2 + 8 = 0\,, \qquad x_1 x_2^2 + x_1 + 8 = 0\,.$$

2 Differentiation and Integration

2.1 Differentiation

Why is there a need to find approximation formulas for the derivative of a function?

- They are needed for numerically solving differential equations (see later).
- They can be useful if a function is only known at discrete points.

We start from the definition of the derivative

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h} \,.$$

This definition suggests the following approximation for small h

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} \,.$$

This is a **forward difference approximation** if $h > 0$. The corresponding **backward difference approximation** is

$$f'(x) \approx \frac{f(x) - f(x-h)}{h} \,, \qquad h > 0 \,.$$

What is the error of these approximations? It can be obtained from Taylor's theorem. Assume that $f \in C^2[a,b]$, $x_0, x_0 + h \in [a,b]$ and $h > 0$. Then

$$f(x_0 + h) = f(x_0) + h\, f'(x_0) + \frac{h^2}{2} f''(\xi) \,,$$

where $x_0 < \xi < x_0 + h$. Solving this equation for $f'(x_0)$ results in

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} - \frac{h}{2} f''(\xi) \,.$$

This shows that the error E of the approximation is of order h

$$E = -\frac{h}{2} f''(\xi) = O(h) \,,$$

and the approximation is a **first order scheme**. The error E here is a **truncation error** because it arises from a truncation of the Taylor series.

If we use more points we can get better approximations.

Three-point central difference approximation for $f'(x_0)$: We try to find an approximation for $f'(x_0)$ in the following form

$$f'(x_0) \approx \alpha f(x_0 + h) + \beta f(x_0) + \gamma f(x_0 - h)\,.$$

We assume $f \in C^3[a,b]$, $x, x \pm h \in [a,b]$, and $h > 0$ and apply Taylor's theorem

$$\begin{aligned}
& \alpha f(x_0 + h) + \beta f(x_0) + \gamma f(x_0 - h) \\
= \;& \alpha \left[f(x_0) + h f'(x_0) + \frac{h^2}{2} f''(x_0) + \frac{h^3}{6} f'''(\xi_+) \right] + \beta f(x_0) \\
& + \gamma \left[f(x_0) - h f'(x_0) + \frac{h^2}{2} f''(x_0) - \frac{h^3}{6} f'''(\xi_-) \right] \\
= \;& f(x_0)\,(\alpha + \beta + \gamma) + h f'(x_0)\,(\alpha - \gamma) + \frac{h^2}{2} f''(x_0)\,(\alpha + \gamma) \\
& + \frac{h^3}{6}\,[\alpha f'''(\xi_+) - \gamma f'''(\xi_-)]\,,
\end{aligned}$$

where $x_0 < \xi_+ < x_0 + h$ and $x_0 - h < \xi_- < x_0$.

To obtain an approximation for the derivative $f'(x_0)$ we require

$$\alpha + \beta + \gamma = 0\,, \qquad h(\alpha - \gamma) = 1\,, \qquad \alpha + \gamma = 0\,.$$

The solution to this system of equations is given by

$$\alpha = \frac{1}{2h}\,, \qquad \beta = 0\,, \qquad \gamma = -\frac{1}{2h}\,.$$

After inserting these values and solving for $f'(x_0)$ we obtain the approximation

$$f'(x_0) = \frac{f(x_0 + h) - f(x_0 - h)}{2h} - \frac{h^2}{12}[f'''(\xi_+) + f'''(\xi_-)]\,.$$

The first term on the right-hand side is the approximation for $f'(x_0)$ and the second term is the error. We can slightly simplify the error term. By using the intermediate value theorem we can conclude that there is a point $\xi \in [\xi_-, \xi_+]$, and thus $\xi \in (x_0 - h, x_0 + h)$, that takes the value

$$f'''(\xi) = \frac{1}{2}[f'''(\xi_+) + f'''(\xi_-)]\,.$$

For the application of the intermediate value theorem we used the fact that the mean value lies between the two values. We find that the error of the approximation is

$$E = -\frac{h^2}{6} f'''(\xi) = O(h^2)\,,$$

and the approximation is a second order scheme.

Three-point central difference approximation for $f''(x_0)$: One can use the same form of approximation to obtain an approximation for the second derivative $f''(x_0)$

$$f''(x_0) \approx \alpha f(x_0+h) + \beta f(x_0) + \gamma f(x_0-h)\,.$$

In this case we have to include one more term in the Taylor expansion as will become clear in the following. We assume $f \in C^4[a,b]$, $x, x\pm h \in [a,b]$, and $h>0$ and apply Taylor's theorem

$$\begin{aligned}
& \alpha f(x_0+h) + \beta f(x_0) + \gamma f(x_0-h) \\
= \;& \alpha\left[f(x_0) + hf'(x_0) + \frac{h^2}{2}f''(x_0) + \frac{h^3}{6}f'''(x_0) + \frac{h^4}{24}f^{(4)}(\xi_+)\right] + \beta f(x_0) \\
& + \gamma\left[f(x_0) - hf'(x_0) + \frac{h^2}{2}f''(x_0) - \frac{h^3}{6}f'''(x_0) + \frac{h^4}{24}f^{(4)}(\xi_-)\right] \\
= \;& f(x_0)\,(\alpha+\beta+\gamma) + hf'(x_0)\,(\alpha-\gamma) + \frac{h^2}{2}f''(x_0)\,(\alpha+\gamma) \\
& + \frac{h^3}{6}f'''(x_0)\,(\alpha-\gamma) + \frac{h^4}{24}\left[\alpha f^{(4)}(\xi_+) + \gamma f^{(4)}(\xi_-)\right],
\end{aligned}$$

where $x_0 < \xi_+ < x_0+h$ and $x_0-h < \xi_- < x_0$.

To obtain an approximation for the derivative $f''(x_0)$ we now require

$$\alpha+\beta+\gamma = 0\,, \qquad \alpha-\gamma = 0\,, \qquad \frac{h^2}{2}(\alpha+\gamma) = 1\,.$$

The solution to this system of equations is given by

$$\alpha = \frac{1}{h^2}\,, \qquad \beta = -\frac{2}{h^2}\,, \qquad \gamma = \frac{1}{h^2}\,.$$

After inserting these values and solving for $f''(x_0)$ we obtain the approximation

$$f'(x_0) = \frac{f(x_0+h) - 2f(x_0) + f(x_0-h)}{h^2} - \frac{h^2}{24}[f^{(4)}(\xi_+) + f^{(4)}(\xi_-)]\,.$$

The first term on the right-hand side is the approximation for $f''(x_0)$ and the second term is the error. We can again slightly simplify the error term. By using the intermediate value theorem we can conclude that there is a point $\xi \in [\xi_-, \xi_+]$, and thus $\xi \in (x_0-h, x_0+h)$, that takes the value

$$f^{(4)}(\xi) = \frac{1}{2}[f^{(4)}(\xi_+) + f^{(4)}(\xi_-)]\,.$$

Hence the error of the approximation is

$$E = -\frac{h^2}{12}f^{(4)}(\xi) = O(h^2)\,,$$

and the approximation is a second order scheme.

There are many possibilities to generalise these approximations

- Forward difference formulas (use x_0, x_0+h, x_0+2h).
- Backward difference formulas (use x_0, x_0-h, x_0-2h).
- More point formulas.

Round-off errors

The theoretical errors get arbitrarily small as h is decreased, however this is not true when doing numerical calculations on the computer because of round-off errors. Numerical differentiation is particularly vulnerable to this because one takes differences of almost equal numbers.

Consider the central difference approximation

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} - \frac{h^2}{6} f'''(\xi)\,, \qquad \xi \in (x_0-h, x_0+h)\,.$$

Assume we calculate with finite digit precision. Then $f(x_0 \pm h)$ have errors $e(x_0 \pm h)$ when calculated on the computer. The total error is a sum of the **round-off errors** and the **truncation error**

$$E = \frac{e(x_0+h) - e(x_0-h)}{2h} - \frac{h^2}{6} f'''(\xi)\,.$$

Assume that $e(x_0 \pm h)$ are bounded by ϵ: $|e(x_0 \pm h)| < \epsilon$.

(ϵ depends on the number of digits used in the calculation.)

Assume further that the third derivative is bounded by $|f'''(\xi)| < M$ for $\xi \in (x_0-h, x_0+h)$.

Then

$$|E| \le \left|\frac{e(x_0+h)}{2h}\right| + \left|\frac{e(x_0-h)}{2h}\right| + \left|\frac{h^2}{6} f'''(\xi)\right| < \frac{\epsilon}{h} + \frac{h^2}{6} M\,.$$

The right-hand side of this inequality is minimal if

$$0 = \frac{\mathrm{d}}{\mathrm{d}h}\left(\frac{\epsilon}{h} + \frac{h^2}{6}M\right) = -\frac{\epsilon}{h^2} + \frac{hM}{3}\,, \qquad \Longrightarrow \qquad h_{\mathrm{opt}} = \left(\frac{3\epsilon}{M}\right)^{1/3}\,.$$

This gives an estimate for the optimal choice of h for which the total error is smallest.

Example in lecture: Calculate the derivative of $f(x) = x^9$ at $x = 1$ with the central difference formula.

What can one do to improve the error?

- Higher order formulas.
- Richardson extrapolation (see next section).

2.1.1 Richardson extrapolation

Richardson extrapolation is a method that is generally useful in Numerical Analysis. It can be used if we know how the error of an approximation depends on a parameter h of an approximation, usually the step size h. Assume that a quantity N is approximated by an expression $N_1(h)$ such that the error has the following expansion in h

$$N = N_1(h) + a_1 h + a_2 h^2 + a_3 h^3 + \dots\,, \qquad (a)$$

The quantities a_i, $i = 1, 2, \ldots$ are constants that are not known numerically. We can now halve the step size to obtain another approximation for N

$$N = N_1\left(\frac{h}{2}\right) + a_1\frac{h}{2} + a_2\frac{h^2}{4} + a_3\frac{h^3}{8} + \ldots . \qquad (b)$$

We can eliminate the linear term in h by forming the linear combination $2\,(b) - (a)$

$$\begin{aligned} N &= \left[2N_1\left(\frac{h}{2}\right) - N_1(h)\right] - \frac{a_2}{2}h^2 - \frac{3a_3}{4}h^3 + \ldots \\ &= N_2(h) + b_2h^2 + b_3h^3 + \ldots , \qquad (c) \end{aligned}$$

where $N_2(h) = 2N_1(h/2) - N_1(h)$ and $b_2 = -a_2/2$, ... We obtain a new approximation $N_2(h)$ that is now accurate to order h^2. Note that the only information that we used was the fact that $N_1(h)$ is a first order approximation (the expansion of the error starts with a linear term in h). This process can be repeated. Consider halving h again in equation (c).

$$N = N_2\left(\frac{h}{2}\right) + b_2\frac{h^2}{4} + b_3\frac{h^3}{8} + \ldots , \qquad (d)$$

where $N_2(h/2) = 2N_1(h/4) - N_1(h/2)$. Form the linear combination $[4\,(d) - (c)]/3$ we obtain

$$N = \frac{1}{3}\left[4N_2\left(\frac{h}{2}\right) - N_2(h)\right] - \frac{1}{6}\,b_3h^3\,.$$

The error is now of order h^3, and the approximation is

$$N \approx \frac{1}{3}\left[4N_2\left(\frac{h}{2}\right) - N_2(h)\right] = \frac{1}{3}\left[8N_1\left(\frac{h}{4}\right) - 6N_1\left(\frac{h}{2}\right) + N_1(h)\right] .$$

This process works even better if the expansion involves only even powers of h like for the central difference approximation for $f'(x_0)$, as we will see later. Assume

$$N = N_1(h) + a_2h^2 + a_4h^4 + a_6h^6 + \dots , \qquad (a)$$

Halve the step size to obtain another approximation for N

$$N = N_1\left(\frac{h}{2}\right) + a_2\left(\frac{h}{2}\right)^2 + a_4\left(\frac{h}{2}\right)^4 + a_6\left(\frac{h}{2}\right)^6 + \dots . \qquad (b)$$

The quadratic term in h is eliminated by forming the linear combination $[4\,(b) - (a)]/3$

$$\begin{aligned} N &= \frac{1}{3}\left[4N_1\left(\frac{h}{2}\right) - N_1(h)\right] + \frac{a_4}{3}\left(-\frac{3}{4}h^4\right) + \frac{a_6}{3}\left(-\frac{15}{16}h^3\right) + \dots \\ &= N_2(h) + b_4h^4 + b_6h^6 + \dots . \qquad (c) \end{aligned}$$

We obtain a new approximation $N_2(h)$ that is now accurate to order h^4. Repeat the process and halve h again in equation (c).

$$N = N_2\left(\frac{h}{2}\right) + b_4\left(\frac{h}{2}\right)^4 + b_6\left(\frac{h}{2}\right)^6 + \dots , \qquad (d)$$

and form the linear combination $[16\,(d) - (c)]/15$.

$$N = \frac{1}{15}\left[16N_2\left(\frac{h}{2}\right) - N_2(h)\right] + \frac{b_6}{15}\left(-\frac{3}{4}h^6\right) .$$

The error is now of order h^6, and the approximation is

$$N \approx \frac{1}{15}\left[16N_2\left(\frac{h}{2}\right) - N_2(h)\right] = \frac{1}{45}\left[64N_1\left(\frac{h}{4}\right) - 20N_1\left(\frac{h}{2}\right) + N_1(h)\right] .$$

Remark: Instead of halving h at each step we can also double it as we will see in later examples.

Example: We can apply this, for example to the central difference approximation for the derivative $f'(x_0)$

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} - \frac{h^2}{6}f'''(\xi)\,.$$

Note that the value of ξ in this formula depends on h. We can, however, obtain a complete expansion in h by expanding $f(x_0 \pm h)$ in all orders in a Taylor expansion. This results in

$$f'(x_0) = \frac{f(x_0+h) - f(x_0-h)}{2h} - \frac{1}{h}\sum_{k=1}^{\infty}\frac{h^{2k+1}}{(2k+1)!}f^{(2k+1)}(x_0)\,.$$

If we apply this to our previous example $f(x) = x^9$ with $x_0 = 1$ and $h = 10^{-2}$ we find that the error for the approximation $N_3(h)$ is only 0.33×10^{-12}.

2.2 Integration

Many types of integrals cannot be evaluated analytically and one needs numerical methods. These involve discretisations, and they approximate integrals of the form

$$\int_a^b f(x)\,\mathrm{d}x \qquad \text{by sums of the form} \qquad \sum_{i=1}^{n} c_i f(x_i)\,.$$

We start with methods that are based on interpolating polynomials.

2.2.1 Lagrange polynomials

Theorem: Given a function $f(x)$ whose value is known at $(n+1)$ distinct points $x_0, \ldots, x_n$ there exists a unique polynomial $P(x)$ of degree at most n such that

$$f(x_k) = P(x_k) \qquad \text{for} \qquad k = 0, 1, \ldots, n\,.$$

This polynomial is

$$P(x) = \sum_{j=0}^{n} \prod_{\substack{i=0 \\ i\neq j}}^{n} \frac{(x - x_i)}{(x_j - x_i)} f(x_j)\,.$$

Proof:

$$P(x_k) = \sum_{j=0}^{n} \prod_{\substack{i=0 \\ i\neq j}}^{n} \frac{(x_k - x_i)}{(x_j - x_i)} f(x_j) = \prod_{\substack{i=0 \\ i\neq k}}^{n} \frac{(x_k - x_i)}{(x_k - x_i)} f(x_k) = f(x_k)\,,$$

where we used the fact that the sum gives only a non-vanishing contribution if $j = k$. For uniqueness we assume that there exist two polynomials of degree at most n, $P_1(x)$ and $P_2(x)$, that satisfy

$$P_1(x_k) = P_2(x_k) = f(x_k) \qquad \text{for} \qquad k = 0, 1, \ldots, n\,.$$

Then $P_3(x) = P_1(x) - P_2(x)$ is a polynomial of degree at most n that satisfies

$$P_3(x_k) = 0 \qquad \text{for} \qquad k = 0, 1, \ldots, n\,.$$

But a polynomial of degree at most n that vanishes at $n+1$ points has to vanish identically (because a polynomial of degree n can have at most n different zeros). Hence $P_1(x)$ and $P_2(x)$ are identical and uniqueness follows.

Example 1: Between two points $(x_0, f(x_0))$ and $(x_1, f(x_1))$ the straight line is

$$P_1(x) = \frac{(x - x_1)}{(x_0 - x_1)} f(x_0) + \frac{(x - x_0)}{(x_1 - x_0)} f(x_1)\,.$$

Example 2: Between three points the unique quadratic is

$$P_1(x) = \frac{(x - x_1)\,(x - x_2)}{(x_0 - x_1)\,(x_0 - x_2)} f(x_0) + \frac{(x - x_0)\,(x - x_2)}{(x_1 - x_0)\,(x_1 - x_2)} f(x_1) + \frac{(x - x_0)\,(x - x_1)}{(x_2 - x_0)\,(x_2 - x_1)} f(x_2)\,.$$

One can also specify the error of the approximation by interpolating polynomials

Theorem: Let $x_0, \ldots, x_n$ be distinct points in the interval $[a, b]$ and let $f \in C^{n+1}[a, b]$. Then for each $x \in [a, b]$ there exists $\xi(x) \in (a, b)$ such that

$$f(x) = P(x) + \frac{f^{(n+1)}(\xi(x))}{(n+1)!}(x - x_0)\,(x - x_1)\ldots(x - x_n)\,,$$

where $P(x)$ is the interpolating Lagrange polynomial.

Note the similarity of the error term with the one in Taylor's theorem.

$$f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k + \frac{f^{n+1}(\xi(x))}{(n+1)!}(x - x_0)^{n+1}\,.$$

A proof of the theorem can be found in the book of Burden and Faires, but we will not discuss it here.

In what follows we will mainly use the results for linear and quadratic fits.

2.2.2 The trapezoidal rule

In our first approximation for the integral $\int_a^b f(x)\,\mathrm{d}x$ we use the linear Lagrange polynomial

$$P_1(x) = \frac{(x - b)}{(a - b)}f(a) + \frac{(x - a)}{(b - a)}f(b)\,.$$

Thus we obtain

$$\int_a^b f(x)\,\mathrm{d}x = \int_a^b P_1(x)\,\mathrm{d}x + \frac{1}{2}\int_a^b f''(\xi(x))\,(x - a)\,(x - b)\,\mathrm{d}x\,.$$

The integral over the polynomial can be easily evaluated and results in

$$\int_a^b P_1(x)\,\mathrm{d}x = \left[\frac{1}{2}\frac{(x - b)^2}{(a - b)}f(a) + \frac{1}{2}\frac{(x - a)^2}{(b - a)}f(b)\right]_a^b = \frac{b - a}{2}f(a) + \frac{b - a}{2}f(b)\,.$$

For the integration of the error term we need the

Weighted mean value theorem for integrals: Suppose $f \in C[a, b]$, g is integrable on $[a, b]$, and $g(x)$ does not change sign on $[a, b]$. Then there exists a number $c \in (a, b)$ such that

$$\int_a^b f(x)\,g(x)\,\mathrm{d}x = f(c)\int_a^b g(x)\,\mathrm{d}x\,.$$

We briefly discuss a proof in the lecture (not examinable).

We apply this theorem to the integration of the error term. We assume that $f \in C^2[a,b]$ and we define $g(x) = (x-a)(x-b)$. Note that $g(x)$ does not change sign in $[a,b]$. Then there exists a number $\xi \in (a,b)$ such that

$$\begin{aligned}\frac{1}{2}\int_a^b f''(\xi(x))\,(x-a)\,(x-b)\,\mathrm{d}x &= \frac{1}{2}f''(\xi)\int_a^b (x-a)\,(x-b)\,\mathrm{d}x \\ &= \frac{1}{2}f''(\xi)\left[\frac{1}{3}x^3 - \frac{(a+b)}{2}x^2 + abx\right]_a^b \\ &= \frac{1}{2}f''(\xi)\,\frac{1}{6}\left[2b^3 - 2a^3 - 3(ab^2+b^3) + 3(ba^2+a^3) + 6ab^2 - 6a^2b\right] \\ &= -\frac{1}{12}(b-a)^3 f''(\xi)\end{aligned}$$

Let us express the result with $a = x_0$, $b = x_1$ and $h = x_1 - x_0$, where $x_1 > x_0$,

$$\int_{x_0}^{x_1} f(x)\,\mathrm{d}x = \frac{h}{2}\left[f(x_0) + f(x_1)\right] - \frac{h^3}{12}f''(\xi)\,,$$

where $\xi \in (x_0, x_1)$. The name of the method arises from the fact that the area under the function $f(x)$ is estimated by a trapezium.

2.2.3 Composite trapezoidal rule

The trapezoidal rule does not in general give a good approximation unless the integration range h is small. For large intervals one can improve the approximation by dividing the interval into smaller subintervals and applying the trapezoidal rule to each of the smaller intervals. This is the composite trapezoidal rule.

We consider an interval $[a,b]$ that we divide into n equal subintervals of length $h = (b-a)/n$. We denote the borders of the subintervals by $x_i = a + ih$, $i = 0, 1, \ldots, n$, and apply the trapezoidal rule to each subinterval

$$\int_a^b f(x)\,\mathrm{d}x = \sum_{i=1}^{n}\left[\frac{h}{2}\left(f(x_{i-1}) + f(x_i)\right) - \frac{h^3}{12}f''(\xi_i)\right]\,,$$

where $x_{i-1} < \xi_i < x_i$. We can use the intermediate value theorem to simplify the sum over the error terms

$$\frac{1}{n}\sum_{i=1}^{n} f''(\xi_i) = f''(\xi)\,, \qquad n = \frac{(b-a)}{h}\,,$$

for some $\xi \in (a,b)$. For the application of the intermediate value theorem we used the fact that the average of the error terms has to lie between the minimum and the maximum of the error terms. The final result can be written as

$$\int_a^b f(x)\,\mathrm{d}x = T_n - \frac{h^2}{12}(b-a)\,f''(\xi)\,,$$

where $\xi \in (a,b)$ and the composite trapezoidal rule for the integral is given by

$$T_n = \frac{h}{2}\left[f(x_0) + 2f(x_1) + 2f(x_2) + \ldots + 2f(x_{n-1}) + f(x_n)\right]\,.$$

We saw that the local truncation error for a subinterval is $-h^3\, f''(\xi_i)/12$.

The global truncation error on the other hand is $-h^2(b-a)\, f''(\xi)/12$.

This is one order in h less due to adding up $n = (b-a)/h$ errors.

In the global error the term ξ is a function of h. One can derive a complete expansion of the error term in powers of h. This is the content of the following formula

Euler-Maclaurin formula (1735)

Assume that $f \in C^{2m+2}[a,b]$ and let $h = (b-a)/n$. Then the following holds

$$\int_a^b f(x)\,\mathrm{d}x = \frac{h}{2}\left[2\sum_{i=0}^{n} f(a+ih) - f(a) - f(b)\right] - \sum_{k=1}^{m} \frac{h^{2k} B_{2k}}{(2k)!}\left[f^{(2k-1)}(b) - f^{(2k-1)}(a)\right] + E_m\,.$$

Here the first term on the right-hand side is the composite trapezoidal rule

$$T_n = \frac{h}{2}\left[2\sum_{i=0}^{n} f(a+ih) - f(a) - f(b)\right].$$

The next term contains the Bernoulli numbers B_j that are defined by the generating function

$$\frac{t}{\mathrm{e}^t - 1} = \sum_{j=0}^{\infty} B_j \frac{t^j}{j!}\,.$$

The first few terms in the expansion of the generating function are

$$\frac{t}{\mathrm{e}^t - 1} = \frac{t}{\left(t + \frac{1}{2}t^2 + \frac{1}{6}t^3 + \frac{1}{24}t^4 + \ldots\right)} = 1 - \frac{t}{2} + \frac{t^2}{12} - \frac{t^4}{720} + \ldots$$

and the first few Bernoulli numbers follow as

$$B_0 = 1\,, \qquad B_1 = -\frac{1}{2}\,, \qquad B_2 = \frac{1}{6}\,, \qquad B_3 = 0\,, \qquad B_4 = -\frac{1}{30}\,, \qquad \ldots$$

The error term E_m in the Euler-Maclaurin formula is of order $O(h^{2m+2})$, and the formula provides a powerful connection between sums and integrals. One important feature is that the formula allows to obtain an expansion of the error term of the composite trapezoidal rule. In particular we see that the expansion of the correction to the composite trapezoidal rule involves only even powers in h

$$\int_a^b f(x)\,\mathrm{d}x = T_n + c_2 h^2 + c_4 h^4 + c_6 h^6 + \ldots$$

We have seen before that Richardson extrapolation can be used to improve an approximation if an expansion of the error term in the step size h is known, and that it is particularly powerful if the expansion contains only even powers of h. We will later apply Richardson extrapolation to the composite trapezoidal rule. This is known as **Romberg integration**.

2.2.4 Simpson's rule

One can improve the approximation of the trapezoidal rule by using a quadratic Lagrange polynomial instead of a linear polynomial to approximate the function $f(x)$

$$f(x) = P_2(x) + \frac{1}{3!} f^{(3)}(\xi(x)) \, (x - x_0) \, (x - x_1) \, (x - x_2) \, .$$

We consider now an interval $[a, b]$ and define $x_0 = a$, $x_2 = b$, $x_1 = (x_0 + x_2)/2$ so that the point x_1 divides the interval into two equal parts. The three points x_0, x_1 and x_2 are used to specify the Lagrange polynomial. Furthermore $h = (b - a)/2$. One can then proof the following

$$\int_{x_0}^{x_2} f(x) \, \mathrm{d}x = \frac{h}{3} \left[f(x_0) + 4f(x_1) + f(x_2) \right] - \frac{h^5}{90} f^{(4)}(\xi) \, ,$$

where $\xi \in (x_0, x_2)$. The first term on the right-hand side is obtained by integrating over the Lagrange polynomial $P_2(x)$. This is not difficult but lengthy. The derivation of the error term on the other hand is tricky. Integrating the error term of the Lagrange polynomial one would expect an error term of order $O(h^4)$ whereas the exact error term is of order $O(h^5)$. This is due to some cancellation.

We will not derive the error term here but remark that one can easily check that the error term is indeed of order $O(h^5)$. This can be done in the following way. Define $F(x) = \int f(x) \, \mathrm{d}x$. Then the error term can be written as

$$E = F(x_1 + h) - F(x_1 - h) - \frac{h}{3} \left[F'(x_1 + h) + 4F'(x_1) + F'(x_1 - h) \right] \, .$$

Inserting Taylor expansions one obtains after some calculations

$$E = -\frac{h^5}{90} F^{(5)}(x_1) + \ldots = -\frac{h^5}{90} f^{(4)}(x_1) + \ldots$$

(The difference to the form $-h^5 f^{(4)}(\xi)/90$ comes from the fact that ξ depends on h).

One sees that the error term depends on the fourth derivative of the function $f(x)$. This means that Simpson's rule is exact for polynomials of degree 3 or less.

One can continue this process and use higher and higher degree polynomials to approximate integrals. This results in the **Newton-Cotes formulas**. This process is, however, problematic due to **Runge's phenomenon**. It occurs if one uses high-degree polynomials to interpolate a function at **equidistant points**. This leads to oscillations at the edge of the interval similar to Gibbs phenomenon when using Fourier series.

The conclusion is that increasing the degree does not always improve the accuracy. It is better to use the trapezoidal rule or Simpson's rule over subdivisions of intervals.

2.2.5 Composite Simpson's rule

For the composite Simpson's rule we divide the integration range $[a, b]$ into smaller intervals and apply Simpson's rule to the subintervals. We need an even number of subintervals, because Simpson's rule involves two adjacent subintervals.

Divide the interval $[a, b]$ into $n = 2m$ subintervals and set

$$h = \frac{b-a}{n}\,, \qquad x_i = a + ih\,, \qquad f_i = f(x_i)\,, \qquad i = 0, 1, \ldots, n\,.$$

Apply Simpson's rule to each pair of subintervals. This results in

$$\begin{aligned}\int_a^b f(x)\,\mathrm{d}x = \; & \frac{h}{3}\left[f(x_0) + 4f(x_1) + f(x_2)\right] - \frac{h^5}{90} f^{(4)}(\xi_1) \\ & + \frac{h}{3}\left[f(x_2) + 4f(x_3) + f(x_4)\right] - \frac{h^5}{90} f^{(4)}(\xi_2) \\ & + \ldots \\ & + \frac{h}{3}\left[f(x_{n-2}) + 4f(x_{n-1}) + f(x_n)\right] - \frac{h^5}{90} f^{(4)}(\xi_m)\end{aligned}$$

The error term can again be simplified by using the intermediate value theorem

$$\frac{1}{m}\sum_{k=1}^{m} f^{(4)}(\xi_k) = f^{(4)}(\xi) \qquad \text{where} \qquad \xi \in (a, b)\,, \qquad m = \frac{n}{2} = \frac{b-a}{2h}\,.$$

The final result can be written as

$$\int_a^b f(x)\,\mathrm{d}x = S_n - \frac{h^4\,(b-a)}{180} f^{(4)}(\xi)\,,$$

where $\xi \in (a, b)$ and S_n is the approximation according to the composite Simpson's rule

$$S_n = \frac{h}{3}\left[f_0 + 4f_1 + 2f_2 + 4f_3 + 2f_4 + \ldots + 2f_{n-2} + 4f_{n-1} + f_n\right]\,.$$

2.2.6 Romberg integration

Romberg integration is Richardson's extrapolation applied to the composite trapezoidal rule. The Euler-Maclaurin formula showed us that

$$\int_a^b f(x)\,\mathrm{d}x = T_n + c_2 h^2 + c_4 h^4 + c_6 h^6 + \dots\,,$$

where T_n is given by the composite trapezoidal rule. The idea of Richardson's extrapolation is to consider an approximation for different values of h, and to eliminate error terms one by one by forming linear combinations of these approximations.

To do this systematically one chooses the number of subintervals n to be a power of two, i. e. one considers the cases $n = 2^i$ for $i = 0, 1, \dots$. The corresponding values of the step size h are

$$h = \frac{(b-a)}{n}\,, \qquad n = 2^i\,, \qquad \Longrightarrow \qquad h = (b-a)\,2^{-i}\,, \qquad i = 0, 1, 2, \dots\,.$$

The starting point for the Romberg integration is the composite trapezoidal rule. One denotes

$$R_{i,0} = T_n \qquad \text{where} \qquad n = 2^i\,.$$

This approximation is accurate to order h^2 and satisfies

$$\int_a^b f(x)\,\mathrm{d}x = R_{i,0} + c_2 h^2 + c_4 h^4 + c_6 h^6 + \dots\,, \qquad (a)$$

In order to derive the next level of the Romberg integration one compares this approximation to the one for $i-1$ which has double the step size

$$\int_a^b f(x)\,\mathrm{d}x = R_{i-1,0} + c_2 (2h)^2 + c_4 (2h)^4 + c_6 (2h)^6 + \dots\,, \qquad (b)$$

Comparing (a) and (b) shows that the error term that is quadratic in h can be eliminated by forming the linear combination (4(a)-(b))/3, which results in

$$\int_a^b f(x)\,\mathrm{d}x = R_{i,1} + c_4' h^4 + c_6' h^6 + \dots\,, \qquad (c)$$

where $R_{i,1}$ is the next level of the Romberg integration. It is given by

$$R_{i,1} = \frac{1}{3}\left(4R_{i,0} - R_{i-1,0}\right)\,.$$

The new approximation $R_{i,1}$ is now accurate to order h^4. The coefficients c_4' and c_6' can be obtained from the previous coefficients c_4 and c_6 but their exact values are not important.

This process can be repeated. Again one compares the approximation for a certain value of i with the approximation for $i-1$ which has double the step size

$$\int_a^b f(x)\,\mathrm{d}x = R_{i-1,1} + c_4'(2h)^4 + c_6'(2h)^6 + \dots\,, \qquad (d)\,.$$

Comparing equations (c) and (d) shows that the error term proportional to h^4 can be eliminated by forming the linear combination (16(c) - (d))/15. This leads to

$$\int_a^b f(x)\,\mathrm{d}x = R_{i,2} + c_6''h^6 + \dots\,,$$

where

$$R_{i,2} = \frac{1}{15}\left(16R_{i,1} - R_{i-1,1}\right)\,.$$

It is not too difficult to see how this process continues. At the j-th iteration of the Romberg integration an error term of order h^{2j} is eliminated. This is achieved by forming the linear combination

$$R_{i,j} = \frac{1}{4^j - 1}\left(4^j R_{i,j-1} - R_{i-1,j-1}\right)\,.$$

Discussion of a numerical example: On the next page there is a numerical example that is discussed in the lecture. It contains an example for the composite trapezoidal rule, the composite Simpson's rule and the Romberg integration.

The numerical example seems to indicate that the first iteration of the Romberg integration gives the composite Simpson's rule. This is indeed correct as is shown in the following

$$\begin{aligned}
\frac{4}{3}T_n &= \frac{h}{3}\left[2f_0 + 4f_1 + 4f_2 + 4f_3 + 4f_4 + \ldots + 4f_{n-1} + 2f_n\right] \\
-\frac{1}{3}T_{n/2} &= \frac{2h}{6}\left[-f_0 - 2f_2 - 2f_4 - \ldots - 2f_{n-2} - f_n\right] \\
\frac{1}{3}\left(4T_n - T_{n/2}\right) &= \frac{h}{3}\left[f_0 + 4f_1 + 2f_2 + 4f_3 + 2f_4 + \ldots + 2f_{n-2} + 4f_{n-1} + f_n\right] = S_n
\end{aligned}$$

One can show that the second iteration of the Romberg integration also corresponds to a Newton-Cotes formula, but higher ones don't. They are more stable than the Newton-Cotes formulas.

Numerical example:

We will evaluate the integral $\int_0^1 x^7\,\mathrm{d}x = 0.125$ by different numerical methods. These methods are based on subdivisions of the interval $[0,1]$ into n equal pieces. We choose n to be powers of 2 and denote $h = 1/n$. Furthermore, $f_j = f(x_j)$ where $x_j = jh$ and $f(x) = x^7$.

Composite Trapezoidal Rule: $T_n = h[f_0 + 2f_1 + 2f_2 + 2f_3 + 2f_4 + \ldots + 2f_{n-1} + f_n]/2$

n	T_n	error E_n
2^0	0.5000000000	0.3750000000
2^1	0.2539062500	0.1289062500
2^2	0.1603393555	0.0353393555
2^3	0.1340436935	0.0090436935
2^4	0.1272742003	0.0022742003
2^5	0.1255693834	0.0005693834
2^6	0.1251423980	0.0001423980
2^7	0.1250356028	0.0000356028
2^8	0.1250089009	0.0000089009
2^9	0.1250022252	0.0000022252

Composite Simpson's Rule: $S_n = h[f_0 + 4f_1 + 2f_2 + 4f_3 + 2f_4 + \ldots + 4f_{n-1} + f_n]/3$

n	S_n	error E_n
2^1	0.1718750000	0.0468750000
2^2	0.1291503906	0.0041503906
2^3	0.1252784729	0.0002784729
2^4	0.1250177026	0.0000177026
2^5	0.1250011111	0.0000011111
2^6	0.1250000695	0.0000000695
2^7	0.1250000043	0.0000000043

Romberg Integration: The starting point is $R_{i,0} = T_n$, where $n = 2^i$, and we have the iteration rules: $R_{i,1} = (4R_{i,0} - R_{i-1,0})/3$, $R_{i,2} = (16R_{i,1} - R_{i-1,1})/15$, and $R_{i,3} = (64R_{i,2} - R_{i-1,2})/63$.

i	$\mathcal{O}(h^2)$	$\mathcal{O}(h^4)$	$\mathcal{O}(h^6)$	$\mathcal{O}(h^8)$
1	$R_{1,0} = 0.2539062500$			
2	$R_{2,0} = 0.1603393555$	$R_{2,1} = 0.1291503906$		
3	$R_{3,0} = 0.1340436935$	$R_{3,1} = 0.1252784729$	$R_{3,2} = 0.1250203451$	
4	$R_{4,0} = 0.1272742003$	$R_{4,1} = 0.1250177026$	$R_{4,2} = 0.1250003179$	$R_{4,3} = 0.1250000000$

Observe that $R_{i,1} = S_n$. The first iteration agrees with the composite Simpson's rule.

2.2.7 Gaussian quadrature

We are now going to discuss a very powerful numerical integration method. Up to now the integral of a function has been approximated by a sum of its functional values at a set of **equally spaced points** multiplied by a certain aptly chosen weighting function. The idea of the Gaussian quadratures is to free up the location of the sampling points so that these may now be chosen optimally. We start with a definition.

Definition: An integrable function w is called a **weight function** on an interval $I = (a, b)$ if $w(x) \geq 0$ on I but $w(x) \not\equiv 0$ on any subinterval of I.

Remark: $w(x)$ can vanish at most at isolated points in the interval. Here and in the following it is allowed that the interval extends to $a = -\infty$ and/or $b = +\infty$.

The general idea of the Gaussian quadrature is to find sampling points x_i and weights w_i such that the integral approximation

$$\int_a^b w(x)\, f(x)\, \mathrm{d}x = \sum_{i=1}^{n} w_i\, f(x_i)\,,$$

is exact if $f(x)$ is a polynomial of degree $(2n-1)$ or less. Here $w(x)$ is a weight function on the interval (a, b). The reason why this might be possible is that a polynomial of degree $(2n-1)$ has $2n$ coefficients a_i

$$f(x) = \sum_{i=0}^{2n-1} a_i x^i = a_{2n-1}x^{2n-1} + a_{2n-2}x^{2n-2} + \ldots + a_1 x + a_0\,,$$

and we have $2n$ variables to select (n weights w_i and n locations x_i). Before we start with the discussion of Gaussian quadrature we make a brief excursion to orthogonal polynomials.

Orthogonal polynomials

A set of functions $\{\phi_i,\, i = 0, \ldots, n\}$ is said to be **orthogonal** on an interval $[a, b]$ with respect to a continuous weight function $w(x)$ provided that

$$\int_a^b w(x)\, \phi_j(x)\, \phi_k(x)\, \mathrm{d}x = \begin{cases} 0 & \text{when } j \neq k\,, \\ \alpha_j > 0 & \text{when } j = k\,. \end{cases}$$

If in addition $\alpha_j = 1$ for each $j = 0, 1, \ldots, n$ the set is said to be **orthonormal**. In the following we assume that $\{\phi_i,\, i = 0, \ldots, n\}$ is a set of orthogonal polynomials such that ϕ_j is of degree j. In the following we will need three properties of orthogonal polynomials.

Properties of orthogonal polynomials

(a) Any polynomial of degree $k \leq n$ can be written as

$$P(x) = \sum_{j=0}^{k} a_j \phi_j(x) \,.$$

Proof: Choose a_k such that the x^k terms on both sides of the equation have the same coefficients. Then $P(x) - a_k \phi_k(x)$ is a polynomial of degree $(k-1)$.

$$P(x) - a_k \phi_k(x) = \sum_{j=0}^{k-1} a_j \phi_j(x) \,.$$

Continue this procedure downwards.

Remark: The coefficients can be calculated directly by using the orthogonality property. Consider

$$\int_a^b w(x)\, P(x)\, \phi_j(x)\, \mathrm{d}x = \sum_{i=0}^{k} a_i \int_a^b w(x)\, \phi_i(x)\, \phi_j(x)\, \mathrm{d}x = a_j \int_a^b w(x)\, \phi_j(x)\, \phi_j(x)\, \mathrm{d}x \,.$$

It follows that

$$a_j = \frac{\int_a^b w(x)\, P(x)\, \phi_j(x)\, \mathrm{d}x}{\int_a^b w(x)\, \phi_j(x)\, \phi_j(x)\, \mathrm{d}x} \,.$$

(b) $\phi_k(x)$ is orthogonal to any polynomial $P(x)$ of lower degree $l < k$.

Proof:

$$\int_a^b w(x)\, P(x)\, \phi_k(x)\, \mathrm{d}x = \sum_{i=0}^{l} a_i \int_a^b w(x)\, \phi_i(x)\, \phi_k(x)\, \mathrm{d}x = 0 \,,$$

because $i \leq l < k$.

(c) Any of the polynomials $\phi_k(x)$ has k distinct zeros in (a, b).

Proof: Let z_i, $i = 1, \ldots, l$ denote all zeros of $\phi_k(x)$ of odd multiplicity in (a, b). The function $S(x) = \prod_{i=1}^{l} (x - z_i)$ is a polynomial of degree $l \leq k$ since $\phi_k(x)$ cannot have more than k zeros. The function $S(x)$ changes sign at the same positions as $\phi_k(x)$. This implies that the product $S(x)\phi_k(x)$ never changes sign on (a, b) and consequently

$$\int_a^b w(x)\, S(x)\, \phi_k(x)\, \mathrm{d}x \neq 0 \,.$$

However, $S(x)$ is a polynomial of degree $l \leq k$ and from property (b) follows that $l = k$.

We are now in the position to prove the main theorem for Gaussian quadrature.

Theorem (Gaussian quadrature): Let $\{\phi_i,\, i = 0, \ldots, n\}$ be a set of orthogonal polynomials with respect to a continuous weight function $w(x)$ on an interval (a, b) as before ($\phi_i(x)$ is of degree i). Then the integration formula

$$\int_a^b w(x)\, f(x)\, \mathrm{d}x = \sum_{j=1}^{n} w_j f(x_j)\,,$$

where

$$w_j = \int_a^b w(x) \prod_{\substack{i=1 \\ i\neq j}}^{n} \frac{(x - x_i)}{(x_j - x_i)} \mathrm{d}x$$

and x_i, $i = 1, \ldots, n$ are the n zeros of $\phi_n(x)$, is exact if $f(x)$ is a polynomial of degree $(2n - 1)$ or less.

Remark: If $n = 1$ then $w_1 = \int_a^b w(x)\, \mathrm{d}x$.

Proof: Assume that $f(x)$ is a polynomial of degree $(2n - 1)$ or less. We can write the fraction $f(x)/\phi_n(x)$ as

$$\frac{f(x)}{\phi_n(x)} = q(x) + \frac{r(x)}{\phi_n(x)}\,,$$

where $q(x)$ and $r(x)$ are polynomials of degree $(n - 1)$ or less. Then

$$\int_a^b w(x)\, f(x)\, \mathrm{d}x = \int_a^b w(x)\, q(x)\, \phi_n(x)\, \mathrm{d}x + \int_a^b w(x)\, r(x)\, \mathrm{d}x\,.$$

The first integral on the right-hand side vanishes by property (b) of orthogonal polynomials.

Now use the points $x_1, \ldots, x_n$ to express $r(x)$ by an interpolating Lagrange polynomial

$$\int_a^b w(x)\, r(x)\, \mathrm{d}x = \sum_{j=1}^{n} \int_a^b w(x) \prod_{\substack{i=1 \\ i\neq j}}^{n} \frac{(x - x_i)}{(x_j - x_i)} r(x_j)\, \mathrm{d}x = \sum_{j=1}^{n} w_j\, r(x_j)\,.$$

But

$$r(x_j) = f(x_j) - q(x_j)\phi_n(x_j) = f(x_j)\,,$$

and the result follows.

Summary: The idea of the Gaussian integration is to find an approximation of the form

$$\int_a^b w(x)\, f(x)\, \mathrm{d}x \approx \sum_{i=1}^{n} w_i f(x_i)\,,$$

with weight function $w(x)$ on an interval (a, b) that is exact if $f(x)$ is a polynomial of degree $(2n - 1)$ or less.

This is achieved by introducing polynomials $\{\phi_i,\, i = 0, \ldots, n\}$ that are orthogonal with respect to the weight function $w(x)$ on the interval (a, b). The sampling points x_j and weights w_j are then obtained from the zeros of $\phi_n(x)$.

There are many classical sets of orthogonal polynomials that have been studied for a long time. They depend on the interval (a, b) and the weight function $w(x)$. The orthogonality condition determines the polynomials up to a multiplicative constant. This constant is usually specified by a standardisation condition. As a first example we discuss Legendre polynomials.

Example: Legendre polynomials $P_n(x)$

Defined on the interval $(-1, 1)$.

Weight function: $w(x) = 1$.

Standardisation: $P_n(1) = 1$.

First polynomials (see homework):

$$P_0(x) = 1\,, \qquad P_1(x) = x\,, \qquad P_2(x) = \frac{1}{2}(3x^2 - 1)\,, \qquad P_3(x) = \frac{1}{2}(5x^3 - 3x)\,.$$

Recursion relation:

$$(n+1)P_{n+1}(x) = (2n+1)\,x\,P_n(x) - n\,P_{n-1}(x)\,.$$

Let us consider examples for the three previously discussed properties of orthogonal polynomials.

(a) Other polynomials can be expanded in terms of $P_j(x)$. For example

$$Q(x) = x^2 + x + 1 = \sum_{j=0}^{2} a_j P_j(x) = a_2 \frac{1}{2}(3x^2 - 1) + a_1 x + a_0\,.$$

We first require that the coefficients of x^2 agree on both sides of the equation. This fixes $a_2 = 2/3$. Then we subtract the $a_2 P_2(x)$ term on both sides of the equation

$$x^2 + x + 1 - \frac{1}{3}(3x^2 - 1) = x + \frac{4}{3} = a_1 x + a_0\,,$$

and we can read off the other coefficients $a_1 = 1$ and $a_0 = 4/3$.

Alternatively we could have used the formula

$$a_j = \frac{\int_a^b w(x)\,Q(x)\,P_j(x)\,\mathrm{d}x}{\int_a^b w(x)\,P_j(x)\,P_j(x)\,\mathrm{d}x}\,.$$

For example, for $j = 0$

$$a_0 = \frac{\int_{-1}^{1}(x^2 + x + 1)\,\mathrm{d}x}{\int_{-1}^{1}\mathrm{d}x} = \frac{1}{2}\left[\frac{1}{3}x^3 + \frac{1}{2}x^2 + x\right]_{-1}^{1} = \frac{1}{2}\left[\frac{2}{3} + 2\right] = \frac{4}{3}\,,$$

as before.

(b) $P_n(x)$ is orthogonal to any polynomial of degree $l < n$.

For example, consider $P_2(x)$ and $Q(x) = ax + b$.

$$\int_{-1}^{1}(ax + b)\,\frac{1}{2}(3x^2 - 1)\,\mathrm{d}x = \frac{1}{2}\int_{-1}^{1}(3ax^3 + 3bx^2 - ax - b)\,\mathrm{d}x = \frac{1}{2}\left[bx^3 - bx\right]_{-1}^{1} = 0\,.$$

(c) $P_n(x)$ has n distinct zeros in $(-1, 1)$. Examples are

$$\begin{aligned}
P_1(x): &\qquad x_1 = 0\,, \\
P_2(x): &\qquad x_1 = -\frac{1}{\sqrt{3}}\,, \qquad x_2 = \frac{1}{\sqrt{3}}\,, \\
P_3(x): &\qquad x_1 = -\sqrt{\frac{3}{5}}\,, \qquad x_2 = 0\,, \qquad x_3 = \sqrt{\frac{3}{5}}\,.
\end{aligned}$$

Gauss-Legendre quadrature

This is Gaussian quadrature with Legendre polynomials (on the interval $(-1, 1)$ with weight function $w(x) = 1$).

Note that a general interval (a, b) can be mapped onto the interval $(-1, 1)$ with the simple linear transformation $t = (2x - a - b)/(b - a)$.

In order to obtain the approximation formula for n points one has to determine the polynomial $P_n(x)$. This can be done, for example, by using the recurrence relation. Then one has to find its n zeros $x_1, \ldots, x_n$, and calculate the weights

$$w_j = \int_a^b w(x) \prod_{\substack{i=1 \\ i \neq j}}^{n} \frac{(x - x_i)}{(x_j - x_i)} \mathrm{d}x \,.$$

For practical applications, the weights w_j and the zeros x_j have been calculated long time ago and are tabulated. They can be found, for example, in the "Handbook of Mathematical Functions" by Abramowitz and Stegun, or in section 3.5 of http://dlmf.nist.gov.

For the first few Legendre polynomials we have

$$\begin{aligned}
P_1(x): &\quad x_1 = 0\,, \quad w_1 = 2\,, \\
P_2(x): &\quad x_1 = -\frac{1}{\sqrt{3}}\,, \quad x_2 = \frac{1}{\sqrt{3}}\,, \quad w_1 = w_2 = 1\,, \\
P_3(x): &\quad x_1 = -\sqrt{\frac{3}{5}}\,, \quad x_2 = 0\,, \quad x_3 = \sqrt{\frac{3}{5}}\,, \quad w_1 = w_3 = \frac{5}{9}\,, \quad w_2 = \frac{8}{9}\,.
\end{aligned}$$

Example in lecture: Evaluate $\int_1^{3/2} \exp(-x^2)\, \mathrm{d}x$.

Other classic orthogonal polynomials are:

Laguerre polynomials $L_n(x)$: Defined on the interval $(0, \infty)$.

Weight function $w(x) = \exp(-x)$.

Standardisation: coefficient of x^n in $L_n(x)$ is $(-1)^n/n!$.

Hermite polynomials $H_n(x)$: Defined on the interval $(-\infty, \infty)$.

Weight function: $w(x) = \exp(-x^2)$.

Standardisation: coefficient of x^n in $H_n(x)$ is 2^n.

In the following we will discuss **Chebyshev Polynomials**. They have the advantage that explicit formulas are known for the polynomials, their zeros, and the weights in the Gaussian-Chebyshev quadrature.

Chebyshev polynomials (of the first kind) $T_n(x)$

Defined on the interval $(-1, 1)$.

Weight function: $w(x) = (1 - x^2)^{-1/2}$.

Standardisation: $T_n(1) = 1$.

Claim: The Chebyshev polynomials are given by $T_n(x) = \cos(n \arccos x)$.

Proof: We first check the standardisation condition

$$T_n(1) = \cos(n \arccos 1) = \cos(0) = 1\,.$$

Next we show by induction that the functions defined by $T_n(x) = \cos(n \arccos x)$ are indeed polynomials of degree n. The first two functions are

$$T_0(x) = \cos(0) = 1\,, \qquad T_1(x) = \cos(\arccos x) = x\,.$$

They are indeed polynomials of degree 0 and 1, respectively. Now consider

$$\begin{aligned} T_{n+1}(x) + T_{n-1}(x) &= \cos((n+1)\arccos x) + \cos((n-1)\arccos x) \\ &= 2\cos(n\ \arccos x)\ \cos(\arccos x) \\ &= 2x\,T_n(x)\,, \end{aligned}$$

where we used an addition theorem for the cosine function. We obtained the following recursion relation

$$T_{n+1}(x) = 2x\,T_n(x) - T_{n-1}(x)\,.$$

This shows that $T_{n+1}(x)$ is a polynomial of degree $(n+1)$ if $T_n(x)$ and $T_{n-1}(x)$ are polynomials of degree n and $n-1$, respectively. We have seen that the first two functions are polynomials of degree 0 and 1, respectively, and this completes the proof.

Finally, we have to show that the functions $T_n(x)$ are orthogonal on the interval $(-1, 1)$ with respect to the weight function $w(x) = (1 - x^2)^{-1/2}$. In the following we make the substitution $x = \cos\theta$, $\mathrm{d}x = -\sin\theta\,\mathrm{d}\theta$.

$$\begin{aligned} \int_{-1}^{1} \frac{1}{\sqrt{1-x^2}}\,T_j(x)\,T_k(x)\,\mathrm{d}x &= \int_0^{\pi} \cos(j\theta)\ \cos(k\theta)\,\mathrm{d}\theta \\ &= \frac{1}{2}\int_0^{\pi} \left[\cos((k+j)\theta) + \cos((k-j)\theta)\right]\,\mathrm{d}\theta \\ &= \frac{1}{2}\left[\frac{\sin((k+j)\theta)}{(k+j)} + \frac{\sin((k-j)\theta)}{(k-j)}\right]_0^{\pi} \\ &= 0 \qquad \text{if} \qquad j \neq k\,. \end{aligned}$$

This concludes the proof that the Chebyshev polynomials are given by $T_n(x) = \cos(n \arccos x)$.

Gauss-Chebyshev quadrature is particularly simple since the zeros x_j and weights w_j are explicitly known. The zeros follow from

$$0 = T_n(x) = \cos(n\ \arccos x) \qquad \Longrightarrow \qquad n\ \arccos x = j\pi - \frac{\pi}{2}\,, \qquad j = 1, \ldots, n\,.$$

Hence the zeros are given by

$$x_j = \cos\left(\frac{(2j-1)\,\pi}{2n}\right), \qquad j = 1, \ldots, n\,.$$

We state without proof that the weights are (see homework for $n = 1, 2, 3.$)

$$w_j = \frac{\pi}{n}\,, \qquad j = 1, \ldots, n\,.$$

2.2.8 Problems in the evaluation of integrals

In many cases the numerical methods that we discussed yield very good results. However, the methods may become inaccurate if the function $f(x)$ is not well behaved. Examples are

- function is discontinuous.
- function is continuous, but derivatives are discontinuous or singular.
- function has integrable singularity.
- range of integration is infinite.
- function is highly oscillatory.

Various techniques are available, based primarily on transformations and breaking integrals into smaller pieces. We consider some examples

(a) An integrable singularity

$$I = \int_0^1 \frac{\exp(x)}{\sqrt{x}} \mathrm{d}x\,.$$

The singularity can be removed with a suitable transformation of variables. Set $x = t^2$, $\mathrm{d}x = 2t\,\mathrm{d}t$

$$I = \int_0^1 2\exp(t^2)\,\mathrm{d}t\,.$$

(b) Integrand is infinite near one endpoint

$$I = \int_{0.1}^1 \frac{1}{x^3}\mathrm{e}^x\,\mathrm{d}x\,.$$

Using Taylor expansion, the integral is split up into two part. The first part, which is the problematic one, can be done exactly. The second part is suitable for numerics.

$$I = \int_{0.1}^1 \frac{1}{x^3}\left(1 + x + \frac{x^2}{2}\right)\,\mathrm{d}x + \int_{0.1}^1 \frac{1}{x^3}\left(\mathrm{e}^x - 1 - x - \frac{x^2}{2}\right)\,\mathrm{d}x\,.$$

Further example in lecture:

$$\int_{-\infty}^{\infty} \exp(-x^2)\,\mathrm{d}x\,.$$

3 Ordinary differential equations: initial value problems (IVPs)

Differential equations play a central role in many scientific fields like mathematics, physics, biology, chemistry, engineering as well as finance and economy. Most differential equations cannot be solved in closed form. The numerical solutions of differential equations is an enormous field, in particular for PDEs. We will deal only with ODEs.

Numerical methods started with Leonhard Euler in the mid 18th century. Primary motivation was an approximate integration of Newton's differential equations for the motion of planets and comets.

We consider the initial value problem (IVP)

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y)\,, \qquad a \le t \le b\,.$$

subject to the initial condition $y(a) = \alpha$.

This is the one-dimensional version of the general first order IVP

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{y} = \mathbf{f}(t, \mathbf{y})\,, \qquad t \in [a, b]\,.$$

subject to the initial condition $\mathbf{y}(a) = \boldsymbol{\alpha}$, where $\mathbf{y}$ is an n-dimensional vector.

This is the most general case since any n-th order ODE can be converted to first order form as will be shown in the following. Consider

$$y^{(n)}(t) = f(t, y, y^{(1)}, \ldots, y^{(n-1)})\,, \qquad t \in [a, b]\,,$$

with initial conditions

$$y(a) = \alpha_1\,, \qquad y'(a) = \alpha_2\,, \qquad y''(a) = \alpha_3\,, \qquad \ldots \qquad y^{(n-1)}(a) = \alpha_n\,.$$

Now set $u_j(t) = y^{(j-1)}(t)$ for $j = 1, \ldots, n$. Then we obtain a system of differential equations

$$\frac{\mathrm{d}u_1}{\mathrm{d}t} = u_2\,, \qquad \frac{\mathrm{d}u_2}{\mathrm{d}t} = u_3\,, \qquad \ldots \qquad \frac{\mathrm{d}u_{n-1}}{\mathrm{d}t} = u_n\,, \qquad \frac{\mathrm{d}u_n}{\mathrm{d}t} = f(t, u_1, u_2, \ldots, u_n)\,,$$

with initial condition $\boldsymbol{u}(a) = \boldsymbol{\alpha}$.

Before we try to solve a problem numerically we should know if there is a unique solution.

Definition: A function $f(t, y)$ is said to satisfy a **Lipschitz condition** in the variable y on a set $D \subset \mathbb{R}^2$ if a constant $L > 0$ exists with

$$|f(t, y_1) - f(t, y_2)| < L\,|y_1 - y_2|\,,$$

whenever (t, y_1) and (t, y_2) are in D.

L is called a **Lipschitz constant** for f.

A sufficient condition for the Lipschitz condition is that $\partial f(t, y)/\partial y$ exists and is bounded for all $(t, y) \in D$. This can be proved by using the mean value theorem (see homework).

Theorem: Suppose that $D = \{(t, y)\,|\,a \le t \le b\,, \quad -\infty < y < \infty\}$ and $f(t, y)$ is continuous on D. If f satisfies also a Lipschitz condition on D in the variable y, then the initial value problem

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y)\,, \qquad a \le t \le b\,, \qquad y(a) = \alpha\,,$$

has a unique solution $y(t)$ for $a \le t \le b$.

The following two examples don't satisfy the Lifschitz condition in the variable y on the domain $D = \{(t, y)\,|\,0 \le t \le b\,, \quad -\infty < y < \infty\}$.

Example of non-existence for $t \ge \log 2$:

$$\frac{\mathrm{d}y}{\mathrm{d}t} = y + y^2\,, \qquad \text{with} \qquad y(0) = 1\,.$$

The solution

$$y(t) = \frac{1}{2\mathrm{e}^{-t} - 1}\,,$$

blows up for $t = \log 2$.

Example of non-uniqueness:

$$\frac{\mathrm{d}y}{\mathrm{d}t} = y^{1/2}\,, \qquad \text{with} \qquad y(0) = 0\,.$$

There is an infinity of solutions given by

$$y(t) = \begin{cases} \frac{(t-a)^2}{4} & \text{if} \quad t \ge a\,, \\ 0 & \text{if} \quad t < a\,. \end{cases}$$

Here a is an arbitrary non-negative real constant.

3.1 Euler's method

Consider the initial value problem (IVP)

$$\frac{\mathrm{d}y}{\mathrm{d}t} = f(t, y)\,, \qquad t \in [a, b]\,, \qquad y(a) = \alpha\,.$$

Euler's method is the simplest method to find an approximate solution. The interval $[a, b]$ is divided into N pieces and the solution is approximated at discrete time steps $t_i = a + ih$, $i = 0, \ldots, N$, called **mesh points**, and $h = (b - a)/N$ is the **step size**.

Euler's method is derived by using Taylor's theorem

$$y(t_{i+1}) = y(t_i + h) = y(t_i) + hy'(t_i) + \frac{h^2}{2}y''(\xi_i)\,,$$

for some $\xi_i \in (t_i, t_i + h)$. We ignore the h^2 term and denote the approximation for $y(t_i)$ by y_i. The derivative of y is given by the function f, and we obtain

$$y_{i+1} = y_i + h\, f(t_i, y_i)\,, \qquad i = 0, 1, \ldots, N - 1\,, \qquad y_0 = y(a) = \alpha\,.$$

This is a difference equation. It can be solved numerically by iteration.

Alternatively, Euler's method can also be derived by replacing the derivative in the ODE by the forward difference approximation

$$\frac{\mathrm{d}y}{\mathrm{d}t}(t_i) = f(t_i, y(t_i))\,, \qquad \frac{\mathrm{d}y}{\mathrm{d}t}(t_i) \approx \frac{y(t_i + h) - y(t_i)}{h}\,.$$

This leads again to

$$y_{i+1} = y_i + h\, f(t_i, y_i)\,, \qquad y_0 = \alpha\,.$$

Example in lecture: Discussion of the IVP $y' = -y + t$, $t \in [0, 1]$, $y(0) = 1$.

The plot in the example shows that the error at the final point $t = 1$ seems to be of order h. In the following we analyse the error. One distinguishes between two different errors.

Local truncation error: This is the error that is introduced at one step of the method, assuming that the approximation at the beginning of the step is exact. For Euler's method it is

$$\tau_{i+1} = y(t_{i+1}) - [y(t_i) + hf(t_i, y(t_i))] = y(t_{i+1}) - [y(t_i) + hy'(t_i)]\,.$$

Taylor expansion shows that the local truncation error for Euler's method is $h^2 y''(\xi_i)/2$ and hence it is $O(h^2)$.

Note that in the literature there are different definitions of the local truncation error. They can differ by a factor h from our definition. The definition in the book of Burden and Faires is also different from our definition.

Global error: This is the error at a fixed time, for example at the final time $t_N = b$. In general it is a difficult problem to determine the global error, because it depends on how the errors accumulate and propagate during the iteration process. A rough estimate is the following. The number of steps is $N = (b-a)/h = O(1/h)$, and the error for a single step is $O(h^2)$. The error at the end of the interval is then estimated to be $O(1/h)\,O(h^2) = O(h)$ if we assume that the errors accumulate linearly.

For Euler's method it can be proved that the global error is indeed $O(h)$ using the Lipschitz condition and assuming that $y''(x)$ is bounded on $[a, b]$. (Without Lipschitz condition the error can become infinite. We have seen that the solution to $y' = y^2 + y$, $y(0) = 1$, diverges at $t = \ln 2$, whereas Euler's approximation always stays finite.)

Euler's method is **first order accurate** because the local truncation error is $O(h^2)$ and the global error $O(h)$.

Sometimes one can solve Euler's difference equation exactly and compare it to the exact solution (see the following example and the homework).

Example in lecture: Solve Euler's difference equation exactly for the previous example

$$y' = -y + t, \qquad t \in [0, 1]\,, \qquad y(0) = 1\,.$$

Application of Euler's method to higher order ODEs.

Euler's method can also be applied to higher order ODEs. We'll discuss here second order ODEs, but it is straightforward to generalise the discussion to higher-order ODEs. Consider

$$y'' = f(t, y, y')\,, \qquad t \in [a, b]\,, \qquad y(a) = \alpha\,, \qquad y'(a) = \beta\,.$$

This IVP can be transformed into a system of first order ODEs by setting $u(t) = y'(t)$. This yields

$$\begin{aligned} y' &= u\,, & y(a) &= \alpha\,, & t &\in [a, b]\,, \\ u' &= f(t, y, u)\,, & u(a) &= \beta\,. \end{aligned}$$

We apply Euler's method to each of these lines and let u_i and y_i denote the approximations for $u(t_i)$ and $y(t_i)$, respectively. This results in

$$\begin{aligned} y_{i+1} &= y_i + h\, u_i\,, & y_0 &= \alpha\,, \\ u_{i+1} &= u_i + h\, f(t_i, y_i, u_i)\,, & u_0 &= \beta\,. \end{aligned}$$

This is a system of first order difference equations which can be solved by iteration. Alternatively, we can obtain from the first line $u_i = (y_{i+1} - y_i)/h$ and use this relation to eliminate the u variable. This leads to a second order difference equation in the variable y (see homework sheet 7).

$$\frac{y_{i+2} - y_{i+1}}{h} = \frac{y_{i+1} - y_i}{h} + h\, f\left(t_i, y_i, \frac{y_{i+1} - y_i}{h}\right)\,, \qquad y_0 = \alpha\,, \qquad \frac{y_1 - y_0}{h} = \beta\,.$$

or

$$y_{i+2} = 2y_{i+1} - y_i + h^2\, f\left(t_i, y_i, \frac{y_{i+1} - y_i}{h}\right)\,, \qquad y_0 = \alpha\,, \qquad y_1 = \alpha + h\beta\,.$$

We mentioned that Euler's method is first order accurate and that its global error is $\mathcal{O}(h)$. This is not accurate enough for most applications. We will discuss in the following methods to improve the approximation by reducing the local truncation error.

3.2 Runge-Kutta methods

One way to improve the approximation is by including more terms in the Taylor expansion of

$$y(t_i + h) = y(t_i) + h\, y'(t_i) + \frac{h^2}{2} y''(t_i) + \dots$$

For example, let us include one more term than in Euler's method. We can use the ODE to replace the first derivative by $y'(t) = f(t, y)$, but we also need an expression for the unknown second derivative. It is obtained by differentiating $y' = f(t, y)$

$$y''(t) = \frac{d}{dt} f(t, y(t)) = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial y} y' = f_t + f_y f\,.$$

Using this expression we obtain the following iteration scheme

$$y_{i+1} = y_i + h f(t_i, y_i) + \frac{h^2}{2} \left[f_t(t_i, y_i) + f_y(t_i, y_i)\, f(t_i, y_i) \right] , \qquad y_0 = \alpha\,. \qquad (*)$$

where y_i is an approximation for $y(t_i)$. This is an example of a higher order Taylor method. It is a Taylor method of order 2 and has local truncation error $\mathcal{O}(h^3)$. (Euler's method is Taylor's method of order 1.)

For further improvement one needs to include the next term in the Taylor expansion $h^3 y'''(t_i)/6$ where

$$y'''(t) = \frac{\mathrm{d}}{\mathrm{d}t} y''(t) = \frac{\mathrm{d}}{\mathrm{d}t} [f_t + f_y f] = f_{tt} + f_{ty} f + f_{ty} f + f_{yy} f^2 + f_y f_t + f_y f_y f\,.$$

One sees the problem with this method. Higher order Taylor methods have the advantage of higher order local truncation errors, but they require the evaluation of higher order partial derivatives of the function $f(t, y)$. This can be complicated and time consuming. Runge-Kutta methods try to obtain local truncation errors of the same order but use only evaluations of $f(t, y)$ (and not its derivatives).

To introduce the idea of the Runge-Kutta method we consider an iteration scheme of the form

$$y_{i+1} = y_i + a f(t_i + b, y_i + c)\,.$$

If $a = h$ and $b = c = 0$ this agrees with Euler's method, but by choosing these parameters optimally one can achieve that this scheme differs from Taylor's method only in terms of order $\mathcal{O}(h^3)$. To show this let us apply a Taylor expansion in the two arguments of the function $f(t, y)$

$$y_{i+1} = y_i + a f(t_i, y_i) + a f_t(t_i, y_i) b + a f_y(t_i, y_i) c + \frac{a}{2} \left[f_{tt} b^2 + 2 f_{ty} bc + f_{yy} c^2 \right] + \dots \qquad (**)$$

Comparison with equation (*) shows that we should choose

$$a = h,\ ab = \frac{h^2}{2},\ ac = \frac{h^2}{2} f(t_i, y_i), \quad \Longrightarrow \quad a = h,\ b = \frac{h}{2},\ c = \frac{h}{2} f(t_i, y_i)\,.$$

Since a, b and c are of order h it follows that the next terms in the Taylor expansion in (**) are $\mathcal{O}(h^3)$ as required. Hence the Runge-Kutta iteration scheme has the following form

$$y_{i+1} = y_i + h f\left(t_i + \frac{h}{2}, y_i + \frac{hk}{2} \right) \qquad \text{where} \qquad k = f(t_i, y_i)\,.$$

Note that $t_i + h/2$ is in the middle between t_i and $t_i + 1$, and $y_i + h f(t_i, y_i)/2$ is in the middle between y_i and y_{i+1} in Euler's method. For this reason it is called the midpoint method. It is a

Runge-Kutta method of order 2, because its local truncation error is $\mathcal{O}(h^3)$ and hence we expect a global error of $\mathcal{O}(h^2)$. Its application requires two functional evaluations per step, $k = f(t_i, y_i)$ and $f(t_i + h/2, y_i + hk/2)$.

The following table shows how it compares to the Euler method when applied to our previous example $y' = -y + t$ with $y(0) = 1$. In both cases the step size is $h = 0.1$. The second order method is clearly more accurate.

t	$y(t)$, Euler	$y(t)$, RK2	$y(t)$, exact
0.2	0.82000	0.838050	0.83746
0.4	0.71220	0.741604	0.74064
0.6	0.66288	0.698807	0.69762
0.8	0.66093	0.699950	0.69866
1.0	0.69736	0.737082	0.73575

There are other Runge-Kutta methods of order 2. They can be obtained by starting from

$$y_{i+1} = y_i + af(t_i, y_i) + bf(t_i + c, y_i + d)\,,$$

and determining the parameters a, b, c, and d as before by requiring that the iteration scheme differs from the higher order Taylor method only in terms of order $\mathcal{O}(h^3)$. In this way one obtains a one-parameter family of Runge-Kutta methods of order 2. Some of these methods are discussed in the homework.

If we allow more functional evaluations per step then we can achieve higher order accuracy. Very popular is the fourth order Runge-Kutta method known as RK4. It requires four functional evaluations per step. The derivation is somewhat involved and we cite here only the result. The functional evaluations are

$$\begin{aligned}
k_1 &= f\left(t_i, y_i\right) \\
k_2 &= f\left(t_i + \frac{h}{2}, y_i + \frac{hk_1}{2}\right) \\
k_3 &= f\left(t_i + \frac{h}{2}, y_i + \frac{hk_2}{2}\right) \\
k_4 &= f\left(t_i + h, y_i + hk_3\right)
\end{aligned}$$

and the iteration scheme has the form

$$y_{i+1} = y_i + \frac{h}{6}[k_1 + 2k_2 + 2k_3 + k_4]\,.$$

The next order Runge-Kutta method already requires six functional evaluations.

3.3 Multistep methods

The methods that we have considered so far are called one-step methods. They can be written in the general form

$$y_{i+1} = y_i + h\,\Phi(t_i, y_i, h)\,.$$

They use the approximation y_i from the previous mesh point t_i to evaluate y_{i+1}. After the calculation of y_{i+1} this information is discarded, and the next iteration uses information from the following mesh point.

Multistep methods employ a different philosophy. They use a number of previous approximations $y_i, y_{i-1}, y_{i-2}, \ldots$ and previous mesh points $t_i, t_{i-1}, t_{i-2}, \ldots$ to arrive at an estimate of the solution at t_{i+1}. In this way one can gain efficiency because one can use functional evaluations from previous steps without needing to calculate them again.

An *linear* k-step multistep method uses the approximations from the previous k steps and can be written as

$$\begin{aligned} y_{i+1} =& \sum_{j=1}^{k} \alpha_j\, y_{i+1-j} + h \sum_{j=0}^{k} \beta_j\, f(t_{i+1-j}, y_{i+1-j}) \\ =& \alpha_1 y_i + \alpha_2 y_{i-1} + \ldots + \alpha_k y_{i+1-k} \\ &+ h\beta_0 f(t_{i+1}, y_{i+1}) + h\beta_1 f(t_i, y_i) + \ldots + h\beta_k f(t_{i+1-k}, y_{i+1-k})\,. \end{aligned}$$

Note that if $\beta_0 \neq 0$ then y_{i+1} appears on both sides of the equation. The above equation is then an implicit equation for y_{i+1} that must be solved and the method is called *implicit*. On the other hand, if $\beta_0 = 0$ then the method is *explicit* and above equation can be applied directly to find y_{i+1}. The reason for including implicit equations is that they are often more accurate than explicit methods as we will see later.

Multistep formulas arise naturally by applying approximation formulas for derivatives that we discussed in section 2.1 on Differentiation. For example, if one replaces the derivative in the equation

$$y'(t_i) = f(t_i, y(t_i))$$

by the forward difference formula $y'(t_i) \approx [y(t_{i+1}) - y(t_i)]/h$ one arrives at Euler's method. If one instead uses the more accurate central difference approximation $y'(t_i) \approx [y(t_{i+1}) - y(t_{i-1})]/2h$ one arrives at an example of a multistep method

$$y_{i+1} = y_{i-1} + 2hf(t_i, y_i)\,.$$

The fact that the multistep method requires the approximate solution at k mesh points raises a new difficulty. How do we start the iteration? The initial condition specifies the solution only at one mesh point. The answer is that one can use one-step methods of the required accuracy, like Runge-Kutta, to approximate the solution at the first k mesh points and then continue with the multistep method.

In practise many of the coefficients in the linear multistep formula are chosen to be zero. We will continue with a discussion of the most commonly used linear multistep methods.

The most popular multistep formulas are (with the notation $f_j = f(t_j, y_j)$):

(a) Adams-Bashforth (k-step AB), explicit: $\alpha_1 = 1$, $\alpha_j = 0$ for $j = 2, \ldots, k$ and $\beta_0 = 0$.

$$y_{i+1} = y_i + h\beta_1 f_i + h\beta_2 f_{i-1} + \ldots + h\beta_k f_{i+1-k}\,.$$

(b) Adams-Moulton (k-step AM), implicit: $\alpha_1 = 1$ and $\alpha_j = 0$ for $j = 2, \ldots, k$.

$$y_{i+1} = y_i + h\beta_0 f_{i+1} + h\beta_1 f_i + \ldots + h\beta_k f_{i+1-k}\,.$$

(c) Backward Differentiation formulas (k-step BD), implicit: $\beta_j = 0$ for $j = 1, \ldots, k$.

$$y_{i+1} = \alpha_1 y_i + \alpha_2 y_{i-1} + \ldots + \alpha_k y_{i+1-k} + h\beta_0 f_{i+1}\,.$$

We will discuss in the following how one chooses the coefficients in these formulas. Some examples are

$$\begin{aligned}
&\textbf{AB1:} && y_{i+1} = y_i + hf_i \\
&\textbf{AB2:} && y_{i+1} = y_i + \frac{h}{2}(3f_i - f_{i-1}) \\
&\textbf{AB4:} && y_{i+1} = y_i + \frac{h}{24}(55f_i - 59f_{i-1} + 37f_{i-2} - 9f_{i-3}) \\
&\textbf{AM2:} && y_{i+1} = y_i + \frac{h}{12}(5f_{i+1} + 8f_i - f_{i-1}) \\
&\textbf{BD1:} && y_{i+1} = y_i + hf_{i+1}
\end{aligned}$$

Example: Three methods are applied to the following problem

$$y'(t) = y(t)\,, \qquad t \in [0, 2]\,, \qquad y(0) = 1\,,$$

which has the solution $y(t) = \exp(t)$. The three methods are Euler's method, the multistep method that is obtained by replacing the derivative by the central difference approximation, and AB4. The table below shows the errors at the end point $t = 2$ as a function of h.

$E(h)$	$E(0.2)$	$E(0.1)$	$E(0.05)$	$E(0.1)/E(0.05)$
Euler	1.20	0.662	0.349	$1.9 \approx 2$
CD	0.0906	0.0238	0.00607	$3.92 \approx 4$
AB4	0.0042	0.00038	0.00003	$13.6 \approx 16$

The numerical results indicate that the errors are of order $O(h)$, $O(h^2)$ and $O(h^4)$, respectively, for the three methods. In the following we discuss the errors in more detail.

Order of Accuracy for linear k-step multistep methods: Similar as for Euler's method one defines the **local truncation error** τ_{i+1} as the error that is introduced at the step $(i+1)$ if one assumes that the approximations at the beginning of the step are exact. Let $y(t)$ be a solution of

$$y'(t) = f(t, y(t)) \,, \qquad y(a) = \alpha \,.$$

Then the local truncation error is obtained from the difference between the two sides of the multistep formula if one replaces the approximations y_j by the solution $y(t_j)$

$$\begin{aligned} \tau_{i+1} &= y(t_{i+1}) - \sum_{j=1}^{k} \alpha_j \, y(t_{i+1-j}) - h \sum_{j=0}^{k} \beta_j \, f(t_{i+1-j}, y(t_{i+1-j})) \\ &= y(t_{i+1}) - \sum_{j=1}^{k} \alpha_j \, y(t_{i+1-j}) - h \sum_{j=0}^{k} \beta_j \, y'(t_{i+1-j}) \end{aligned}$$

where we used the differential equation. The local truncation error is then calculated by using Taylor expansion.

Definition: If the local truncation error is $\tau_{i+1} = O(h^{p+1})$ the multistep method is said to have an **accuracy of order p**.

Definition: The multistep method is said to be **consistent** if the local truncation error goes to zero faster than h: $\lim_{h\to 0} \tau_{i+1}/h = 0$.

The definition of the local truncation error agrees with our previous definition for Euler's method which has a local truncation error $O(h^2)$ and an order of accuracy 1. The consistency condition is necessary so that the numerical solution can converge to the exact solution. This will be discussed in the next section.

The order of accuracy p gives us a criterion for the derivation of the multistep methods. The coefficients α_j and β_j for the Adams-Bashforth, Adams-Moulton and Backward Differentiation formulas are chosen in such a way that one obtains the largest possible value of p.

Example: Derivation of AB2. It has the form

$$y_{i+1} = y_i + h\beta_1 f_i + h\beta_2 f_{i-1} \,.$$

Hence we obtain the local truncation error from

$$\begin{aligned} \tau_{i+1} &= y(t_i + h) - y(t_i) - h\beta_1 \, y'(t_i) - h\beta_2 \, y'(t_i - h) \\ &= \left[y + h\, y' + \frac{h^2}{2} y'' + \frac{h^3}{6} y''' + \ldots \right] - y - h\, \beta_1 \, y' - h\, \beta_2 \left[y' - h y'' + \frac{h^2}{2} y''' + \ldots \right] \\ &= h\, y' \left[1 - \beta_1 - \beta_2 \right] + h^2 \, y'' \, \left[\frac{1}{2} + \beta_2 \right] + h^3 y''' \left[\frac{1}{6} - \frac{\beta_2}{2} \right] + \ldots \end{aligned}$$

where the functions were expanded around t_i. We require

$$1 - \beta_1 - \beta_2 = 0 \,, \qquad \frac{1}{2} + \beta_2 = 0 \qquad \Longrightarrow \qquad \beta_1 = \frac{3}{2} \,, \qquad \beta_2 = -\frac{1}{2} \,.$$

This agrees with the formula on the previous page. The coefficient of h^3 does not vanish and we conclude that $\tau_{i+1} = O(h^3)$ and $p = 2$. This shows that AB2 is 2nd order accurate.

Similarly one can show that AB4 is 4th order accurate and AM2 is 3rd order accurate.

In general: One can show that ABk is k-th order accurate and AMk is $(k+1)$-th order accurate (because one has one more free coefficient β_0).

Question: Given $k > 0$ one has $(2k+1)$ coefficients $\alpha_1, \ldots, \alpha_k$ and $\beta_0, \ldots \beta_k$. Why does one not pick them to eliminate the first $(2k+1)$ terms in the expansion of the local truncation error which are proportional to $y^{(0)}$, $y^{(1)}$, ..., $y^{(2k)}$. In this way one could achieve that $\tau_{i+1} = O(h^{2k+1})$ and $p = 2k$.

Answer: The resulting formulas are unstable (for $k > 2$) as we will discuss below, and therefore useless. A famous theorem by Dahlquist asserts that a stable k-step multistep formula can have order of accuracy at most

- k: if explicit.
- $k+1$: if implicit and k is odd.
- $k+2$: if implicit and k is even.

For this reason many of the α_j and β_j are usually set equal to zero.

3.4 Stability

We discussed the local truncation error. It is the error of y_{i+1} if we assume that the previous y_i, y_{i-1}, ... agree with the exact solution. It is a much more difficult problem to determine how the error propagate and accumulate during the iteration process to give the global error. If the local truncation error is $O(h^{p+1})$ this does not always mean that the global error is $O(1/h) \times O(h^{p+1}) = O(h^p)$. We have seen that this is true for Euler's method (under certain conditions), but it is not true in general. In order to investigate the global error we need the notion of **stability**.

One distinguishes between two different forms of stability

(A) Stability (also zero-stability): If $T > 0$ is fixed, does the approximation for $y(T)$ stay bounded as $h \to 0$? This notion of stability is investigated in the present section 3.4.

(B) Time stability (also absolute stability, A-stability): If $h > 0$ is held fixed, does the approximation $y(T)$ remain bounded as $T \to \infty$ (assuming that the exact solution does)? This notion of stability is investigated in the next section.

Example: Consider the following IVP

$$\frac{\mathrm{d}y}{\mathrm{d}t} = y\,, \qquad y(0) = 1\,, \qquad t \in [0,1]\,.$$

It has the solution $y(t) = \exp(t)$ and its value at the end point is $y(1) = \mathrm{e} = 2.71828$. We apply two different multistep methods to it, AB2 and second one that we denote by $(*)$.

$$\begin{aligned} \textbf{AB2:} \qquad & y_{i+1} = y_i + \frac{h}{2}(3f_i - f_{i-1}) \\ \textbf{(*):} \qquad & y_{i+1} = -4y_i + 5y_{i-1} + h(4f_i + 2f_{i-1}) \end{aligned}$$

AB2 has order of accuracy 2, and (*) has order of accuracy 3. We start the iteration in both cases with the exact starting values $y_0 = 1$ and $y_1 = \exp(h)$. The following table shows the results for the value $y(1)$ of the solution at the end point.

h	0.2	0.1	0.05	0.025
AB2	2.68771	2.70881	2.71568	2.71760
(*)	2.73433	-0.12720	-1.62×10^6	-9.34×10^{18}

Clearly the second method does not yield any sensible results. What happened? We will see in the following that AB2 is stable and $(*)$ is not.

In the previous week we considered the example

$$\frac{\mathrm{d}y}{\mathrm{d}t} = y\,, \qquad y(0) = 1\,, \qquad t \in [0,1]\,.$$

We applied the following two multistep methods to it

$$\begin{aligned} \textbf{AB2:} \qquad & y_{i+1} = y_i + \frac{h}{2}(3f_i - f_{i-1})\,, \\ \textbf{(*):} \qquad & y_{i+1} = -4y_i + 5y_{i-1} + h(4f_i + 2f_{i-1})\,. \end{aligned}$$

We found that the first method converged to the exact solution in the limit $h \to 0$, whereas the second method $(*)$ did not yield any sensible results in this limit. We will see in the following that the first method is stable whereas the second method is not.

In order to investigate stability we have to consider the limit $h \to 0$. In this limit the α_j terms in the multistep formula are important and the $h\beta_j$ terms can be neglected. An equivalent approach is to apply our multistep method to a very simple model problem if we want to investigate its stability. This model problem is given by

$$y'(t) = 0\,, \qquad y(0) = c\,, \qquad t \in [0,1]\,,$$

where c is a constant. It has the solution $y(t) = c$ and satisfies $y(1) = c$ at the end point.

Let apply the method $(*)$ to this model problem. We obtain the following difference equation

$$y_{i+1} + 4y_i - 5y_{i-1} = 0\,.$$

This is a second order difference equation with constant coefficients. It can be solved exactly by trying $y_i = Az^i$

$$Az^{i+1} + 4Az^i - 5Az^{i-1} = 0\,.$$

We divide by Az^{i-1} (we are not interested in the trivial solution $z = 0$)

$$0 = z^2 + 4z - 5 = (z+5)\,(z-1) \qquad \Longrightarrow \qquad z = -5 \qquad \text{or} \qquad z = 1\,.$$

The general solution is given by the linear combination

$$y_i = A\,(-5)^i + B\,1^i\,.$$

The value at the end point follows by choosing $t = ih = 1$ or $i = 1/h$.

If we start with the exact values $y_0 = c$ and $y_1 = c$ we obtain

$$c = A + B\,, \qquad c = A(-5) + B \qquad \Longrightarrow \qquad A = 0 \qquad \text{and} \qquad B = c\,.$$

This gives indeed the exact solution.

However, if we make a small error in one of the iteration steps, or due to round-off errors, then A will not be exactly zero. For example let us assume that the second initial value has a small error such that $y_0 = c$ and $y_1 = c + \epsilon$. Then we would have $A = -\epsilon/6$ and $B = c + \epsilon/6$. But a non-vanishing A leads to an error at the end point $t = 1$ that is exponentially increasing as h goes to zero

$$y(1) \approx A\,(-5)^{1/h} + B1^{1/h} = A\,(-1)^{1/h} \exp(\log(5)/h) + B\,.$$

The solution $y_i = A\,(-5)^i$ is known as **parasitic solution**. It is introduced by the discretisation rather than by the ODE itself. If such a mode exists, it will always be excited by errors in the calculation such as round-off errors or errors in the initial data.

Let us now apply the same analysis to the general k-step multistep method. Consider

$$y_{i+1} = \sum_{j=1}^{k} \alpha_j\, y_{i+1-j} + h \sum_{j=0}^{k} \beta_j\, f_{i+1-j} \qquad \text{where} \qquad f_i = f(t_i, y_i)\,.$$

We apply this method to the model problem $y'(t) = 0$, $y(0) = c$ and obtain

$$0 = y_{i+1} - \alpha_1 y_i - \alpha_2 y_{i-1} - \ldots - \alpha_k y_{i+1-k}\,. \qquad (a)$$

As before we obtain a linear difference equation with constant coefficients. We look for solutions of the form $y_i = z^i$ and obtain

$$0 = z^{i+1} - \alpha_1 z^i - \alpha_2 z^{i-1} - \ldots - \alpha_k z^{i+1-k}\,.$$

We are not interested in the trivial solution $z = 0$ and divide by z^{i+1-k}

$$0 = z^k - \alpha_1 z^{k-1} - \alpha_2 z^{k-2} - \ldots - \alpha_{k-1} z - \alpha_k\,.$$

The polynomial on the right-hand side is the **characteristic polynomial**. It has k roots $z_1, \ldots, z_k$. If all are distinct, then the general solution of (a) is

$$y_i = c_1 z_1^i + c_2 z_2^i + \ldots + c_k z_k^i\,.$$

(If not all are distinct, e.g. if $z_1 = z_2$, then $z_1^i(c_1 + c_2 i)$ is also a solution. The general case is discussed in the handout on linear difference equations.)

For the solution to stay bounded as $i \to \infty$ (this corresponds to $h \to 0$ for fixed time) we need the following condition: all roots of the characteristic polynomial have to satisfy $|z| \leq 1$ and any root with $|z| = 1$ has to be simple. This is the so-called **root condition**.

We derived this stability criterion (root condition) for the differential equation $y'(t) = 0$, but one can show that it applies to all well-behaved differential equations. This was shown in the 1950s by Dahlquist.

Summary: Consider a linear k-step multistep method.

$$y_{i+1} = \sum_{j=1}^{k} \alpha_j \, y_{i+1-j} + h \sum_{j=0}^{k} \beta_j \, f_{i+1-j} \, .$$

The behaviour of the method in the limit $h \to 0$ is determined by the roots of the characteristic polynomial

$$0 = z^k - \alpha_1 z^{k-1} - \alpha_2 z^{k-2} - \ldots - \alpha_{k-1} z - \alpha_k \, .$$

Theorem (root condition): A linear multistep formula is **stable** if and only if all roots of its characteristic polynomial satisfy $|z| \leq 1$, and any root with $|z| = 1$ has multiplicity one.

If the method is stable then one can specify the order of the global error of the method. This is the content of the following theorem.

Theorem (Dahlqvist): If a linear multistep formula has local truncation error $O(h^{p+1})$ and is stable, then the global error is $O(h^p)$.

If $p > 0$, the method will converge to the exact solution as $h \to 0$ and is called convergent with order of accuracy p.

From the definition of *consistent* (week 8) follows also that a method is convergent if it is consistent and stable.

Example (a): Investigate stability and convergence of the following method

$$y_{i+1} = \frac{1}{2} y_i + \frac{1}{2} y_{i-1} + 2h f_i \, .$$

We apply the method to the ODE $y' = 0$ to investigate its stability

$$y_{i+1} - \frac{1}{2} y_i - \frac{1}{2} y_{i-1} = 0 \, .$$

Setting $y_i = z^i$ leads to the characteristic polynomial

$$0 = z^2 - \frac{1}{2} z - \frac{1}{2} = (z - 1) \left(z + \frac{1}{2} \right) \, .$$

Both roots satisfy $|z| \leq 1$, and the root with $|z| = 1$ is simple. It follows from the root condition that the method is stable. Let us now investigate the order of accuracy. The local truncation error is

$$\begin{aligned}\tau_{i+1} &= y(t_i + h) - \frac{1}{2}y(t_i) - \frac{1}{2}y(t_i - h) - 2hy'(t_i) \\ &= y + hy' + \ldots - \frac{1}{2}y - \frac{1}{2}\left[y - hy' + \ldots\right] - 2hy' \\ &= -\frac{1}{2}hy' + \ldots\end{aligned}$$

The local truncation error is $O(h)$, the order of accuracy is $p = 0$. The method is not consistent and also not convergent.

Example (b): Investigate stability and convergence of the following multistep method

$$y_{i+1} = y_{i-3} + \frac{4}{3}h\,\left(f_i + f_{i-1} + f_{i-2}\right)\,.$$

Here the number of steps is $k = 4$. The characteristic polynomial is

$$z^4 - 1 = 0 \qquad \Longrightarrow \qquad z = 1\,, \quad z = -1\,, \quad z = i\,, \quad z = -i\,.$$

All roots satisfy $|z| = 1$ and are simple. Hence the method is stable. One can show that the local truncation error is $O(h^3)$. Consequently, the order of accuracy $p = 2$ and the method is convergent.

The ABk and AMk formulas: The Adams-Bashforth ($\beta_0 = 0$) and Adams-Moulton ($\beta_0 \neq 0$) formulas are given by

$$y_{i+1} = y_i + h\beta_0 f_{i+1} + h\beta_1 f_i + \ldots + h\beta_k f_{i+1-k}\,.$$

The characteristic polynomial is in all cases $z - 1 = 0$, and from the root condition follows that the methods are all stable. As mentioned before, one can show that the order of accuracy of ABk is $p = k$, and the order of accuracy of AMk is $p = k + 1$. Hence the methods are convergent (since $k \geq 1$).

The maximal order of accuracy that can be achieved is determined by the following theorem that was mentioned already in the previous week.

Theorem (first Dahlquist stability barrier): A **stable** k-step multistep formula can have order of accuracy at most

- k: if explicit.
- $k + 1$: if implicit and k is odd.
- $k + 2$: if implicit and k is even.

For this reason many of the α_j and β_j are usually set equal to zero.

3.5 Time stability (absolute stability)

We have seen in the previous section that stability is defined by the behaviour in the limit $h \to 0$. A method is stable if the approximation for the solution $y(t)$ at a fixed time t remains bounded as $h \to 0$.

Another question is how small one should choose h in practice to get reasonable results. We will see that this question is connected to a different notion of stability that is called **time stability** or **absolute stability**. It is determined by the behaviour of the approximate solution for fixed h as time $t \to \infty$.

Time stability is investigated by looking at another canonical problem that is given by

$$y'(t) = \lambda\, y(t)\,, \qquad \lambda \in \mathbb{C}\,, \quad \operatorname{Re}\lambda < 0\,, \qquad y(0) = c\,.$$

It has the exact solution $y(t) = c\,\exp(\lambda\, t)$.

If $\operatorname{Re}\lambda < 0$ then $|y(t)| \to 0$ as $t \to \infty$. A method is called time stable for fixed values of $h > 0$ and $\lambda \in \mathbb{C}$, $\operatorname{Re}\lambda < 0$, if it mimics this behaviour, i. e. if the approximate numerical solution also approaches zero as $t \to \infty$. Let us consider a simple example.

Euler's method: If we apply Euler's method to the canonical problem we obtain

$$y_{i+1} = y_i + h\lambda y_i = (1 + h\lambda)\, y_i\,, \qquad y_0 = c\,.$$

The solution to this difference equation is

$$y_i = (1 + h\lambda)^i c\,.$$

We see that $y_i \to 0$ as $i \to \infty$ if and only if $|1 + h\lambda| < 1$ or $|(-1) - h\lambda| < 1$. This condition describes a circle of radius 1 around $h\lambda = -1$ in the complex $h\lambda$-plane. This is the **time stability region** of Euler's method. For a particular value of λ it restricts the choice of h for which the method is time stable or absolutely stable.

For example, if $\lambda \in \mathbb{R}$ and $\lambda < 0$ then the above condition leads to $-1 < 1 + h\lambda < 1$ or $-2 < h\lambda < 0$, and we obtain the following condition for h

$$0 < h < \frac{-2}{\lambda} = \left|\frac{2}{\lambda}\right|\,.$$

We see that if $|\lambda|$ is very big then h has to be very small to get a reasonable result.

Backward (implicit) Euler method: It is given by

$$y_{i+1} = y_i + h f(t_{i+1}, y_{i+1})\,.$$

We apply it to the canonical problem $y' = \lambda y$, $\operatorname{Re}\lambda < 0$, $y(0) = c$ and obtain

$$y_{i+1} = y_i + h\lambda y_{i+1} \qquad \text{or} \qquad y_{i+1} = \frac{1}{1 - h\lambda} y_i\,.$$

It has the solution

$$y_i = (1 - h\lambda)^{-i} c\,.$$

It satisfies $y_i \to 0$ as $i \to \infty$ if and only if $|1 - h\lambda| > 1$. This condition denotes the exterior of a circle with radius 1 and centre $h\lambda = 1$ in the $h\lambda$-plane. Since we assumed that $\operatorname{Re}\lambda < 0$ and $h > 0$ we are only interested in the left half of the complex $h\lambda$-plane ($\operatorname{Re} h\lambda < 0$). Hence the condition $|1 - h\lambda| > 1$ does not lead to any restriction on h and the method is time stable for any value $h > 0$.

In general we seek a linear multistep method with the following properties:

- high order truncation error
- stability
- time stability region contains as much of the left half plane as possible

Let us now apply the general multistep method to our canonical problem $y' = \lambda y$, $\operatorname{Re}\lambda < 0$, $y(0) = c$. We obtain the following iteration

$$y_{i+1} = \alpha_1 y_i + \ldots + \alpha_k y_{i+1-k} + h\lambda\beta_0 y_{i+1} + h\lambda\beta_1 y_i + \ldots + h\lambda\beta_k y_{i+1-k}\,.$$

This is again a difference equation with constant coefficients, and we can find solutions of the form $y_i = z^i$. After division by z^{i+1-k} we obtain the following equation

$$(1 - h\lambda\beta_0)\, z^k - (\alpha_1 + h\lambda\beta_1)\, z^{k-1} - \ldots - (\alpha_{k-1} + h\lambda\beta_{k-1})\, z - (\alpha_k + h\lambda\beta_k) = 0\,. \qquad (*)$$

This is the **stability polynomial**. It has roots $z_1, \ldots, z_k$ that depend on the value of $h\lambda$.

Definition: A linear multistep method is **time-stable (absolutely stable)** for a given value of $h\lambda$ if all roots of the stability polynomial have modulus less than one.

The **time stability region** in the complex $h\lambda$-plane is the set of those values of $h\lambda$ for which all roots of the stability polynomial have modulus less than one.

What is new here is that everything depends on the value of $h\lambda$.

Remark: The border of the stability region can often be obtained by solving equation $(*)$ for $h\lambda$ and setting $z = \exp(i\theta)$ where $0 \le \theta < 2\pi$.

The investigation of time stability is particularly important for so-called **stiff equations**. These are differential equations for which many numerical methods require a very small step size h for them to work. It is difficult to give an exact definition of stiff equations but they often contain a term that leads to rapidly varying terms in the solution, as for example a rapidly decaying exponential term. We consider the following example.

Example: The linear IVP

$$\frac{\mathrm{d}y}{\mathrm{d}t} = -100\,(y - \cos t) - \sin t\,, \qquad y(0) = 1\,.$$

The general solution is $y(t) = \cos t + c\,\exp(-100t)$ and the initial condition $y(0) = 1$ requires $c = 0$.

We apply two methods

$$AB2: \qquad y_{i+1} = y_i + \frac{h}{2}(3f_i - f_{i-1})$$
$$BD2: \qquad y_{i+1} = \frac{4}{3}y_i - \frac{1}{3}y_{i-1} + \frac{2}{3}hf_{i+1}$$

The second method is implicit, but for a linear ODE we can easily solve it for y_{i+1}. We apply both methods to compute the value of $y(1)$. The exact solution satisfies $y(1) = \cos(1) \approx 0.5403023$.

h	AB2	BD2
0.2	14.40	0.5404
0.1	-5.7×10^4	0.54033
0.05	-1.91×10^9	0.540309
0.02	-5.77×10^{10}	0.5403034
0.01	0.5403020	0.5403026
0.005	0.5403022	0.5403024

BD2 behaves well for all h, but AB2 works only if h is small enough. Let us investigate the time stability region. The substitution $x(t) = y(t) - \cos t$ transforms the differential equation into our canonical problem $x'(t) = \lambda x(t)$ with $\lambda = -100$. We consider the two methods separately.

AB2: The stability polynomial with $\lambda = -100$ has the form $z^2 - (1 - 150h)z - 50h = 0$. This is a quadratic equation with two solutions. The method is time stable for those values of h for which these two solutions have modulus $|z| < 1$. The boundary of the stability region can be obtained by solving the equation for h and setting $z = 1$ and $z = -1$ respectively.

$$h = \frac{z^2 - z}{50 - 150z} \qquad \Longrightarrow \qquad h = \begin{cases} 0 & \text{if} \quad z = 1 \\ 0.01 & \text{if} \quad z = -1 \end{cases}$$

One can show that in the intermediate region $0.01 > h > 0$ the method is time stable, i. e. the two solutions of the quadratic equation satisfy $|z| < 1$. This agrees with the numerical results in the table above.

BD2: The stability polynomial with $\lambda = -100$ has the form $(3 + 200h)z^2 - 4z + 1 = 0$. One can show that the two solutions of this quadratic equation satisfy $|z| < 1$ for all $h > 0$. Hence the method is time stable for all $h > 0$. This agrees with the numerical results in the table above.

In general, implicit methods work better for stiff problems.

3.6 Systems of ODEs and higher order ODEs

The methods that we discussed so far can be generalised to systems of initial value problems or higher order initial value problems. We briefly discussed this for Euler's methods. In the following we will discuss it more systematically for two examples of one-step methods: for Euler's method and for the midpoint Runge-Kutta method. One-step methods can be written in the form

$$y_{i+1} = y_i + h\Phi(t_i, y_i, h)\,.$$

For Euler's method we have $\Phi(t_i, y_i, h) = f(t_i, y_i)$, and for the midpoint Runge-Kutta method $\Phi(t_i, y_i, h) = f(t_i + h/2, y_i + hK/2)$ where $K = f(t_i, y_i)$.

Consider a general system of m first order ODEs for $t \in [a, b]$.

$$\begin{aligned} \frac{\mathrm{d}u_1}{\mathrm{d}t} &= f_1(t, u_1, u_2, \ldots, u_m)\,, & u_1(a) &= \alpha_1\,, \\ &\vdots \\ \frac{\mathrm{d}u_m}{\mathrm{d}t} &= f_m(t, u_1, u_2, \ldots, u_m)\,, & u_m(a) &= \alpha_m\,. \end{aligned}$$

As in the case of one equation we discretise time by setting $t_i = a + ih$ where h is the step size, and we denote the approximations to $u_j(t_i)$ by $u_{j,i}$. The application of Euler's method to the m equations then yields

$$\begin{aligned} u_{1,i+1} &= u_{1,i} + hK_1\,, & K_1 &= f_1(t_i, u_{1,i}, u_{2,i}, \ldots, u_{m,i})\,, & u_{1,0} &= \alpha_1\,, \\ &\vdots \\ u_{m,i+1} &= u_{m,i} + hK_m\,, & K_m &= f_m(t_i, u_{1,i}, u_{2,i}, \ldots, u_{m,i})\,, & u_{m,0} &= \alpha_m\,. \end{aligned}$$

In the case of the midpoint Runge-Kutta method we have

$$\begin{aligned} u_{1,i+1} &= u_{1,i} + hf_1\left(t_i + \frac{h}{2}, u_{1,i} + h\frac{K_1}{2}, u_{2,i} + h\frac{K_2}{2}, \ldots, u_{m,i} + h\frac{K_m}{2}\right)\,, & u_{1,0} &= \alpha_1\,, \\ &\vdots \\ u_{m,i+1} &= u_{m,i} + hf_m\left(t_i + \frac{h}{2}, u_{1,i} + h\frac{K_1}{2}, u_{2,i} + h\frac{K_2}{2}, \ldots, u_{m,i} + h\frac{K_m}{2}\right)\,, & u_{m,0} &= \alpha_m\,. \end{aligned}$$

In both cases the systems can be solved numerically by iteration, starting from the initial values.

In a similar way one can treat a general m-th order initial value problem. Consider

$$y^{(m)}(t) = f(t, y, y^{(1)}, \ldots, y^{(m-1)})\,, \qquad t \in [a, b]\,,$$

with initial conditions

$$y(a) = \alpha_1\,, \qquad y'(a) = \alpha_2\,, \qquad y''(a) = \alpha_3\,, \qquad \ldots \qquad y^{(m-1)}(a) = \alpha_m\,.$$

As was discussed at the beginning of chapter 3, one can convert this equation into a system of first-order equations by setting $u_j(t) = y^{(j-1)}(t)$ for $j = 1, \ldots, m$. Then we obtain the system

$$\frac{\mathrm{d}u_1}{\mathrm{d}t} = u_2\,, \qquad \frac{\mathrm{d}u_2}{\mathrm{d}t} = u_3\,, \qquad \ldots \qquad \frac{\mathrm{d}u_{m-1}}{\mathrm{d}t} = u_m\,, \qquad \frac{\mathrm{d}u_m}{\mathrm{d}t} = f(t, u_1, u_2, \ldots, u_m)\,,$$

with initial conditions $u_1(a) = \alpha_1\,, \ldots, u_m(a) = \alpha_m$, and we can continue with the methods for systems that we discussed above.

4 Ordinary differential equations: boundary value problems (BVPs)

We consider problems of the type

$$y'' = f(x, y, y')\,, \qquad x \in [a, b]\,, \qquad y(a) = \alpha\,, \qquad y(b) = \beta\,.$$

Instead of two conditions at the initial point $x = a$ we now have one condition at either end point. New techniques are required for solving such problems. These types of equations often occur in physical problems, for example for describing a loaded beam, a hanging chain or a vibrating string. These physical problems are often position dependent rather than time dependent, and for this reason we denote the independent variable in this chapter by x instead of t.

The problem with this type of BVPs is that the methods of chapter 3 require two initial conditions to start the numerical iteration process for second order ODEs, whereas the BVP provides only one initial condition at $x = a$. One would like to find the second initial condition for which the solution will arrive at the correct end point $y(b) = \beta$.

4.1 The linear shooting method

We first consider the linear problem

$$y'' = f(x, y, y') = p(x)y' + q(x)y + r(x)\,, \qquad y(a) = \alpha\,, \qquad y(b) = \beta\,. \qquad (*)$$

This problem does not always have a unique solution.

Example: Consider

$$y'' + y = 0\,, \qquad x \in [a, b]\,, \qquad y(0) = 0\,, \qquad y(b) = \beta\,.$$

The general solution is

$$y(x) = A \sin x + B \cos x\,,$$

and the condition $y(0) = 0$ requires $B = 0$ and $y(x) = A \sin x$. The solution depends now on the second boundary condition. We consider three cases.

- $y(\pi/2) = 0$: In this case $A = 0$ and we have the unique solution $y(x) = 0$.
- $y(\pi) = 0$: In this case A is arbitrary and we have infinitely many solutions $y(x) = A \sin x$.
- $y(\pi) = 1$: In this case we obtain the condition $1 = A \sin \pi$ which has no solution.

Theorem (existence and uniqueness): If the problem $(*)$ satisfies

i) $p(x)$, $q(x)$ and $r(x)$ are continuous on $[a,b]$,

ii) $q(x) > 0$ on $[a,b]$,

then the problem has a unique solution.

In the following we assume that the assumptions of this theorem hold. We can solve the linear BVP $(*)$ by transforming it into two IVPs as will be shown in the following. These two IVPs are given by

$$\begin{aligned} y_1'' &= p(x)y_1' + q(x)y_1 + r(x)\,, \qquad x \in [a,b]\,, \qquad y_1(a) = \alpha\,, \qquad y_1'(a) = 0\,, \\ y_2'' &= p(x)y_2' + q(x)y_2\,, \qquad\quad x \in [a,b]\,, \qquad y_2(a) = 0\,, \qquad y_2'(a) = 1\,. \end{aligned}$$

Now consider the linear combination $y(x) = y_1(x) + Cy_2(x)$. Because of the linearity of the ODE it satisfies the following equation

$$y'' = y_1'' + Cy_2'' = p(x)\,(y_1' + Cy_2') + q(x)\,(y_1 + Cy_2) + r(x) = p(x)y' + q(x)y + r(x)\,,$$

and the initial condition

$$y(a) = y_1(a) + Cy_2(a) = \alpha\,.$$

Hence it satisfies the correct ODE and the correct first boundary condition. It remains to choose the constant C such that the second boundary condition is satisfied

$$\beta = y(b) = y_1(b) + Cy_2(b)\,, \qquad \Longrightarrow \qquad C = \frac{\beta - y_1(b)}{y_2(b)}\,,$$

and in this way we obtain the solution to the boundary value problem $(*)$.

Remark: Note that the constant C can always be found because $y_2(b) \neq 0$. This follows from the fact that the BVP for $y_2(x)$ with $y_2(a) = y_2(b) = 0$ would have the unique solution $y_2(x) = 0$ (we assumed that the assumptions of the existence and uniqueness theorem are satisfied). This would be in contradiction to $y_2'(a) = 1$. Hence we can conclude conversely that the IVP for $y_2(x)$ with $y_2(a) = 0$ and $y_2'(a) = 1$ cannot satisfy $y_2(b) = 0$.

4.2 Shooting method for nonlinear problems

Consider now the general second-order BVP

$$y'' = f(x, y, y')\,, \qquad x \in [a, b]\,, \qquad y(a) = \alpha\,, \qquad y(b) = \beta\,.$$

To solve it we consider the corresponding IVP

$$y'' = f(x, y, y')\,, \qquad x \in [a, b]\,, \qquad y(a) = \alpha\,, \qquad y'(a) = \lambda\,,$$

which depends on the parameter λ. We write its solution in the form $y = y(x, \lambda)$ to make its dependence on the parameter λ explicit. The idea behind the shooting method is to adjust the parameter λ until $y(b, \lambda) = \beta$, and then we have a solution to the original BVP.

This is a root finding problem for the function $g(\lambda) = y(b, \lambda) - \beta$. For a given value of λ this function is evaluated by solving the IVP for $y(x, \lambda)$ numerically. We can apply the methods of chapter 1 for solving the root finding problem for $g(\lambda)$. In the following we'll discuss the secant method and Newton's method.

Secant method: The secant method needs two initial guesses λ_0 and λ_1. It then proceeds with the iteration (see chapter 1)

$$\lambda_k = \lambda_{k-1} - g(\lambda_{k-1}) \frac{\lambda_{k-1} - \lambda_{k-2}}{g(\lambda_{k-1}) - g(\lambda_{k-2})}\,.$$

Newton's method: Newton's method is more powerful. It needs one initial guess λ_0 and continues with the iteration

$$\lambda_k = \lambda_{k-1} - \frac{g(\lambda_{k-1})}{\frac{\mathrm{d}g}{\mathrm{d}\lambda}(\lambda_{k-1})}\,,$$

and with $g(\lambda) = y(b, \lambda) - \beta$ we obtain

$$\lambda_k = \lambda_{k-1} - \frac{y(b, \lambda_{k-1}) - \beta}{\frac{\mathrm{d}y}{\mathrm{d}\lambda}(b, \lambda_{k-1})}\,,$$

The problem with this method is that we need to know $\frac{\mathrm{d}y}{\mathrm{d}\lambda}(b, \lambda)$.

The solution to this problem consists in solving another IVP for $\frac{\mathrm{d}y}{\mathrm{d}\lambda}(x, \lambda) = z(x, \lambda)$. To see this let us go back to the previous IVP

$$y''(x, \lambda) = f(x, y(x, \lambda), y'(x, \lambda))\,, \qquad x \in [a, b]\,, \qquad y(a, \lambda) = \alpha\,, \qquad y'(a, \lambda) = \lambda\,,$$

where the prime denotes differentiation with respect to x, and where we have written the dependence on λ explicitly. Let us differentiate this ODE and the boundary conditions with respect to λ. By using the chain rule we obtain

$$\frac{\partial y''}{\partial \lambda} = \frac{\partial f}{\partial \lambda} = \frac{\partial f}{\partial y}\frac{\partial y}{\partial \lambda} + \frac{\partial f}{\partial y'}\frac{\partial y'}{\partial \lambda}\,, \qquad x \in [a, b]\,, \qquad \frac{\partial y}{\partial \lambda}(a, \lambda) = 0\,, \qquad \frac{\partial y'}{\partial \lambda}(a, \lambda) = 1\,.$$

Now let $z(x,\lambda) = \frac{\mathrm{d}y}{\mathrm{d}\lambda}(b,\lambda)$ and assume that the differentiations with respect to x and λ can be interchanged. Then

$$z'' = \frac{\partial f}{\partial \lambda} = \frac{\partial f}{\partial y} z + \frac{\partial f}{\partial y'} z' \,, \qquad x \in [a,b] \,, \qquad z(a,\lambda) = 0 \,, \qquad z'(a,\lambda) = 1 \,.$$

This IVP for $z(x,\lambda)$ can be solved numerically alongside the original IVP, and then Newton's method gives the next iteration for the value of the parameter λ

$$\lambda_k = \lambda_{k-1} - \frac{y(b,\lambda_{k-1}) - \beta}{z(b,\lambda_{k-1})} \,,$$

Newton's method is generally faster than the secant method.

Example: Consider the BVP

$$y'' = \frac{1}{8}(32 + 2x^3 - yy') \,, \qquad x \in [1,3] \,, \qquad y(1) = 17 \,, \qquad y(3) = \frac{43}{3} \,.$$

The exact solution is

$$y(x) = x^2 + \frac{16}{x} \,.$$

Applying the shooting method in combination with Newton's method requires to solve at each step of the iteration the two IVPs

$$y'' = \frac{1}{8}(32 + 2x^3 - yy') \,, \qquad x \in [1,3] \,, \qquad y(1) = 17 \,, \qquad y'(1) = \lambda \,,$$

and

$$z'' = \frac{\partial f}{\partial y} z + \frac{\partial f}{\partial y'} z' = -\frac{1}{8}(y'z + yz') \,, \qquad x \in [1,3] \,, \qquad z(1,\lambda) = 0 \,, \qquad z'(1,\lambda) = 1 \,.$$

Using $h = 0.1$ and the Runge-Kutta method RK4 with starting value

$$\lambda_0 = \frac{\beta - \alpha}{b - a} = \frac{1}{2}\left(\frac{43}{3} - \frac{51}{3}\right) = -\frac{4}{3} \,,$$

yields after 4 iterations $\lambda_4 = -14.000203$. This is close to the actual value $\lambda = -14$.

4.3 Finite difference methods for linear problems

We consider again the linear problem

$$y'' = f(x, y, y') = p(x)y' + q(x)y + r(x)\,, \qquad y(a) = \alpha\,, \qquad y(b) = \beta\,.$$

The idea of the finite difference methods is to replace the derivatives by finite difference approximations, as we did in chapter 2. This results in a linear system of equations that can be solved. The finite difference methods have in general a better stability than the shooting methods, but they require more work for the same accuracy. We start with a simple example.

Example: Consider the BVP

$$y'' = 0\,, \qquad x \in [0, 6]\,, \qquad y(0) = -1\,, \qquad y(6) = 5\,.$$

It has the general solution $y(x) = Ax + B$ and from the boundary conditions follows $B = -1$ and $A = 1$. The exact solution follows as $y(x) = x - 1$.

We now apply the finite difference method to this BVP. The interval $[0, 6]$ is divided into n equal pieces. The mesh points are $x_i = i\,h$, $i = 0, 1, \ldots, n$ where $h = 6/n$. We then approximate the second derivative y'' at each interior mesh point by using the central difference approximation from chapter 2.

$$y''(x_i) = \frac{y(x_i + h) - 2y(x_i) + y(x_i - h)}{h^2} - \frac{h^2}{12}y^{(4)}(\xi_i),$$

where $x_i - h < \xi_i < x_i + h$. Let y_i denote the approximation to $y(x_i)$. We insert the approximation for the second derivative into the ODE $y'' = 0$, neglect the error term, and obtain

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} = 0,\ 1 \le i \le n - 1, \qquad y_0 = -1, \quad y_n = 5.$$

Inserting y_0 and y_n into the first and last of these equations, respectively, leads to

$$\begin{aligned} y_2 - 2y_1 &= -y_0 = 1, \\ y_{i+1} - 2y_i + y_{i-1} &= 0, \qquad 1 < i < n - 1, \\ -2y_{n-1} + y_{n-2} &= -y_n = -5. \end{aligned}$$

These are $(n-1)$ equations for the $(n-1)$ unknowns $y_1, \ldots, y_{n-1}$. We can write these equations in matrix form and solve by Gaussian elimination. Take $n = 6$.

$$\begin{aligned} y_2 - 2y_1 &= 1, \\ y_3 - 2y_2 + y_1 &= 0, \\ y_4 - 2y_3 + y_2 &= 0, \\ y_5 - 2y_4 + y_3 &= 0, \\ -2y_5 + y_4 &= -5. \end{aligned}$$

In matrix form we obtain

$$\begin{bmatrix} -2 & 1 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & 0 \\ 0 & 1 & -2 & 1 & 0 \\ 0 & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 1 & -2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ -5 \end{bmatrix}$$

The solutions can be found by Gaussian elimination and are given by

$$y_0 = -1\,, \qquad y_1 = 0\,, \qquad y_2 = 1\,, \qquad y_3 = 2\,, \qquad y_4 = 3\,, \qquad y_5 = 4\,, \qquad y_6 = 5\,.$$

They have the form $y_i = i - 1$, and we can easily check that they solve the difference equation

$$y_{i+1} - 2y_i + y_{i-1} = i - 2(i-1) + i - 2 = 0\,.$$

They also agree with the exact solution ($h = 1$)

$$y_i = y(x_i) = y(i) = i - 1\,.$$

One can show that the solution is exact also for other values of n.

Back to the general problem: Let us now consider the general linear problem.

$$y'' = f(x, y, y') = p(x)y' + q(x)y + r(x)\,, \qquad y(a) = \alpha\,, \qquad y(b) = \beta\,.$$

We set $x_i = a + ih$, $i = 0, 1, \ldots, n$ where $h = (b-a)/n$. We now approximate the first and second derivatives of y by using the central difference approximations of chapter 2.

$$\begin{aligned} y'(x_i) &= \frac{y(x_i + h) - y(x_i - h)}{2h} + O(h^2) \\ y''(x_i) &= \frac{y(x_i + h) - 2y(x_i) + y(x_i - h)}{h^2} + O(h^2) \end{aligned}$$

We insert these approximations into the ODE. We denote the approximation to $y(x_i)$ by y_i, and we set $p(x_i) = p_i$, $q(x_i) = q_i$ and $r(x_i) = r_i$. This leads to

$$\frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} = p_i \frac{y_{i+1} - y_{i-1}}{2h} + q_i y_i + r_i\,, \qquad 1 \le i \le n-1\,, \qquad y_0 = \alpha\,, \qquad y_n = \beta\,.$$

We insert the boundary values $y_0 = \alpha$ and $y_n = \beta$ into the first and last of these equations and obtain

$$\begin{aligned} y_2\left(1 - \frac{h}{2}p_1\right) - y_1\left(2 + h^2 q_1\right) &= h^2 r_1 - \alpha\left(1 + \frac{h}{2}p_1\right) \\ y_{i+1}\left(1 - \frac{h}{2}p_i\right) - y_i\left(2 + h^2 q_i\right) + y_{i-1}\left(1 + \frac{h}{2}p_i\right) &= h^2 r_i \qquad\qquad 1 < i < n-1 \\ -y_{n-1}\left(2 + h^2 q_{n-1}\right) + y_{n-2}\left(1 + \frac{h}{2}p_{n-1}\right) &= h^2 r_{n-1} - \beta\left(1 - \frac{h}{2}p_{n-1}\right) \end{aligned}$$

This can be written more compactly with the abbreviations

$$a_i = -2 - h^2 q_i\,, \qquad b_i = 1 + \frac{h}{2} p_i\,, \qquad c_i = 1 - \frac{h}{2} p_i\,.$$

We then obtain

$$\begin{aligned} y_2 c_1 + y_1 a_1 &= h^2 r_1 - \alpha b_1 \\ y_{i+1} c_i + y_i a_i + y_{i-1} b_i &= h^2 r_i \qquad\qquad 1 < i < n-1 \\ y_{n-1} a_{n-1} + y_{n-2} b_{n-1} &= h^2 r_{n-1} - \beta c_{n-1} \end{aligned}$$

This can be written in matrix form

$$\begin{bmatrix} a_1 & c_1 & 0 & \cdots & \cdots & 0 \\ b_2 & a_2 & c_2 & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & \ddots & b_{n-2} & a_{n-2} & c_{n-2} \\ 0 & \cdots & \cdots & 0 & b_{n-1} & a_{n-1} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ y_{n-2} \\ y_{n-1} \end{bmatrix} = \begin{bmatrix} h^2 r_1 - \alpha b_1 \\ h^2 r_2 \\ \vdots \\ \vdots \\ h^2 r_{n-2} \\ h^2 r_{n-1} - \beta c_{n-1} \end{bmatrix}$$

This is a tridiagonal system that can be solved by Gaussian elimination.

Remark: The local truncation error of the method is $O(h^2)$. This is not as good as the shooting method with RK4. For this reason, one usually chooses a smaller step size, but there is a limit because of the instability of finite difference approximations for derivatives. In order to improve the approximation one can use Richardson extrapolation for the finite difference approximations.

4.4 Spectral methods for linear problems

We consider again the linear boundary value problem

$$y''(x) = p(x)\, y'(x) + q(x)\, y(x) + r(x), \quad x \in [a,b], \quad y(a) = \alpha, \quad y(b) = \beta.$$

We can write the ODE in a compact form by introducing the differential operator

$$L = \frac{\mathrm{d}^2}{\mathrm{d}x^2} - p(x)\,\frac{\mathrm{d}}{\mathrm{d}x} - q(x).$$

Then the ODE becomes

$$Ly = r.$$

The idea behind the spectral methods in this section is to expand the function $y(x)$ in the form

$$y(x) = \sum_{n=1}^{\infty} c_n \phi_n(x), \tag{1}$$

where $\{\phi_n(x), n = 1, 2, \ldots\}$ is a complete set of functions over the interval $[a,b]$. For example, one might expand $y(x)$ in a Fourier series on the interval $[a,b]$, but the functions $\phi_n(x)$ can be

chosen much more generally. It is convenient to require that the functions $\phi_n(x)$ are orthogonal on the interval $[a, b]$ with respect to some weight function $w(x)$:

$$\langle \phi_m, \phi_n \rangle := \int_a^b w(x)\, \phi_m(x)\, \phi_n(x)\, \mathrm{d}x = 0 \qquad \text{if} \qquad m \neq n\,,$$

where the above integral can be considered as an inner product of the functions ϕ_m and ϕ_n that is denoted by $\langle \phi_m, \phi_n \rangle$.

Now we introduce an approximation that consists of a truncation of the expansion (1) at finite $n = N$, and we substitute the truncated expansion into the ODE $Ly = r$. This results in

$$L\left(\sum_{n=1}^{N} c_n \phi_n(x)\right) = \sum_{n=1}^{N} c_n L\,\phi_n(x) = r(x)\,. \tag{2}$$

We need N conditions to determine the N coefficients c_n. Two are provided by the two boundary conditions

$$\sum_{n=1}^{N} c_n \phi_n(a) = \alpha\,, \qquad \text{and} \qquad \sum_{n=1}^{N} c_n \phi_n(b) = \beta\,.$$

For the remaining $(N-2)$ conditions we take the inner product of both sides of equation (2) with $\phi_m(x)$ where $m = 1, \ldots, N-2$.

$$\sum_{n=1}^{N} c_n \langle \phi_m, L\,\phi_n \rangle = \langle \phi_m, r \rangle\,, \qquad m = 1, \ldots, N-2\,.$$

In this way we obtain a linear algebra problem consisting of N equations for N unknowns. It can be written in matrix/vector form:

$$\begin{bmatrix} \langle \phi_1, L\,\phi_1 \rangle & \langle \phi_1, L\,\phi_2 \rangle & \ldots & \langle \phi_1, L\,\phi_N \rangle \\ \langle \phi_2, L\,\phi_1 \rangle & \langle \phi_2, L\,\phi_2 \rangle & \ldots & \langle \phi_2, L\,\phi_N \rangle \\ \vdots & \vdots & & \vdots \\ \langle \phi_{N-2}, L\,\phi_1 \rangle & \langle \phi_{N-2}, L\,\phi_2 \rangle & \ldots & \langle \phi_{N-2}, L\,\phi_N \rangle \\ \phi_1(a) & \phi_2(a) & \ldots & \phi_N(a) \\ \phi_1(b) & \phi_2(b) & \ldots & \phi_N(b) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{N-2} \\ c_{N-1} \\ c_N \end{bmatrix} = \begin{bmatrix} \langle \phi_1, r \rangle \\ \langle \phi_2, r \rangle \\ \vdots \\ \langle \phi_{N-2}, r \rangle \\ \alpha \\ \beta \end{bmatrix}$$

This linear system of equations can be solved with Gaussian elimination.

There are two main advantages of this method

(A) It delivers a global representation of the solution $y(x)$ (see equation (1)) that can be evaluated anywhere, not just at the mesh points.

(B) One can show that the method has a fast (exponential) convergence to the exact solution as N increases.

Example: Let us choose Chebyshev polynomials of the first kind as our basis functions, $\phi_n(x) = T_{n-1}(x)$, $n = 1, 2, 3, \ldots$. We know that they are defined on the interval $[-1, 1]$ with weight function $w(x) = 1/\sqrt{1-x^2}$. If the interval $[a, b]$ does not coincide with the interval $[-1, 1]$ then we can perform a linear variable change of the form $x' = 1 + 2(x-b)/(b-a)$ to map it onto the interval $[-1, 1]$. The Chebyshev polynomials are explicitly given by

$$T_n(x) = \cos(n \arccos x)\,.$$

They satisfy the relation

$$\langle T_m, T_n \rangle = \begin{cases} 0 & m \neq n \\ \pi/2 & m = n \neq 0 \\ \pi & m = n = 0 \end{cases}$$

The first few polynomials have the form

$$T_0(x) = 1, \qquad T_1(x) = x, \qquad T_2(x) = 2x^2 - 1, \qquad T_3(x) = 4x^3 - 3x.$$

At the end points they take the values $T_n(1) = \cos(n \cdot 0) = 1$ and $T_n(-1) = \cos(n\,\pi) = (-1)^n$.

Let us apply these orthogonal polynomials to the following simple boundary value problem

$$y'' = 0, \qquad y(-1) = \alpha, \qquad y(+1) = \beta.$$

The ODE corresponds to $L\,y = 0$ with the differential operator $L = \mathrm{d}^2/\mathrm{d}x^2$. We choose $N = 4$ and substitute the expansion

$$y(x) \approx \sum_{n=1}^{4} c_n\, T_{n-1}(x)$$

into the ODE with the result

$$\sum_{n=1}^{4} c_n\, T''_{n-1}(x) = 0.$$

Taking the inner product of both sides of this equation with T_m for $m = 0$ and $m = 1$ we obtain two equations for the coefficients c_n

$$\sum_{n=1}^{4} c_n\, \langle T_m, T''_{n-1} \rangle = 0, \qquad m = 0, 1.$$

The remaining two equations are obtained from the boundary conditions

$$\sum_{n=1}^{4} c_n\, T_{n-1}(-1) = \alpha, \qquad \sum_{n=1}^{4} c_n\, T_{n-1}(+1) = \beta.$$

Altogether, we obtain the linear system

$$\begin{bmatrix} \langle T_0, T''_0 \rangle & \langle T_0, T''_1 \rangle & \langle T_0, T''_2 \rangle & \langle T_0, T''_3 \rangle \\ \langle T_1, T''_0 \rangle & \langle T_1, T''_1 \rangle & \langle T_1, T''_2 \rangle & \langle T_1, T''_3 \rangle \\ T_0(-1) & T_1(-1) & T_2(-1) & T_3(-1) \\ T_0(+1) & T_1(+1) & T_2(+1) & T_3(+1) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \alpha \\ \beta \end{bmatrix}$$

Note that $T''_0 \equiv 0$ and $T''_1 \equiv 0$. We further have $\langle T_0, T''_3 \rangle = 0$ and $\langle T_1, T''_2 \rangle = 0$ because these are inner products between even and odd functions, and they hence vanish. Finally

$$\begin{aligned} T''_2(x) = 4 = 4\,T_0(x) \qquad &\Longrightarrow \qquad \langle T_0, T''_2 \rangle = 4\,\pi, \\ T''_3(x) = 24\,x = 24\,T_1(x) \qquad &\Longrightarrow \qquad \langle T_1, T''_3 \rangle = 12\,\pi. \end{aligned}$$

All in all, we obtain the following system

$$\begin{bmatrix} 0 & 0 & 4\pi & 0 \\ 0 & 0 & 0 & 12\pi \\ 1 & -1 & 1 & -1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \alpha \\ \beta \end{bmatrix}$$

The solutions of this system of linear equations are

$$c_1 = \frac{\alpha + \beta}{2}, \qquad c_2 = \frac{\beta - \alpha}{2}, \qquad c_3 = c_4 = 0.$$

The resulting function

$$y(x) = \frac{\alpha + \beta}{2} + \frac{\beta - \alpha}{2}\, x$$

agrees with the exact solution.