Introduction to Numerical Mathematics.

Mathias Sawall

Institut für Mathematik, Universität Rostock

WS 2025/2026

Lectures:

- Tue 11.00-13.00, room 11, Albert-Einstein-Str. 2,
- Thu 9.00-11.00, room 17, Albert-Einstein-Str. 2,
- volume of 56 hours lecture and 28h tutorial,
- slides on the web page https://www.numerik.mathematik.uni-rostock.de/sawall/

Tutorials:

- by Jiss Mariam Babu,
- Mon 13.00 - 15.00, room 111, Albert-Einstein-Str. 2.

Contact:

- mathias.sawall@uni-rostock.de,
- room 431, Ulmenstraße 69, Haus 3.

Structure of the module

Exam:

- final examination of 120 min,
- allowed are 7 leaves DIN-A4, hand written on both sides, simple pocket calculator
- simple pocket calculator: no graphic, no programming, no matrix- and vector calculus, no solution of linear systems of equations, no numerical differentiation, no numerical integration.

Exercises:

- available on the web page,
- e. g. print them and think about them at home for your own,
- the tasks are discussed in the tutorials.

Web:

- https://www.numerik.mathematik.uni-rostock.de/sawall/
- Tutorials, slides and more.

Table of contents

Outline

Outline

Machine numbers:

- Storage/representation of numbers on a computer,
- computer do not work with real numbers (for example $\sqrt{2}$), but with a finite subset: *floating point numbers*,
- for example $1/3$ cannot be represented without an error in the binary system.

Normalized floating point:

- representation of a number $d \neq 0$ on a digital computer using base p,
- normalized floating point for a mantissa of length l reads

$$d = \pm 0\,.\underbrace{d_1\, d_2\, d_3 \ldots d_l}_{\text{mantissa}} \cdot p^e, \qquad 0 \leq d_i < p,\ d_1 \neq 0$$

with exponent $e \in \mathbb{Z}$, $-m \leq e \leq M$.

Definition 1.1

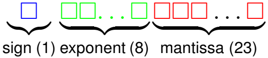
The set of machine numbers $\mathbb{F}(p, l, m)$ contains all numbers in normalized floating point representation for the base p, a mantissa of length l and an exponent of length m.

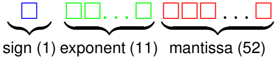
Remarks:

1. The set $\mathbb{F}(p, l, m)$ is an finite subset of $\mathbb{Q}$.
2. Machine numbers are not equidistant. The distance between machine numbers are related to their values. There is a „gap" around 0.
3. For $p = 2$ the first digit in normalized floating point representation is always 1. Often this leading 1 is omitted („hidden bit") in the number representation on a computer. For $x = 0$, a special representation is needed.

<u>Standardization of normalized floating point numbers</u>:

- IEEE arithmetic (Institute of Electrical and Electronic Engineers, 1985),
- dual system $p = 2$ using 32bit (single) or 64bit (double)





- representable numbers (depending on the length of the exponent)

$$\text{32bit}: \; 1.2 \cdot 10^{-38} \leq |x| \leq 3.4 \cdot 10^{38}, \qquad \text{64bit}: \; 2.2 \cdot 10^{-308} \leq |x| \leq 1.8 \cdot 10^{308},$$

- rounding erors (depending on the length of the mantissa)

$$\text{32bit}: \; 2^{-23} \approx 1.19 \cdot 10^{-7}, \qquad \text{64bit}: \; 2^{-52} \approx 2.22 \cdot 10^{-16},$$

- further errors appear if the numbers are converted from decimal (classical numbers) to binary system (storage).

Outline

Converting a number to IEEE:

- as a rule, a given $x \in \mathbb{R}$ is not a member of the set $\mathbb{F}(p,l,m)$,
- rounding operater (to convert an $x \notin \mathbb{F}$)

$$\mathrm{rd}: \ \mathbb{R} \to \mathbb{F}(p,l,m) \qquad \text{with the property} \qquad |x - \mathrm{rd}(x)| = \min_{f \in \mathbb{F}(p,l,m)} |x - f|,$$

- for

$$x = \pm p^b \sum_{k=-\infty}^{-1} \alpha_k p^k, \qquad \alpha_1 \neq 0$$

is

$$\mathrm{rd}(x) = \begin{cases} \pm(\sum_{k=-l}^{-1} \alpha_k p^k) p^b & \text{if } \alpha_{-l-1} < p/2 \\ \pm(\sum_{k=-l}^{-1} \alpha_k p^k + p^{-l}) p^b & \text{if } \alpha_{-l-1} \geq p/2 \end{cases},$$

- if $|x|$ is smaller than the smallest number in the system, then $\mathrm{rd}(x) = 0$,
- if $|x|$ is larger than the largest number in the system, then $\mathrm{rd}(x) = \pm\mathrm{Inf}$,
- for the absolute and relative rounding errors in the binary system holds

$$|x - \mathrm{rd}(x)| \le \frac{p^{-l}}{2} p^e, \qquad \frac{|x - \mathrm{rd}(x)|}{|x|} \le \frac{p}{2} p^{-l}.$$

Rounding errors:

- occur while reading a number into a computer, converting a number from one number system to another, computing $+$, $-$, $*$, $/$,
- different numerical results for math. equivalent expressions in pseudoarithmetic,
- select suitable algorithms to reduce the lack of precision.

Machine epsilon

Definition 1.2

We name the number

$$\text{eps} = \frac{p}{2} p^{-l} \qquad \text{(or macheps)}$$

Machine epsilon or roundoff unit. For IEEE arithmetic with 64 bit holds

$$\text{eps} = \frac{2}{2} 2^{-52} \approx 2.22 \cdot 10^{-16}.$$

Remark:

- the machine epsilon is the number with the smallest absolute value that can still be added to 1 so without getting 1,
- for all x with $|x| < \text{eps}$ holds $1 \oplus x = 1$.

Computation of the machine epsilon:

```
x = 1;
eps = 1;
while x+eps>1
   eps = eps/2;
end
2*eps
```

Outline

Error analysis

Errors during a calculation:

- each calculation step includes a (small) generated error and a propagated error

$$\delta_y = \underbrace{\varepsilon_f \cdot f(\tilde{x})}_{\text{generated error}} + \underbrace{f(\tilde{x}) - f(x)}_{\text{propagated error}},$$

- e. g. the relative error for $(x + \Delta x) \pm (y + \Delta y) = x \pm y + (\Delta x \pm \Delta y)$ is

$$\frac{\Delta x \pm \Delta y}{x \pm y} = \frac{x}{x \pm y} \frac{\Delta x}{x} \pm \frac{y}{y \pm y} \frac{\Delta y}{y},$$

- the relative error for $x \pm y$ increases if $|x \pm y| \approx 0$ (catastrophic cancellation),
- try to design algorithms avoiding catastrophic cancellation,
- backward and forward error analysis.

Definition 1.3 (Condition numbers)

The absolute condition number of the problem of calculating $y = f(x)$ *is the multiplying factor of the absolute initial error*

$$\operatorname{acond}(f) := f'(\tilde{x}).$$

The relative condition number of the problem to calculate $y = f(x)$ *is the multiplying factor of the relative initial error*

$$\operatorname{cond}(f) := \frac{\tilde{x} \cdot f'(\tilde{x})}{f(\tilde{x})}.$$

Approximation of the relative error of a function value y

$$\varepsilon_y := \frac{\delta_y}{f(\tilde{x})} \approx \varepsilon_f + \underbrace{\frac{\delta_x}{\tilde{x}}}_{\varepsilon_{\tilde{x}} \approx \varepsilon_x} \cdot \underbrace{\frac{\tilde{x} f'(\tilde{x})}{f(\tilde{x})}}_{\operatorname{cond}(x)} = \varepsilon_f + \varepsilon_x \cdot \operatorname{cond}(f).$$

Outline

1. Machine computing

2. Nonlinear equations & optimization

3. Interpolation

4. Numerical integration

5. Ordinary differential equations

6. Systems of linear equations

7. Least squares problems

8. Numerical approximation of eigenvalues

9. Literature

Problem:

- given function

$$f : \mathbb{R}^n \to \mathbb{R}^n,$$

- determine the zeros of $f(x)$, this means x^* with $f(x^*) = 0$, whereas

$$f(x) = 0 \quad \Leftrightarrow \quad \begin{pmatrix} f_1(x) \\ \vdots \\ f_n(x) \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Possible solutions:

- there exists no solution, for example $x^2 + 1$,
- there exists a finite number of solutions, for example $x^2 - 1$,
- there exists an infinite number of solutions, for example $x^2 \sin(\frac{1}{x})$.

Outline

Fixed-point iteration

Definition 2.1

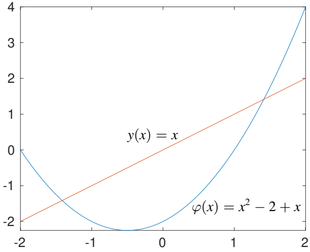
Let $\varphi : I \to I$ with $I \subset \mathbb{R}^n$ be a function mapping the set I into itself. An $x^* \in I$ is called a fixpoint of φ if

$$\varphi(x^*) = x^*.(*)$$

Furthermore, $(*)$ is called a fixpoint form.

Geometric meaning:

- fixed points are the intersections of $\varphi(x)$ with the straight line $y = x$.



Definition 2.2

A mapping $\varphi : I \to I$ with $I \subset \mathbb{R}^n$ is called contraction if there exists a constant $L \in [0, 1)$ such that for all $x, y \in I$

$$\|\varphi(x) - \varphi(y)\| \leq L\|x - y\|.(*)$$

Remarks:

- if $L < 1$, then images of two points are always closer to each other than the originals (contraction),
- for differentiable $f : \mathbb{R} \to \mathbb{R}$ with $I \subset \mathbb{R}$, the map f is a contraction, if

$$\max_{x \in I} |f'(x)| \leq L < 1.$$

- in general, i. e. without the restriction $L < 1$, one calls $(*)$ a Lipschitz constant and L a Lipschitz constant.

Banach fixed-point theorem

Theorem 2.3 (Banach fixed-point theorem)

Let I be a closed subset of $\mathbb{R}^n$ and $\varphi : I \to I$ be a self-mapping, i.e., it holds that $\varphi(I) \subset I$. Furthermore, let φ on I be a contraction.
Then φ has exactly one fixed-point $x^* \in I$ and the sequence $\{x_n\}_{n=0,1,2,\dots}$ generated by the fixed-point iteration

$$x_{n+1} = \varphi(x_n)$$

converges for each starting iteration $x_0 \in I$ to this fixed-point.

Therefore, the following must be checked:

- Is I complete?
- Does $\varphi(I) \subset I$ apply?
- Holds $\|\varphi(x) - \varphi(y)\| \leq L\|x - y\|$ for all $x, y \in I$ as well as $L < 1$?

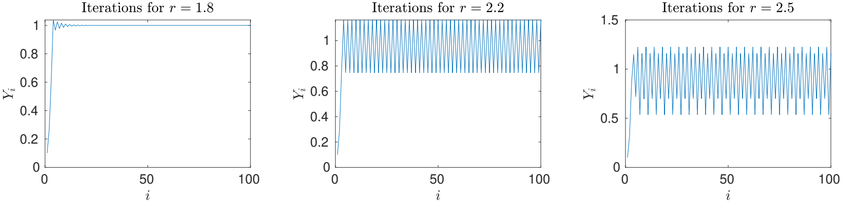
Fixed-point problem by logistic map

Example:

- consider the iteration $Y_{n+1} = \varphi(Y_n)$ with $Y_0 = 0.1$ and

$$\varphi(y) = (1+r)y - ry^2,$$

- fixed points and alternating points after a short start-up phase of at least 10 steps

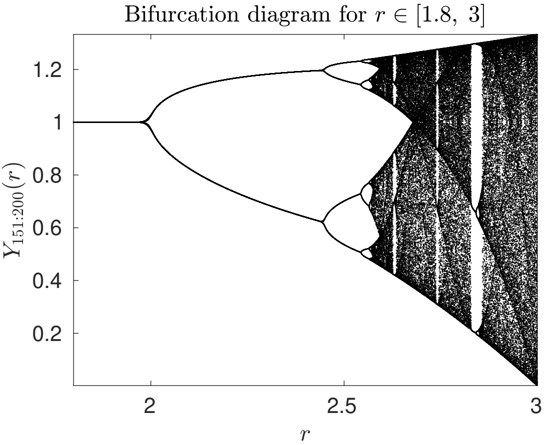


Analysis of the fixed points:

- fixed-point form $y_{i+1} = \varphi(y_i) = (1+r)y - ry^2$,
- fixed-points are $y_1 = 0$, $y_2 = 1$, the first value y_1 is not of interest,
- for y_2 holds $\varphi'(y_2) = 1 - r$, thus y_2 is stable only for $r < 2$.

Bifurcation diagram (Feigenbaum constants):

- using the iterations $\varphi(y) = (1+r)y - ry^2$,
- plot the iterations Y_i for $r \geq 150$,
- several alternating situations.



Fixed-point problem by logistic map

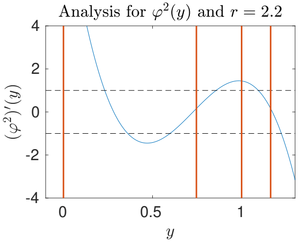
Alternations between two values:

- fixed-points for

$$\varphi^2(y) = y \cdot (1 + (1-y)r) \cdot (1 + (y^2 - y)r^2 + (1-y)r)$$

$$y_1 = 0,\ y_2 = 1,\ y_{3,4} = \frac{0.5r + 1 \pm 0.5\sqrt{r^2-4}}{r}$$

- stability analysis by $(\varphi^2)'(y)$, e. g. for $r = 2.2$ only y_3 and y_4 are stable fixed points.



Fixed-points of period 4:

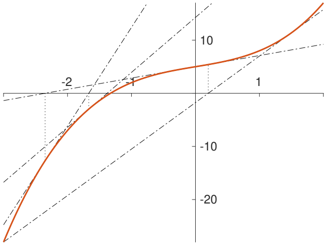
- thus $y = \varphi^4(y)$, e. g. for $r = 2.5$ there are four stable points

$$y_3 = 0.6, \quad y_4 = 0.7012, \quad y_6 = 1.1576, \quad y_7 = 1.2.$$

Outline

Idea:

- put a tangent to $f(x_i)$,
- the zero of the tangent as a new iterate x_{i+1}.



Iteration of Newton's method:

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}.$$

Definition 2.4

Let $(x_i)_{i=0,1,2,\dots}$ be a sequence with $x_i \in \mathbb{R}^n$, which converges to $x^* \in \mathbb{R}^n$ and let $x_i \neq x^*$ for all i. Further let $\|\cdot\|$ be a vector norm for $\mathbb{R}^n$.
The sequence is called convergent to x^* with at least the convergence order p if there is a $c > 0$ with

$$\|x_{i+1} - x^*\| \leq c\|x_i - x^*\|^p$$

for all sufficiently large $i \in \mathbb{N}$.

Theorem 2.5

Let x^* be a simple zero of f and further let $U \subset \mathbb{R}$ be an open neighbourhood around x^* as well as f be two times continuous differentiable on U.
In a neighbourhood of x^* holds

$$x_{k+1} - x^* = \frac{1}{2}\frac{f''(\xi)}{f'(x_k)}(x_k - x^*)^2, \qquad \text{for an } \xi \in U.$$

Outline

Newton's method for systems

Newton's iteration:

$$x_{k+1} = x_k + \Delta = x_k - \left(J_f(x_k)\right)^{-1} f(x_k), \qquad k = 0, 1, 2, \ldots.$$

Jacobian matrix of partial derivatives of first order of f:

$$J_f(x) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{pmatrix}.$$

Stopping criterion:

$$\|f(x_k)\|_2 \leq \varepsilon_f \quad \text{and/or} \quad \|\Delta\|_2 < \varepsilon_x$$

with $\varepsilon_f, \varepsilon_x > 0$, e.g. $\varepsilon_f = 10^{-6}$ and $\varepsilon_x = 10^{-4}$.

Example (fractal):

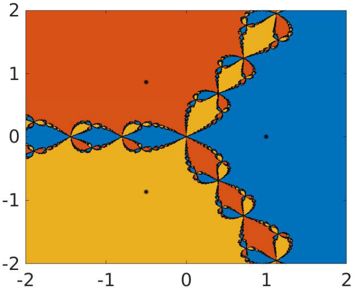
- 2×2 system of equations

$$f(x) = \begin{pmatrix} x_1^3 - 3x_1x_2^2 - 1 \\ 3x_1^2x_2 - x_2^3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

- three solutions

$$(1,0), \qquad (-1/2, \sqrt{3}/2), \qquad (-1/2, -\sqrt{3}/2),$$

- iteration converges to one of the solutions, depending on the starting vector,
- application for starting vectors in $[-2,2] \times [-2,2]$.



Example:

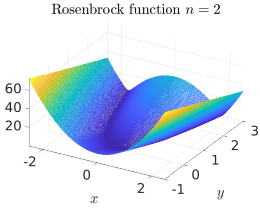
- consider the problem of Rosenbrock (with the solution $x^* = (1,1)^T$)

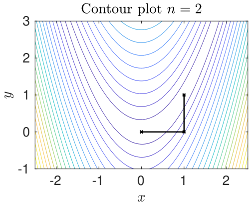
$$f : \mathbb{R}^2 \to \mathbb{R}^2, \quad f(x_1, x_2) = \begin{pmatrix} 1 - x_1 \\ 10(x_2 - x_1^2) \end{pmatrix} \quad \text{with} \quad J_f = \begin{pmatrix} -1 & 0 \\ -20x_1 & 10 \end{pmatrix},$$

- applying Newton's method to $x_1 = (0,0)^T$ we get

$$x^{(1)} = x^{(0)} - J_f(x^{(0)})^{-1} f(x^{(0)}) = \begin{pmatrix} 0 \\ 0 \end{pmatrix} - \begin{pmatrix} -1 & 0 \\ 0 & 10 \end{pmatrix}^{-1} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$x^{(2)} = x^{(1)} - J_f(x^{(1)})^{-1} f(x^{(1)}) = \begin{pmatrix} 1 \\ 0 \end{pmatrix} - \begin{pmatrix} -1 & 0 \\ -2 & 0.1 \end{pmatrix} \begin{pmatrix} 0 \\ -10 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$





Outline

2. Nonlinear equations & optimization

Damped Newton's method:

- do not take the full step, but only a part,
- for $0 < \lambda \leq 1$ and $n = 1$ this means

$$x_{i+1} = x_i - \lambda \frac{f(x_i)}{f'(x_i)},$$

- the same for systems

$$x_{k+1} = x_k - \lambda \big(J_f(x_k)\big)^{-1} f(x_k),$$

- e. g. divide λ by 2 until

$$\|f\big(x_k - \lambda \big(J_f(x_k)\big)^{-1} f(x_k)\big)\| \leq (1 - \frac{\lambda}{2})\|f(x_k)\|.$$

Simplified Newton's method:

- update the computation/approximation of $J_f(x_k)$ not in each step,
- work with the computed $J_f(x_k)$ for a number of m_{fix} iterations,
- use LU-decomposition.

Advantage:

- less computations of $J_f(x_k)$ (high effort).

Disadvantage:

- linear convergence only.

Outline

Nonlinear least-squares problem:

$$F(x) = \frac{1}{2}\sum_{i=1}^{m}\left(f_i(x)\right)^2 \to \min, \qquad f : \mathbb{R}^n \to \mathbb{R}^m,\ m \geq n.$$

Necessary condition for a local minimum:

$$\nabla F(x) = \left(J_f^T f\right)(x) = 0. \tag{1}$$

Gauss-Newton step:

$$x_{k+1} = x_k - \left(J_f^T(x_k)J_f(x_k)\right)^{-1}\left(J_f^T(x_k)f(x_k)\right).$$

<u>Levenberg-Marquardt procedure</u>:

- additional regularization of the solution,
- iteration reads

$$x_{k+1} = x_k - \alpha \left(J_f^T(x_k) J_f(x_k) + M \right)^{-1} \left(J_f^T(x_k) f(x_k) \right),$$

- regularization by $M \in \mathbb{R}^{n \times n}$ e.g. $M = \beta I$,
- for $\beta = 0$ and $\alpha = 1$ we get Gauss-Newton's method,
- otherwise the additional term acts regulating e.g. to avoid too large steps.

Outline

Problem:

- find the minimum of $f : \Omega \to \mathbb{R}$,
- unconstrained if $\Omega = \mathbb{R}^n$ otherwise constrained.

Gradient/steepest descent:

- for a differentiable $f(x)$ the gradient $\nabla f(x_k)$ is the direction of steepest ascent,
- use $v_k = -\nabla f(x_k)$ to get the steepest descent,
- the function $s \mapsto f(x_k + sv_k)$ is monotonously decreasing for $s \in [0, \tilde{s})$ for some positive $\tilde{s}$,
- start with $s = 1$ and divide s by 2 until

$$f(x_k + sv_k) < f(x_k),$$

- use $x_{k+1} = x_k + sv_k$.

Curve-fitting problem:

- measured values for $c(t)$ are

t	0	1	2	3
c	9	5	4	2

- approach

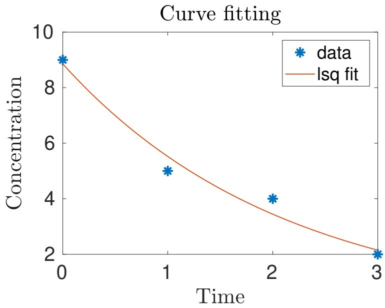
$$g(t) = g(t;x) = x_1 \exp(x_2 t),$$

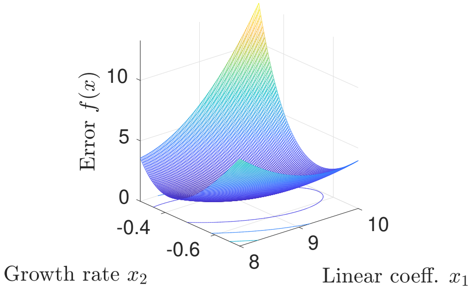
- compute the parameters x_1 and x_2 by minimizing the error

$$f(x) = \frac{1}{2}\|g(t,x) - c\|_2^2,$$

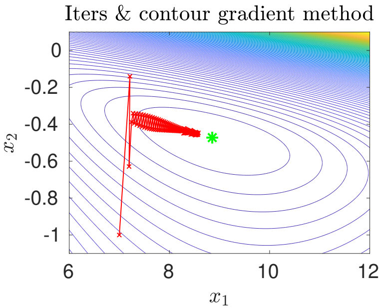
- the optimum is at $x^*(8.8551,\ -0.4722)^T$.

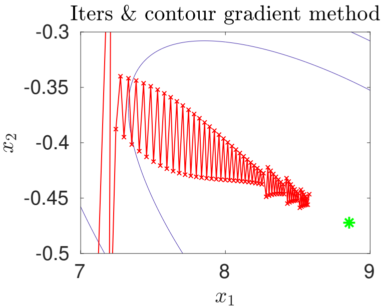
Curve fitting problem and cost function:





A number of 100 steps of steepest descent for $x_0 = (7, -1)^T$:

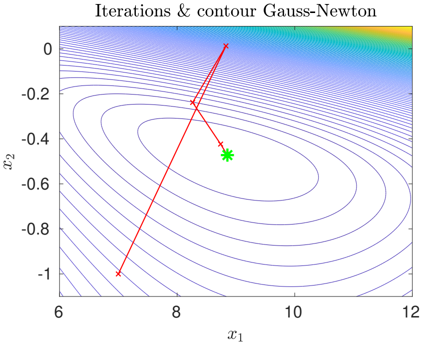




Gauss-Newton's:

- proper method of curve fitting,
- starting vector $x_0 = (7, -1)^T$,
- after 6 iterations we get an approximation with

$$\|x_6 - x^*\|_2 = 3.6 \cdot 10^{-4}$$



Outline

Polynomial interpolation

Given:

$$(x_i, f_i), \quad i = 0, \ldots, n.$$

To compute: Polynomial $p(x)$ of degree smaller than or equal to n with

$$p(x_i) = f_i, \quad i = 0, \ldots, n.$$

Interpolation condition: The nodes $x_0, \ldots, x_n$ have to be pairwise different.

Theorem 3.1

For pairwise different grid points the interpolation polynomial is unique.

A simple approach to compute the polynomial:

$$p(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_n x^n.$$

System of linear equations (Vandermonde's matrix):

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \ldots & x_0^n \\ 1 & x_1 & x_1^2 & \ldots & x_1^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \ldots & x_n^n \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{pmatrix}. \tag{2}$$

with

$$\det(V_{n+1}) = \prod_{0 \le i < j \le n} (x_i - x_j).$$

But: The problem is ill-conditioned. Use other approaches.

Outline

Lagrange interpolation

Definition 3.2

According to $x_0, \dots, x_n$ the Lagrange basis functions are

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

Obviously holds

$$l_i(x_\ell) = \prod_{j=0, j\neq i}^{n} \frac{x_\ell - x_j}{x_i - x_j} = \begin{cases} 1 & \text{for } \ell = i \\ 0 & \text{otherwise.} \end{cases}$$

Lagrange interpolation

Definition 3.3

The construction

$$p(x) = \sum_{i=0}^{n} f_i l_i(x) = \sum_{i=0}^{n} f_i \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j}$$

is called Lagrange polynomial.

Outline

Newton interpolation (1)

- successive structure, start with one interpolation point, then 2, 3, etc.
- degree of the polynomial increases by one per step,
- initialisation $p_0(x) = y_0$
- first iteration step

$$p_1(x) = p_0(x) + c_1(x - x_0), \quad \text{with} \quad c_1 = \frac{y_1 - y_0}{x_1 - x_0}.$$

Newton interpolation (2)

- further iteration from $p_{m-1}(x)$ to $p_m(x)$ by

$$p_m(x) = p_{m-1}(x) + c_m \prod_{i=0}^{m-1} (x - x_i),$$

- to have $p_m(x_m) = y_m$ next to $p_m(x_j) = y_j$ for $j = 0, \ldots, m-1$ use

$$c_m = \frac{y_m - p_{m-1}(x_m)}{\prod_{i=0}^{m-1}(x_m - x_i)}.$$

Definition 3.4

Using $f[x_0] = y_0$ the divided difference are

$$f[x_m, \ldots, x_0] := c_m = \frac{y_m - p_{m-1}(x_m)}{\prod_{i=0}^{m-1}(x_m - x_i)}, \qquad m = 1, 2, \ldots, n.$$

The representation of $p_m(x)$ in Newton's form is

$$p_m(x) = \sum_{i=0}^{m} f[x_i, \ldots, x_0] \prod_{j=0}^{i-1}(x - x_j).$$

Theorem 3.5

Let $x_0, \ldots, x_m$ be pairwise different and $f[x_0] = y_0, \ldots, f[x_m] = y_m$. We can apply the recursion

$$f[x_m, \ldots, x_0] = \frac{f[x_m, \ldots, x_1] - f[x_{m-1}, \ldots, x_0]}{x_m - x_0}.$$

Computation of coefficients using Newton's scheme:

- initialisation $f[x_i] = y_i$ for $i = 0, \ldots, n$,
- triangle scheme

$$
\begin{array}{c|ccccc}
x_0 & f[x_0] & & & \\
x_1 & f[x_1] & \frac{f[x_1]-f[x_0]}{x_1-x_0} =: f[x_1,x_0] & & \\
x_2 & f[x_2] & \frac{f[x_2]-f[x_1]}{x_2-x_1} =: f[x_2,x_1] & \frac{f[x_2,x_1]-f[x_1,x_0]}{x_2-x_0} =: f[x_2,x_1,x_0] & \\
x_3 & f[x_3] & \frac{f[x_3]-f[x_2]}{x_3-x_2} =: f[x_3,x_2] & \frac{f[x_3,x_2]-f[x_2,x_1]}{x_3-x_1} =: f[x_3,x_2,x_1] & f[x_3,x_2,x_1,x_0] \\
\vdots & \vdots & \vdots & \vdots & \vdots
\end{array}
$$

- the interpolation polynomial reads

$$
p(x) = f[x_0] + f[x_1,x_0](x-x_0) + f[x_2,x_1,x_0](x-x_0)(x-x_1) + \ldots
$$

Example:

- interpolation of

$$(-1,5),\ (0,4),\ (1,-3),\ (2,3),$$

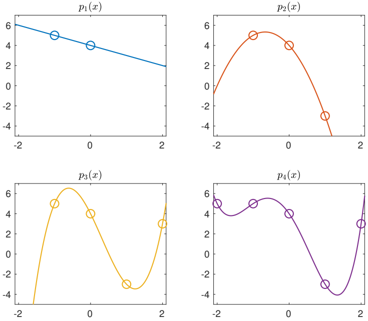
- polynomial of degree ≤ 3,
- Newton's scheme by divided differences

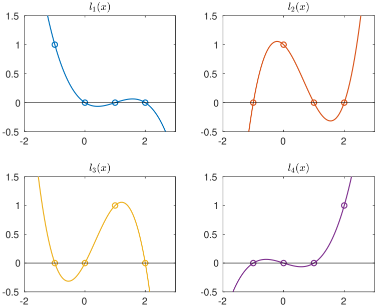
$$\begin{array}{r|rrrr} -1 & 5 & & & \\ 0 & 4 & -1 & & \\ 1 & -3 & -7 & -3 & \\ 2 & 3 & 6 & \frac{13}{2} & \frac{19}{6} \end{array}$$

- interpolating polynomial is

$$\begin{aligned} p(x) &= 5 - 1(x+1) - 3(x+1)x + \frac{19}{6}(x+1)x(x-1) \\ &= 4 - \frac{43}{6}x - 3x^2 + \frac{19}{6}x^3. \end{aligned}$$

The single Newton steps (left) as well as the single Lagrange basis functions (right)





Both approaches result in the same polynomial

$$\begin{aligned} p(x) &= 5 - 1(x+1) - 3(x+1)x + \frac{19}{6}(x+1)x(x-1) \\ &= 5\,\frac{x(x-1)(x-2)}{-6} + 4\,\frac{(x+1)(x-1)(x-2)}{2} + (-3)\,\frac{(x+1)x(x-2)}{-2} + 3\,\frac{(x+1)x(x-1)}{6} \\ &= 4 - \frac{43}{6}x - 3x^2 + \frac{19}{6}x^3. \end{aligned}$$

Extension:

- consider that we add $(-2, 5)$ as additional point to the previous example,
- the extended scheme is

$$
\begin{array}{c|cccccc}
-1 & 5 & & & & \\
0 & 4 & -1 & & & \\
1 & -3 & -7 & -3 & & \\
2 & 3 & 6 & \frac{13}{2} & \frac{19}{6} & \\
\hline
-2 & 5 & -\frac{1}{2} & \frac{13}{6} & \frac{13}{6} & 1
\end{array}
$$

- the new polynomial is

$$\tilde{p}(x) = 5 - 1(x+1) - 3(x+1)x + \frac{19}{6}(x+1)x(x-1) + (x+1)x(x-1)(x-2).$$

How about $\max_{x\in[a,b]} |f(x) - p(x)|$ for a function $f(x)$?

Theorem 3.6

Let $f(x)$ be $(n+1)$ times continously differentiable on the interval $[a,b]$. Further let $a \leq x_0 < x_1 < \cdots < x_n \leq b$ and $p(x)$ be the interpolating polynomial for $(x_i, f(x_i))$, $i = 0\ldots,n$. Then it holds

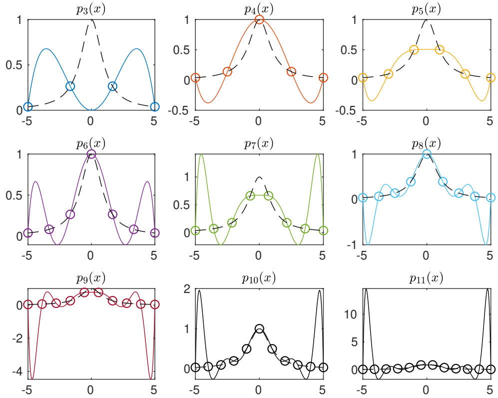
$$|f(\tilde{x}) - p(\tilde{x})| \leq \frac{|w(\tilde{x})|}{(n+1)!} \max_{\xi\in[a,b]} \left|f^{(n+1)}(\xi)\right|$$

for

$$w(x) = \prod_{j=0}^{n} (x - x_j).$$

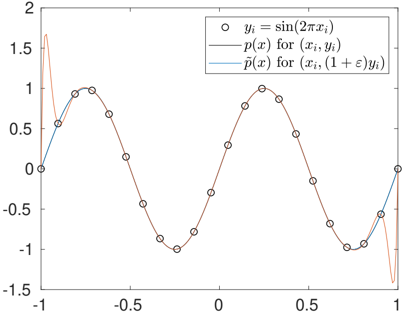
Runge, 1901:

- interpol. of $(1+x^2)^{-1}$, polynomials of nth degree, equidistant nodes in $[-5,5]$,
- very bad approximations for large n, oscillations at the boundary.



Sesitivity for noise:

- interpolation of $f(x) = \sin(2\pi x)$ using 22 equidistant knots on $[-1, 1]$,
- comparison of $p(x)$ for noisy and noise free data $\tilde{y}_i = (1+\varepsilon) f(x_i)$ with $\varepsilon = 10^{-4}$,
- large differences for the polynomial and high oscillations at the boundaries.



Outline

Modified problem:

- interpolation also for values of derivates of $f(x)$,
- nodes $x_0 < x_1 < \cdots < x_m$ and values $f_i^{(j)} = f^{(j)}(x_i)$ for $j = 0, \ldots, n_i - 1$,
- compute an interpolation polynomial $p(x)$ with

$$\deg(p(x)) \leq n, \quad n + 1 = \sum_{i=0}^{m} n_i$$

such that

$$p^{(j)}(x_i) = f_i^{(j)}, \qquad j = 0, \ldots, n_i - 1, \; i = 0, \ldots, m. \tag{3}$$

- the interpolating polynomial is unique as $n + 1$ degrees of freedom equals the number of conditions.

Hermite Interpolation

Scheme for a block to one node:

$$
\begin{array}{c|ccccc}
x_0 & f[x_0] & & & & \\
x_0 & f[x_0] & \frac{f'(x_0)}{1!} & & & \\
x_0 & f[x_0] & \frac{f'(x_0)}{1!} & \frac{f''(x_0)}{2!} & & \\
\vdots & \vdots & & & \ddots & \\
x_0 & f[x_0] & \frac{f'(x_0)}{1!} & \dots & & \frac{f^{(n)}(x_0)}{n!}
\end{array}
$$

Interpolating polynomial

$$p(x) = f(x_0) + \frac{f'(x_0)}{1!}(x - x_0) + \cdots + \frac{f^{(n)}(x_0}{n!}(x - x_0)^n$$

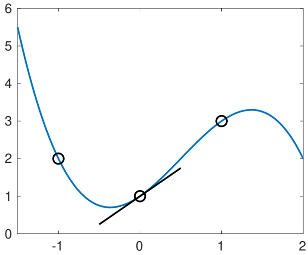
Example:

- interpolation of $(-1,2)$, $(0,1)$, $(1,3)$ with the additional condition $p'(0) = 1.5$,
- Newton's scheme

$$\begin{array}{c|cccc} -1 & 2 & & & \\ 0 & 1 & -1 & & \\ 0 & 1 & 1.5 = \frac{1.5}{1!} & 2.5 & \\ 1 & 3 & 2 & 0.5 & -1 \end{array}$$

- interpolation polynomial

$$p(x) = 2 - (x+1) + 2.5(x+1)x - 1(x+1)x^2.$$



Example:

- interpolation polynomial for $f(x) = 1/x$ using $x_0 = 1$, $x_1 = 2$,
- conditions

$$f(x_0) = 1,\ f'(x_0) = -1,\ f''(x_0) = 2,\ f'''(x_0) = -6$$
$$f(x_1) = 0.5,\ f'(x_1) = -0.25$$

- Newton's scheme

$$\begin{array}{c|ccccccc}
1 & 1 & & & & & \\
1 & 1 & -1 = \frac{-1}{1!} & & & & \\
1 & 1 & -1 & 1 = \frac{2}{2!} & & & \\
1 & 1 & -1 & 1 & -1 = -\frac{6}{3!} & & \\
2 & 1/2 & -\frac{1}{2} & \frac{1}{2} & -\frac{1}{2} & \frac{1}{2} & \\
2 & 1/2 & -\frac{1}{4} & \frac{1}{4} & -\frac{1}{4} & \frac{1}{4} & -\frac{1}{4}
\end{array}$$

- interpolation polynomial

$$p(x) = 1 - (x-1) + (x-1)^2 - (x-1)^3 + \frac{1}{2}(x-1)^4 - \frac{1}{4}(x-1)^4(x-2).$$

Outline

Splines:

- many interpolating points,
- local interpolations for a few neighboring points instead of a global polynomial,
- piecewise composed polynomial using smoothness requirements at transitions.

Definition 3.7

The <u>*cubic splines*</u> *are defined as follows: For the pairs* (x_i, y_i), $i = 0, \ldots, n$, *compute a spline* S *that interpolates these points such that*

$$S_{|[x_i,\, x_{i+1}]}(x) := p_i(x) \qquad \text{is a polynomial of degree } 3$$

and to make sure that the function is twice continously differentiable at the transition points. This means S *has to fulfill the four conditions*

$$p_i(x_i) = y_i, \quad p_i(x_{i+1}) = y_{i+1}, \quad p_i'(x_i) = p_{i+1}'(x_i), \quad p_i''(x_i) = p_{i+1}''(x_i).$$

For the first and the last polynomial there are no boundary conditions. The selection

$$S''(a) = S''(b) = 0$$

results in the <u>*natural splines*</u>.

Computation of the splines:

- a linear system of equations (moment equations) to calculate the cubic splines,
- quite technical, see the following theorem.

Theorem 3.8

If the twice continuously differentiable cubic spline S satisfies the interpolation condition

$$S(x_j) = f_j, \quad j = 0, \dots, n,$$

then on $[x_j, x_{j+1}]$ its form reads

$$\begin{aligned} S(x) = {} & f_j \frac{x_{j+1} - x}{x_{j+1} - x_j} + f_{j+1} \frac{x - x_j}{x_{j+1} - x_j} \\ & - \frac{1}{6} S''(x_j) \frac{(x_{j+1} - x)(x - x_j)}{x_{j+1} - x_j} \big((x_{j+1} - x) + (x_{j+1} - x_j)\big) \\ & - \frac{1}{6} S''(x_{j+1}) \frac{(x_{j+1} - x)(x - x_j)}{x_{j+1} - x_j} \big((x - x_j) + (x_{j+1} - x_j)\big). \end{aligned}$$

Theorem 3.9

Under the conditions of Thm. 3.8 it holds for the moments $S''(x_j)$ *for* $j = 1, \dots, n-1$ *that*

$$\begin{aligned} &\frac{x_{j+1} - x_j}{x_{j+1} - x_{j-1}} S''(x_{j+1}) + 2S''(x_j) + \frac{x_j - x_{j-1}}{x_{j+1} - x_{j-1}} S''(x_{j-1}) \\ &= \frac{6}{x_{j+1} - x_{j-1}} \left(\frac{f_{j+1} - f_j}{x_{j+1} - x_j} - \frac{f_j - f_{j-1}}{x_j - x_{j-1}} \right) \\ &= 6f[x_{j-1}, x_j, x_{j+1}]. \end{aligned}$$

Theorem 3.10

Together with the two conditions, e.g. $S''(x_0) = S''(x_n) = 0$ *for the natural splines, the system of linear equations has a unique solutions. The matrix of the system has a condition not larger then 3.*

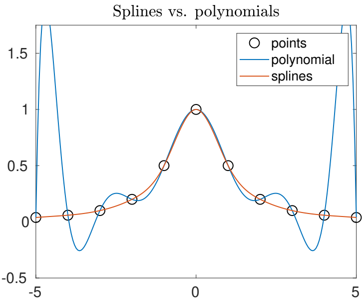
Example:

- consider once again the example of Runge

$$f(x) = \frac{1}{1+x^2},$$

- cubic splines for equidistant nodes and $[-5, 5]$ using MatLab,
- much better results than for polynomials, maximal error

$$\max_{x\in[-5,5]} \left|S(x) - f(x)\right| \approx 0.022.$$



Splines in MatLab:

```
f = @(x) 1./(1+x.^2);
X = -5:5;
Y = f(X);
xx = linspace(-5,5,401);
yy = spline(X,Y,xx);
plot(X,Y,'o',xx,yy)
max(abs(yy-f(xx)))
```

Numerical differentiation

Definition 3.11 (Finite differences)

The approximations

$$D^+f(x) = \frac{f(x+h)-f(x)}{h},$$
$$D^-f(x) = \frac{f(x)-f(x-h)}{h},$$
$$D^0f(x) = \frac{f(x+h)-f(x-h)}{2h}$$

for $f'(x)$ are called forward difference, backward difference and central difference.

Theorem 3.12 (Approximation orders)

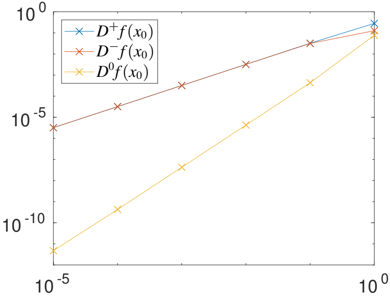
For the finite differences holds

$$f'(x) - D^+f(x) = \mathcal{O}(h), \qquad \textit{(order 1)}$$
$$f'(x) - D^-f(x) = \mathcal{O}(h), \qquad \textit{(order 1)}$$
$$f'(x) - D^0f(x) = \mathcal{O}(h^2), \qquad \textit{(order 2).}$$

Example:

- approximation of $f'(x_0)$ for $f(x) = \arctan(x)$ and $x_0 = 0.5$
- approximations errors for different h

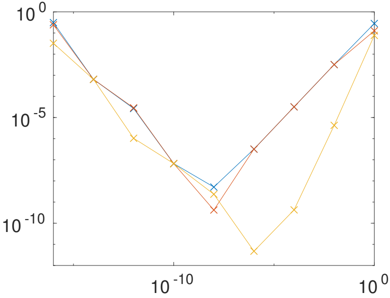
h	$\lvert D^+f(x_0) - f'(x_0)\rvert$	$\lvert D^-f(x_0) - f'(x_0)\rvert$	$\lvert D^0f(x_0) - f'(x_0)\rvert$
0.1	$3.2 \cdot 10^{-2}$	$3.1 \cdot 10^{-2}$	$4.3 \cdot 10^{-4}$
0.01	$3.2 \cdot 10^{-3}$	$3.2 \cdot 10^{-3}$	$4.3 \cdot 10^{-6}$
10^{-5}	$3.2 \cdot 10^{-6}$	$3.2 \cdot 10^{-6}$	$4.8 \cdot 10^{-12}$.



Example:

- approximation of $f'(x_0)$ for $f(x) = \arctan(x)$ and $x_0 = 0.5$
- but we cannot reduce h arbitrarily, as

h	$\lvert D^+f(x_0) - f'(x_0)\rvert$	$\lvert D^-f(x_0) - f'(x_0)\rvert$	$\lvert D^0f(x_0) - f'(x_0)\rvert$
10^{-12}	$2.7 \cdot 10^{-5}$	$2.9 \cdot 10^{-5}$	$1.0 \cdot 10^{-6}$
10^{-14}	$6.4 \cdot 10^{-4}$	$6.4 \cdot 10^{-4}$	$6.4 \cdot 10^{-4}$.



Definition 3.13 (Finite differences)

The approximation

$$D^2 f(x) = \frac{f(x-h) - 2f(x) + f(x+h)}{h^2}$$

for $f''(x)$ is called central difference of second order.

The error of the second order central difference is in $\mathcal{O}(h^2)$, so $D^2 f$ is an approximation second order, thus

$$f''(x) = D^2 f(x) + \mathcal{O}(h^2).$$

Partial derivatives of higher order in 2D:

- mixed partial derivatives for $h_x = h_y = h$

$$u_{xy} \approx \frac{u(x+h,y+h) - u(x+h,y-h) - u(x-h,y+h) + u(x-h,y-h)}{4h^2},$$

- central difference of second order for $h_x = h$

$$u_{xx} \approx \frac{u(x+h,y) - 2u(x,y) + u(x-h,y)}{h^2}$$

- for the Laplace operator or Laplacian for $h_x = h_y = h$ we get

$$\begin{aligned}\Delta u &= u_{xx} + u_{yy}\\ &\approx \frac{u(x+h,y) + u(x-h,y) + u(x,y+h) + u(x,y-h) - 4u(x,y)}{h^2}.\end{aligned}$$

Outline

Literature

- Atkinson, K.: Elementary Numerical Analysis, John Wiley & Sons, 1993.
- Burden, R., Faries, J. D.: Numerical Analysis, Brooks Cole Publishing Company, 1997.
- Friedmann, M., Kandel, A.: Fundamentals of Computer Numerical Analysis, CRC Press, 1993.
- Golub, G. H., Ortega, J. M.: Scientific Computing and Differential Equations: An Introduction to Numerical Methods, Academic Press, 1992.
- Kharab, A., Guenther, R. B.: An Introduction to Numerical Methods: A Matlab Approach, Chapman & Hall / CRC, 2002.
- Quarteroni, A., Sacco, R., Saleri, F.: Numerical Mathematics, Springer, 2007.
- Stoer, J., Bulirsch, R.: Introduction to Numerical Analysis, Springer, 2002.
- Süli, E., Mayers, D.: An Introduction to Numerical Analysis, Cambridge University Press, 2003.