1.2. Finite and divided differences

Finite differences

Let $M = \{a_i \mid a_i = a + ih, \text{ with } i = 0, ..., m;\ \ a, h \in \mathbb{R}^*,\ \ m \in \mathbb{N}^*\}$ and $\mathcal{F} = \{f \mid f : M \to \mathbb{R}\}$.

Definition 7 *For* $f \in \mathcal{F}$,

$$(\triangle_h f)(a_i) = f(a_{i+1}) - f(a_i), \qquad i < m$$

is called **the finite difference of the first order** *of the function* f*, with step* h*, at point* a_i*.*

Theorem 8 *The operator* $\triangle_h$ *is a linear operator with respect to* f*.*

Proof. If $f, g : M \to \mathbb{R}$; $A, B \in \mathbb{R}$ and $i < m$, we have

$$\begin{aligned}(\triangle_h(Af + Bg))(a_i) &= (Af + Bg)(a_{i+1}) - (Af + Bg)(a_i) \qquad (1)\\ &= A[f(a_{i+1}) - f(a_i)] + B[g(a_{\ i+1}) - g(a_i)]\\ &= A(\triangle_h f)(a_i) + B(\triangle_h g)(a_i).\end{aligned}$$

■

Definition 9 *Let* $0 \leq i < m,\ k \in \mathbb{N}$ *and* $1 \leq k \leq m - i$

$$\begin{aligned}(\triangle_h^k f)(a_i) &= (\triangle_h(\triangle_h^{k-1} f))(a_i) \\ &= (\triangle_h^{k-1} f)(a_{i+1}) - (\triangle_h^{k-1} f)(a_i), \quad \textit{with } \triangle_h^0 = I \textit{ și } \triangle_h^1 = \triangle_h\end{aligned} \tag{2}$$

is called **the k-th order finite difference** *of the function* f, *with step* h, *at point* a_i.

Theorem 10 *If* $0 \leq i < m;\ k, p \in \mathbb{N}$ *and* $1 \leq p + k \leq m - i$, *then*

$$(\triangle_h^p(\triangle_h^k f))(a_i) = \triangle_h^k(\triangle_h^p f)(a_i) = (\triangle_h^{p+k} f)(a_i). \tag{3}$$

Finite differences table: (f_i denotes $f(a_i)$)

a	f	$\triangle_h f$	$\triangle_h^2 f$	...	$\triangle_h^{m-1} f$	$\triangle_h^m f$
a_0	f_0	$\triangle_h f_0$	$\triangle_h^2 f_0$	...	$\triangle_h^{m-1} f_0$	$\triangle_h^m f_0$
a_1	f_1	$\triangle_h f_1$	$\triangle_h^2 f_1$	...	$\triangle_h^{m-1} f_1$	
	...					
a_{m-3}	f_{m-3}	$\triangle_h f_{m-3}$	$\triangle_h^2 f_{m-3}$			
a_{m-2}	f_{m-2}	$\triangle_h f_{m-2}$	$\triangle_h^2 f_{m-2}$			
a_{m-1}	f_{m-1}	$\triangle_h f_{m-1}$				
a_m	f_m					

where

$$\triangle_h^k f_i = \triangle_h^{k-1} f_{i+1} - \triangle_h^{k-1} f_i,\ \ k = 1, ..., m;\ \ i = 0, 1, ..., m-k.$$

Examples.

1. Considering $h = 0.25$, $a = 1$, $a_i = a + ih$, $i = \overline{0, 4}$, and $f_0 = 0$, $f_1 = 2$, $f_2 = 6$, $f_3 = 14$, $f_4 = 17$ form the finite differences table.

Sol.: We get:

a	f	$\triangle_h f$	$\triangle_h^2 f$	$\triangle_h^3 f$	$\triangle_h^4 f$
1	0	2	2	2	-11
1.25	2	4	4	-9	
1.50	6	8	-5		
1.75	14	3			
2	17				

2. For $f(x) = e^x$ find $(\triangle_h^k f)(a_i)$, with $a_i = a + ih$, $i \in \mathbb{N}$.

Divided differences

Let $X = \{x_i \mid x_i \in \mathbb{R},\ i = 0, 1, ..., m,\ m \in \mathbb{N}^*\}$ and $f : X \to \mathbb{R}$.

Definition 11 *For $r \in \mathbb{N}$, $r < m$,*

$$(\mathcal{D}f)(x_r) := [x_r, x_{r+1}; f] = \frac{f(x_{r+1}) - f(x_r)}{x_{r+1} - x_r}$$

is called **the first order divided difference** *of the function f, regarding the points x_r and x_{r+1}.*

Theorem 12 *The operator $\mathcal{D}$ is linear with respect to f.*

Proof.

$$\begin{aligned}(\mathcal{D}(\alpha f + \beta g))(x_r) &= \frac{(\alpha f + \beta g)(x_{r+1}) - (\alpha f + \beta g)(x_{r)}}{x_{r+1} - x_r} \qquad (4)\\ &= \alpha(\mathcal{D}f)(x_r) + \beta(\mathcal{D}g)(x_r), \qquad \text{for } \alpha, \beta \in \mathbb{R}.\end{aligned}$$

■

Definition 13 *Let* $r, k \in \mathbb{N}, 0 \le r < m$ *and* $1 \le k \le m - r,\ m \in \mathbb{N}^*$. *The quantity*

$$(\mathcal{D}^k f)(x_r) = \frac{(\mathcal{D}^{k-1} f)(x_{r+1}) - (\mathcal{D}^{k-1} f)(x_r)}{x_{r+k} - x_r}, \quad \text{with } \mathcal{D}^0 = 1,\ \mathcal{D}^1 = \mathcal{D}, \tag{5}$$

is called **the** k**-th order divided difference** *of the function* f, *at* x_r.

$(\mathcal{D}^k f)(x_r)$ is also denoted by $\left[x_r, ..., x_{r+k}; f\right]$. Relation (5) can be written as

$$\left[x_r, ..., x_{r+k}; f\right] = \frac{\left[x_{r+1}, ..., x_{r+k}; f\right] - \left[x_r, ..., x_{r+k-1}; f\right]}{x_{r+k} - x_r}. \tag{6}$$

Remark 14 *The operator* $\mathcal{D}^k$ *is linear with respect to* f.

For $r = 0$ and $k = m$ we have

$$(\mathcal{D}^m f)(x_0) = \sum_{i=0}^{m} \frac{f(x_i)}{(x_i - x_0)...|...(x_i - x_m)}. \tag{7}$$

Theorem 15 *If $f, g : X \to \mathbb{R}$ then*

$$[x_0, ..., x_m; fg] = \sum_{k=0}^{m} [x_0, ..., x_k; f][x_k, ..., x_m; g].$$

Proof. The proof follows by complete induction with respect to m. ■

Table of divided differences:

x	f	$\mathcal{D}f$	$\mathcal{D}^2 f$	...	$\mathcal{D}^{\mathbf{m-1}} f$	$\mathcal{D}^m f$
x_0	f_0	$\mathcal{D}f_0$	$\mathcal{D}^2 f_0$	...	$\mathcal{D}^{m-1} f_0$	$\mathcal{D}^m f_0$
x_1	f_1	$\mathcal{D}f_1$	$\mathcal{D}^2 f_1$		$\mathcal{D}^{m-1} f_1$	
x_2	f_2	$\mathcal{D}f_2$	$\mathcal{D}^2 f_2$			
...	...	...				
x_{m-2}	f_{m-2}	$\mathcal{D}f_{m-2}$	$\mathcal{D}^2 f_{m-2}$			
x_{m-1}	f_{m-1}	$\mathcal{D}f_{m-1}$				
x_m	f_m					

with $f_i = f(x_i), \quad i = 0, 1, ..., m.$

Example 16 *For* $x_0 = 0$, $x_1 = 1$, $x_2 = 2$, $x_3 = 4$ *and* $f_0 = 3$, $f_1 = 4$, $f_2 = 7$, $f_3 = 19$ *form the divided differences table.*

x	f	Df	D^2f	D^3f
0	3	1	1	0
1	4	3	1	
2	7	6		
4	19			

Example 17 *Form the divided differences table for* $x_0 = 2$, $x_1 = 4$, $x_2 = 6$, $x_3 = 8$ *and* $f_0 = 4$, $f_1 = 8$, $f_2 = 20$, $f_3 = 48$.

Example 18 *Form the divided differences table for* $x_0 = 1$, $x_1 = 2$, $x_2 = 3$, $x_3 = 5$, $x_4 = 7$ *and* $f_0 = 3$, $f_1 = 5$, $f_2 = 9$, $f_3 = 11$, $f_4 = 15$.

2.1. Taylor interpolation

Theorem 21 *(Taylor theorem) Let $f \in C^n[a,b]$, such that there exists $f^{(n+1)}$ on $[a,b]$ and consider $x_0 \in [a,b]$. The Taylor polynomial is*

$$T_n(x) = \sum_{k=0}^{n} \frac{(x-x_0)^k}{k!} f^{(k)}(x_0) \tag{8}$$

and we have the approximation formula

$$f(x) = T_n(x) + R_n(x),$$

R_n denoting the remainder (the error).

For $\forall x \in [a,b]$ there exists a number ξ between x_0 and x such that

$$R_n(x) = \frac{(x-x_0)^{n+1}}{(n+1)!} f^{(n+1)}(\xi).$$

Remark 22 *Taylor polynomials agree as closely as possible with a given function around the specific point x_0, but not on the entire interval.*

COURSE 2

2.2. Lagrange interpolation

Let $[a,b] \subset \mathbb{R},\ x_i \in [a,b],\ i = 0,1,...,m$ such that $x_i \neq x_j$ for $i \neq j$ and consider $f : [a,b] \to \mathbb{R}$.

The Lagrange interpolation problem (LIP) consists in determining the polynomial P of the smallest degree for which

$$P(x_i) = f(x_i),\ i = 0,1,...,m \tag{1}$$

i.e., the polynomial of the smallest degree which passes through the distinct points $(x_i, f(x_i))$, $i = 0,1,...,m$.

Definition 1 *A solution of (LIP) is called* **Lagrange interpolation polynomial**, *denoted by $L_m f$.*

Remark 2 *We have $(L_m f)(x_i) = f(x_i),\ i = 0,1,...,m$.*

$L_m f \in \mathbb{P}_m$ *($\mathbb{P}_m$ is the space of polynomials of at most m-th degree).*

The Lagrange interpolation polynomial is given by

$$(L_m f)(x) = \sum_{i=0}^{m} \ell_i(x) f(x_i), \tag{2}$$

where by $\ell_i(x)$ denote **the Lagrange fundamental interpolation polynomials.**

We have

$$u(x) = \prod_{j=0}^{m} (x - x_j),$$

$$u_i(x) = \frac{u(x)}{x - x_i} = (x - x_0)...(x - x_{i-1})(x - x_{i+1})...(x - x_m) = \prod_{\substack{j=0 \\ j \neq i}}^{m} (x - x_j)$$

and

$$\ell_i(x) = \frac{u_i(x)}{u_i(x_i)} = \frac{(x - x_0)...(x - x_{i-1})(x - x_{i+1})...(x - x_m)}{(x_i - x_0)...(x_i - x_{i-1})(x_i - x_{i+1})...(x_i - x_m)} = \prod_{\substack{j=0 \\ j \neq i}}^{m} \frac{x - x_j}{x_i - x_j}, \tag{3}$$

for $i = 0, 1, ..., m$.

How do we know that the interpolation polynomial expanded in powers of x (Course 1) and the polynomial constructed as in (2) represent the same polynomial?

Assume we have computed two interpolating polynomials $Q(x)$ and $P(x)$ each of degree m such that

$$Q(x_j) = f(x_j) = P(x_j), \quad j = 0, ..., m.$$

Then we can form the difference

$$d(x) = Q(x) - P(x),$$

that is a polynomial of degree less or equal to m.

Because of the interpolation property of P and Q, we have

$$d(x_j) = Q(x_j) - P(x_j) = 0, \quad j = 0, ..., m.$$

A non-zero polynomial of degree less than or equal to m cannot have more than m zeros. But d has $m+1$ distinct zeros, hence it must be identically zero, so $Q(x) = P(x)$.

Proposition 3 *We also have*

$$\ell_i(x) = \frac{u(x)}{(x - x_i)u'(x_i)},\ \ i = 0, 1, ..., m. \tag{4}$$

Proof. We have $u_i(x) = \frac{u(x)}{x - x_i}$, so $u(x) = u_i(x)(x - x_i)$. We get $u'(x) = u_i(x) + (x - x_i)u_i'(x)$, whence it follows $u'(x_i) = u_i(x_i)$. So, as

$$\ell_i(x) = \frac{u_i(x)}{u_i(x_i)}$$

we get

$$\ell_i(x) = \frac{u_i(x)}{u'(x_i)} = \frac{u(x)}{(x - x_i)u'(x_i)},\ \ i = 0, 1, ..., m. \tag{5}$$

■

Theorem 4 *The operator L_m is linear.*

Proof.

$$\begin{aligned} L_m(\alpha f + \beta g)(x) &= \sum_{i=0}^{m} \ell_i(x)(\alpha f + \beta g)(x_i) = \sum_{i=0}^{m} [\ell_i(x)\alpha f(x_i) + \ell_i(x)\beta g(x_i)] \\ &= \alpha(L_m f)(x) + \beta(L_m g)(x), \end{aligned}$$

so

$$L_m(\alpha f + \beta g) = \alpha L_m f + \beta L_m g, \qquad \forall f, g : [a, b] \to \mathbb{R} \text{ and } \alpha, \beta \in \mathbb{R}.$$

■

Example 5 *a) Consider the nodes x_0, x_1 and a function f to be interpolated.*

b) Find the Lagrange polynomial that interpolates the data in the following table and find the approximative value of $f(-0.5)$.

x	-1	0	3
$f(x)$	8	-2	4

Sol.

a) We have $m = 1$,

$$\begin{aligned} u(x) &= (x - x_0)(x - x_1) \\ u_0(x) &= x - x_1 \\ u_1(x) &= x - x_0 \end{aligned}$$

$$\begin{aligned} (L_1 f)(x) &= l_0(x) f(x_0) + l_1(x) f(x_1) \\ &= \frac{x - x_1}{x_0 - x_1} f(x_0) + \frac{x - x_0}{x_1 - x_0} f(x_1), \end{aligned}$$

which is the line passing through the given points $(x_0, f(x_0))$ and $(x_1, f(x_1))$.

b) We have $m = 2$. The Lagrange polynomial is

$$(L_2 f)(x) = l_0(x) f(x_0) + l_1(x) f(x_1) + l_2(x) f(x_2).$$

$u(x) = (x+1)(x-0)(x-3)$ and it follows

$$l_0(x) = \frac{(x-0)(x-3)}{(-1-0)(-1-3)} = \frac{1}{4}x(x-3)$$
$$l_1(x) = \frac{(x+1)(x-3)}{(0+1)(0-3)} = -\frac{1}{3}(x+1)(x-3)$$
$$l_2(x) = \frac{(x+1)(x-0)}{(3+1)(3-0)} = \frac{1}{12}x(x+1),$$

The polynomial is

$$(L_2f)(x) = 2x(x-3) + \frac{2}{3}(x+1)(x-3) + \frac{1}{3}x(x+1).$$

and $(L_2f)(-0.5) = 2.25$.

Remark 6 *Disadvantages of the form (2) of Lagrange polynomial: requires many computations and if we add or substract a point we have to start with a complete new set of computations.*

Remark 7 *Formula (6) needs half of the number of arithmetic operations needed for (2) and it is easier to add or substract a point.*

The Lagrange polynomial generates **the Lagrange interpolation formula**

$$f = L_m f + R_m f,$$

where $R_m f$ denotes **the remainder** (**the error**).

Theorem 8 *Let* $\alpha = \min\{x, x_0, ..., x_m\}$ *and* $\beta = \max\{x, x_0, ..., x_m\}$. *If* $f \in C^m[\alpha, \beta]$ *and* $f^{(m)}$ *is derivable on* (α, β) *then* $\forall x \in (\alpha, \beta)$, *there exists* $\xi \in (\alpha, \beta)$ *such that*

$$(R_m f)(x) = \frac{u(x)}{(m+1)!} f^{(m+1)}(\xi). \tag{7}$$

Proof. Consider

$$F(z) = \begin{vmatrix} u(z) & (R_m f)(z) \\ u(x) & (R_m f)(x) \end{vmatrix}.$$

whence $(R_m f)(x) = \frac{u(x)}{(m+1)!} f^{(m+1)}(\xi)$. ■

Corrolary 9 *If $f \in C^{m+1}[a,b]$ then*

$$|(R_m f)(x)| \leq \frac{|u(x)|}{(m+1)!} \left\| f^{(m+1)} \right\|_\infty, \qquad x \in [a,b]$$

where $\|\cdot\|_\infty$ denotes the uniform norm, and $\|f\|_\infty = \max\limits_{x\in[a,b]} |f(x)|$.

Example 10 *If we know that* $\lg 2 = 0.301$, $\lg 3 = 0.477$, $\lg 5 = 0.699$, *find* $\lg 76$. *Study the approximation error.*

Example 11 *Which is the limit of the error for computing $\sqrt{115}$ using Lagrange interpolation formula for the nodes $x_0 = 100$, $x_1 = 121$ and $x_2 = 144$? Find the approximative value of $\sqrt{115}$.*

COURSE 3

The Aitken's algorithm

Let $[a,b] \subset \mathbb{R},\ x_i \in [a,b],\ i = 0,1,...,m$ such that $x_i \neq x_j$ for $i \neq j$ and consider $f : [a,b] \to \mathbb{R}$.

Usually, for a practical approximation problem, for a given function $f : [a,b] \to \mathbb{R}$ we have to find the approximation of $f(\alpha)$, $\alpha \in [a,b]$ with an error not greater than a given $\varepsilon > 0$.

If we have enough information about f and its derivatives, we use the inequality $|(R_m f)(x)| \leq \varepsilon$ to find m such that $(L_m f)(\alpha)$ approximates $f(\alpha)$ with the given precision.

We may use the condition $\frac{|u(x)|}{(m+1)!} \left\| f^{(m+1)} \right\|_\infty \leq \varepsilon$, but it should be known $\left\| f^{(m+1)} \right\|_\infty$ or a majorant of it.

A practical method for computing the Lagrange polynomial is **the Aitken's algorithm.** This consists in generating the table:

$$\begin{array}{c|c|c|cccc}
x_0 & f_{00} & & & & & \\
x_1 & f_{10} & f_{11} & & & & \\
x_2 & f_{20} & f_{21} & f_{22} & & & \\
x_3 & f_{30} & f_{31} & f_{32} & f_{33} & & \\
\vdots & \vdots & \vdots & \vdots & \vdots & & \\
x_m & f_{m0} & f_{m1} & f_{m2} & f_{m3} & \cdots & f_{mm}
\end{array}$$

where

$$f_{i0} = f(x_i), \quad i = 0, 1, ..., m,$$

and

$$f_{i,j+1} = \frac{1}{x_i - x_j} \begin{vmatrix} f_{jj} & x_j - x \\ f_{ij} & x_i - x \end{vmatrix}, \qquad i = 0, 1, ..., m; j = 0, ..., i-1.$$

For example,

$$f_{11} = \frac{1}{x_1 - x_0} \left| \begin{array}{ll} f_{00} & x_0 - x \\ f_{10} & x_1 - x \end{array} \right|$$
$$= \frac{1}{x_1 - x_0} [f_{00}(x_1 - x) - f_{10}(x_0 - x)]$$
$$= \frac{x - x_1}{x_0 - x_1} f(x_0) + \frac{x - x_0}{x_1 - x_0} f(x_1) = (L_1 f)(x),$$

so f_{11} is the value in x of Lagrange polynomial for the nodes x_0, x_1. We have

$$f_{ii} = (L_i f)(x),$$

$L_i f$ being Lagrange polynomial for the nodes $x_0, x_1, ..., x_i$.

So $f_{11}, f_{22}, ..., f_{ii}, ..., f_{mm}$ is a sequence of approximations of $f(x)$.

If the interpolation procedure is convergent then the sequence is also convergent, i.e., $\lim_{m\to\infty} f_{mm} = f(x)$. By Cauchy convergence criterion it follows

$$\lim_{i\to\infty} |f_{ii} - f_{i-1,i-1}| = 0.$$

This could be used as a stopping criterion, i.e.,

$$\left|f_{ii} - f_{i-1,i-1}\right| \leq \varepsilon, \quad \text{for a given precision } \varepsilon > 0.$$

Recommendation is to sort the nodes $x_0, x_1, ..., x_m$ with respect to the distance to x, such that

$$|x_i - x| \leq \left|x_j - x\right| \text{ if } i < j, \quad i, \quad j = 1, ..., m.$$

Example 1 *Approximate* $\sqrt{115}$ *with precision* $\varepsilon = 10^{-3}$, *using Aitken's algorithm.*

Newton interpolation polynomial

A useful representation for Lagrange interpolation polynomial is

$$(L_m f)(x) := (N_m f)(x) = f(x_0) + \sum_{i=1}^{m} (x - x_0)...(x - x_{i-1})(D^i f)(x_0) \tag{1}$$

$$= f(x_0) + \sum_{i=1}^{m} (x - x_0)...(x - x_{i-1})[x_0, ..., x_i; f],$$

which is called **Newton interpolation polynomial;** where $(D^i f)(x_0)$(or denoted $[x_0, ..., x_i; f]$) is the i-th order divided difference of the function f at x_0, given by the table

	f	$\mathcal{D}f$	$\mathcal{D}^2 f$	...	$\mathcal{D}^{\mathbf{m-1}} f$	$\mathcal{D}^m f$
x_0	f_0	$\mathcal{D}f_0$	$\mathcal{D}^2 f_0$	...	$\mathcal{D}^{m-1} f_0$	$\mathcal{D}^m f_0$
x_1	f_1	$\mathcal{D}f_1$	$\mathcal{D}^2 f_1$		$\mathcal{D}^{m-1} f_1$	
x_2	f_2	$\mathcal{D}f_2$	$\mathcal{D}^2 f_2$			
...	...	...				
x_{m-2}	f_{m-2}	$\mathcal{D}f_{m-2}$	$\mathcal{D}^2 f_{m-2}$			
x_{m-1}	f_{m-1}	$\mathcal{D}f_{m-1}$				
x_m	f_m					

Newton interpolation formula is

$$f = N_m f + R_m f,$$

where $R_m f$ denotes the remainder.

Assume that we add the point $(x, f(x))$ at the top of the table of divided differences:

	f	Df	...	$D^{m+1}f$
x	$f(x)$	$(Df)(x) = [x, x_0; f]$		$[x, x_0, ..., x_m; f]$
x_0	$f(x_0)$	$(Df)(x_0) = [x_0, x_1; f]$	...	
x_1	$f(x_1)$	$(Df)(x_1) = [x_1, x_2; f]$		
...	...	...		
x_{m-1}	$f(x_{m-1})$	$(Df)(x_{m-1}) = [x_{m-1}, x_m; f]$		
x_m	$f(x_m)$			

For obtaining the interpolation polynomial we consider

$$[x, x_0; f] = \frac{f(x_0) - f(x)}{x_0 - x} \Longrightarrow f(x) = f(x_0) + (x - x_0)[x, x_0; f] \quad (2)$$

$$[x, x_0, x_1; f] = \frac{[x_0, x_1; f] - [x, x_0; f]}{x_1 - x} \quad (3)$$

$$\Longrightarrow [x, x_0; f] = [x_0, x_1; f] + (x - x_1)[x, x_0, x_1; f].$$

Inserting (3) in (2) we get

$$f(x) = f(x_0) + (x - x_0)[x_0, x_1; f] + (x - x_0)(x - x_1)[x, x_0, x_1; f].$$

If we continue eliminating the divided differences involving x in the same way, we get

$$f(x) = (N_m f)(x) + (R_m f)(x)$$

with

$$(N_m f)(x) = f(x_0) + \sum_{i=1}^{m} (x - x_0)...(x - x_{i-1})[x_0, ..., x_i; f]$$

and the remainder (the error) given by

$$(R_m f)(x) = (x - x_0)...(x - x_m)[x, x_0, ..., x_m; f]. \tag{4}$$

We notice that

$$(N_i f)(x) = (N_{i-1} f)(x) + (x - x_0)...(x - x_{i-1})[x_0, ..., x_i; f]$$

so the Newton polynomials of degree $2, 3, ...,$ can be iteratively generated, similarly to Aitken's algorithm.

Remark 2 *The remainder for Lagrange interpolation formula is also given by*

$$(R_m f)(x) = \frac{(x - x_0)...(x - x_m)}{(m+1)!} f^{(m+1)}(\xi),$$

with ξ *between* $x, x_0, ..., x_m$, *so, by (4), it follows that* **the divided differences are approximations of the derivatives**

$$[x, x_0, ..., x_m; f] = \frac{f^{(m+1)}(\xi)}{(m+1)!}.$$

Example 3 *Find* $L_2 f$ *for* $f(x) = \sin \pi x$, *and* $x_0 = 0, x_1 = \frac{1}{6}, x_2 = \frac{1}{2}$, *in both forms.*

Sol. a) We have $u(x) = x(x - \frac{1}{6})(x - \frac{1}{2})$; $u_0(x) = (x - \frac{1}{6})(x - \frac{1}{2})$; $u_1(x) = x(x - \frac{1}{2})$; $u_2(x) = x(x - \frac{1}{6})$

$$\begin{aligned}(L_2 f)(x) &= \sum_{i=0}^{2} l_i(x) f(x_i) = \sum_{i=0}^{2} \frac{u_i(x)}{u_i(x_i)} f(x_i) \\ &= \frac{(x - \frac{1}{6})(x - \frac{1}{2})}{(-\frac{1}{6})(-\frac{1}{2})} 0 + \frac{x(x - \frac{1}{2})}{\frac{1}{6}(-\frac{1}{3})} \frac{1}{2} + \frac{x(x - \frac{1}{6})}{\frac{1}{2} \cdot \frac{1}{3}} 1 \\ &= -3x^2 + \frac{7}{2}x.\end{aligned}$$

b)

$$
\begin{aligned}
(N_2f)(x) &= f(0) + \sum_{i=1}^{2}(x-x_0)...(x-x_{i-1})(D^i f)(x_0) \\
&= f(0) + (x-x_0)(Df)(x_0) + (x-x_0)(x-x_1)(D^2 f)(x_0) \\
&= x(Df)(x_0) + x(x-\frac{1}{6})(D^2 f)(x_0)
\end{aligned}
$$

The table of divided differences:

x	f	Df	D^2f
0	0	3	-3
$\frac{1}{6}$	$\frac{1}{2}$	$\frac{3}{2}$	
$\frac{1}{2}$	1		

so

$$(N_2f)(x) = 3x - 3x(x-\frac{1}{6}) = -3x^2 + \frac{7}{2}x.$$

- Useful if you have the Lagrange polynomial based on some set of data points $(x_i, f(x_i)), i = 0, 1, \ldots, n$, and you get a new data point, $(x_{n+1}, f(x_{n+1}))$.

Definition 4 *Let f be a function defined at x_0, $x_1, \ldots,$ x_n and suppose that m_1, $\ldots$, m_k are k distinct integers with $0 \leq m_i \leq n$, for every i. The Lagrange polynomial that interpolates $f(x)$ at the k points $x_{m_1}, \ldots,$ x_{m_k} is denoted by $P_{m_1,\ldots,m_k}(x)$.*

Example 5 *Consider $x_0 = 1$, $x_1 = 2$, $x_2 = 3$, $x_3 = 4$, $x_4 = 6$ and $f(x) = e^x$. Determine $P_{1,2,4}(x)$ and use it to approximate $f(5)$.*

$$\begin{aligned} P_{1,2,4}(x) &= \frac{(x-x_2)(x-x_4)}{(x_1-x_2)(x_1-x_4)}f(x_1) + \frac{(x-x_1)(x-x_4)}{(x_2-x_1)(x_2-x_4)}f(x_2) + \frac{(x-x_1)(x-x_2)}{(x_4-x_1)(x_4-x_1)}f(x_4) \\ &= \frac{(x-3)(x-6)}{(2-3)(2-6)}f(2) + \frac{(x-2)(x-6)}{(3-2)(3-6)}f(3) + \frac{(x-2)(x-3)}{(6-2)(6-3)}f(6) \end{aligned}$$

So, $f(5) \approx P_{1,2,4}(5)$, more specifically

$$\begin{aligned} P_{1,2,4}(5) &= \frac{(5-3)(5-6)}{(2-3)(2-6)}e^2 + \frac{(5-2)(5-6)}{(3-2)(3-6)}e^3 + \frac{(5-2)(5-3)}{(6-2)(6-3)}e^6 \\ &= -0.5e^2 + e^3 + 0.5e^6 \approx 218.105. \end{aligned}$$

The following results present a method for recursively generating Lagrange's polynomial approximation.

Theorem 6 *Let f be a function defined at the points x_0, $x_1, \ldots,$ x_k and let x_i and x_j be two distinct points in this set. Then*

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

is the kth Lagrange polynomial that approximates f at the $k+1$ nodes x_0, $x_1, \ldots,$ x_k.

Denote by $P_j(x) = f(x_j)$.

The previous Theorem implies that the interpolating polynomials can be generated recursively. For example

$$P_{0,1} = \frac{1}{x_1 - x_0}\left[(x - x_0)P_1 - (x - x_1)P_0\right]$$

$$P_{1,2} = \frac{1}{x_2 - x_1}\left[(x - x_1)P_2 - (x - x_2)P_1\right]$$

$$P_{0,1,2} = \frac{1}{x_2 - x_0}\left[(x - x_0)P_{1,2} - (x - x_2)P_{0,1}\right]$$

and so on. In the table below it can be seen how they are generated, taking into account the fact that each row is completed before the succeeding rows are begun.

x_0	P_0				
x_1	P_1	$P_{0,1}$			
x_2	P_2	$P_{1,2}$	$P_{0,1,2}$		
x_3	P_3	$P_{2,3}$	$P_{1,2,3}$	$P_{0,1,2,3}$	
x_4	P_4	$P_{3,4}$	$P_{2,3,4}$	$P_{1,2,3,4}$	$P_{0,1,2,3,4}$

This procedure is called the **Neville's method**.

- Going down in the table $\rightarrow$ using consecutive points x_i, with larger i

- Going to the right $\rightarrow$ increasing the degree of the interpolating polynomial.

- The points appear consecutively in each entry $\Longrightarrow$ we need to give only a starting point and the number of additional points used.

- Let $Q_{i,j}(x) = P_{i-j,i-j+1,\ldots,i-1,i}(x)$, with $0 \leq j \leq i$.

- The table becomes

x_0	$P_0 = Q_{0,0}$				
x_1	$P_1 = Q_{1,0}$	$P_{0,1} = Q_{1,1}$			
x_2	$P_2 = Q_{2,0}$	$P_{1,2} = Q_{2,1}$	$P_{0,1,2} = Q_{2,2}$		
x_3	$P_3 = Q_{3,0}$	$P_{2,3} = Q_{3,1}$	$P_{1,2,3} = Q_{3,2}$	$P_{0,1,2,3} = Q_{3,3}$	
x_4	$P_4 = Q_{4,0}$	$P_{3,4} = Q_{4,1}$	$P_{2,3,4} = Q_{4,2}$	$P_{1,2,3,4} = Q_{4,3}$	$P_{0,1,2,3,4} = Q_{4,4}$

with $Q_{i,j} = \frac{(x_i - x)Q_{i-1,j-1} + (x - x_{i-j})Q_{i,j-1}}{x_i - x_{i-j}}$, for $i = 1, 2, \ldots, n$; $j = 1, 2, \ldots, i$.

Neville's Iterated Interpolation Algorithm. Evaluate the polynomial P on $n+1$ given nodes, $x_0, \ldots, x_n$, at a given point x, for a function f.

Input : The nodes $x, x_0, x_1, \ldots, x_n$; the values of the function $f(x_0), \ldots, f(x_n$
as the first column of Q $(Q_{0,0}, Q_{1,0}, \ldots, Q_{n,0})$.

Output : the table Q with $P(x) = Q_{n,n}$.

```
Step 1 for $i = 1, 2, \ldots, n$
       for $j = 1, 2, \ldots, i$
       $Q_{i,j} = \frac{(x_i - x)Q_{i-1,j-1} + (x - x_{i-j})Q_{i,j-1}}{x_i - x_{i-j}}$.

Step 2 Output(Q); Stop.
```

It can be additionally added a stopping criterion

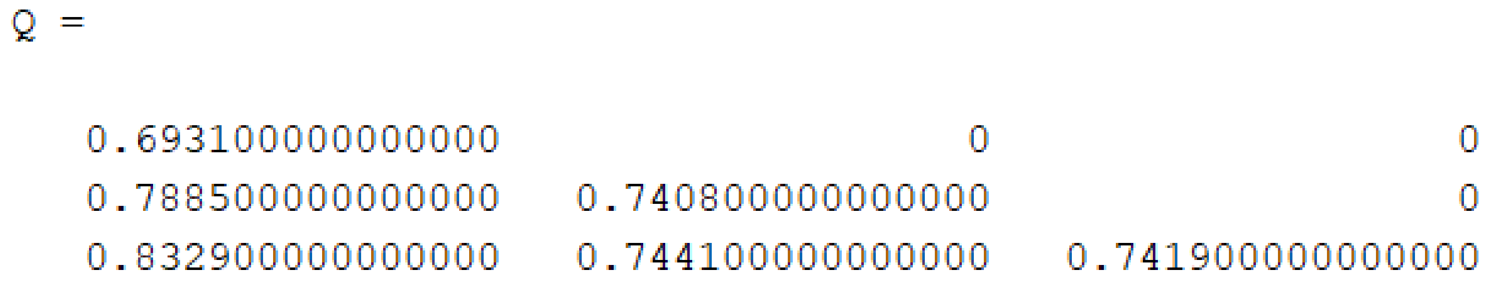
$$|Q_{i,j} - Q_{i-1,j-1}| < \varepsilon,$$

with ε a given error tolerance. If the inequality is not fulfilled, a new interpolation node x_{i+1} is added.

Example 7 *Approximate* ln(2.1) *using the function* $f(x) = \ln x$ *and the data*

x	*2*	*2.2*	*2.3*
$\ln x$	*0.6931*	*0.7885*	*0.8329*

$\ln(2.1) = f(2.1)$ *and we have obtained the table*



```
Q =

   0.693100000000000                   0                   0
   0.788500000000000   0.740800000000000                   0
   0.832900000000000   0.744100000000000   0.741900000000000
```

So, $\ln(2.1) \approx 0.7419$, with an error of $3.7344 \cdot 10^{-5}$.

Example 8 *Approximate* $\sqrt{115}$ *using Neville's algorithm, with 3 given nodes.*

COURSE 4

2.3. Hermite interpolation

Example 1 *In the following table there are some data regarding a moving car. We may estimate the position (and the speed) of the car when the time is $t = 10$ using Hermite interpolation.*

Time	0	3	5	8	13
Distance	0	225	383	623	993
Speed	75	77	80	74	72

Let $x_k \in [a,b],\ k = 0,1,...,m$ be such that $x_i \neq x_j$, for $i \neq j$ and let $r_k \in \mathbb{N},\ k = 0,1,...,m$. Consider $f : [a,b] \to \mathbb{R}$ such that there exist $f^{(j)}(x_k),\ k = 0,1,...,m;\ j = 0,1,...,r_k$ and $n = m + r_0 + ... + r_m$.

The Hermite interpolation problem (HIP) consists in determining the polynomial P of the smallest degree for which

$$P^{(j)}(x_k) = f^{(j)}(x_k), \quad k = 0,...,m;\ j = 0,...,r_k.$$

Definition 2 *A solution of (HIP) is called* **Hermite interpolation polynomial**, *denoted by $H_n f$.*

Hermite interpolation polynomial, $H_n f$, satisfies the interpolation conditions:

$$(H_n f)^{(j)}(x_k) = f^{(j)}(x_k), \quad k = 0, ..., m; \; j = 0, ..., r_k.$$

Hermite interpolation polynomial is given by

$$(H_n f)(x) = \sum_{k=0}^{m} \sum_{j=0}^{r_k} h_{kj}(x) f^{(j)}(x_k) \in \mathbb{P}_n, \tag{1}$$

where $h_{kj}(x)$ denote **the Hermite fundamental interpolation polynomials.** They fulfill the relations:

$h_{kj}^{(p)}(x_\nu) = 0, \; \nu \neq k, \quad p = 0, 1, ..., r_\nu$
$h_{kj}^{(p)}(x_k) = \delta_{jp}, \; p = 0, 1, ..., r_k, \quad$ for $j = 0, 1, ..., r_k$ and $\nu, k = 0, 1, ..., m$,

with $\delta_{jp} = \begin{cases} 1, & j = p \\ 0, & j \neq p. \end{cases}$

We denote by

$$u(x) = \prod_{k=0}^{m} (x - x_k)^{r_k+1} \qquad \text{and } u_k(x) = \frac{u(x)}{(x - x_k)^{r_k+1}}.$$

We have

$$h_{kj}(x) = \frac{(x - x_k)^j}{j!} u_k(x) \sum_{\upsilon=0}^{r_k-j} \frac{(x - x_k)^\nu}{\nu!} \left[\frac{1}{u_k(x)}\right]_{x=x_k}^{(\nu)}. \tag{2}$$

Example 3 *Find the Hermite interpolation polynomial for a function f for which we know $f(0) = 1, f'(0) = 2$ and $f(1) = -3$ (equivalent with $x_0 = 0$ multiple node of order 2 or double node, $x_1 = 1$ simple node).*

Sol. We have $x_0 = 0, x_1 = 1, m = 1, r_0 = 1, r_1 = 0, n = m + r_0 + r_1 = 2$

$$\begin{aligned}(H_2 f)(x) &= \sum_{k=0}^{1} \sum_{j=0}^{r_k} h_{kj}(x) f^{(j)}(x_k) \\ &= h_{00}(x) f(0) + h_{01}(x) f'(0) + h_{10}(x) f(1).\end{aligned}$$

We have h_{00}, h_{01}, h_{10}. These fulfills relations:

$h_{kj}^{(p)}(x_\nu) = 0,\ \nu \neq k, \quad p = 0, 1, ..., r_\nu$
$h_{kj}^{(p)}(x_k) = \delta_{jp},\ p = 0, 1, ..., r_k, \quad$ for $j = 0, 1, ..., r_k$ and $\nu, k = 0, 1, ..., m$.

We have $h_{00}(x) = a_1 x^2 + b_1 x + c_1 \in \mathbb{P}_2$, with $a_1, b_1, c_1 \in \mathbb{R}$, and the system

$$\begin{cases} h_{00}(x_0) = 1 \\ h'_{00}(x_0) = 0 \\ h_{00}(x_1) = 0 \end{cases} \Leftrightarrow \begin{cases} h_{00}(0) = 1 \\ h'_{00}(0) = 0 \\ h_{00}(1) = 0 \end{cases}$$

that becomes

$$\begin{cases} c_1 = 1 \\ b_1 = 0 \\ a_1 + b_1 + c_1 = 0. \end{cases}$$

Solution is: $a_1 = -1, b_1 = 0, c_1 = 1$ so $h_{00}(x) = -x^2 + 1$.

We have $h_{01}(x) = a_2 x^2 + b_2 x + c_2 \in \mathbb{P}_2$, with $a_2, b_2, c_2 \in \mathbb{R}$. The system is

$$\left\{\begin{array}{l} h_{01}(x_0) = 0 \\ h'_{01}(x_0) = 1 \\ h_{01}(x_1) = 0 \end{array}\right. \Leftrightarrow \left\{\begin{array}{l} h_{01}(0) = 0 \\ h'_{01}(0) = 1 \\ h_{01}(1) = 0 \end{array}\right.$$

and we get $h_{01}(x) = -x^2 + x$.

We have $h_{10}(x) = a_3 x^2 + b_3 x + c_3 \in \mathbb{P}_2$, with $a_3, b_3, c_3 \in \mathbb{R}$. The system is

$$\left\{\begin{array}{l} h_{10}(x_0) = 0 \\ h'_{10}(x_0) = 0 \\ h_{10}(x_1) = 1 \end{array}\right. \Leftrightarrow \left\{\begin{array}{l} h_{10}(0) = 0 \\ h'_{10}(0) = 0 \\ h_{10}(1) = 1 \end{array}\right.$$

and we get $h_{10}(x) = x^2$.

The Hermite polynomial is

$$(H_2 f)(x) = -x^2 + 1 - 2x^2 + 2x - 3x^2 = -6x^2 + 2x + 1.$$

The Hermite interpolation formula is

$$f = H_n f + R_n f,$$

where $R_n f$ denotes the remainder term (the error).

Theorem 4 *If $f \in C^n[\alpha, \beta]$ and $f^{(n)}$ is derivable on (α, β), with $\alpha =$* min$\{x, x_0, ..., x_m\}$ *and $\beta =$* max$\{x, x_0, ..., x_m\}$, *then there exists $\xi \in (\alpha, \beta)$ such that*

$$(R_n f)(x) = \frac{u(x)}{(n+1)!} f^{(n+1)}(\xi). \tag{3}$$

Proof. Consider

$$F(z) = \begin{vmatrix} u(z) & (R_n f)(z) \\ u(x) & (R_n f)(x) \end{vmatrix}.$$

$F \in C^n[\alpha, \beta]$ and there exists $F^{(n+1)}$ on (α, β).

We have

$$F(x) = 0, \quad F^{(j)}(x_k) = 0, \qquad k = 0, ..., m; \; j = 0, ..., r_k;$$

COURSE 5

Hermite interpolation with double nodes

Example 1 *In the following table there are some data regarding a moving car. We may estimate the position (and the speed) of the car when the time is $t = 10$ using Hermite interpolation.*

Time	0	3	5	8	13
Distance	0	225	383	623	993
Speed	75	77	80	74	72

Consider $f : [a, b] \to \mathbb{R}$, $x_0, x_1, ..., x_m \in [a, b]$

and the values $f(x_0), f(x_1), ..., f(x_m), f'(x_0), f'(x_1), ..., f'(x_m)$.

The Hermite interpolation polynomial with double nodes, H_{2m+1}, satisfies the interpolation properties:

$$H_{2m+1}(x_i) = f(x_i), \;\; i = \overline{0, m},$$
$$H'_{2m+1}(x_i) = f'(x_i), \;\; i = \overline{0, m}.$$

It is a polynomial of $n = 2m + 1$ degree.

For computation: use Lagrange polynomial written in Newton form, with divided differences table having each node x_i written twice.

Consider $z_0 = x_0$, $z_1 = x_0$, $z_2 = x_1$, $z_3 = x_1$, $\ldots$, $z_{2m} = x_m$, $z_{2m+1} = x_m$.

Form divided differences table: each node appear twice, in the first column write the values of f for each node twice; in the second column, at the odd positions put the values of the derivatives of f; the other elements are computed using the rule from divided differences.

We obtain the following table:

z_0	$f(z_0)$	$(\mathcal{D}^1 f)(z_0) = f'(x_0)$	$(\mathcal{D}^2 f)(z_0)$		$(\mathcal{D}^{2m} f)(z_0)$	$(\mathcal{D}^{2m+1} f)(z_0)$
z_1	$f(z_1)$	$(\mathcal{D}^1 f)(z_1)$	$\vdots$		$(\mathcal{D}^{2m} f)(z_1)$	
z_2	$f(z_2)$	$(\mathcal{D}^1 f)(z_2) = f'(x_1)$				
z_3	$f(z_3)$	$\vdots$				
$\vdots$	$\vdots$	$(\mathcal{D}^1 f)(z_{2m-1})$	$(\mathcal{D}^2 f)(z_{2m-1})$	$\ddots$		
z_{2m}	$f(z_{2m})$	$(\mathcal{D}^1 f)(z_{2m}) = f'(x_m)$		$\ldots$		
z_{2m+1}	$f(z_{2m+1})$			$\ldots$		

Newton interpolation polynomial for the nodes $x_0, ..., x_n$ is

$$(N_n f)(x) = f(x_0) + \sum_{i=1}^{n} (x - x_0)...(x - x_{i-1})(\mathcal{D}^i f)(x_0),$$

and similarly, Hermite interpolation polynomial is

$$(H_{2m+1} f)(x) = f(z_0) + \sum_{i=1}^{2m+1} (x - z_0)...(x - z_{i-1})(\mathcal{D}^i f)(z_0),$$

where $(\mathcal{D}^i f)(z_0)$, $i = 1, ..., 2m+1$ are the elements from the first line and columns $2, ..., 2m+1$.

Example 2 *Consider the double nodes* $x_0 = -1$ *and* $x_1 = 1$, *and* $f(-1) = -3, f'(-1) = 10, f(1) = 1, f'(1) = 2$. *Find the Hermite interpolation polynomial, that approximates the function* f, *in both forms, using the classical formula and using divided differences.*

Sol. We present here the method with divided differences. We have $m = 1, r_0 = r_1 = 1 \Rightarrow n = 3$

$z_0 = -1$	$f(-1) = -3$	$f'(-1) = 10$	$\frac{\frac{f(1)-f(-1)}{2} - f'(-1)}{z_2 - z_0} = -4$	$\frac{0-(-4)}{z_3 - z_0} = 2$
$z_1 = -1$	$f(-1) = -3$	$\frac{f(1)-f(-1)}{z_2 - z_1} = 2$	$\frac{f'(1) - \frac{f(1)-f(-1)}{2}}{z_3 - z_1} = 0$	
$z_2 = 1$	$f(1) = 1$	$f'(1) = 2$		
$z_3 = 1$	$f(1) = 1$			

The Hermite interpolation polynomial is

$$
\begin{aligned}
(H_3 f)(x) &= f(z_0) + \sum_{i=1}^{3} (x - z_0) \ldots (x - z_{i-1}) (\mathcal{D}^i f)(z_0) \\
&= f(z_0) + (x - z_0)(\mathcal{D}^1 f)(z_0) + (x - z_0)(x - z_1)(\mathcal{D}^2 f)(z_0) \\
&+ (x - z_0)(x - z_1)(x - z_2)(\mathcal{D}^3 f)(z_0)
\end{aligned}
$$

i.e.,

$$
\begin{aligned}
(H_3 f)(x) =& f(-1) + (x+1) f'(-1) + (x+1)^2 \tfrac{f(1) - f(-1) - 2f'(-1)}{4} \\
&+ (x+1)^2 (x-1) \tfrac{2f'(1) - f(1) + f(-1)}{4} \\
=& -3 + 10(x+1) - 4(x+1)^2 + 2(x+1)^2(x-1) \\
=& 2x^3 - 2x^2 + 1.
\end{aligned}
$$

2.4. Birkhoff interpolation

Let $x_k \in [a,b]$, $k = 0,1,...,m$, $x_i \neq x_j$ for $i \neq j, r_k \in \mathbb{N}$ and $I_k \subset \{0,1,...,r_k\}$, $k = 0,1,...,m$, $f : [a,b] \to \mathbb{R}$ s.t. $\exists f^{(j)}(x_k)$, $k = 0,...,m$, $j \in I_k$, and denote $n = |I_0| + ... + |I_m| - 1$, where $|I_k|$ is the cardinal of the set I_k.

The Birkhoff interpolation problem (BIP) consists in determining the polynomial P of the smallest degree such that

$$P^{(j)}(x_k) = f^{(j)}(x_k), \quad k = 0,...,m; \; j \in I_k.$$

Remark 4 *If $I_k = \{0,1,...,r_k\}$, $k = 0,...,m$, then (BIP) reduces to a (HIP). Birkhoff interpolation is also called* lacunary Hermite interpolation.

In order to check if (BIP) has an unique solution, we consider the polynomial $P(x) = a_n x^n + ... + a_0$ and the $(n+1) \times (n+1)$ linear system

$$P^{(j)}(x_k) = f^{(j)}(x_k), \quad k = 0,...,m; \; j \in I_k, \tag{1}$$

having as unknowns the coefficients of the polynomial. If the determinant of the system (1) is nonzero than (BIP) has an unique solution.

Definition 5 *A solution of (BIP), if exists, is called* **Birkhoff interpolation polynomial**, *denoted by* $B_n f$.

Birkhoff interpolation polynomial is given by

$$(B_n f)(x) = \sum_{k=0}^{m} \sum_{j \in I_k} b_{kj}(x) f^{(j)}(x_k), \tag{2}$$

where $b_{kj}(x)$ denote the Birkhoff fundamental interpolation polynomials. They fulfill relations:

$$\begin{aligned} & b_{kj}^{(p)}(x_\nu) = 0,\ \nu \neq k, \quad p \in I_\nu \\ & b_{kj}^{(p)}(x_k) = \delta_{jp},\ p \in I_k, \quad \text{for } j \in I_k \text{ and } \nu, k = 0, 1, ..., m, \end{aligned} \tag{3}$$

with $\delta_{jp} = \begin{cases} 1, & j = p \\ 0, & j \neq p. \end{cases}$

Remark 6 *Because of the gaps of the interpolation conditions, it is hard to find an explicit expression for* b_{kj}, $k = 0, ..., m;\ j \in I_k$. *They are found using relations (3).*

Birkhoff interpolation formula is

$$f = B_n f + R_n f,$$

where $R_n f$ denotes the remainder term.

Example 7 *Let* $f \in C^2[0,1]$, *the nodes* $x_0 = 0$, $x_1 = 1$ *and we suppose that we know* $f(0) = 1$ *and* $f'(1) = \frac{1}{2}$. *Find the corresponding interpolation formula.*

Sol. *We have* $m = 1, \ I_0 = \{0\}, \ I_1 = \{1\}$, *so* $n = 1 + 1 - 1 = 1$.

We check if there exists a solution of the problem.

Consider $P(x) = a_1 x + a_0 \in \mathbb{P}_1$ *and the system*

$$\begin{cases} P(0) = f(0) \\ P'(1) = f'(1) \end{cases} \Longleftrightarrow \begin{cases} a_0 = f(0) \\ a_1 = f'(1) \end{cases} .$$

The determinat of the system is

$$\begin{vmatrix} 0 & 1 \\ 1 & 0 \end{vmatrix} = -1 \neq 0,$$

so the problem has an unique solution.

The Birkhoff polynomial is

$$(B_1 f)(x) = b_{00}(x) f(0) + b_{11}(x) f'(1) \in \mathbb{P}_1.$$

We have $b_{00}(x) = ax + b \in \mathbb{P}_1$ *and*

$$\begin{cases} b_{00}(x_0) = 1 \\ b'_{00}(x_1) = 0 \end{cases} \Longleftrightarrow \begin{cases} b_{00}(0) = 1 \\ b'_{00}(1) = 0 \end{cases} \Leftrightarrow \begin{cases} b = 1 \\ a = 0 \end{cases} ,$$

whence

$$b_{00}(x) = 1.$$

For $b_{11}(x) = cx + d \in \mathbb{P}_1$ *we have*

$$\begin{cases} b_{11}(x_0) = 0 \\ b'_{11}(x_1) = 1 \end{cases} \Longleftrightarrow \begin{cases} b_{11}(0) = 0 \\ b'_{11}(1) = 1 \end{cases} \Leftrightarrow \begin{cases} d = 0 \\ c = 1 \end{cases}$$

whence

$$b_{11}(x) = x.$$

So,

$$(B_1 f)(x) = f(0) + x f'(1) = 1 + \frac{1}{2}x.$$

Example 8 *Considering* $f'(0) = 1$, $f(1) = 2$ *and* $f'(2) = 1$. *Find the approximative value of* $f(\frac{1}{2})$.

- spline interpolation: piecewise polynomials that require no specific derivative information, except perhaps at the endpoints of the interval.

Definition 9 *The piecewise-polynomial approximation that uses cubic spline polynomials between each successive pair of nodes is called* **cubic spline interpolation.**

(The word "spline" was used to refer to a long flexible strip, generally of metal, that could be used to draw continuous smooth curves by forcing the strip to pass through specified points and tracing along the curve.)

Definition 10 *Let $f : [a, b] \to \mathbb{R}$ and the nodes $a = x_0 < x_1 < ... < x_n = b$, a* **cubic spline interpolant** *S for f is the function that satisfies the following conditions:*

(a) $S(x)$ *is a cubic polynomial, denoted* $S_j(x)$ *on the subinterval* $[x_j, x_{j+1}]$, $\forall j = 0, 1, ..., n-1$, *i.e.,*

$$S(x) = \begin{cases} S_0(x), & x \in [x_0, x_1] \\ S_1(x), & x \in [x_1, x_2] \\ \quad ... & \\ S_{n-1}(x), & x \in [x_{n-1}, x_n] \end{cases}$$

(b) $S_j(x_j) = f(x_j)$ *and* $S_j(x_{j+1}) = f(x_{j+1})$, $\forall j = 0, 1, ..., n-1$;

(c) $S_j(x_{j+1}) = S_{j+1}(x_{j+1})$, $\forall j = 0, 1, ..., n-2$;

(d) $S_j^{'}(x_{j+1}) = S_{j+1}^{'}(x_{j+1})$, $\forall j = 0, 1, ..., n-2$;

(e) $S_j^{''}(x_{j+1}) = S_{j+1}^{''}(x_{j+1})$, $\forall j = 0, 1, ..., n-2$;

(f) *One of the following boundary conditions is satisfied:*

(i) $S''(x_0) = S''(x_n) = 0$ *(*$\Longleftrightarrow S_0''(x_0) = S_{n-1}''(x_n) = 0$ *natural (or free) boundary)* **natural spline***;*

(ii) $S^{'}(x_0) = f'(x_0)$ *and* $S'(x_n) = f'(x_n)$ *(*$\Longleftrightarrow S_0^{'}(x_0) = f'(x_0)$ *and* $S_{n-1}'(x_n) = f'(x_n)$ *clamped boundary)* **clamped spline***;*

(iii) $S_0(x) = S_1(x)$ *and* $S_{n-2} = S_{n-1}$ *(***de Boor spline***).*

Remark 11 *A cubic spline function defined on an interval divided into* n *subintervals will require determining* $4n$ *constants.*

We have the following expression of a cubic spline:

$$S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3, \quad \forall j = 0, 1, ..., n-1. \quad (4)$$

Theorem 12 *If* f *is defined at* $a = x_0 < x_1 < ... < x_n = b$*, then* f *has an unique natural spline interpolant* S *on the nodes* x_0*,* $x_1, ..., x_n$*; that satisfies the natural boundary conditions* $S''(a) = 0$ *and* $S''(b) = 0$*.*

Example 17 *Construct* **a natural cubic spline** *that passes through the points* $(1,2)$, $(2,3)$ *and* $(3,5)$.

Sol. (Sketch of the solution) *We follow Definition 10:*

Here $S(x)$ *consists of two cubic splines,* $S_j(x)$ *on the subinterval* $[x_j, x_{j+1}]$, $\forall j = 0, 1$, *i.e.,*

$$S(x) = \begin{cases} S_0(x), & x \in [x_0, x_1] \\ S_1(x), & x \in [x_1, x_2] \end{cases}$$

given by (4),

$$S_0(x) = a_0 + b_0(x-1) + c_0(x-1)^2 + d_0(x-1)^3,$$
$$S_1(x) = a_1 + b_1(x-2) + c_1(x-2)^2 + d_1(x-2)^3.$$

There are 8 constants $(a_i, b_i, c_i, d_i,\ i = 0, 1)$ *to be determined, which requires 8 conditions, that come from (b),(c),(d),(e),(i).*

$$\begin{cases} s_0(x_0) = f(x_0) \\ s_0(x_1) = f(x_1) \\ s_1(x_1) = f(x_1) \\ s_1(x_2) = f(x_2) \\ s_0'(x_1) = s_1'(x_1) \\ s_0''(x_1) = s_1''(x_1) \\ s_0''(x_0) = 0 \\ s_1''(x_2) = 0 \end{cases}$$

Example 18 *Construct* **a clamped spline** S *that passes through the points* $(1,2)$, $(2,3)$ *and* $(3,5)$ *and that has* $S'(1) = 2$ *and* $S'(3) = 1$.

General case. Solution of the least squares problem is

$$\varphi(x) = \sum_{i=1}^{n} a_i g_i(x),$$

where $\{g_i,\ i = 1, ..., n\}$ is a basis of the space and the coefficients a_i are obtained solving **the normal equations**:

$$\sum_{i=1}^{n} a_i \langle g_i, g_k \rangle = \langle f, g_k \rangle, \qquad k = 1, ..., n.$$

In the discrete case

$$\langle f, g \rangle = \sum_{k=0}^{m} w(x_k)\, f(x_k)\, g(x_k)$$

and in the continuous case

$$\langle f, g \rangle = \int_a^b w(x)\, f(x)\, g(x),$$

where w is a weight function.

3. Numerical integration of functions

The need: for evaluating definite integrals of functions that has no explicit antiderivatives or whose antiderivatives are not easy to obtain.

Let $f : [a, b] \to \mathbb{R}$ be an integrable function, x_k, $k = 0, ..., m$, distinct nodes from $[a, b]$.

Definition 4 *A formula of the form*

$$\int_a^b f(x)dx = \sum_{k=0}^{m} A_k f(x_k) + R(f),$$

is **a numerical integration formula** *or* **a quadrature formula***.*

A_k - **the coefficients**; $x_k -$**the nodes**; $R(f)$ - **the remainder (the error)**.

COURSE 7

3.1. Interpolatory quadrature formulas

Definition 1 *A quadrature formula*

$$\int_a^b f(x)dx = \sum_{k=0}^{m} A_k f(x_k) + R(f),$$

is **an interpolatory quadrature formula** *if it is obtained by integrating each member of an interpolation formula regarding the function f and the nodes x_k.*

Remark 2 *An interpolatory quadrature formula has its degree of exactness at least the degree of the corresponding interpolation polynomial.*

Definition 3 *The quadrature formulas with equidistant nodes, $x_k = a + kh$, $h = \frac{b-a}{m}$, are called* **Newton-Cotes formulas.**

have that there exist $\xi \in (a,b)$ such that

$$\int_a^b f(x)dx = \frac{b-a}{2}[f(a)+f(b)] + \frac{f''(\xi)}{2}\left[\frac{x^3}{3} - \frac{(a+b)x^2}{2} + abx\right]\Bigg|_a^b$$

We obtain **the trapezium's quadrature formula**

$$\int_a^b f(x)dx = \frac{b-a}{2}[f(a)+f(b)] + R_1(f), \tag{1}$$

where the remainder (the error) is

$$R_1(f) = -\frac{(b-a)^3}{12} f''(\xi), \quad \xi \in (a,b).$$

This formula is called the trapezium's formula because the integral is approximated by the area of a trapezium.

Remark 4 *The error from (1) involves f'', so the rule gives exact result when is applied to function whose second derivative is zero (polynomial of first degree or less). So its degree of exactness is 1.*

Example 5 *Approximate the integral $\int_1^3 (2x+1)dx$ using the trapezium's formula.*

(*Remark.* The result is the exact value of the integral because $f(x) = 2x+1$ is a linear function and the degree of exactness of the trapezium's formula is 1.)

For $m = 2$ ($(x_0 = a, x_1 = a + \frac{b-a}{2}, x_2 = b, h = \frac{b-a}{2}$) one obtains **the Simpson's quadrature formula**

$$\int_a^b f(x)dx = \frac{b-a}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right] + R_2(f), \tag{2}$$

where

$$R_2(f) = -\frac{(b-a)^5}{2880} f^{(4)}(\xi),\ a \le \xi \le b. \tag{3}$$

Example 6 *Approximate the integral $\int_1^3 (2x+1)dx$ using Simpson's formula.*

Remark 7 *The error from (2) involves $f^{(4)}$, so the rule gives exact result when is applied to any polynomial of third degree or less. So degree of exactness of Simpson's formula is 3.*

Remark 8 *A Newton-Cotes quadrature formula has degree of exactness equal to* $\begin{cases} m, & \text{if } m \text{ is an odd number} \\ m+1, & \text{if } m \text{ is an even number.} \end{cases}$

Remark 9 *The coefficients of the Newton-Cotes quadrature formulas have the symmetry property:*

$$A_i = A_{m-i}, i = 0, ..., m.$$

For $m = 3$, **Newton's formula**

$$\int_a^b f(x)dx = \frac{b-a}{8}\left[f(a) + 3f\left(\frac{2a+b}{3}\right) + 3f\left(\frac{a+2b}{3}\right) + f(b)\right] + R_3(f),$$

with

$$R_3(f) = -\frac{(b-a)^5}{648} f^{(4)}(\xi).$$

Example 10 *Compare the trapezium's rule and Simpson's rule approximations for*

$$\int_0^2 x^2 dx.$$

Sol. *The exact value is* 2.667*; for trapezium rule the value is* 4*, for Simpson's rule the value is* 2.667*. (The approximation from Simpson's rule is exact because the error involves* $f^{(4)}(x) = 0$*.)*

An efficient way of constructing a practical quadrature formula: repeated application of a simple formula.

Let $x_k = a + kh,\ k = 0, ..., n$ with $h = \frac{b-a}{n}$, be the nodes of a uniform grid of $[a, b]$. By the additivity property of the integral we have

$$\int_a^b f(x)dx = \sum_{k=1}^{n} I_k, \text{ with } I_k = \int_{x_{k-1}}^{x_k} f(x)dx$$

Applying a quadrature formula to I_k, one obtains **the repeated quadrature formula.**

Applying to each integral I_k the trapezium's formula, we get

$$\int_a^b f(x)dx = \sum_{k=1}^{n} \left\{ \frac{x_k - x_{k-1}}{2} \left[f(x_{k-1}) + f(x_k) \right] - \frac{(x_k - x_{k-1})^3}{12} f''(\xi_k) \right\},$$

where $x_{k-1} \leq \xi_k \leq x_k$, or

$$\int_a^b f(x)dx = \frac{b-a}{2n} \left[f(a) + f(b) + 2 \sum_{k=1}^{n-1} f(x_k) \right] + R_n(f), \qquad (4)$$

with

$$R_n(f) = -\frac{(b-a)^3}{12n^3} \sum_{k=1}^{n} f''(\xi_k).$$

There exists $\xi \in (a, b)$ such that

$$\frac{1}{n} \sum_{k=1}^{n} f''(\xi_k) = f''(\xi).$$

So **the repeated trapezium's quadrature formula** is

$$\int_a^b f(x)dx = \frac{b-a}{2n}\left[f(a) + f(b) + 2\sum_{k=1}^{n-1} f(x_k)\right] + R_n(f), \tag{5}$$

with

$$R_n(f) = -\frac{(b-a)^3}{12n^2} f''(\xi),\ a < \xi < b \tag{6}$$

We have

$$|R_n(f)| \le \frac{(b-a)^3}{12n^2} M_2 f,$$

where $M_2 f = \max\limits_{a \le x \le b} |f''(x)|$. By

$$|R_n(f)| \le \frac{(b-a)^3}{12n^2} M_2 f, \tag{7}$$

it follows that the repeated trapezium quadrature formula allows the approx. of an integral with arbitrary small given error, if n is taken sufficiently large. If we want that the absolute error to be smaller than ε, we determine the smallest solution n of the inequation

$$\frac{(b-a)^3}{12n^2} M_2 f < \varepsilon, \ \ n \in \mathbb{N},$$

and using this value in (4), leads to desired approximation.

Similarly, there is obtained **the repeated Simpson's quadrature formula**

$$\int_a^b f(x)dx = \tfrac{b-a}{6n}\left[f(a) + f(b) + 4\sum_{k=1}^{n} f\left(\tfrac{x_{k-1}+x_k}{2}\right) + 2\sum_{k=1}^{n-1} f(x_k)\right] + R_n(f) \tag{8}$$

where

$$R_n(f) = -\frac{(b-a)^5}{2880n^4} f^{(4)}(\xi),\ a < \xi < b,$$

and

$$|R_n(f)| \leq \frac{(b-a)^5}{2880n^4} M_4 f.$$

Example 12 *1. Approximate the integral* $\int_1^3 (2x+1)dx$ *with repeated trapezium's formula for* $n = 2$.

2. Approximate $\frac{\pi}{4}$ *with repeated trapezium's formula, considering precision* $\varepsilon = 10^{-2}$.

Sol. 1. *Remark.* The result is the exact value of the integral because $f(x) = 2x+1$ is a linear function and the degree of exactness of the trapezium's formula is 1.

2. We have

$$\frac{\pi}{4} = arctg(1) = \int_0^1 \frac{dx}{1+x^2},$$

so $f(x) = \frac{1}{1+x^2}$. Using (7), we get

$$|R_n(f)| \leq \frac{(1-0)^3}{12n^2} M_2 f.$$

We have

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

$$f''(x) = \frac{6x^2-2}{(1+x^2)^3}$$

and

$$M_2 f = \max_{x\in[0,1]} |f''(x)| = 2,$$

so

$$|R_n(f)| \leq \frac{1}{6n^2} < 10^{-2} \Rightarrow n^2 > \frac{10^2}{6} = 16.66 \Rightarrow n = 5.$$

We have $x_0 = 0, x_1 = \frac{1}{5}, x_2 = \frac{2}{5}, x_3 = \frac{3}{5}, x_4 = \frac{4}{5}, x_5 = 1$ $(h = \frac{1}{5})$. The integral will be

$$\int_a^b f(x)dx \approx \frac{1}{10}\left\{f(0) + f(1) + 2\left[f(\frac{1}{5}) + f(\frac{2}{5}) + f(\frac{3}{5}) + f(\frac{4}{5})\right]\right\} = 0.7837.$$

3.5. Quadrature formulas of Gauss type

All the previous rules can be written in the form

$$\int_a^b f(x)dx = \sum_{k=1}^{m} A_k f(x_k) + R_m(f), \tag{9}$$

where the coefficients A_k, $k = 1, ..., m$, do not depend on the function f. We have picked the nodes x_k, $k = 1, ..., m$ equispaced and have then calculated the coefficients A_k, $k = 1, ..., m$. This guarantees that the rule is exact for polynomials of degree $\leq m$.

It is possible to make such a rule exact for polynomials of degree $\leq 2m - 1$, by choosing also the nodes appropriately. This is the basic idea of the gaussian rules.

Let $f : [a, b] \to \mathbb{R}$ be an integrable function and $w : [a, b] \to \mathbb{R}_+$ a weight function, integrable on $[a, b]$.

Definition 2 *A formula of the following form*

$$\int_a^b w(x)f(x)dx = \sum_{k=1}^{m} A_k f(x_k) + R_m(f) \tag{10}$$

is called **a quadrature formula of Gauss type** *or* **with maximum degree of exactness** *if the coefficients A_k and the nodes x_k, $k = 1, ..., m$ are determined such that the formula has the maximum degree of exactness.*

Remark 3 *The coefficients and the nodes are determined such that to minimize the error, to produce exact results for the largest class of polynomials.*

A_k and x_k, $k = 1, ..., m$ from (10) are $2m$ unknown parameters $\Rightarrow$ $2m$ equations obtained such that the formula (10) is exact for any polynomial degree at most $2m - 1$.

It is often possible to rewrite the integral $\int_a^b g(x)dx$ as $\int_a^b w(x)f(x)dx$, where $w(x)$ is a nonnegative integrable function, and $f(x) = \frac{g(x)}{w(x)}$ is smooth, or it is possible to consider the simple choice $w(x) = 1$.

For the general case, consider the elementary polynomials $e_k(x) = x^k$; $k = 0, ..., 2m-1$ and obtain the system s.t. $R_m(e_k) = 0$:

$$\begin{cases} \sum\limits_{k=1}^{m} A_k e_0(x_k) = \int_a^b w(x) e_0\,(x)\,dx \\ \sum\limits_{k=1}^{m} A_k e_1(x_k) = \int_a^b w(x) e_1\,(x)\,dx \\ ... \\ \sum\limits_{k=1}^{m} A_k e_{2m-1}(x_k) = \int_a^b w(x) e_{2m-1}\,(x)\,dx \end{cases}$$

$\Longleftrightarrow$

$$\begin{cases} A_1 + A_2 + ... + A_m = \mu_0 \\ A_1 x_1 + A_2 x_2 + ... + A_m x_m = \mu_1 \\ ... \\ A_1 x_1^{2m-1} + A_2 x_2^{2m-1} + ... + A_m x_m^{2m-1} = \mu_{2m-1} \end{cases} \tag{11}$$

with

$$\mu_k = \int_a^b w(x) x^k dx.$$

As, the system (11) is difficult to solve, there have been found other ways to find the unknown parameters.

If $w(x) = 1$ (this case was studied by Gauss), then the nodes are the roots of Legendre orthogonal polynomial

$$u(x) = \frac{m!}{(2m)!}\left[(x-a)^m(x-b)^m\right]^{(m)}$$

and for finding the coefficients we use the first m equations from the system (11).

Consider $m = 1$ and obtain the following Gauss type quadrature formula

$$\int_a^b f(x)dx = A_1 f(x_1) + R_1(f).$$

The system (11) becomes

$$\begin{cases} A_1 = \int_a^b dx = b - a \\ A_1 x_1 = \int_a^b x dx = \frac{b^2 - a^2}{2}. \end{cases}$$

The unique solution of this system is $A_1 = b - a$, $x_1 = \frac{a+b}{2}$.

Example 4 *The same result is obtained considering* x_1 *the root of the Legendre polynomial of the first degree,*

$$u(x) = \frac{1}{2}[(x-a)(x-b)]' = x - \frac{a+b}{2}.$$

The Gauss type quadrature formula with one node is

$$\int_a^b f(x)dx = (b-a)f\left(\frac{a+b}{2}\right) + R_1(f),$$

with

$$R_1(f) = \frac{(b-a)^3}{24} f''(\xi), \qquad \xi \in [a,b]$$

which is called **the rectangle quadrature rule** (also called **the mid-point rule).**

The repeated rectangle (midpoint) quadrature formula is

$$\int_a^b f(x)dx = \frac{b-a}{n}\sum_{i=1}^{n} f(x_i) + R_n(f),$$

$$R_n(f) = \frac{(b-a)^3}{24n^2} f''(\xi), \qquad \xi \in [a,b]$$

with $x_1 = a + \frac{b-a}{2n}$, $x_i = x_1 + (i-1)\frac{b-a}{n}$, $i = 2, ..., n$.

We have

$$|R_n(f)| \le \frac{(b-a)^3}{24n^2} M_2 f, \quad \text{with } M_2 f = \max_{x\in[a,b]} |f''(x)|.$$

Remark 5 *Another* **rectangle rule** *is the following*:

$$\int_a^b f(x)dx = (b-a)f(a) + R(f),$$

with

$$R(f) = \frac{(b-a)^2}{2} f'(\xi), \qquad \xi \in [a,b].$$

Example 6 *Approximate* $\ln 2 = \int_1^2 \frac{1}{x}dx$, *with* $\varepsilon = 10^{-2}$, *using the repeated rectangle (midpoint) method.*

Solution. We have

$$\int_a^b f(x)dx = \frac{b-a}{n}\sum_{i=1}^{n} f(x_i) + R_n(f),$$

$$R_n(f) = \frac{(b-a)^3}{24n^2}f''(\xi), \qquad \xi \in [a,b].$$

$$\ln 2 = \int_1^2 \frac{dx}{x},$$

so $f(x) = \frac{1}{x}$ and we get

$$\ln 2 = \frac{b-a}{n}\left[f(a+\frac{b-a}{2n}) + \sum_{i=2}^{n} f(a+\frac{b-a}{2n}+(i-1)\frac{b-a}{n})\right] + \frac{(b-a)^3}{24n^2}f''(\xi$$

We have $f(x) = \frac{1}{x}$, $f'(x) = -\frac{1}{x^2}$, $f''(x) = \frac{2}{x^3}$, and $|f''(\xi)| \le 2$, for $\xi \in [1,2]$ so it follows

$$|R_n(f)| \le \frac{1}{24n^2}2 < 10^{-2} \Rightarrow 12n^2 > 100 \Rightarrow n = 3.$$

Therefore,

$$\ln 2 \approx \frac{1}{3}\left(\frac{1}{1+\frac{1}{6}}+\frac{1}{1+\frac{1}{6}+\frac{1}{3}}+\frac{1}{1+\frac{1}{6}+\frac{2}{3}}\right)=\frac{1}{3}\left(\frac{6}{7}+\frac{6}{9}+\frac{6}{11}\right)=0.6897$$

(real value is 0.693...)

Example 7 *For* $m=2$, *Gauss quadrature formula is*

$$\int_a^b f(x)dx = A_1 f(x_1) + A_2 f(x_2) + R_2(f).$$

Find A_1, A_2, x_1, x_2.

Sol. The corresponding Legendre polynomial is

$$\begin{aligned} u(x) &= \frac{2}{4!}\left[(x-a)^2(x-b)^2\right]'' \\ &= x^2-(a+b)x+\frac{1}{6}(a^2+b^2+4ab), \end{aligned}$$

with the roots

$$x_1 = \frac{a+b}{2} - \frac{(b-a)\sqrt{3}}{6},$$
$$x_2 = \frac{a+b}{2} + \frac{(b-a)\sqrt{3}}{6}.$$

For finding A_1 and A_2 we use the first two equations:

$$\begin{cases} A_1 + A_2 = b - a \\ A_1 x_1 + A_2 x_2 = (b^2 - a^2)/2. \end{cases}$$

We get

$$A_1 = A_2 = (b-a)/2,$$

so the quadrature formula of Gauss type with two nodes is

$$\int_a^b f(x)dx = \frac{b-a}{2}\left[f\left(\frac{a+b}{2} - \frac{b-a}{6}\sqrt{3}\right) + f\left(\frac{a+b}{2} + \frac{b-a}{6}\sqrt{3}\right)\right] + R_2(f).$$

For the interval $[-1,1]$ we get $A_1 = A_2 = 1$ and $x_1 = -\frac{\sqrt{3}}{3}, x_2 = \frac{\sqrt{3}}{3}$, which gives the fomula

$$\int_{-1}^{1} f(x)dx \simeq f(-\frac{\sqrt{3}}{3}) + f(\frac{\sqrt{3}}{3}).$$

This formula has degree of precision 3, i.e., it gives exact result for every polynomial of the 3−rd degree or less.

Remark 8 *The resulting rules look more complicated than the interpolatory rules. Both nodes and weights for gaussian rules are, in general, irrational numbers. But, on a computer, it usually makes no difference whether one evaluates a function at* $x = 3$ *or at* $x = 1/\sqrt{3}$*. Once the nodes and weights of such a rule are stored, these rules are as easily used as the trapezium rule or Simpson's rule. At the same time, these gaussian rules are usually much more accurate when compared with the last ones on the basis of number of function values used.*

Remark 9 *a) The coefficients* A_k, $k = 1, ..., m$ *of a Gauss type formula are positive.*

b) The coefficients A_k, $k = 1, ..., m$ and the roots of the Legendre polynomials can be found in tables, for $a = -1$, $b = 1$. For example, for $m = 2$ and $m = 3$:

m	*nodes*	*coefficients*
2	0.577	1
	−0.577	1
3	0.774	0.555
	0	0.888
	−0.774	0.555

c) For different weight functions, tables are available for both the nodes and the coefficients.

Example 10 *Approximate $\int_{-1}^{1} e^x \cos x dx$ using a Gauss type quadrature formula for $m = 3$.*

Sol.

$$\int_{-1}^{1} e^x \cos x dx \simeq 0.55e^{0.77} \cos 0.77 + 0.88 \cos 0 + 0.55e^{-0.77} \cos(-0.77)$$
$$= 1.9333904$$

(Exact value is 1.9334214.) Absolute error is $<3.2 \cdot 10^{-5}$.

The integral $\int_a^b f(x)dx$ for an arbitrary interval $[a,b]$ could be transfomed in an integral on $[-1,1]$ using the change of variable

$$t = \frac{2x-a-b}{b-a} \Leftrightarrow x = \frac{1}{2}\left[(b-a)t + a + b\right].$$

The Gauss type quadrature formulas may be applied on the following way:

$$\int_a^b f(x)dx = \int_{-1}^{1} f\left(\frac{(b-a)t+(b+a)}{2}\right)\frac{(b-a)}{2}dt. \qquad (13)$$

Example 11 *Consider the integral* $\int_1^3 (x^6 - x^2 \sin(2x))dx = 317.3442466$.

a) Compare the result obtained for Newton-Cotes type formula with $m = 1$ *(trapezium formula) and Gauss-Legendre formula for* $m = 2$;

b) Compare the result obtained for Newton-Cotes type formula with $m = 2$ *(Simpson formula) and Gauss-Legendre formula for* $m = 3$.

Sol. a) Each formula needs 2 evaluations of the function $f(x) = x^6 - x^2 \sin(2x)$. We have

$$\text{Trapezium formula } (m = 1): \frac{2}{2}[f(1) + f(3)] = 731.6054420;$$

and a Gauss type formula for $m = 2$, using (13):

$$\int_1^3 (x^6 - x^2 \sin(2x))dx = \int_{-1}^1 ((t+2)^6 - (t+2)^2 \sin(2(t+2)))dt$$
$$\simeq f(-0.577 + 2) + f(0.577 + 2) = 306.8199344.$$

b) Each formula needs 3 evaluations of the function. We have

$$\text{F. Simpson } (m = 2): \frac{1}{3}[f(1) + 4f(2) + f(3)] = 333.23;$$

and a Gauss type formula for $m = 3$, using (13):

$$\int_1^3 (x^6 - x^2 \sin(2x))dx = \int_{-1}^1 ((t+2)^6 - (t+2)^2 \sin(2(t+2)))dt$$
$$\simeq 0.55f(-0.77 + 2) + 0.88f(2) + 0.55f(0.77 + 2)$$
$$= 317.2641516$$

3.6. General quadrature formulas

Using the interpolation formulas, there are obtained a large variety of quadrature formulas.

In the case of some concrete applications, the choosing of the quadrature formula is made according to the information about the function f.

A general quadrature formula is given by:

$$\int_a^b f(x)dx = \sum_{k=0}^{m} \sum_{j \in I_k} A_{kj} f^{(j)}(x_k) + R(f).$$

For example, consider the Hermite interpolation formula for $f : [a, b] \to \mathbb{R}$, the nodes $x_k \in [a, b]$, $k = 0, ..., m$ multiple of orders $r_0, ..., r_m \in \mathbb{N}$,

$$f = H_n f + R_n f, \tag{14}$$

with $\dot{n} = m + r_0 + ... + r_m$, and

$$(H_n f)(x) = \sum_{k=0}^{m} \sum_{j=0}^{r_k} h_{kj} f^{(j)}(x_k),$$

h_{jk} being the Hermite fundamental polynomials.

Formula (14) generates the quadrature formula with multiple nodes:

$$\int_a^b f(x)dx = \sum_{k=0}^{m} \sum_{j=0}^{r_k} A_{kj} f^{(j)}(x_k) + R_n(f),$$

with

$$A_{kj} = \int_a^b h_{kj}(x)dx.$$

Similarly, there are obtained quadrature formulas starting to Birkhoff interpolation formulas.

4.2.1. Gauss method for solving linear systems

Consider the linear system $Ax = b$, i.e.,

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}. \tag{4}$$

The method consists of two stages:

- reducing the system (4) to an equivalent one, $Ux = d$, with U an upper triangular matrix.

- solving of the upper triangular linear system $Ux = d$ by backward substitution.

At least one of the elements on the first column is nonzero, otherwise A is singular. We choose one of these nonzero elements (using some criterion) and this will be called the first elimination **pivot.**

If the case, we change the line of the pivot with the first line, both in A and in b, and next we successively make zeros under the first pivot:

$$\begin{pmatrix} a_{11}^1 & a_{12}^1 & \dots & a_{1n}^1 \\ 0 & a_{22}^1 & \dots & a_{2n}^1 \\ \vdots & \vdots & & \vdots \\ 0 & a_{n2}^1 & \dots & a_{nn}^1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1^1 \\ b_2^1 \\ \vdots \\ b_n^1 \end{pmatrix}.$$

Analogously, after k steps we obtain the system

$$\begin{pmatrix} a_{11}^1 & a_{12}^1 & \dots & a_{1k}^1 & a_{1,k+1}^1 & \dots & a_{1n}^1 \\ 0 & a_{22}^2 & \dots & a_{2k}^2 & a_{2,k+1}^2 & \dots & a_{2n}^2 \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & a_{kk}^k & a_{k,k+1}^k & \dots & a_{kn}^k \\ 0 & 0 & \dots & 0 & a_{k+1,k+1}^k & \dots & a_{k+1,n}^k \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & 0 & a_{n,k+1}^k & \dots & a_{nn}^k \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \\ x_{k+1} \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1^1 \\ b_2^2 \\ \vdots \\ b_k^k \\ b_{k+1}^k \\ \vdots \\ b_n^k \end{pmatrix}.$$

If $a_{kk}^k \neq 0$, denote $m_{ik} = \frac{a_{ik}^k}{a_{kk}^k}$ and we get

$$a_{ij}^{k+1} = a_{ij}^k - m_{ik} a_{kj}^k, \quad j = k, ..., n$$
$$b_i^{k+1} = b_i^k - m_{ik} b_k^k, \quad i = k+1, ..., n.$$

After $n-1$ steps we obtain the system

$$\begin{pmatrix} a_{11}^1 & a_{12}^1 & ... & a_{1n}^1 \\ 0 & a_{22}^2 & ... & a_{2n}^2 \\ 0 & 0 & ... & a_{3n}^3 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & ... & a_{nn}^{n-1} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1^1 \\ b_2^2 \\ b_3^3 \\ \vdots \\ b_n^{n-1} \end{pmatrix}.$$

Remark 6 *The total number of elementary operations is of order* $\frac{2}{3}n^3$.

Example 7 *Consider the system*

$$\begin{pmatrix} 0.0001 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}.$$

Gauss algorithm yields: $m_{21} = \frac{a_{21}}{a_{11}} = \frac{1}{0.0001}$

$$\left(\begin{array}{cc} 0.0001 & 1 \\ 1 - 0.0001 * m_{21} = 0 & 1 - 1 * m_{21} = -9999 \end{array} \right) \left(\begin{array}{c} x \\ y \end{array} \right)$$
$$= \left(\begin{array}{c} 1 \\ 2 - 1 * m_{21} = -9998 \end{array} \right)$$

$$\Rightarrow y = \frac{9998}{9999} = 0.(9998) \approx 1.$$

Replacing in the first equation we get

$$x = 1.000(1000) \approx 1.$$

By division with a pivot of small absolute value there could be induced errors. For avoiding this there are two ways:

A) Partial pivoting: finding an index $p \in \{k, ..., n\}$ such that:

$$\left| a_{p,k}^{k} \right| = \max_{i=\overline{k,n}} \left| a_{i,k}^{k} \right|.$$

B) Total pivoting: finding $p, q \in \{k, ..., n\}$ such that:

$$\left|a_{p,q}^k\right| = \max_{i,j=\overline{k,n}} \left|a_{ij}^k\right|,$$

Example 8 *Solve the following system of equations using partial pivoting:*

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 3 & 1 & 5 \\ -1 & 1 & -5 & 3 \\ 3 & 1 & 7 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 10 \\ 31 \\ -2 \\ 18 \end{bmatrix}.$$

The pivot is a_{41}. We interchange the $1-$st line and the $4-$th line. We have

$$\begin{bmatrix} 3 & 1 & 7 & -2 \\ 2 & 3 & 1 & 5 \\ -1 & 1 & -5 & 3 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 18 \\ 31 \\ -2 \\ 10 \end{bmatrix},$$

then

$$\begin{array}{r} \textit{pivot element} \rightarrow \\ m_{21} = \frac{2}{3} \\ m_{31} = -\frac{1}{3} \\ m_{41} = \frac{1}{3} \end{array} \left[\begin{array}{ccccc} \mathbf{3} & 1 & 7 & -2 & | \; 18 \\ 0 & 2.33 & -3.66 & 6.33 & | \; 19 \\ 0 & 1.33 & -2.66 & 2.33 & | \; 4 \\ 0 & 0.66 & -1.33 & 1.66 & | \; 4 \end{array}\right].$$

Subtracting multiplies of the first equation from the three others gives

$$\begin{array}{r} \\ \textit{pivot element} \rightarrow \\ m_{32} = \frac{1.33}{2.33} \\ m_{42} = \frac{0.66}{2.33} \end{array} \left[\begin{array}{ccccc} 3 & 1 & 7 & -2 & | \; 18 \\ 0 & \mathbf{2.33} & -3.66 & 6.33 & | \; 19 \\ 0 & 1.33 & -2.66 & 2.33 & | \; 4 \\ 0 & 0.66 & -1.33 & 1.66 & | \; 4 \end{array}\right].$$

Subtracting multiplies, of the second equation from the last two equations, gives

$$\begin{array}{r} \\ \\ \textit{pivot element} \rightarrow \\ m_{43} = \frac{0.28}{0.57} \end{array} \left[\begin{array}{ccccc} 3 & 1 & 7 & -2 & | \; 18 \\ 0 & 2.33 & -3.66 & 6.33 & | \; 19 \\ 0 & 0 & \mathbf{-0.57} & -1.28 & | \; -6.85 \\ 0 & 0 & -0.28 & -0.14 & | \; -1.42 \end{array}\right].$$

Subtracting multiplies, of the third equation form the last one, gives the upper triangular system

$$\left[\begin{array}{cccc|c} 3 & 1 & 7 & -2 & 18 \\ 0 & 2.33 & -3.66 & 6.33 & 19 \\ 0 & 0 & -\mathbf{0.57} & -1.28 & -6.85 \\ 0 & 0 & 0 & 0.5 & 2 \end{array}\right].$$

The process of the back substitution algorithm applied to the triangular system produces the solution

$$\begin{aligned} x_4 &= \frac{2}{0.5} = 4 \\ x_3 &= \frac{-6.85 + 1.28x_4}{-0.57} = 3 \\ x_2 &= \frac{19 + 3.66x_3 - 6.33x_4}{2.33} = 2 \\ x_1 &= \frac{18 - x_2 - 7x_3 + 2x_4}{3} = 1. \end{aligned}$$

Example 9 *Solve the system:*

$$\begin{cases} 2x + y = 3 \\ 3x - 2y = 1 \end{cases}$$

Sol.

$$\begin{cases} 2x + y = 3 \\ 3x - 2y = 1 \end{cases}$$

The extended matrix is

$$\left[\begin{array}{ccc|c} 2 & 1 & & 3 \\ 3 & -2 & & 1 \end{array}\right]$$

and the pivot is 3. We interchange the lines:

$$\left[\begin{array}{ccc|c} 3 & -2 & & 1 \\ 2 & 1 & & 3 \end{array}\right]$$

We have $L_2 - \frac{2}{3}L_1 \to L_2$ and obtain

$$\left[\begin{array}{ccc|c} 3 & -2 & & 1 \\ 0 & \frac{7}{3} & & \frac{7}{3} \end{array}\right]$$

so the system becames

$$\begin{cases} 3x - 2y = 1 \\ \frac{7}{3}y = \frac{7}{3} \end{cases} .$$

Solution is

$$\begin{cases} x = 1 \\ y = 1 \end{cases} .$$

Example 10 *Solve the following system using Gauss elimination method:*

$$\begin{cases} x_1 + x_2 + x_3 = 4 \\ 2x_1 - 2x_2 + 3x_3 = 5 \\ x_1 - x_2 + 4x_3 = 5. \end{cases}$$

4.2.2. **Gauss-Jordan method** ("total elimination" method)

Consider the linear system $Ax = b$, i.e.,

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}. \tag{5}$$

We make transformations, like in Gauss elimination method, to make zeroes in the lines $i+1, i+2, ..., n$ and then, also in the lines $1, 2, ..., i-1$ such that the system to be reducing to:

$$\begin{pmatrix} a_{11}^1 & 0 & \dots & 0 \\ 0 & a_{22}^2 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \vdots & \vdots & & \vdots \\ 0 & 0 & \dots & a_{nn}^n \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1^1 \\ b_2^2 \\ b_3^3 \\ \vdots \\ b_n^n \end{pmatrix}.$$

The solution is obtained by

$$x_i = \frac{b_i^i}{a_{ii}^i}, \quad i = 1, ..., n.$$

Definition 11 *A $n \times n$ matrix A is* **strictly diagonally dominant** *if*

$$|a_{ii}| > \sum_{j=1, j \neq i}^{n} \left|a_{ij}\right|, \text{ for } \; i = 1, 2, ..., n.$$

Theorem 12 *If A is a strictly diagonally dominant matrix, then A is nonsingular and moreover, Gaussian elimination can be performed on any linear system $Ax = b$ without rows or columns interchanges, and the computations are stable with respect to the growth of rounding errors.*

COURSE 10

4.2.3. Factorization methods - LU methods

The matrix A can be factored into the product of a lower triangular matrix L and an upper triangular matrix U, namely $A = LU$.

$$Ax = b \Longleftrightarrow LUx = b,$$

where

$$L = \begin{pmatrix} l_{11} & 0 & \dots & 0 \\ l_{21} & l_{22} & \dots & 0 \\ \vdots & & & \\ l_{n1} & l_{n2} & \dots & l_{nn} \end{pmatrix} \qquad U = \begin{pmatrix} u_{11} & u_{12} & \dots & u_{1n} \\ 0 & u_{22} & \dots & u_{2n} \\ \vdots & & & \\ 0 & 0 & \dots & u_{nn} \end{pmatrix}.$$

We solve the systems in two stages:

First stage: Solve $Lz = b$,

Second stage: Solve $Ux = z$.

Methods for computing matrices L and U : **Doolittle method** where all diagonal elements of L have to be 1; **Crout method** where all diagonal elements of U have to be 1 and **Choleski method** where $l_{ii} = u_{ii}$ for $i = 1, ..., n$.

Remark 1 *LU factorizations are modified forms of Gauss elimination method.*

Doolittle method

We consider $a_{kk} \neq 0$, $k = \overline{1, n-1}$. Denote

$$l_{i,k} := \frac{a_{i,k}^{(k-1)}}{a_{k,k}^{(k-1)}}, \quad i = \overline{k+1, n}$$

$$t^{(k)} = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ l_{k+1,k} \\ \cdots \\ l_{n,k} \end{bmatrix},$$

having zeros for the first k-th lines, and

$$M_k = I_n - t^{(k)} e_k \in \mathcal{M}_{n\times n}(\mathbb{R}) \tag{1}$$

where $e_k = \begin{pmatrix} 0 & ... & 1 & ... & 0 \end{pmatrix}$ is the k-unit vector of dimension n,(has 1 on the k-th position) and $I_n = \begin{pmatrix} 1 & 0 & ... & 0 \\ 0 & 1 & ... & 0 \\ ... & & & \\ 0 & 0 & ... & 1 \end{pmatrix}$ is the identity matrix of order n.

$a_{i,k}^{(0)}$ are elements of A; $a_{i,k}^{(1)}$ are elements of $M_1 \cdot A$; ...; $a_{i,k}^{(k-1)}$ are elements of $M_{k-1} \ldots \cdot M_1 \cdot A$.

Definition 2 *The matrix M_k is called* **the Gauss matrix**, *the components $l_{i,k}$ are called* **the Gauss multiplies** *and the vector $t^{(k)}$ is* **the Gauss vector**.

Remark 3 *If $A \in \mathcal{M}_{n\times n}(\mathbb{R})$, then the Gauss matrices $M_1, \ldots, M_{n-1}$ can be determined such that*

$$U = M_{n-1} \cdot M_{n-2} \ldots M_2 \cdot M_1 \cdot A$$

is an upper triangular matrix. Moreover, if we choose

$$L = M_1^{-1} \cdot M_2^{-1} \dots M_{n-1}^{-1}$$

then

$$A = L \cdot U.$$

Example 4 *Find LU factorization for the matrix*

$$A = \begin{pmatrix} 2 & 1 \\ 6 & 8 \end{pmatrix}.$$

Solve the system $\begin{cases} 2x_1 + x_2 = 3 \\ 6x_1 + 8x_2 = 9 \end{cases}$.

Sol.

$$M_1 = I_2 - t^{(1)} e_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \begin{pmatrix} 0 \\ \frac{6}{2} \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix}$$

$$= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 3 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -3 & 1 \end{pmatrix}.$$

We have

$$U = M_1 A = \begin{pmatrix} 1 & 0 \\ -3 & 1 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 6 & 8 \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 0 & 5 \end{pmatrix}$$

$$L = M_1^{-1} = \begin{pmatrix} 1 & 0 \\ 3 & 1 \end{pmatrix}.$$

So

$$A = \begin{pmatrix} 2 & 1 \\ 6 & 8 \end{pmatrix} = L \cdot U = \begin{pmatrix} 1 & 0 \\ 3 & 1 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 0 & 5 \end{pmatrix}.$$

We have

$$L \cdot U \cdot x = \begin{pmatrix} 3 \\ 9 \end{pmatrix}$$

$$Ux = z$$

and

$$\begin{pmatrix} 1 & 0 \\ 3 & 1 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} 3 \\ 9 \end{pmatrix} \Rightarrow z = \begin{pmatrix} 3 \\ 0 \end{pmatrix}$$
$$\begin{pmatrix} 2 & 1 \\ 0 & 5 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 3 \\ 0 \end{pmatrix} \Rightarrow x = \begin{pmatrix} 1.5 \\ 0 \end{pmatrix}.$$

4.3. Iterative methods for solving linear systems

Because of round-off errors, direct methods become less efficient than iterative methods for large systems ($>$100 000 variables).

An iterative scheme for linear systems consists of converting the system

$$Ax = b \tag{2}$$

to the form

$$x = \tilde{b} - Bx.$$

After an initial guess for $x^{(0)}$, the sequence of approximations of the solution $x^{(0)}, x^{(1)}, ..., x^{(k)}, ...$is generated by computing

$$x^{(k)} = \tilde{b} - Bx^{(k-1)}, \quad \text{for } k = 1, 2, 3, ...$$

4.3.1. Jacobi iterative method

Consider the $n \times n$ linear system,

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_n = b_2 \\ \qquad\qquad ... \\ a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_n = b_n, \end{cases}$$

where we assume that the diagonal terms a_{11}, a_{22}, ..., a_{nn} are all nonzero.

We begin our iterative scheme by solving each equation for one of the variables:

$$\begin{cases} x_1 = u_{12}x_2 + ... + u_{1n}x_n + c_1 \\ x_2 = u_{21}x_1 + ... + u_{2n}x_n + c_2 \\ ... \\ x_n = u_{n1}x_1 + ... + u_{nn-1}x_{n-1} + c_n, \end{cases}$$

where $u_{ij} = -\frac{a_{ij}}{a_{ii}}$, $c_i = \frac{b_i}{a_{ii}}$, $i = 1, ..., n$.

Let $x^{(0)} = (x_1^{(0)}, x_2^{(0)}, ..., x_n^{(0)})$ be an initial approximation of the solution. The $k+1$-th approximation is:

$$\begin{cases} x_1^{(k+1)} = u_{12}x_2^{(k)} + ... + u_{1n}x_n^{(k)} + c_1 \\ x_2^{(k+1)} = u_{21}x_1^{(k)} + u_{23}x_3^{(k)} + ... + u_{2n}x_n^{(k)} + c_2 \\ ... \\ x_n^{(k+1)} = u_{n1}x_1^{(k)} + ... + u_{nn-1}x_{n-1}^{(k)} + c_n, \end{cases}$$

for $k = 0, 1, 2, ...$

An algorithmic form:

$$x_i^{(k)} = \frac{b_i - \sum\limits_{j=1, j\neq i}^{n} a_{ij}x_j^{(k-1)}}{a_{ii}}, \; i = 1, 2, ..., n, \text{ for } k \geq 1.$$

The iterative process is terminated when a convergence criterion is satisfied.

Stopping criterions: $\left|x^{(k)} - x^{(k-1)}\right| < \varepsilon$ or $\frac{\left|x^{(k)} - x^{(k-1)}\right|}{\left|x^{(k)}\right|} < \varepsilon$, with $\varepsilon > 0$ - a prescribed tolerance.

Example 5 *Solve the following system using the Jacobi iterative method. Use $\varepsilon = 10^{-3}$ and $x^{(0)} = (0\ 0\ 0\ 0)$ as the starting vector.*

$$\begin{cases} 7x_1 - 2x_2 + x_3 = 17 \\ x_1 - 9x_2 + 3x_3 - x_4 = 13 \\ 2x_1 + 10x_3 + x_4 = 15 \\ x_1 - x_2 + x_3 + 6x_4 = 10. \end{cases}$$

These equations can be rearranged to give

$$\begin{aligned} x_1 &= (17 + 2x_2 - x_3)/7 \\ x_2 &= (-13 + x_1 + 3x_3 - x_4)/9 \\ x_3 &= (15 - 2x_1 - x_4)/10 \\ x_4 &= (10 - x_1 + x_2 - x_3)/6 \end{aligned}$$

and, for example,

$$\begin{aligned} x_1^{(1)} &= (17 + 2x_2^{(0)} - x_3^{(0)})/7 \\ x_2^{(1)} &= (-13 + x_1^{(0)} + 3x_3^{(0)} - x_4^{(0)})/9 \\ x_3^{(1)} &= (15 - 2x_1^{(0)} - x_4^{(0)})/10 \\ x_4^{(1)} &= (10 - x_1^{(0)} + x_2^{(0)} - x_3^{(0)})/6. \end{aligned}$$

Substitute $x^{(0)} = (0, 0, 0, 0)$ into the right-hand side of each of these equations to get

$$\begin{aligned}
x_1^{(1)} &= (17 + 2 \cdot 0 - 0)/7 = 2.428\ 571\ 429 \\
x_2^{(1)} &= (-13 + 0 + 3 \cdot 0 - 0)/9 = -1.444\ 444\ 444 \\
x_3^{(1)} &= (15 - 2 \cdot 0 - 0)/10 = 1.5 \\
x_4^{(1)} &= (10 - 0 + 0 - 0)/6 = 1.666\ 666\ 667
\end{aligned}$$

and so $x^{(1)} = (2.428\ 571\ 429, -1.444\ 444\ 444, 1.5, 1.666\ 666\ 667)$.
The Jacobi iterative process:

$$\begin{aligned}
x_1^{(k+1)} &= \left(17 + 2x_2^{(k)} - x_3^{(k)}\right)/7 \\
x_2^{(k+1)} &= \left(-13 + x_1^{(k)} + 3x_3^{(k)} - x_4^{(k)}\right)/9 \\
x_3^{(k+1)} &= \left(15 - 2x_1^{(k)} - x_4^{(k)}\right)/10 \\
x_4^{(k+1)} &= \left(10 - x_1^{(k)} + x_2^{(k)} - x_3^{(k)}\right)/6, \qquad k \geq 1.
\end{aligned}$$

We obtain a sequence that converges to

$$\mathbf{x}^{(9)} = (2.000127203, -1.000100162, 1.000118096, 1.000162172).$$

4.3.2. Gauss-Seidel iterative method

Almost the same as Jacobi method, except that each x-value is improved using the most recent approx. of the other variables.

For a $n \times n$ system, the $k+1$-th approximation is:

$$\begin{cases} x_1^{(k+1)} = u_{12}x_2^{(k)} + ... + u_{1n}x_n^{(k)} + c_1 \\ x_2^{(k+1)} = u_{21}x_1^{(k+1)} + u_{23}x_3^{(k)} + ... + u_{2n}x_n^{(k)} + c_2 \\ ... \\ x_n^{(k+1)} = u_{n1}x_1^{(k+1)} + ... + u_{nn-1}x_{n-1}^{(k+1)} + c_n, \end{cases}$$

with $k = 0, 1, 2, ...;$ $u_{ij} = -\frac{a_{ij}}{a_{ii}},$ $c_i = \frac{b_i}{a_{ii}},$ $i = 1, ..., n$ (as in Jacobi method).

Algorithmic form:

$$x_i^{(k)} = \frac{b_i - \sum\limits_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum\limits_{j=i+1}^{n} a_{ij}x_j^{(k-1)}}{a_{ii}}$$

for each $i = 1, 2, ...n,$ and for $k \geq 1$.

Stopping criterions: $\left|x^{(k)} - x^{(k-1)}\right| < \varepsilon$, or $\frac{\left|\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}\right|}{\left|\mathbf{x}^{(k)}\right|} < \varepsilon$, with ε - a prescribed tolerance, $\varepsilon > 0$.

Remark 6 *Because the new values can be immediately stored in the location that held the old values, the storage requirements for* $\mathbf{x}$ *with the Gauss-Seidel method is half than that for Jacobi method and the rate of convergence is faster.*

Example 7 *Solve the following system using the Gauss-Seidel iterative method. Use* $\varepsilon = 10^{-3}$ *and* $\mathbf{x}^{(0)} = (0\ 0\ 0\ 0)$ *as the starting vector.*

$$\begin{cases} 7x_1 - 2x_2 + x_3 = 17 \\ x_1 - 9x_2 + 3x_3 - x_4 = 13 \\ 2x_1 + 10x_3 + x_4 = 15 \\ x_1 - x_2 + x_3 + 6x_4 = 10 \end{cases}$$

We have

$$\begin{aligned} x_1 &= (17 + 2x_2 - x_3)/7 \\ x_2 &= (-13 + x_1 + 3x_3 - x_4)/9 \\ x_3 &= (15 - 2x_1 - x_4)/10 \\ x_4 &= (10 - x_1 + x_2 - x_3)/6, \end{aligned}$$

and, for example,

$$\begin{aligned} x_1^{(1)} &= (17 + 2x_2^{(0)} - x_3^{(0)})/7 \\ x_2^{(1)} &= (-13 + x_1^{(1)} + 3x_3^{(0)} - x_4^{(0)})/9 \\ x_3^{(1)} &= (15 - 2x_1^{(1)} - x_4^{(0)})/10 \\ x_4^{(1)} &= (10 - x_1^{(1)} + x_2^{(1)} - x_3^{(1)})/6, \end{aligned}$$

which provide the following Gauss-Seidel iterative process:

$$\begin{aligned}
x_1^{(k+1)} &= \left(17 + 2x_2^{(k)} - x_3^{(k)}\right)/7 \\
x_2^{(k+1)} &= \left(-13 + x_1^{(k+1)} + 3x_3^{(k)} - x_4^{(k)}\right)/9 \\
x_3^{(k+1)} &= \left(15 - 2x_1^{(k+1)} - x_4^{(k)}\right)/10 \\
x_4^{(k+1)} &= \left(10 - x_1^{(k+1)} + x_2^{(k+1)} - x_3^{(k+1)}\right)/6, \quad \textit{for } k \geq 1.
\end{aligned}$$

Substitute $\mathbf{x}^{(0)} = (0,0,0,0)$ *into the right-hand side of each of these equations to get*

$$\begin{aligned}
x_1^{(1)} &= (17 + 2\cdot 0 - 0)/7 = 2.428\ 571\ 429 \\
x_2^{(1)} &= (-13 + 2.428\ 571\ 429 + 3\cdot 0 - 0)/9 = -1.1746031746 \\
x_3^{(1)} &= (15 - 2\cdot 2.428\ 571\ 429 - 0)/10 = 1.0142857143 \\
x_4^{(1)} &= (10 - 2.428\ 571\ 429 - 1.1746031746 - 1.0142857143)/6 \\
&= 0.8970899472
\end{aligned}$$

and so

$$\mathbf{x}^{(1)} = (2.428571429 - 1.1746031746, 1.0142857143, 0.8970899472).$$

Similar procedure generates a sequence that converges to

$$\mathbf{x}^{(5)} = (2.000025, -1.000130, 1.000020.0.999971).$$

4.3.3. Relaxation method

In case of convergence, the Gauss-Seidel method is faster than Jacobi method. The convergence can be more improved using **relaxation method (SOR method) (**SOR=Succesive Over Relaxation)

Algorithmic form of the method:

$$x_i^{(k)} = \frac{\omega}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}\right) + (1-\omega)x_i^{(k-1)}$$

for each $i = 1, 2, ...n,$ and for $k \geq 1$.

For $0 < \omega < 1$ the procedure is called **under relaxation method,** that can be used to obtain convergence for systems which are not convergent by Gauss-Siedel method.

For $\omega > 1$ the procedure is called **over relaxation method,** that can be used to accelerate the convergence for systems which are convergent by Gauss-Siedel method.

By Kahan's Theorem follows that the method converges for $0 < \omega < 2$.

Remark 8 *For $\omega = 1$, relaxation method is Gauss-Seidel method.*

Example 9 *Solve the following system, using relaxation iterative method. Use $\varepsilon = 10^{-3}$, $\mathbf{x}^{(0)} = (1\ 1\ 1)$ and $\omega = 1.25$,*

$$\begin{array}{rcrcrcr} 4x_1 & + & 3x_2 & & & = & 24 \\ 3x_1 & + & 4x_2 & - & x_3 & = & 30 \\ & - & x_2 & + & 4x_3 & = & -24 \end{array}$$

We have

$$\begin{aligned} x_1^{(k)} &= 7.5 - 0.937x_2^{(k-1)} - 0.25x_1^{(k-1)} \\ x_2^{(k)} &= 9.375 - 9.375x_1^{(k)} + 0.3125x_3^{(k-1)} - 0.25x_2^{(k-1)} \\ x_3^{(k)} &= -7.5 + 0.3125x_2^{(k)} - 0.25x_3^{(k-1)}, \quad \textit{for } k \geq 1. \end{aligned}$$

The solution is $(3, 4, -5)$.

5.1. One-step methods

Let F be a one-step method, i.e., for a given x_i we have $x_{i+1} = F(x_i)$.

Remark 18 *If $p = 1$ the convergence condition is $|F'(x)| < 1$.*

If $p > 1$ there always exists a neighborhood of α where the F-method converges.

All information on f are given at a single point, the starting value $\Rightarrow$ we are lead to Taylor interpolation.

Theorem 19 *Let α be a solution of equation (3), $V(\alpha)$ a neighborhood of α, $x, x_i \in V(\alpha)$, f fulfills the necessary continuity conditions. Then we have the following method, denoted by F_m^T, for approximating α:*

$$F_m^T(x_i) = x_i + \sum_{k=1}^{m-1} \frac{(-1)^k}{k!}[f(x_i)]^k g^{(k)}(f(x_i)), \tag{6}$$

is an approximation of α, and F_m^T is an approximation method for the solution α. ■

Concerning the order of the method F_m^T we state:

Theorem 20 *If* $g = f^{-1}$ *satisfies condition* $g^{(m)}(0) \neq 0$, *then* $\mathrm{ord}(F_m^T) = m$.

Remark 21 *We have an upper bound for the absolute error in approximating* α *by* x_{i+1} :

$$\left|\alpha - F_m^T(x_i)\right| \leq \frac{1}{m!}[f(x_i)]^m M_m g, \quad \text{\textit{with} } M_m g = \max_{y \in V(0)} \left|g^{(m)}(y)\right|.$$

COURSE 11

5. Numerical methods for solving nonlinear equations in $\mathbb{R}$

The roots of the iterative methods are traced back to Egyptians and Babylonians (cc. 1800 B.C.).

Example of nonlinear equation. Kepler's Equation: consider a two-body problem like a satellite orbiting the earth (or a planet revolving around the sun). Kepler discovered that the orbit is an ellipse and the central body F (earth) is in a focus of the ellipse. The speed of the satellite P is not uniform: near the earth it moves faster than far away. It is used Kepler's law to predict where the satellite will be at a given time. If we want to know the position of the satellite for t = 9 minutes, then we have to solve the equation $f(E) = E - 0.8 \sin E - 2\pi/10 = 0$.

Let $f : \Omega \to \mathbb{R}$, $\Omega \subset \mathbb{R}$. Consider the equation

$$f(x) = 0, \quad x \in \Omega. \tag{1}$$

We attach a mapping $F : D \to D,\ D \subset \Omega^n$ to this equation.

Let $(x_0, ..., x_{n-1}) \in D$. Using F and the numbers $x_0, x_1, ..., x_{n-1}$ we construct iteratively the sequence

$$x_0, x_1, ..., x_{n-1}, x_n, ... \tag{2}$$

with

$$x_i = F\left(x_{i-n}, ..., x_{i-1}\right), \quad i = n, \ldots. \tag{3}$$

The problem consists in choosing F and $x_0, ..., x_{n-1} \in D$ such that the sequence (2) to be convergent to the solution of the equation (1).

Definition 1 *The procedure of approximation the solution of equation (1) by the elements of the sequence (2), computed as in (3), is called* F-**method.**

The numbers $x_0, x_1, ..., x_{n-1}$ are called **the starting (initial) points** *and the k-th element of the sequence (2) is called an approximation of k-th index of the solution.*

If the set of starting points has only one element then the F-method is **an one-step method;** if it has more than one element then the F-method is **a multistep method**.

Definition 2 *If the sequence (2) converges to the solution of the equation (1) then the F-method is convergent, otherwise it is divergent.*

Definition 3 *Let $\alpha \in \Omega$ be a solution of the equation (1) and let $x_0, x_1, ..., x_{n-1}, x_n, ...$ be the sequence generated by a given F-method. The number p having the property*

$$\lim_{x_i \to \alpha} \frac{|\alpha - F(x_{i-n}, ..., x_i)|}{|\alpha - x_i|^p} = C \neq 0, \quad C = constant,$$

is called the order *of the F-method.*

For one-step methods, the above condition reduces to

$$\lim_{x_i \to \alpha} \frac{|\alpha - F(x_i)|}{|\alpha - x_i|^p} = \lim_{x_i \to \alpha} \frac{|\alpha - x_{i+1}|}{|\alpha - x_i|^p} = C \neq 0, \quad C = constant,$$

We construct some classes of F-methods based on the interpolation procedures.

Let $\alpha \in \Omega$ be a solution of the equation (1) and $V(\alpha)$ a neighborhood of α. Assume that f has inverse on $V(\alpha)$ and denote $g := f^{-1}$. Since

$$f(\alpha) = 0$$

it follows that

$$\alpha = g(0).$$

This way, the approximation of the solution α is reduced to the approximation of $g(0)$.

Definition 4 *The approximation of g by means of an interpolating method, and of α by the value of g at the point zero is called* **the inverse interpolation procedure.**

5.1. One-step methods

Let F be a one-step method, i.e., for a given x_i we have $x_{i+1} = F(x_i)$.

Remark 5 *If $p = 1$, a sufficient convergence condition is $|F'(x)| < 1$.*

All information on f are given at a single point, the starting value $\Rightarrow$ we are lead to Taylor interpolation.

Theorem 6 *Let α be a solution of equation (1), $V(\alpha)$ a neighborhood of α, $x, x_i \in V(\alpha)$, f fulfills the necessary continuity conditions. Then we have the following method, denoted by F_m^T, for approximating α:*

$$F_m^T(x_i) = x_i + \sum_{k=1}^{m-1} \frac{(-1)^k}{k!}[f(x_i)]^k g^{(k)}(f(x_i)), \tag{4}$$

where $g = f^{-1}$.

Proof. There exists $g = f^{-1} \in C^m[V(0)]$. Let $y_i = f(x_i)$ and consider Taylor interpolation formula

$$g(y) = (T_{m-1}g)(y) + (R_{m-1}g)(y),$$

with

$$(T_{m-1}g)(y) = \sum_{k=0}^{m-1} \tfrac{1}{k!}(y - y_i)^k g^{(k)}(y_i),$$

and $R_{m-1}g$ is the corresponding remainder.

Since $\alpha = g(0)$ and $g \approx T_{m-1}g$, it follows

$$\alpha \approx (T_{m-1}g)(0) = x_i + \sum_{k=1}^{m-1} \tfrac{(-1)^k}{k!} y_i^k g^{(k)}(y_i).$$

Hence,

$$x_{i+1} := F_m^T(x_i) = x_i + \sum_{k=1}^{m-1} \tfrac{(-1)^k}{k!} [f(x_i)]^k g^{(k)}(f(x_i))$$

is an approximation of α, and F_m^T is an approximation method for the solution α. ■

Concerning the order of the method F_m^T we state:

Theorem 7 *If* $g = f^{-1}$ *satisfies* $g^{(m)}(0) \neq 0$, *then* $\mathrm{ord}(F_m^T) = m$.

Remark 8 *We have an upper bound for the absolute error in approximating* α *by* x_{i+1} :

$$\left|\alpha - F_m^T(x_i)\right| \leq \frac{1}{m!}[f(x_i)]^m M_m g, \quad \textit{with } M_m g = \max_{y \in V(0)} \left|g^{(m)}(y)\right|.$$

Particular cases.

1) Case $m = 2$.

$$F_2^T(x_i) = x_i - \frac{f(x_i)}{f'(x_i)}.$$

This method is called **Newton's method** (**the tangent method**). Its order is 2.

2) Case $m = 3$.

$$F_3^T(x_i) = x_i - \frac{f(x_i)}{f'(x_i)} - \frac{1}{2}\left[\frac{f(x_i)}{f'(x_i)}\right]^2 \frac{f''(x_i)}{f'(x_i)},$$

with $\text{ord}(F_3^T) = 3$. So, this method converges faster than F_2^T.

3) Case $m = 4$.

$$F_4^T(x_i) = x_i - \frac{f(x_i)}{f'(x_i)} - \frac{1}{2}\frac{f''(x_i)f^2(x_i)}{[f'(x_i)]^3} + \frac{\left(f'''(x_i)f'(x_i) - 3[f''(x_i)]^2\right)f^3(x_i)}{3![f'(x_i)]^5}.$$

If α is a solution of equation (1) and $x_{n+1} = F_2^T\left(x_n\right)$, for approximation error, Remark 8 gives

$$\left|\alpha - x_{n+1}\right| \leq \tfrac{1}{2}[f\left(x_n\right)]^2 M_2 g.$$

Lemma 10 *Let $\alpha \in (a, b)$ be a solution of equation (1) and let $x_n = F_2^T\left(x_{n-1}\right)$. Then*

$$\left|\alpha - x_n\right| \leq \tfrac{1}{m_1}\left|f\left(x_n\right)\right|, \quad \textit{with } m_1 \leq m_1 f = \min_{a \leq x \leq b}\left|f'\left(x\right)\right|.$$

Proof. We use the mean formula

$$f\left(\alpha\right) - f\left(x_n\right) = f'\left(\xi\right)\left(\alpha - x_n\right),$$

with $\xi \in$ to the interval determined by α and $x_{n.}$ From $f\left(\alpha\right) = 0$ and $\left|f'\left(x\right)\right| \geq m_1$ for $x \in \left(a, b\right)$, it follows $\left|f\left(x_n\right)\right| \geq m_1\left|\alpha - x_n\right|$, that is

$$\left|\alpha - x_n\right| \leq \tfrac{1}{m_1}\left|f\left(x_n\right)\right|.$$

■

In practical applications the following evaluation is more useful:

Lemma 11 *If* $f \in C^2[a,b]$ *and* F_2^T *is convergent, then there exists* $n_0 \in \mathbb{N}$ *such that*

$$|x_n - \alpha| \le |x_n - x_{n-1}|, \quad n > n_0.$$

Proof. We start with Taylor formula

$$f(x_n) = f(x_{n-1}) + (x_n - x_{n-1}) f'(x_{n-1}) + \tfrac{1}{2}(x_n - x_{n-1})^2 f''(\xi),$$

where ξ belongs to the interval determined by x_{n-1} and x_n.

Since $x_n = F_2^T(x_{n-1})$, it follows that

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

thus we obtain

$$f(x_n) = \tfrac{1}{2}(x_n - x_{n-1})^2 f''(\xi).$$

Consequently,

$$|f(x_n)| \leq \tfrac{1}{2}(x_n - x_{n-1})^2 M_2 f,$$

and Lemma 10 yields $|\alpha - x_n| \leq \frac{1}{m_1}|f(x_n)|$ so

$$|\alpha - x_n| \leq \tfrac{1}{2m_1}(x_n - x_{n-1})^2 M_2 f.$$

Since F_2^T is convergent, there exists $n_0 \in \mathbb{N}$ such that

$$\tfrac{1}{2m_1}|x_n - x_{n-1}| M_2 f < 1, \quad n > n_0.$$

Hence,

$$|\alpha - x_n| \leq |x_n - x_{n-1}|, \quad n > n_0.$$

■

Remark 12 *The starting value is chosen randomly. If, after a fixed number of iterations the required precision is not achieved, i.e., condition $|x_n - x_{n-1}| \leq \varepsilon$, does not hold for a prescribed positive ε, the computation has to be started over with a new starting value.*

A modified form of Newton's method: - the same value during the computation of f':

$$x_{k+1} = x_k - \frac{f\left(x_k\right)}{f'\left(x_0\right)}, \quad k = 0, 1, \ldots.$$

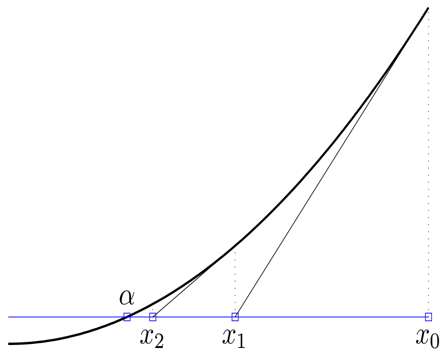
It is very useful because it doesn't request the computation of f' at x_j, $j = 1, 2, \ldots$ but the order is no longer equal to 2.

Another way for obtaining Newton's method.

We start with x_0 as an initial guess, sufficiently close to the α. Next approximation x_1 is the point at which the tangent line to f at $(x_0, f(x_0))$ crosses the Ox-axis. The value x_1 is much closer to the root α than x_0.

We write the equation of the tangent line at $(x_0, f(x_0))$:

$$y - f(x_0) = f'(x_0)(x - x_0).$$



The algorithm:

Let x_0 be the initial approximation.

for $n = 0, 1, ..., ITMAX$

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

A stopping criterion is:

$$|f(x_n)| \leq \varepsilon \text{ or } \left|x_{n+1} - x_n\right| \leq \varepsilon \text{ or } \frac{\left|x_{n+1} - x_n\right|}{\left|x_{n+1}\right|} \leq \varepsilon,$$

where ε is a specified tolerance value.

Example 13 *Use Newton's method to compute a root of* $x^3-x^2-1=0$, *to an accuracy of* 10^{-4}. *Use* $x_0=1$.

Sol. *The derivative of f is* $f'(x)=3x^2-2x$. *Using* $x_0=1$ *gives* $f(1)=-1$ *and* $f'(1)=1$ *and so the first Newton's iterate is*

$$x_1=1-\frac{-1}{1}=2 \text{ and } f(2)=3,\ f'(2)=8.$$

The next iterate is

$$x_2=2-\frac{3}{8}=1.625.$$

Continuing in this manner we obtain the sequence of approximations which converges to 1.465571.

5.2. Multistep methods

Lagrange inverse interpolation

Let $y_k = f(x_k)$, $k = 0, ..., n$, hence $x_k = g(y_k)$. We attach the Lagrange interpolation formula to y_k and $g(y_k)$, $k = 0, ..., n$:

$$g = L_n g + R_n g, \tag{6}$$

where

$$(L_n g)(y) = \sum_{k=0}^{n} \frac{(y-y_0)...(y-y_{k-1})(y-y_{k+1})...(y-y_n)}{(y_k-y_0)...(y_k-y_{k-1})(y_k-y_{k+1})...(y_k-y_n)} g(y_k). \tag{7}$$

Taking

$$F_n^L(x_0, ..., x_n) = (L_n g)(0),$$

F_n^L is a $(n+1)-$ steps method defined by

$$\begin{aligned} F_n^L(x_0, ..., x_n) &= \sum_{k=0}^{n} \frac{y_0 \cdots y_{k-1} y_{k+1} \cdots y_n}{(y_k-y_0)...(y_k-y_{k-1})(y_k-y_{k+1})...(y_k-y_n)} (-1)^n g(y_k) \\ &= \sum_{k=0}^{n} \frac{y_0 \cdots y_{k-1} y_{k+1} \cdots y_n}{(y_k-y_0)...(y_k-y_{k-1})(y_k-y_{k+1})...(y_k-y_n)} (-1)^n x_k. \end{aligned}$$

Concerning the convergence of this method we state:

Theorem 14 *If* $\alpha \in (a,b)$ *is solution of equation* $f(x) = 0$, f' *is bounded on* (a,b), *and the starting values satisfy* $|\alpha - x_k| < 1/c, \quad k = 0,...,n,$ *with* $c =$*constant, then the sequence*

$$x_{i+1} = F_n^L(x_{n-i},...,x_i), \quad i = n, n+1, ...$$

converges to α.

Remark 15 *The order* $ord(F_n^L)$ *is the positive solution of the equation*

$$t^{n+1} - t^n - ... - t - 1 = 0.$$

Particular cases.

1) For $n = 1$, the nodes x_0, x_1, we get **the secant method**

$$F_1^L(x_0,x_1) = x_1 - \frac{(x_1 - x_0) f(x_1)}{f(x_1) - f(x_0)},$$

Thus,

$$x_{k+1} := F_1^L\left(x_{k-1}, x_k\right) = x_k - \frac{\left(x_k - x_{k-1}\right) f\left(x_k\right)}{f\left(x_k\right) - f\left(x_{k-1}\right)}, \quad k = 1, 2, ...$$

is the new approximation obtained using the previous approximations x_{k-1}, x_k.

The *order* of this method is the positive solution of equation:

$$t^2 - t - 1 = 0,$$

so $ord(F_1^L) = \frac{(1+\sqrt{5})}{2} \approx 1.618$. (the golden ratio).

A modified form of the secant method: if we keep x_1 fixed and we change every time the same interpolation node, i.e.,

$$x_{k+1} = x_k - \frac{\left(x_k - x_1\right) f\left(x_k\right)}{f\left(x_k\right) - f\left(x_1\right)}, \quad k = 2, 3, \ldots.$$

2) For $n = 2$, the nodes x_0, x_1, x_2 and we get

$$F_2^L(x_0, x_1, x_2) = \frac{x_0 f(x_1) f(x_2)}{[f(x_0)-f(x_1)][f(x_0)-f(x_2)]} + \frac{x_1 f(x_0) f(x_2)}{[f(x_1)-f(x_0)][f(x_1)-f(x_2)]} + \frac{x_2 f(x_0) f(x_1)}{[f(x_2)-f(x_0)][f(x_2)-f(x_1)]}.$$

Remark 16 *The computation time is not the unique criterion in choosing the method! Newton's method is easier to apply. If $f(x)$ is not explicitly known (for example, it is the solution of the numerical integration of a differential equation), then its derivative is computed numerically. If we consider the following expression for the numerical computation of derivative:*

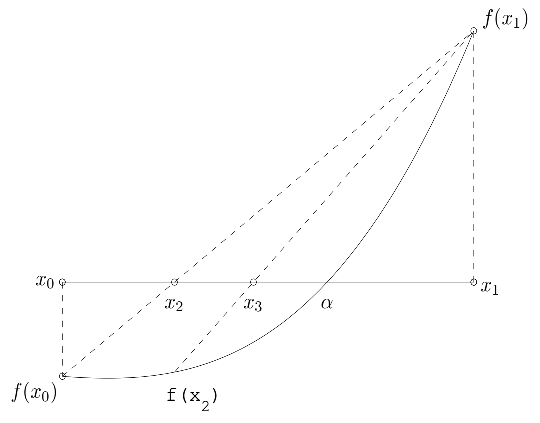
$$f'(x) \approx \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}} \tag{8}$$

then the Newton's method becomes the secant method.

Another way of obtaining secant method.

Based on approx. the function by a straight line connecting two points on the graph of f (not required f to have opposite signs at the initial points).

The first point, x_2, of the iteration is taken to be the point of intersection of the Ox-axis and the secant line connecting two starting points



The algorithm:

Let x_0 and x_1 be two initial approximations.

for $n = 1, 2, ..., ITMAX$

$$x_{n+1} \leftarrow x_n - f(x_n)\left[\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}\right].$$

A suitable stopping criterion is

$$|f(x_n)| \leq \varepsilon \text{ or } \left|x_{n+1} - x_n\right| \leq \varepsilon \text{ or } \frac{\left|x_{n+1} - x_n\right|}{\left|x_{n+1}\right|} \leq \varepsilon,$$

where ε is a specified tolerance value.

Example 17 *Use the secant method with $x_0 = 1$ and $x_1 = 2$ to solve $x^3 - x^2 - 1 = 0$, with $\varepsilon = 10^{-4}$.*

Sol. *With $x_0 = 1$, $f(x_0) = -1$ and $x_1 = 2$, $f(x_1) = 3$, we have*

$$x_2 = 2 - \frac{(2-1)(3)}{3-(-1)} = 1.25$$

from which $f(x_2) = f(1.25) = -0.609375$. The next iterate is

$$x_3 = 1.25 - \frac{(1.25-2)(-0.609375)}{-0.609375-3} = 1.3766234.$$

Continuing in this manner the iterations lead to the approximation 1.4655713.

Examples of other multi-step methods

1. THE BISECTION METHOD

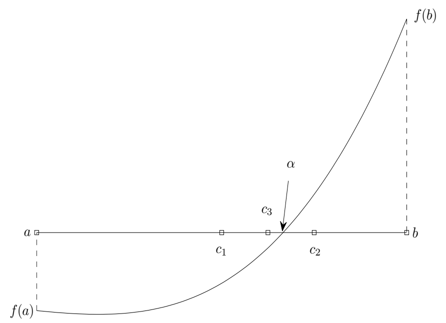
Let f be a given function, continuous on an interval $[a, b]$, such that

$$f(a)f(b) < 0. \tag{10}$$

By the Mean Value Theorem, it follows that there exists at least one zero α of f in (a, b).

The bisection method is based on halving the interval $[a, b]$ to determine a smaller and smaller interval within α must lie.

First we give the midpoint of $[a, b]$, $c = (a + b)/2$ and then compute the product $f(c)f(b)$. If the product is negative, then the root is in the interval $[c, b]$ and we take $a_1 = c$, $b_1 = b$. If the product is positive, then the root is in the interval $[a, c]$ and we take $a_1 = a$, $b_1 = c$. Thus, a new interval containing α is obtained.



Bisection method

The algorithm:

Suppose $f(a)f(b) \leq 0$. Let $a_0 = a$ and $b_0 = b$.

for $n = 0, 1, ...,$ITMAX

$c \leftarrow \frac{a_n + b_n}{2}$

if $f(a_n)f(c) \leq 0$, set $a_{n+1} = a_n, b_{n+1} = c$

else, set $a_{n+1} = c, b_{n+1} = b_n$

The process of halving the new interval continues until the root is located as accurately as desired, namely

$$\frac{|a_n - b_n|}{|a_n|} < \varepsilon,$$

where a_n and b_n are the endpoints of the n-th interval $[a_n, b_n]$ and ε is a specified precision. The approximation of the solution will be $\frac{a_n+b_n}{2}$.

Some other stopping criterions: $|a_n - b_n| < \varepsilon$ or $|f(a_n)| < \varepsilon$.

Example 18 *The function $f(x) = x^3 - x^2 - 1$ has one zero in $[1, 2]$. Use the bisection algorithm to approximate the zero of f with precision 10^{-4}.*

Sol. *Since $f(1) = -1 < 0$ and $f(2) = 3 > 0$, then (10) is satisfied. Starting with $a_0 = 1$ and $b_0 = 2$, we compute*

$$c_0 = \frac{a_0 + b_0}{2} = \frac{1+2}{2} = 1.5 \textit{ and } f(c_0) = 0.125.$$

Since $f(1.5)f(2) > 0$, the function changes sign on $[a_0, c_0] = [1, 1.5]$.

To continue, we set $a_1 = a_0$ *and* $b_1 = c_0$; *so*

$$c_1 = \frac{a_1 + b_1}{2} = \frac{1 + 1.5}{2} = 1.25 \text{ and } f(c_1) = -0.609375$$

Again, $f(1.25)f(1.5) < 0$ *so the function changes sign on* $[c_1, b_1] = [1.25,\ 1.5]$. *Next we set* $a_2 = c_1$ *and* $b_2 = b_1$. *Continuing in this manner we obtain a sequence* $(c_i)_{i>0}$ *which converges to* 1.465454, *the solution of the equation.*

2. THE METHOD OF FALSE POSITION

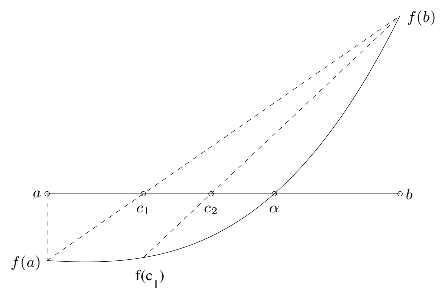
This method is also known as *regula falsi*, is similar to the Bisection method but has the advantage of being slightly faster than the latter. The function have to be continuous on $[a, b]$ with

$$f(a)f(b) < 0.$$

The point c is selected as point of intersection of the Ox-axis, and the straight line joining the points $(a,\ f(a))$ and $(b, f(b))$. From the equation of the secant line, it follows that

$$c = b - f(b)\frac{b-a}{f(b)-f(a)} = \frac{af(b)-bf(a)}{f(b)-f(a)} \tag{11}$$

Compute $f(c)$ and repeat the procedure between the values at which the function changes sign, that is, if $f(a)f(c) < 0$ set $b = c$, otherwise set $a = c$. At each step we get a new interval that contains a root of f and the generated sequence of points will eventually converge to the root.



Method of false position.

The algorithm:

Given a function f continuous on $[a_0, b_0]$, with $f(a_0)f(b_0) < 0$,

input: a_0, b_0

for $n = 0, 1, ..., ITMAX$

$c \leftarrow \frac{f(b_n)a_n - f(a_n)b_n}{f(b_n) - f(a_n)}$

if $f(a_n)f(c) < 0$, set $a_{n+1} = a_n, b_{n+1} = c$ **else** set $a_{n+1} = c, b_{n+1} = b_n$.

Stopping criterions: $|f(a_n)| \le \varepsilon$ or $|a_n - a_{n-1}| \le \varepsilon$, where ε is a specified tolerance value.

Remark 19 *The bisection and the false position methods converge at a very low speed compared to the secant method.*

Example 20 *The function* $f(x) = x^3 - x^2 - 1$ *has one zero in* $[1, 2]$*. Use the method of false position to approximate the zero of* f *with precision* 10^{-4}*.*

Sol. *A root lies in the interval* $[1, 2]$ *since* $f(1) = -1$ *and* $f(2) = 3$*. Starting with* $a_0 = 1$ *and* $b_0 = 2$*, we get using (11)*

$$c_0 = 2 - \frac{3(2-1)}{3-(-1)} = 1.25 \text{ and } f(c_0) = -0.609375.$$

Here, $f(c_0)$ *has the same sign as* $f(a_0)$ *and so the root must lie on the interval* $[c_0, b_0] = [1.25, 2]$*. Next we set* $a_1 = c_0$ *and* $b_1 = b_0$ *to get the next approximation*

$$c_1 = 2 - \frac{3-(2-1.25)}{3-(-0.609375)} = 1.37662337 \text{ and } f(c_1) = -0.2862640.$$

Now $f(x)$ change sign on $[c_1, b_1] = [1.37662337, 2]$. *Thus we set* $a_2 = c_1$ *and* $b_2 = b_{1.}$ *Continuing in this manner the iterations lead to the approximation* 1.465558.

Example 21 *Compare the false position method, the secant method and Newton's method for solving the equation $x = cos x$, having as starting points $x_0 = 0.5$ și $x_1 = \pi/4$, respectively $x_0 = \pi/4$.*

n	**(a) x_n False position**	**(b) x_n Secant**	**(c) x_n Newton**
0	0.5	0.5	0.5
1	0.785398163397	0.785398163397	0.785398163397
2	0.736384138837	0.736384138837	0.739536133515
3	0.739058139214	0.739058139214	0.739085178106
4	0.739084863815	0.739085149337	0.739085133215
5	0.739085130527	0.739085133215	
6	0.739085133188		
7	0.739085133215		

The extra condition from the false position method usually requires more computation than the secant method, and the simplifications