Numerical Methods - Lecture 1

University Leipzig

April 2023

Contents

1 Number Systems **2**
1.1 Switching between different systems . . . 2
1.2 Transformation of fractions . . . 3
1.3 Special Systems . . . 4

2 Number Systems **5**
2.1 Integer numbers . . . 5
2.2 Floating point numbers (fractions) . . . 6

3 Stability and fixed-point theorem **8**
3.1 Stability . . . 8
3.2 Fixed-point iteration and fixed-point theorem . . . 9
3.3 Rate of convergence . . . 11
3.4 Newtons method (for root finding) . . . 11
3.5 Rate of convergence for Newton's method . . . 12
3.6 Halley's method . . . 12

4 (Compolex) Roots of polynomials **14**
4.1 Fundamental theorem of algebra . . . 14
4.2 Müllers method . . . 14
4.3 Laguerre's Method . . . 16
4.4 Deflation . . . 17
4.5 Eigenvalue methods . . . 17
4.6 Aithen's Methods . . . 18

5 Interpolation **19**
5.1 Polynomial interpolation . . . 19
5.1.1 Vandermonde-Matrix . . . 19
5.1.2 Lagrange polynomials . . . 20
5.1.3 Recursive determination of the interpolating polynomial / Neville's method . 21
5.1.4 Newton's method . . . 21

Chapter 1

Number Systems

Standard decimal number:

$$x = 1709.3_{10} = 1 \cdot 10^3 + 7 \cdot 10^2 + 0 \cdot 10^1 + 9 \cdot 10^0 + 3 \cdot 10^{-1}$$

The 10 indicates the base $b = 10$. Generally we can write a number as follows:

$$x = \pm a_n \cdot b^n + a_{n-1} \cdot b^{n-1} + \ldots + a_0 b^0 + a_{-1} b^{-1} + \ldots$$

Where $b \in \mathbb{N}$, $b > 1$ and $a_i \in \{0, 1, \ldots, b-1\}$

If $b \leq 10$ the usual symbols can be used. If $b > 10$ new symbols are needed. E.g.: hexadecimal ($b = 16$):

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

1.1 Switching between different systems

1709_{10} in base 8: $8^0 = 1 \quad 8^1 = 8 \quad 8^2 = 64 \quad 8^3 = 512 \quad 8^4 = 4096$

$$\begin{aligned} 1709_{10} &= 3 \cdot 512_{10} + 2 \cdot 64_{10} + 5 \cdot 8_{10} + 5_{10} \\ &= 3 \cdot 8^3 + 2 \cdot 8^2 + 5 \cdot 8^1 + 5 \cdot 8^0 \\ &= 3255_8 \end{aligned}$$

alternatively: calculate in the new system:

$$\begin{aligned} 10_{10} &= 12_8 \\ 1709_{10} &= 1 \cdot {12_8}^3 + 7 \cdot {12_8}^2 + 0 \cdot {12_8}^1 + 9 \cdot {12_8}^0 \end{aligned}$$

better alternative: division with remainder:

$$1709_{10} = 3255_8 = ((3 \cdot 8 + 2) \cdot 8 + 5) \cdot 8 + 5$$

generally

$$x = (\dots(((a_n b + a_{n-1})b + a_{n-2})b + \dots)b + a_1)b + a_0$$

a_0 is remainder from division x/b generally a_i is the remainder from

$$(\dots((x - a_0)/b - a_1)/b \dots - a_{i-1})/b$$

$$\begin{aligned}
1709 : 8 &= 213 \quad |5 \\
213 : 8 &= 26 \quad |5 \\
26 : 8 &= 3 \quad |2 \\
3 : 8 &= 0 \quad |3 \\
1709_{10} &= 3255_8
\end{aligned}$$

1.2 Transformation of fractions

Fraction $0.3_{10} = 3/10_{10}$

$$3 : 12_8 = 0.2\overline{3146}_8$$

better alternative: repeated multiplication ($0 \leq x < 1$)

$$x = (a_{-1} + (a_{-2} + (a_{-3} + \dots)/b)/b)/b$$

for example: a_{-1} is position in front of the dot in $x \cdot 8$

$$\begin{aligned}
&0.3 \cdot 8 = 2.4 \to 2 \\
&0.4 \cdot 8 = 3.2 \to 3 \\
&0.2 \cdot 8 = 1.6 \to 1 \\
&0.6 \cdot 8 = 4.8 \to 4 \\
&0.8 \cdot 8 = 6.4 \to 6 \\
&0.4 \cdot 8 = 3.2 \to 3 \\
&\dots \\
&0.3_{10} = 0.2\overline{3146}_8
\end{aligned}$$

1.3 Special Systems

The transformation becomes particularly simple if the base of one system is power or root of the other base. E.g.: $b = 2 = \sqrt[3]{8}$ from octal to binary.

$$
\begin{aligned}
1709_{10} =& 3 \cdot 8^3 + 2 \cdot 8^2 + 5 \cdot 8^1 + 5 \\
=& 3 \cdot 2^9 + 2 \cdot 2^6 + 5 \cdot 2^3 + 5 \\
=& (0 \cdot 2^2 + 1 \cdot 2^1 + \cdot\, 2^0) \cdot 2^9 + (0 \cdot 2^2 + 1 \cdot 2^1 + 0 \cdot 2^0) \cdot 2^6 \\
& + (1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0) \cdot 2^3 + (1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0) \cdot 2^0 \\
=& \underbrace{011}_{3_8}\underbrace{010}_{2_8}\underbrace{101}_{5_8}\underbrace{101}_{5_8}{}_2
\end{aligned}
$$

From binary to hexadecimal:

$$\underbrace{0110}_{6_{16}}\underbrace{1010}_{A_{16}}\underbrace{1101}_{D_{16}}2 = 6AD_{16}$$

Chapter 2

Number Systems

In the computer, the smallest units of memory is the bit, which can assume just 2 different states {0, 1} {on, off}, {true, false} , not necessary numbers. Larger units of memory are:

name	value	
Byte	8	bits
Kibibyte	2^{10}	bytes
kilobyte	10^3	bytes
Mebibyte	2^{20}	bytes
Megabyte	10^6	bytes

2.1 Integer numbers

Integer numbers can easily be represented. The data “standards“ are called “data types“ and specify the size and value function. E.g.: 1 byte unsigned integer:

memory state	value
11111111	255
...	...
00000001	1
00000000	0

with negative numbers $\rightarrow$ 1 byte signed integer:

memory state	value
01111111	127
...	...
00000001	1
00000000	0
11111111	-1
11111110	-2
11111101	-3
...	...
10000000	-128

2.2 Floating point numbers (fractions)

Floating-point numbers are the product of a number $m \in (0,1)$ called mantissa, an integer power e of a base b (usually b = 2) and a sign s.

$$x = (-1)^s \cdot m \cdot b^e$$

Datatype “single precision“ / 32-bit float

s	1 bit	e	8 bits	m	23 bits

- sign $(-1)^s$
- exponent e is an integer, but it is stored differently than “normal“ integers - binary number is shifted by 127

memory state	binary number	value of e
11111111	255	128
11111110	254	127
...	...	...
100000000	128	1
01111111	127	0
...	...	...
00000010	2	-125
00000001	1	-126
00000000	0	-127

  $x = e - 127$ (Where x represents the exponent)

  The values $e = 127$ and $e = 128$ are not used in the usual way. Instead, they are reserved for special cases (e.g. ± 0, $\pm\infty$, signaling NaN)

- mantissa is a positive binary number. In order to achieve maximal precision e is chosen such that all positions of m are used (no leading zeros). The point is after the first digit.

  E.g.: 4-digit decimal mantissa:

$$310678 \to 3.107 \cdot 10^5$$

$$0.00043136 \to 4.314 \cdot 10^{-4}$$

Hence, if $x \neq 0$ there is always a non-zero digit in front of the point. Since $b = 2$ this digit is a 1. It is therefore not stored, but set implicitly. (This is only not the case if $e = -127$)

$$m = 1 + \sum_{i=1}^{23} m_i \cdot 2^{-i}$$

The datatype double precision (float64, double)

s 1 bit	e 11 bits	m 52 bits

$$e \in \{-1023, \ldots, 1023\}$$
$$(-1022, 1024 \text{ reserved})$$

Precision: The relative precision with which a number can be represented is determined by the number of bits of the mantissa $m \in [1, 2)$. 23 bits $\rightarrow 2^{23} \approx 10^7$ different states of m $\rightarrow$ relative distance of a possible number x to its “neighbor” is 10^{-7}. double precision: 52 bits $\rightarrow 2^{-52} \approx 10^{-15}$

Chapter 3

Stability and fixed-point theorem

3.1 Stability

Consider

$$\int_0^1 \frac{x^{10}}{x+10}dx$$

This integral can be solved iteratively:

$$\begin{aligned} y_n &= \int_0^1 \frac{x^n}{x+10}dx \\ y_0 &= \int_0^1 \frac{1}{x+10}dx = [\ln(x+10)]_0^1 = \ln\left(\frac{11}{10}\right) \approx 0.095310 \\ y_n + 10 \cdot y_{n-1} &= \int_0^1 \frac{x^n + 10 \cdot x^{n-1}}{x+10}dx = \int_0^1 x^{n-1}dx = \frac{1}{n} \end{aligned}$$

No we could use $y_n = \frac{1}{n} - 10 \cdot y_{n-1}$ to obtain $y_1, \ldots, y_{10}$:

n	y_n
0	0.0953102
1	0.0468982
2	0.0310181
3	0.0231520
4	0.0184804
5	0.0151955
6	0.0147117
7	-0.00425944
8	0.167594
9	-1.56483 ...
10	15...
11	-157...

If we use $y_n = \frac{1}{n} - 10 \cdot y_{n-1}$, the initial error of y_0 grows like 10^n and since $y_n < y_{n+1}$, the error soon dominates. This is unstable behaviour.

If we use

$$y_n = \frac{1}{10} \cdot \left(\frac{1}{n+1} - y_{n+1} \right)$$

and start with the crude approximation $y_{30} = 0$, the large error of y_{30} is reduced by a factor of 10 with each iteration. This is a stable algorithm.

n	y_n
30	0
29	0.00333
28	0.00311
27	0.00324
...	
20	0.00434704
...	
0	0.0953102

A numerical algorithm is called stable, if the provided solution to a problem P is the exact solution to a Problem Q that can be derived from P by a slight variation of the input data. Else it is called unstable.

3.2 Fixed-point iteration and fixed-point theorem

We search the root of $f(x) = 2x - \tan(x)$. $2x - \tan(x) = 0$ can be transformed into a fixpoint equation:

$$x = \Phi_1(x) = \frac{1}{2}\tan(x)$$
$$x = \Phi_2(x) = \arctan(2x)$$

With $\hat{x} = \Phi(\hat{x})$ The idea of the fixed-point iteration is to create a sequence by repeated insertion in Φ

$$x_{i+1} = \Phi(x_i)$$

which converges against $\hat{x}$

i	$x_{i+1} = \frac{1}{2}\tan(x)$	$x_{i+1} = \arctan(2x_i)$
0	1.2	1.2
1	1.2861	1.1760
2	1.7084	1.1688
3	-3.6108	1.1666
4	!	1.1659
5		1.1657
6		1.1656

Consider the difference

$$|x_{i+1} - x_i| = |\Phi(x_i) - \Phi(x_{i-1})|$$

Definition. Let $I = [a, b] \subset \mathbb{R}$ be an interval. $\Phi : I \to \mathbb{R}$ is a contraction on I if there is a $0 \leq \theta < 1$ such that $|\Phi(x) - \Phi(y)| \leq \theta |x - y|$.

Remark. θ is called Lipschitz constant $\to$ Lipschitz continuity if $0 \leq \theta < \infty$

Lemma. If $\Phi : I \to \mathbb{R}$ is continuously differentiable on I ($\Phi \in C^1(I)$) then

$$\sup_{x,y \in I} \frac{|\Phi(x) - \Phi(y)|}{|x - y|} = \sup_{z \in I} \Phi'(z)$$

Proof. mean value theorem: For all $x, y \in I$, $x < y$ there is a $\xi \in I$ such that

$$\Phi(x) - \Phi(y) = \Phi'(\xi)(x - y)$$

□

Theorem (Fixed-point theorem)**.** Let $I = [a, b] \subset \mathbb{R}$ be an interval and $\Phi : I \to I$ a contraction with Lipschitz constant $0 < \theta < 1$ Then:

(i) there is exactly one fixed-point $\hat{x}$ of Φ, such that $\Phi(\hat{x}) = \hat{x}$

(ii) For any initial value $x_0 \in I$ the fixed-point iteration $x_{i+1} = \Phi(x_i)$ converges against $\hat{x}$ with

$$|x_{i+1} - x_i| \leq \theta \cdot |x_i - x_{i-1}| \text{ and}$$
$$|\hat{x} - x_i| \leq \frac{\theta^i}{1 - \theta} \cdot |x_1 - x_0|$$

Proof. For all $x_0 \in I$

$$|x_{i+1} - x_i| = |\Phi(x_i) - \Phi(x_{i-1})| \leq \theta \cdot |x_i - x_{i-1}| \text{ therefore}$$
$$|x_{i+1} - x_i| \leq \theta^i \cdot |x_1 - x_0|$$

Now we show that x_i is a Cauchy sequence:

$$\begin{aligned}
|x_{i+k} - x_i| &\leq |x_{i+k} - x_{i+k-1}| + |x_{i+k-1} - x_{i+k-2}| + \ldots + |x_{i+1} - x_i| \\
&\leq (\theta^{i+k-1} + \theta^{i+k-2} + \ldots + \theta^i) \cdot |x_1 - x_0| \\
&= \theta^i \cdot (\theta^{k-1} + \theta^{k-2} + \ldots + \theta^0) \cdot |x_1 - x_0| \\
&\leq \frac{\theta^i}{1 - \theta} \cdot |x_1 - x_0|
\end{aligned}$$

Therefore, x_i is a Cauchy sequence which implies convergence. $\hat{x} = \lim_{i \to \infty} x_i$ is also a fixpoint of Φ, since:

$$\begin{aligned}
|\hat{x} - \Phi(\hat{x})| &= |\hat{x} - x_{i+1} + x_{i+1} - \Phi(\hat{x})| \\
&= |\hat{x} - x_{i+1} + \Phi(x_i) - \Phi(\hat{x})| \\
&\leq |\hat{x} - x_{i+1}| + |\Phi(x_i) - \Phi(\hat{x})| \\
&\leq |\hat{x} - x_{i+1}| + \theta \cdot |\hat{x} - x_i| \\
&\to 0 \text{ for } i \to \infty
\end{aligned}$$

Show that there is only one fixed-point by assuming the opposite. If $\hat{x}$ and $\hat{y}$ are distinct fixed-points, then

$$0 \leq |\hat{x} - \hat{y}| = |\Phi(\hat{x}) - \Phi(\hat{y})| \leq \theta \cdot |\hat{x} - \hat{y}|$$

but $\theta < 1$, therefore this is only possible if $|\hat{x} - \hat{y}| = 0 \implies \hat{x} = \hat{y}$ □

3.3 Rate of convergence

A sequence $x_i \in \mathbb{R}$ converges with order (at least) $P \geq 1$ against $\hat{x}$ if there is a constant $C \geq 0$ such that

$$|x_{i+1} - \hat{x}| \leq C \cdot |x_i - \hat{x}|^P$$

where if $P = 1$ it is additionally required that $C < 1$. If $P = 1$ we speak of linear convergence, if $P = 2$ of quadratic convergence.

Alternatively the rate of convergence can also be defined through the inequalities of differences of adjacent elements of the sequence.

$$|x_{i+1} - x_i| \leq C \cdot |x_i - x_{i-1}|^P$$

In general, the fixed-point iteration converges only linearly due to

$$|x_{i+1} - \hat{x}| \leq \theta \cdot |x_i - \hat{x}|$$

3.4 Newtons method (for root finding)

We are searching the root $\hat{x}$ of f, but now we also have access to the first order derivative $f'(x)$. First order taylor expansion at current position x:

$$\begin{aligned}
&f(x) \approx f(x_i) + f'(x_i)(x - x_i) \\
&f(\hat{x}) = 0 \\
&x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}
\end{aligned}$$

This is a fixed-point equation with the iteration function $\Phi(x) = x - \dfrac{f(x)}{f'(x)}$

Example. Calculation of the squareroot of C.

$$f(x) = x^2 - C$$

In floating point calculation the exponent of C should be treated separately

$$\sqrt{C} = \sqrt{m \cdot 2^e} = \sqrt{m} \cdot 2^{e/2}$$

and $\sqrt{2}$ (for odd e) be calculated once and stored

$$\begin{aligned}
\sqrt{m} &= ? \\
f(x) = x^2 - m &= 0 \\
f'(x) &= 2x \neq 0
\end{aligned}$$

Where $m \in [1, 2)$ and $x \in \left[1, \sqrt{2}\right]$

$$\Phi(x) = x - \frac{f(x)}{f'(x)} = x - \frac{x^2 - m}{2x} = \frac{x}{2} + \frac{m}{2x} = \frac{1}{2}\left(x + \frac{m}{x}\right)$$
$$x_{i+1} = \frac{1}{2} \cdot \left(x_i + \frac{m}{x_i}\right)$$

E.g: $m = 1.96$

i	x_i	# correct digits
0	1	≤ 1
1	1.48	$= 1$
2	1.420216	$= 3$
3	1.40000167	$= 6$
4	1.40000000000099	$= 12$
5	1.4	$= 15$

(number of correct digits approximately doubles every iteration $\rightarrow$ quadratic convervenge)

3.5 Rate of convergence for Newton's method

$$\begin{aligned} 0 = f(\hat{x}) &= f(x_i) + f'(x_i)(\hat{x} - x_i) + \frac{1}{2} f(x_i)''(\hat{x} - x_i)^2 + \dots \\ &= f(x_i) + f'(x_i)(\hat{x} - x_i) + \frac{1}{2} f(\xi)''(\hat{x} - x_i)^2 \text{ for } \xi \in [\hat{x}, x_i] \end{aligned}$$

$$\hat{x} \underbrace{- x_i + \frac{f(x_i)}{f'(x_i)}}_{x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}} = \frac{1}{2} \frac{f''(\xi)}{f'(x_i)} (\hat{x} - x_i)^2$$

$$\hat{x} - x_{i+1} = \frac{f''(\xi)}{2f'(x)} (\hat{x} - x_i)^2$$

Hence if f' is limited and if $|f'(x)| > \epsilon > 0$ on an interval that contains the root, we can expect quadratic convergence:

$$|\hat{x} - x_{i+1}| \leq M \cdot |\hat{x} - x_i|^2$$

3.6 Halley's method

If the second derivative $f''(x)$ can be evaluated easily, then Halley's method can be useful. Consider:

$$g(x) = \frac{f(x)}{\sqrt{|f'(x)|}}$$
$$|f'(x)| = \operatorname{sign}(f'(x)) \cdot f'(x)$$

Roots of f are roots of g and vice versa if $|f'(x)| < \infty$. Now calculate the derivative of $g(x)$

$$\begin{aligned} g'(x) &= \frac{f'(x)\sqrt{|f'(x)|} - f(x) \cdot \frac{f''(x)}{2 \cdot \sqrt{|f'(x)|}} \cdot \operatorname{sign}(f'(x))}{|f'(x)|} \\ &= \frac{f'(x) \cdot |f'(x)| - \frac{1}{2} f(x) f''(x) \cdot \operatorname{sign}(f'(x))}{|f'(x)| \cdot \sqrt{|f'(x)|}} \\ &= \frac{f'(x)^2 - \frac{1}{2} f(x) f''(x)}{f'(x) \cdot \sqrt{|f'(x)|}} \end{aligned}$$

Apply Newtons method to $g(x)$

$$\begin{aligned} x_{i+1} &= x_i - \frac{g(x_i)}{g'(x_i)} \\ &= x_i - \frac{f(x_i)}{f'(x_i) - \frac{f(x_i) \cdot f''(x_i)}{2 \cdot f'(x_i)}} = \frac{f(x_i)}{f'(x_i) \cdot \left(1 - \frac{f(x_i) \cdot f''(x_i)}{2 \cdot f'(x_i)^2}\right)} \end{aligned}$$

This term can be considered to be a correction to Newtons method. (Without calculation) Halley's method has cubic convergence.

Chapter 4

(Compolex) Roots of polynomials

A Polynomial $P_n = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_1 x + a_0$ should be evaluated according to

$$P_n = ((\ldots((a_n x + a_{n-1})x + a_{n-2})x + \ldots)x + a_1)x + a_0$$

which has better stability and requires fewer operations.

4.1 Fundamental theorem of algebra

Let $P(z) = \sum_{k=0}^{n} a_k \cdot z^k$ be a non-constant polynomial of order $n \geq 1$ and complex coefficients $a_k \in \mathbb{C}$. Then P has at least one root $\hat{z}$ with $P(\hat{z}) = 0$ If roots are counted according to their multiplicity, P has n roots. If the roots $\hat{z_i}$ are known the polynomial can be written as

$$P(z) = a_n(z - \hat{z_1})(z - \hat{z_2})\ldots(z - \hat{z_n})$$

$\hat{z_i} = \hat{z_j}$ for $i \neq j$ is possible $\rightarrow$ two fold root in the representation of P.

$$P(z) = a_n(z - \hat{z_1})^{m_1}(z - \hat{z_2})^{m_2}\ldots(z - \hat{z_n})^{m_n}$$

Where the roots z_i are pairwise distinct. $m_i \in \mathbb{N}$ is called the multiplicity of the root z_i.

$$\sum_{i=1}^{k} m_i = n$$

4.2 Müllers method

Approximates the function by a parabolic (also works for non-polynomial functions) If 3 initial values x_1, x_2, x_3 are given. Then:

$$P(x) = a(x - x_3)^2 + b(x - x_3) + c$$

$$
\begin{aligned}
y_i &= f(x_i) \\
y_1 &= a(x_1 - x_3)^2 + b(x_1 - x_3) + c \\
y_2 &= a(x_2 - x_3)^2 + b(x_2 - x_3) + c \\
y_3 &= c
\end{aligned}
$$

Task: Get the a, b, c find root of $\tilde{P}(\tilde{x}) = a\tilde{x}^2 + b\tilde{x} + c$ and iterate again with $x_4 = \hat{\tilde{x}} + x_3$

First we eliminate the parameter b to calculate a:

$$
\begin{aligned}
\frac{y_1 - c}{x_1 - x_3} &= a(x_1 - x_3) + b \\
\frac{y_2 - c}{x_2 - x_3} &= a(x_2 - x_3) + b
\end{aligned}
$$

$$
\begin{aligned}
a(x_1 - x_2) &= \frac{y_1 - c}{x_1 - x_3} - \frac{y_2 - c}{x_2 - x_3} \\
a &= \frac{(y_1 - y_3)(x_2 - x_3) - (y_2 - y_3)(x_1 - x_3)}{(x_1 - x_3)(x_2 - x_3)(x_1 - x_2)}
\end{aligned}
$$

Now we can calculate the parameter b:

$$
\begin{aligned}
b &= \frac{y_1 - y_3}{x_1 - x_3} - \frac{(y_1 - y_3)(x_2 - x_3) - (y_2 - y_3)(x_1 - x_3)}{(x_1 - x_3)(x_2 - x_3)(x_1 - x_2)} \cdot (x_1 - x_3) \\
&= \frac{(x_1 - x_3)^2(y_2 - y_3) - (x_2 - x_3)^2(y_1 - y_3)}{(x_1 - x_3)(x_2 - x_3)(x_1 - x_2)}
\end{aligned}
$$

Now we calculate the root of $\tilde{p}$

$$
x_4 - x_3 = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}
$$

We select the solution that leads to the smaller $|x_4 - x_3|$ (The next estimate should be close to the current one). Hence, the difference in the numerator and potential rounding issues. Solution:

$$
\begin{aligned}
x_4 - x_3 = \hat{\tilde{x}} &= \frac{(-b \pm \sqrt{b^2 - 4ac}) \cdot (-b \mp \sqrt{b^2 - 4ac})}{2a \cdot (-b \mp \sqrt{b^2 - 4ac})} \\
&= \frac{b^2 - (b^2 - 4ac)}{2a \cdot (-b \mp \sqrt{b^2 - 4ac})} \\
&= \frac{-2c}{b \pm \sqrt{b^2 - 4ac}}
\end{aligned}
$$

Remember, we want to make $|x_4 - x_3|$ small

$$
x_4 - x_3 = \frac{-2c}{b + \operatorname{sign}(b)\sqrt{b^2 - 4ac}}
$$

$$
\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \to \begin{bmatrix} y_1 = f(x_1) \\ y_2 = f(x_2) \\ y_3 = f(x_3) \end{bmatrix} \to \begin{bmatrix} a & b & c \end{bmatrix} \to \begin{bmatrix} x_4 \end{bmatrix} \to \begin{bmatrix} x_2 & x_3 & x_4 \end{bmatrix}
$$

4.3 Laguerre's Method

$$P_n(x) = a_n(x-\xi_1)(x-\xi_2)\dots(x-\xi_n)$$

with unknown roots ξ_i ($p_n(\xi_i) = 0$)

$$\begin{aligned}
\ln|P_n(x)| &= \ln(a_n) + \ln|x-\xi_1| + \ln|x-\xi_2| + \ldots + \ln|x-\xi_n| \\
\frac{d}{dx}(\ln|P_n(x)|) = \frac{P_n'(x)}{P_n(x)} &= \frac{1}{x-\xi_1} + \frac{1}{x-\xi_2} + \ldots + \frac{1}{x-\xi_n} \coloneqq \mathcal{G}(x) \\
-\frac{d^2}{dx^2}(\ln|P_n(x)|) = \left(\frac{P_n''(x)}{P_n(x)}\right)^2 - \frac{P_n''(x)}{P_n(x)} &= \frac{1}{(x-\xi_1)^2} + \frac{1}{(x-\xi_2)^2} + \ldots + \frac{1}{(x-\xi_n)^2} \coloneqq \mathcal{H}(x)
\end{aligned}$$

$\mathcal{G}(x)$ and $\mathcal{H}(x)$ can be calculated since p_n, p_n', p_n'' are available. Assumptions:

- root closest to current estimate x_i is ξ_1. Define as a distance $a = x_i - \xi_1$
- all other roots $\xi_2, \ldots, \xi_n$ have the same distance to x_i. $b = x_i - \xi_k$ for $k > 1$

Now we can find an approximation for $\mathcal{G}, \mathcal{H}$:

$$\begin{aligned}
\mathcal{G} &= \frac{1}{a} + \frac{n-1}{b} \\
\mathcal{H} &= \frac{1}{a^2} + \frac{n-1}{b^2} \\
b &= \frac{n-1}{\mathcal{G} - \frac{1}{a}} = \sqrt{\frac{n-1}{\mathcal{H} - \frac{1}{a^2}}} \\
\frac{\left(\mathcal{G} - \frac{1}{a}\right)^2}{(n-1)^2} &= \frac{\mathcal{H} - \frac{1}{a^2}}{n-1} \\
\mathcal{G}^2 - \frac{2}{a}\mathcal{G} + \frac{1}{a^2} &= \left(\mathcal{H} - \frac{1}{a^2}\right)(n-1)
\end{aligned}$$

Now we can solve this quadratic equation to get the parameter a:

$$\begin{aligned}
0 &= \frac{n}{a^2} - 2\,\mathcal{G}\,\frac{1}{a} + \mathcal{G}^2 - \mathcal{H}(n-1) \\
0 &= \frac{1}{a^2} - \frac{2\,\mathcal{G}}{n} \cdot \frac{1}{a} + \frac{\mathcal{G}^2}{n} - \frac{\mathcal{H}(n-1)}{n} \\
\frac{1}{a} &= \frac{\mathcal{G}}{n} \pm \sqrt{\left(\frac{\mathcal{G}}{n}\right)^2 - \frac{\mathcal{G}^2}{n} + \frac{\mathcal{H}(n-1)}{n}} \\
a &= \frac{n}{\mathcal{G} \pm \mathcal{G}^2(1-n) + \mathcal{H}(n(n-1))}
\end{aligned}$$

$x_{i+1} = x_i - a$ We want a to be small. For real $\mathcal{G}$:

$$a = \frac{n}{\mathcal{G} + \text{sign}(\mathcal{G})\sqrt{\mathcal{G}^2(1-n) + \mathcal{H}\, n(n-1)}}$$

for complex values we choose sign that maximize the absolute value of the denominator. $\rightarrow$ Cubic convergence in the proximity of a single root, reliable convergence to some root.

4.4 Deflation

If one root $\hat{x}$ of the polynomial p_n has been found and the simplified polynomial $Q_{n-1} = {}^{p_n}\!/_{(x-\hat{x})}$ can be calculated, Q_{n-1} can be evaluated since it is of a lower order and the roots of Q_{n-1} are the remaining unknown roots. One avoids to find $\hat{x}$ a second time. This process can be repeated for each new root. If p_n has only real coefficients and if the root $\hat{x}$ is complex, then the complex conjugate $\bar{\hat{x}}$ is also a root of p_n. In that case, we can divide by the product $(x - \bar{\hat{x}})(x - \hat{x})$ which reduces the order of p_n by 2.

Deflation is stable if one goes from absolutely small to large roots and starts the polynomial division with the coefficients of highest order or if one goes from large to small roots and starts division of the scalar term. In any case, at the end all roots should be optimized using the original polynomial.

4.5 Eigenvalue methods

It can be shown that the polynomial $p_n = \sum_{i=0}^{n} a_i \cdot x^i$ has the same roots as the characteristic polynom of:

$$A = \begin{pmatrix} \frac{a_{n-1}}{a_n} & \frac{a_{n-2}}{a_n} & \dots & \frac{a_0}{a_n} \\ 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & 0 \\ 0 & \dots & 1 & 0 \end{pmatrix}$$

Therefore, roots of p are Eigenvalues of A and can be found using Eigenvalue-methods.

4.6 Aithen's Methods

If one already uses an algorithm that shows linear convergence, Aithen's method might be useful to accelerate convergence. Linear convergence:

$$|x_{i+1} - \hat{x}| \leq C \cdot |x_i - \hat{x}|$$

Assuming:

$$\begin{aligned}
\frac{x_{i+1} - \hat{x}}{x_i - \hat{x}} &\approx \frac{x_{i+2} - \hat{x}}{x_{i+1} - \hat{x}} \\
(x_{i+1} - \hat{x})^2 &\approx (x_{i+2} - \hat{x})(x_i - \hat{x}) \\
x_{i+1}^2 - 2x_{i+1}\hat{x} &\approx x_i x_{i+2} - (x_i + x_{i+2})\hat{x} \\
\hat{x}(x_i - 2x_{i+1} + x_{i+2}) &\approx x_i x_{i+2} - x_{i+1}^2 \\
\hat{x} &\approx \frac{x_i x_{i+2} - x_{i+1}^2}{x_i - 2x_{i+1} + x_{i+2}}
\end{aligned}$$

$$\hat{x} \approx \frac{x_i x_{i+2} - 2x_i x_{i+1} + x_i^2 - x_{i+1}^2 + 2x_i x_{i+2} - x_i^2}{x_i - 2x_{i+1} + x_{i+2}} = x_i - \frac{(x_{i+1} - x_i)^2}{x_{i+2} - 2x_{i+1} + x_i}$$

This method is also called Δ^2-method.

$$\begin{aligned}
\Delta y_n &= y_{n+1} - y_n \\
\Delta^2 y_n &= \Delta(\Delta y_n) = \Delta y_{n+1} - \Delta y_n = (y_{n+2} - y_{n+1}) - (y_{n+1} - y_n) \\
&= y_{n+2} - 2y_{n+1} + y_n
\end{aligned}$$

$$\rightarrow \hat{x} \approx x_i - \frac{(\Delta x_i)^2}{\Delta^2 x_i}$$

Chapter 5

Interpolation

Often instead of a function f only individual values $f(x_i)$ and perhaps values of the derivatives $f'(x_i), f''(x_i)$ are available. This is for instance the case if differential equations are being solved numerically or with experimental data. However, for experimental data, one typically uses fitting and not interpolation) If one wants to obtain function values at intermediate position, integrate or simply get a smooth representation, one has to interpolate. That means one searches for a function φ that agrees with f, $f'(x), \dots$ at the nodes x_i:

$$\begin{aligned} f(x_i) &= \varphi(x_i) \\ f'(x_i) &= \varphi'(x_i) \end{aligned}$$

5.1 Polynomial interpolation

Simple problem (no derivatives): $y_i = f(x_i)$ with $i = 0, \dots, n$ given, looking for polynomial $P \in \mathbb{P}_n$ of degree $\leq n$ such that $P(x_i) = y_i$

P is unambiguous. To show this, we assume the negation:

$$P, Q \in \mathbb{P}_n \text{ such that } P(x_i) = Q(x_i) = y_i, \text{ for all } i = 0, \dots, n$$

Then $P - Q$ is a polynomial of degree $\leq n$ with roots at $x_0, x_1, \dots x_n$. But non-zero polynomials of degree n have at most exactly n roots. Hence, $P(x) - Q(x) = 0$ There are different methods for finding P, which can be associated with different bases of the space of polynomials.

5.1.1 Vandermonde-Matrix

Using the monomial basis $\{1, x, x^2, x^3, \dots, x^n\}$ $P(x_i) = y_i$ forms a linear system of equations:

$$\begin{pmatrix} 1 & x_0 & x_0^2 & \dots & x_0^n \\ 1 & x_1 & x_1^2 & \dots & x_1^n \\ \vdots & & & & \\ 1 & x_n & x_n^2 & \dots & x_n^2 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ \vdots \\ y_n \end{pmatrix}$$

The solution (inversion of U) is computationally expensive ($\propto n^3$) operations).

5.1.2 Lagrange polynomials

An alternative basis is given by the Lagrange-polynomials $L_0, L_1, \dots, L_n$ which implicitly are defined through $L_i(x_j) = \delta_{ij}$.
The explicit form is:

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

The interpolating polynomial P is obtained by superposition

$$\begin{aligned} P(x) &= \sum_{i=0}^{n} y_i \cdot L_i(x) \\ \rightarrow P(x_j) &= \sum_{i=0}^{n} y_i L_i(x_j) = \sum_{i=0}^{n} y_i \cdot \delta_{ij} = y_j \end{aligned}$$

$$\begin{aligned} P(x) &= y_0 L_0(x) + y_1 L_1(x) + y_2 L_2(x) + \dots \\ P(x_2) &= y_0 \underbrace{L_0(x_2)}_{0} + y_1 \underbrace{L_1(x_2)}_{0} + y_2 \underbrace{L_2(x_2)}_{1} + y_3 \underbrace{L_3(x_2)}_{0}, \dots \end{aligned}$$

Remark. The Lagrange polynomials form an orthonormal basis with respect to the inner product $\langle P, Q\rangle = \sum_{i=0}^{n} P(x_i) \cdot Q(x_i)$

$$\rightarrow \langle P, L_i\rangle = P(x_i)$$

$$P = \sum_{i=0}^{n} \langle P, L_i\rangle L_i = \sum_{i=0}^{n} P(x_i) L_i = \sum_{i=0}^{n} y_i L_i(x)$$

Example.

$$f(x) = \frac{1}{x}$$

i	x_i	y_i
0	2	$1/2$
1	2.5	$2/5$
2	4	$1/4$

$$\begin{aligned} L_0(x) &= \frac{(x-2.5)(x-4)}{(2-2.5)(2-4)} = (x-6.5)\cdot x + 10 \\ L_1(x) &= \frac{(x-2)(x-4)}{(2.5-2)(2.5-4)} = \frac{(-4x+24)x-32}{3} \\ L_2(x) &= \frac{(x-2)(x-2.5)}{(4-2)(4-2.5)} = \frac{(x-4.5)x+5}{3} \end{aligned}$$

$$P = \sum_{i=0}^{2} y_i \cdot L_i(x) = \frac{1}{2}((x-6.5)x+10) + \frac{2}{5}\left(\frac{(-4x+24)x-32}{3}\right) + \frac{1}{4}\left(\frac{(x-4.5)x+5}{3}\right) = (0.05x - 0.425)x + 1.15$$

5.1.3 Recursive determination of the interpolating polynomial / Neville's method

Let $P_{m_0,m_1,\ldots,m_k}(x)$ be the polynomial of degree k that interpolates $f(x)$ at the nodes $x_{m_0}, x_{m_1}, \ldots, x_{m_k}$

$$P(x) = \frac{(x_j - x)P_{0,1,\ldots,j-1,j+1,\ldots,n}(x) - (x_i - x)P_{0,1,\ldots,i-1,i+1,\ldots,n}(x)}{(x_j - x_i)}$$

is the polynomial of degree n that interpolates f at the nodes $x_o, \ldots, x_n$ (proof by plugging in).

Involves a lot of operations but is efficient if one is only interested in the value of the interpolating polynomial at one specific position $\hat{x}$. Then $P_{1,2}(\hat{x})$, $P_{0,1,\ldots,n-1}(\hat{x})$, ...are just numbers.

5.1.4 Newton's method

Newton basis: $N_0(x) = 1$, $N_1(x) = (x - x_0)$, $N_2(x) = (x - x_0)(x - x_1)$, ...

$$N_i(x) = \prod_{k=0}^{i-1}(x - x_k)$$

$$\rightarrow P_{0,1,\ldots,i-1}(x) + c_i \cdot N_i(x) = P_{0,1,\ldots,i}(x)$$
$$\text{since } N_i(x_0) = N_i(x_1) = \ldots = N_i(x_{i-1}) = 0$$

The problem becomes what is c_i? effective determination through divided differences($f[\ldots]$). Polynomial of degree 0:

$$\begin{aligned}
P_i(x) &= f[x_i] \coloneqq f_i = f(x_i) \\
P_{i,i+1} &= \frac{(x_i - x)P_{i+1} - (x_{i+1}P_i(x))}{x_i - x_{i+1}} = f[x_i] + (x - x_i)\underbrace{f[x_i, x_i + 1]}_{=c} \\
&= \frac{(x_i - x)f[x_{i+1}] - (x_{i+1} - x)f[x_i]}{x_i - x_i + 1} \\
f[x_i, x_{i+1}] &= \frac{(x_i - x)f[x_{i+1}] - (x_{i+1} - x)f[x_i] - (x_i - x_{i+1})f[x_i]}{(x_i - x_{i+1})(x - x_i)} \\
&= \frac{(x_i - x)f[x_{i+1}] - (x_i - x)f[x_i]}{(x_i - x_{i+1})(x - x_i)} = \frac{f[x_i] - f[x_{i+1}]}{x_i - x_{i+1}}
\end{aligned}$$

Let the $(n-1)$th divided difference $f[x_0, \ldots, x_{n-1}]$ and $f[x_i, \ldots, x_n]$ be known.

$$\begin{aligned}
P(x) = P_{0,\ldots,n}(x) &= P_{0,1,\ldots,n-1}(x) + f[x_0, \ldots, x_n](x - x_0)(x - x_1)\ldots(x - x_{n-1})(*) \\
P_{0,\ldots,n-1}(x) &= f[x_0] + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) + \ldots \\
&\quad + f[x_0, \ldots,]x_{n-1}(x - x_0)(x - x_1)\ldots(x - x_{n-2}) \\
P_{1,\ldots,n} &= f[x_1] + f[x_1, x_2](x - x_1) + f[x_1, x_2, x_3](x - x_1)(x - x_2) + \ldots \\
&\quad + f[x_1, x_2, \ldots, x_n](x - x_1)(x - x_2)\ldots(x - x_{n-1}) \\
P_{0,1,\ldots,n}(x) &= \frac{(x_0 - x)P_{1,\ldots,n}(x) - (x_n - x)P_{0,\ldots,n-1}}{x_0 - x_n}(**) \\
&\text{compare coefficients of } x^n \text{ in } (*) \text{ and } (**) \\
f[x_0, \ldots, x_n] &= \frac{f[x_0, \ldots, x_n - 1] - f[x_1, \ldots, x_n]}{x_0 - x_1}
\end{aligned}$$