Numerical Analysis notes

Foundations of matrix analysis

Numerical Analysis is the *"art of approximating"*. Quoting Wikipedia:

> An approximation is an inexact representation of something that is still close enough to be useful. Although approximation is most often applied to numbers, it is also frequently applied to such things as mathematical functions, shapes, and physical laws.

Approximations should be used when incomplete information prevents use of exact representations. Many problems in physics are either too complex to solve analytically, or impossible to solve using the available analytical tools. Thus, even when the exact representation is known, an approximation may yield a sufficiently accurate solution while reducing the complexity of the problem significantly.

The type of approximation used depends on the available information, the degree of accuracy required, the sensitivity of the problem to this data, and the savings (usually in time and effort) that can be achieved by approximation.

In this course we focus on three main aspects:

- Methodologies (or approximation algorithms)
- Analysis (estimate errors and convergence properties)
- Implementation (through `Python` and `numpy` notebooks)

Approximation is a matter of

- Representation (floating point values VS real numbers, finite dimensional spaces VS infinite dimensional ones, etc.)
- Measure of the Error (how do we know that we did a good job in approximating?)

In general, we will end up working with $\mathbb{R}^n$.

Matrices

Denoting by A_{ij} the matrix of order $n-1$ obtained from A by eliminating the i-th row and the j-th column, we call the **complementary minor** associated with the entry a_{ij} the determinant of the matrix A_{ij}. We call the **k-th principal (dominating) minor** of A, d_k, the determinant of the principal submatrix of order k, $A_k = A(1:k, 1:k)$.

The computational cost is O (ops) and can be constant, linear, polynomial, exponential, factorial, etc.

Total error: $f(x) - \hat{f}(\hat{x}) = \underbrace{\hat{f}(\hat{x}) - f(\hat{x})}_{\text{computational error } e_c = e_t + e_r} + \underbrace{f(\hat{x}) - f(x)}_{\text{propagated data error (independent from } f)}$

Example: finite difference approximation $f'(x) = \lim_{h\to 0} \frac{f(x-h)-f(x)}{h}$

- *truncation error* (obtained through Taylor) $\sim \frac{1}{2}|f''(x)|h + O(h^2)$
- *rounding error* $\sim \frac{2\varepsilon}{h}$, with $\varepsilon =$ machine precision

The optimal h is thus $h = 2\sqrt{\frac{\varepsilon}{|f''(x)|}}$.

Norms of vectors, matrices and functions

Given a vector space V over the field of real $\mathbb{R}$ or complex numbers $\mathbb{C}$ (V might be infinite dimensional), a **semi-norm** on V is a function $|\cdot| : V \to \mathbb{C}$ satisfying:

1. $|cf| = |c||f|$, for all $c \in \mathbb{C}$ [homogeneity];
2. $|f + g| \leq |f| + |g|$ [triangle inequality].

As it can be easily seen (1)–(2) imply that the norm is always non-negative:

$$0 = 0 \cdot |f| = |0 \cdot f| = |(1-1)f| = |f - f| \leq |f| + |(-1)f| = 2\|f|.$$

The semi-norm becomes a **norm** if in addition to (1)–(2) we have also that for all $f \in V$

$$|f| = 0 \text{ if and only if } f = 0.$$

A vector space is said to be **complete** if every Cauchy sequence in that space converges to one of the space's elements.

A complete vector space with a norm is called a **Banach space**.

A scalar product is a function $(\cdot, \cdot) : V \times V \mapsto \mathbb{C}$ which is:

1. **symmetric**: $(f, g) = \overline{(g, f)}$;
2. **linear**:
   - $(\alpha f, g) = \alpha(f, g)$ for all $\alpha \in \mathbb{C}$;
   - $(f + g, h) = (f, h) + (g, h)$;

   equivalent to
   - $(\alpha_1 f + \alpha_2 g, h) = \alpha_1(f, h) + \alpha_2(g, h)$
3. **positive definite**: $(f, f) \geq 0$ and $(f, f) = 0$ if and only if $f = 0$.

The norm is then defined as $\|f\|^2 = (f, f)$. That this is a norm (i.e. satisfies the triangle inequality) is proved by using the **Cauchy-Schwarz** inequality

$$|(f, g)| \leq \sqrt{(f, f)(g, g)}.$$

A Banach space with a scalar product and a norm induced by it is called a **Hilbert space**.

Example: Some norms in Banach spaces

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{\frac{1}{p}}, \quad 1 \leq p < \infty, \qquad \|x\|_1 = \sum_{i=1}^{n} |x_i|$$

$$\|x\|_2 = \sqrt{x_1^2 + x_2^2} \quad [\text{Euclidean norm}] \qquad \|x\|_\infty = \sup_{1 \leq i \leq n} |x_i|.$$

Definition: Two norms $||\cdot||_p, ||\cdot||_q$ on a vector space V are **equivalent** if there exist two positive constants c_1 and c_2 such that:

$$c_1||x||_q \leq ||x||_p \leq c_2||x||_q \quad \forall x \in V$$

In a finite-dimensional normed vector space (the dimension is given by the number of vectors in the basis) all norms are equivalent.

Principles of Numerical Mathematics

In an abstract setting, we describe a generic problem as

$$F(x, d) = 0 \tag{1}$$

where x is the unknown (generally a real number, a vector or a function) and $d \in D$ is the data. For each of the elements above we use an appropriate norm, which will enable us to measure quantities of interests from a numerical point of view, such as errors, stability, and dependency of the solution from the data. In particular we will use the symbols $\|\cdot\|_F$, $\|\cdot\|_x$ and $\|\cdot\|_d$ to indicate the various norms.

Stability of a problem

In general, not all problems can be approximated. If we write a problem as in (1), then its approximation is useful only if the continuous problem has a unique solution which depends continuously on the data. We call these problems *well posed* or *stable*:

Definition: A mathematical problem (1) is **well posed** or **stable** if it

- *admits a unique solution x:*

$$\forall d \in D, \exists! \, x \text{ s.t. } F(x, d) = 0.$$

- *which depends with continuity on the data:*

  it means that small perturbations on the data d of D yield "small" changes in the solution x. Precisely, let $d \in D$ and denote by δd a perturbation admissible of the data, in the sense that $d + \delta d \in D$, and by δx the corresponding change in the solution x, so that $x + \delta x$ is the perturbed solution, in such a way that

$$F(x + \delta x, d + \delta d) = 0, \tag{2}$$

  then we require that

$$\begin{aligned} &\forall d \in D, \quad \exists \eta_0 = \eta_0(d), K_0 \text{ such that} \\ &\text{if } \|\delta d\|_d < \eta_0 \quad \Longrightarrow \quad \|\delta x\|_x < K_0 \|\delta d\|_d. \end{aligned} \tag{3}$$

Example: A simple instance of an ill-posed problem is finding the number of real roots of a polynomial. For example, the polynomial $p(x) = x^4 - x^2(2a - 1) + a(a - 1)$ exhibits a discontinuous variation of the number of real roots as a continuously varies in the real field. We have, indeed, 4 real roots if $a \geq 1$, 2 if $a \in [0, 1)$ while no real roots exist if $a < 0$.

Condition numbers

Definition: A measure of how accurately we can approximate the problem at hand is given by the **condition number**: it measures the problem sensitiveness w.r.t. the input data.

Relative condition number:

$$K := \sup_{d, \delta d} \left\{ \frac{\|\delta x\|_x / \|x\|_x}{\|\delta d\|_d / \|d\|_d}, \ \delta d \neq 0, \ d + \delta d \in D \right\}. \tag{4}$$

Absolute condition number (to be used when either $x = 0$ or $d = 0$):

$$K_{abs} := \sup_{\delta d} \left\{ \frac{\|\delta x\|_x}{\|\delta d\|_d}, \ \delta d \neq 0, \ d + \delta d \in D \right\}. \tag{5}$$

If problem (1) admits a unique solution x to each data d, then we can construct a **resolvent map** G between the sets of the data and of the solutions, such that

$$G(d) = x, \quad \text{that is} \quad F(G(d), d) = 0. \tag{6}$$

According to this definition, (2) yields $x + \delta x = G(d + \delta d)$. Assuming that G is differentiable in d and denoting formally by $G'(d)$ its derivative with respect to d, then the Taylor expansion of G around d truncated at first order ensures that

$$G(d + \delta d) - G(d) = G'(d)\delta d + o(\delta d) \qquad \text{for } \delta d \to 0,$$

where $||\cdot||$ is a suitable vector norm and $o(\cdot)$ is the classical infinitesimal symbol denoting an infinitesimal term of higher order with respect to its argument.

Neglecting the infinitesimal of higher order with respect to δd, since we have that $x + \delta x = G(d + \delta d)$ and $x = G(d)$, we have that

$$x + \delta x - x = G(d + \delta d) - G(d) \quad \Longrightarrow \quad \delta x = G(d) - G(d + \delta d) = G'(d)\delta d$$

Using this and from (4) and (5), we respectively deduce that

$$K \simeq \|G'(d)\| \frac{\|d\|_d}{\|G(d)\|_x} \qquad K_{abs} \simeq \|G'(d)\|.$$

A *stable* problem is **well conditioned** when its condition number is "small", where the meaning of "small" depends on the problem at hand. If $K >> 1$, the problem is **ill-posed (sensitive, unstable)** and thus *not* approximable through numerical methods.

Stability of numerical methods

Once we have a *stable* problem (1), a numerical method for the approximate solution of (1) will consist, in general, of a sequence of approximate problems

$$F_n(x_n, d_n) = 0, \qquad n \geq 1 \tag{7}$$

depending on a certain parameter n (to be defined case by case), such that

$$\lim_{n\to\infty} \|F_n - F\|_F = 0$$
$$\lim_{n\to\infty} \|x_n - x\|_x = 0$$
$$\lim_{n\to\infty} \|d_n - d\|_d = 0,$$

for some appropriate norms $||\cdot||_F$, $||\cdot||_x$ and $||\cdot||_d$, so the understood expectation is that $x_n \to x$ as $n \to \infty$, i.e. that the numerical solution converges to the exact solution.

Equivalently to what happens in the continuous case, we can establish the stability of the approximate n-th problem.

Definition: A mathematical approximation of a stable problem is itself a **stable numerical method** if the following properties are satisfied:

- *Existence and uniqueness of solutions:*

  $$\forall n \geq 1, \forall d_n \in D_n, \ \exists!\, x_n \text{ such that } F_n(x_n, d_n) = 0.$$

- *Continuous dependence on data:*

  Let δd_n be a perturbation of the data, such that $d_n + \delta d_n \in D_n$, and let $x_n + \delta x_n$ be the corresponding perturbed solution, i.e., $F_n(x_n + \delta x_n, d_n + \delta d_n) = 0$, then

$$\begin{aligned} &\forall d_n \in D_n, \quad \exists \eta_0 = \eta_0(d_n), K_0 = K_0(d_n) \text{ such that} \\ &\text{if } \|\delta d_n\|_d < \eta_0 \in D_n \quad \Longrightarrow \quad \|\delta x_n\|_x < K_0 \|\delta d_n\|_d. \end{aligned} \tag{8}$$

Consistency

Whenever the data d is admissible for F_n, then further properties of the approximations can be devised. In particular, we expect that $x_n \to x$ as $n \to \infty$, i.e. that the numerical solution converges to the exact solution. We could also say that the residual (the error produced by plugging the exact solution in the scheme) tends to zero as $n \to \infty$.

Definition: If the datum d of problem (1) is admissible for F_n, so $d \in D_n \ \forall n$, we say that the numerical problem (7) is **consistent** if

$$\lim_{n\to\infty} F_n(x, d) = \lim_{n\to\infty} F_n(x, d) - F(x, d) = 0. \tag{9}$$

where x is the solution to problem (1) corresponding to the datum d.
Moreover, a numerical approximation is said to be **strongly consistent** if

$$F_n(x, d) = 0 \quad \text{for any value of } n.$$

Convergence

Definition: A numerical method (7) is **convergent** iff

$$\begin{aligned} &\forall \varepsilon > 0, \quad \exists\, n_0 = n_0(\varepsilon), \quad \exists \delta = \delta(n_0, \varepsilon) \quad \text{such that} \\ &\forall n > n_0, \quad \forall \delta d_n : \quad \|\delta d_n\|_d \le \delta \quad \Longrightarrow \quad \|x(d) - x_n(d + \delta d_n)\| \le \varepsilon, \end{aligned} \tag{10}$$

where d is an admissible datum for the problem (1), $x(d)$ is the corresponding solution and $x_n(d + \delta d_n)$ is the solution of the numerical problem (7) with datum $d + \delta d_n$.

To verify the implication (10) it suffices to check that under the same assumptions

$$||x(d + \delta d_n) - x_n(d + \delta d_n)|| \le \frac{\varepsilon}{2}. \tag{11}$$

Measures of the convergence of x_n to x are given by the **absolute error** or the **relative error**, respectively defined as

$$E(x_n) = |x - x_n|, \qquad E_{\text{rel}}(x_n) = \frac{|x - x_n|}{|x|} \quad (\text{if } x \neq 0).$$

The concepts of stability and convergence are strongly connected. First of all, if problem (1) is well posed, **a numerical problem (7) which is convergent is also stable**. So if the problem is well posed, a *necessary* condition in order for the numerical problem to be convergent is that it is stable.

Proof. Let us thus assume that the method is convergent, that is, (10) holds for an arbitrary $\varepsilon > 0$. We have

$$\begin{aligned} ||\delta x_n|| &= ||x_n(d + \delta d_n) - x_n(d)|| \\ &\le ||x_n(d) - x(d)|| + ||x(d) - x(d + \delta d_n)|| + ||x(d + \delta d_n) - x_n(d + \delta d_n)|| \\ &\le K(\delta(n_0, \varepsilon), d)||\delta d_n|| + \varepsilon, \end{aligned}$$

having used (3) and (11) twice. Choosing now δd_n such that $||\delta d_n|| \le \eta_0$, we deduce that $||\delta x_n||/||\delta d_n||$ can be bounded by $K_0 = K(\delta(n_0, \varepsilon), d) + 1$, provided that $\varepsilon \le ||\delta d_n||$, so that the method is stable.

□

Thus, we are interested in stable numerical methods since only these can be convergent (if a method is not stable it cannot be convergent, since stability $\Rightarrow$ convergence).

Lax-Richtmyer theorem

The stability of a numerical method becomes a *sufficient* condition for the numerical problem (7) to converge if this latter is also consistent with problem (1).

One of the fundamental theorems (a milestone) of numerical analysis is the so called **Lax-Richtmyer theorem** (or **equivalence theorem**):

Theorem. *For a consistent numerical method, stability is equivalent to convergence.*

Proof. Indeed, under these assumptions we have

$$||x(d+\delta d_n) - x_n(d+\delta d_n)|| \leq ||x(d+\delta d_n) - x(d)|| + ||x(d) - x_n(d)|| + ||x_n(d) - x_n(d+\delta d_n)||.$$

Thanks to (3), the first term at right-hand side can be bounded by $||\delta d_n||$ (up to a multiplicative constant independent of δd_n). A similar bound holds for the third term, due to the stability property (8). Finally, concerning the remaining term, if F_n is differentiable with respect to the variable x, an expansion in a Taylor series gives

$$F_n(x(d), d) - F_n(x_n(d), d) = \left.\frac{\partial F_n}{\partial x}\right|_{(\bar{x}, d)} (x(d) - x_n(d)),$$

for a suitable $\bar{x}$ "between" $x(d)$ and $x_n(d)$. Assuming also that $\partial F_n / \partial x$ is invertible, we get

$$x(d) - x_n(d) = \left.\left(\frac{\partial F_n}{\partial x}\right)^{-1}\right|_{(\bar{x}, d)} \left[F_n(x(d), d) - F_n(x_n(d), d)\right].$$

On the other hand, replacing $F_n(x_n(d), d)$ with $F(x(d), d)$ (since both terms are equal to zero) and passing to the norms, we find

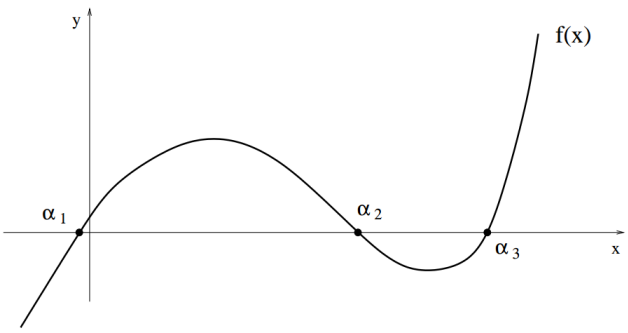
$$||x(d) - x_n(d)|| \leq \left|\left|\left.\left(\frac{\partial F_n}{\partial x}\right)^{-1}\right|_{(\bar{x}, d)}\right|\right| ||F_n(x(d), d) - F(x(d), d)||.$$

Thanks to (9) we can thus conclude that $||x(d) - x_n(d)|| \to 0$ for $n \to \infty$.

□

Nonlinear equations

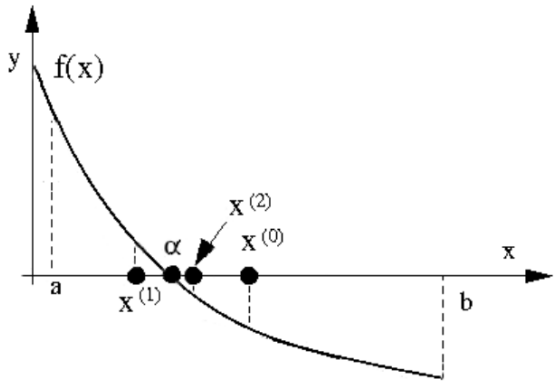
We may want to find the roots of scalar (or vector) non-linear functions, so find $\alpha \in \mathbb{R}$ s.t. $f(\alpha) = 0$, in a computational way. Most common approaches are *iterative*, since there is no explicit solving formula for $p \in \mathbb{R}^n$ with $n \geq 5$ for Abel's theorem.



Bisection method (Linear convergence)

This method is used to compute the root of a *continuous* function f on $[a, b]$, i.e., the point α such that $f(\alpha) = 0$. We assume that $f : [a, b] \to \mathbb{R}$ and $a < b$. If $f(a)f(b) < 0$, since f is continuous, we know that there exists (at least) one root α of f in the interval $[a, b]$, thanks to the following

Property (Theorem of zeros for continuous functions)**.** *Given a continuous function $f : [a, b] \to \mathbb{R}$, such that $f(a)f(b) < 0$, then $\exists \alpha \in (a, b)$ such that $f(\alpha) = 0$.*



Starting from $I^{(0)} = [a, b]$, the **bisection method** generates a sequence of subintervals $I^{(k)} = [a^{(k)}, b^{(k)}]$, $k \geq 0$, with $I^{(k)} \subset I^{(k-1)}$, $k \geq 1$, and enjoys the property that $f(a^{(k)})f(b^{(k)}) < 0$. Precisely,

1. set $a^{(0)} = a$, $b^{(0)} = b$ and $x^{(0)} = \frac{a^{(0)}+b^{(0)}}{2}$,
2. then, for $k \geq 0$, if $f(x^{(k)}) = 0$, then $x^{(k)}$ is the zero.
3. if $f(x^{(k)}) \neq 0$, then:

   a. if $f(x^{(k)})f(b^{(k)}) < 0 \Leftrightarrow f(a^{(k)})f(x^{(k)}) > 0 \Rightarrow$ the zero $\alpha \in (x^{(k)}, b^{(k)})$ and we define $a^{(k+1)} = x^{(k)}, b^{(k+1)} = b^{(k)}$,

   b. if $f(a^{(k)})f(x^{(k)}) < 0 \Leftrightarrow f(x^{(k)})f(b^{(k)}) > 0 \Rightarrow$ the zero $\alpha \in (a^{(k)}, x^{(k)})$ and we define $a^{(k+1)} = a^{(k)}, b^{(k+1)} = x^{(k)}$,

   c. finally, set $x^{(k+1)} = (a^{(k+1)} + b^{(k+1)})/2$.

We generate a sequence of intervals whose length is halved at each step, with $x^{(k)}$ being the midpoint at step k. By the divisions of this type, we construct the sequence $x^{(0)}, x^{(1)}, \dots, x^{(k)}$ such that $\lim_{k\to\infty} x^{(k)} = \alpha$ and that satisfies, for all k,

$$\boxed{|e^{(k)}| = |x^{(k)} - \alpha| \leq \frac{1}{2} I^{(k)} = \frac{b^{(k)} - a^{(k)}}{2} = \frac{b-a}{2^{k+1}}},$$

which is the *absolute error*, th error of estimation, at step k. This implies that $\lim_{k\to\infty} |e^{(k)}| = 0$, so the bisection method is therefore **globally convergent**.

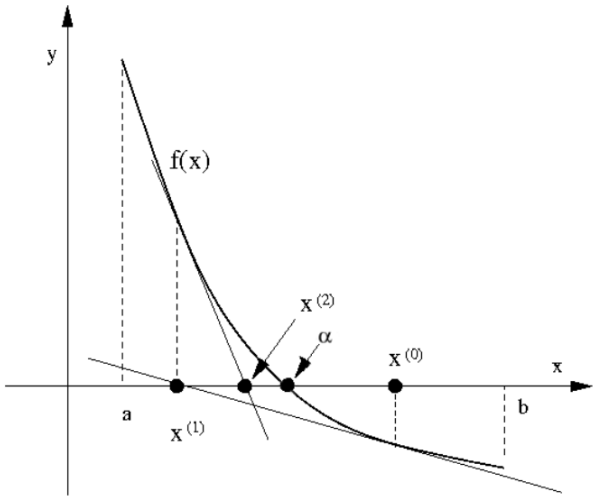
In order to ensure that the error is $|e^{(k)}| < \varepsilon$, we carry out $k_{\min}$ iterations at least:

$$k_{\min} > \log_2 \left(\frac{b-a}{\varepsilon} \right) - 1$$

The error doesn't decrease monotonically. The only possible stopping criterion is controlling the size of $I^{(k)}$.

Newton's method (Quadratic or linear convergence)

It is used to compute the root of a function f by using the values of f and f', and thus it is more efficient than the bisection method.



Let $f : \mathbb{R} \to \mathbb{R}$ be a differentiable function. Let $x^{(0)}$ be an initial guess, which is sufficiently close to α given f (estimated maybe through the graph or the bisection method). Let us consider the equation $y(x)$ which passes through the point $(x^{(k)}, f(x^{(k)}))$ and which has slope $f'(x^{(k)})$ (*linearized* version of problem):

$$y(x) = f'(x^{(k)})(x - x^{(k)}) + f(x^{(k)}).$$

We define $x^{(k+1)}$ by the point where this line intersects the axis x, i.e. $y(x^{(k+1)}) = 0$, since we are trying to approximate the root of the function. We deduce that:

$$\boxed{x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}}, \quad k = 0, 1, 2, \dots$$

Starting from the point $x^{(0)}$, the sequence $\{x^{(k)}\}$ converges the the root of f. This method is called **Newton - Rapson method**. Actually, the convergence of this method depends on the *property of the function* and on the *initial guess*.

Secant method (Super-linear convergence)

Let f be a *continuous* function with root α with $m = 1$ (for super-linearity) and $f'(x) \neq 0 \ \forall x \in I(\alpha)$, and let's select the initial point $x^{(0)}$ in a suitable $I(\alpha)$. In case $f'(x)$ is not available we can replace its value with an incremental ratio based on previous values:

$$\boxed{x^{(k+1)} = x^{(k)} - \left(\frac{f(x^{(k)}) - f(x^{(k-1)})}{x^{(k)} - x^{(k-1)}} \right)^{-1} f(x^{(k)})}, \quad k = 0, 1, 2, \dots$$

If $m = 1$ and $f \in C^2(I(\alpha))$, $\exists c > 0$ s.t. $|x^{(k+1)} - \alpha| \leq c|x^{(t)} - \alpha|^p$, with $p \approx 1.618$.

Otherwise, the method converges linearly.

Systems of nonlinear equations

Given $f_1, \dots, f_n$ nonlinear functions in $x_1, \dots, x_n$, we can set $\bar{f} = (f_1, \dots, f_n)^T$ and $\bar{x} = (x_1, \dots, x_n)^T$ to write a system as

$$\bar{f}(\bar{x}) = 0.$$

We can extend the Newton method to this system by replacing the derivative f' with the *Jacobian matrix* $J_{\bar{f}}$, as $(J_{\bar{f}})_{ij} = \frac{\partial f_i}{\partial x_j}$ for $i, j = 1, \dots, n$.

The secant method can also be adopted by recursively defining matrices B_k which are suitable approximations of $J_{\bar{f}}(x^{(0)})$ (**Broyden method**). This belongs to the family of **Quasi-Newton methods**.

Fixed point iterations

A general method for finding the roots of a nonlinear equation $f(x) = 0$ is the transformation in an equivalent problem $x - \phi(x) = 0$, where the auxiliary function $\phi : [a, b] \to \mathbb{R}$ must have the following property:

$$\phi(\alpha) = \alpha \quad \text{if and only if} \quad f(\alpha) = 0.$$

The point α is called a **fixed point** of ϕ, while ϕ is called the **iteration function**. Searching the zeros of f is reduced to the problem of determining the fixed points of ϕ.

It could be computed by the following algorithm:

$$\boxed{x^{(k+1)} = \phi(x^{(k)})}, \quad k \geq 0.$$

Indeed, if $x^{(k)} \to \alpha$ and if ϕ is *continuous* on $[a, b]$, then the limit α satisfies $\phi(\alpha) = \alpha$. So, starting from the point $x^{(0)}$, the sequence $\{x^{(k)}\}$ converges to the fixed point α.

The Newton method is a fixed point method: $x^{(k+1)} = \phi(x^{(k)})$ for the function

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

Let α be a zero of f, i.e. such that $f(\alpha) = 0$. Note that $\phi'(\alpha) = 0$, when $f'(\alpha) \neq 0$. Indeed,

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

Proposition 1 (Global convergence)**.**

1. *Assume that $\phi(x)$ is continuous on $[a,b]$ and such that $\phi(x) \in [a,b]$ for all $x \in [a,b]$; then there exists at least one fixed point $\alpha \in [a,b]$ of ϕ.*
2. *If ϕ is Lipschitz continue with constant $L < 1$ (**asymptotic convergence factor**), that is, if $\exists L < 1$ such that*

$$|\phi(x_1) - \phi(x_2)| \leq L|x_1 - x_2| \; \forall x_1, x_2 \in [a,b],$$

then there exists a unique fixed point $\alpha \in [a,b]$ and the sequence $x^{(k+1)} = \phi(x^{(k)}), k \geq 0$ converges to α, for any initial guess $x^{(0)} \in [a,b]$.

Proof.

1. The function $g(x) = \phi(x) - x$ is continuous in $[a,b]$ and, thanks to the assumption made on the range of ϕ, it holds $g(a) = \phi(a) - a \geq 0$ and $g(b) = \phi(b) - b \leq 0$. By applying the theorem of zeros of continuous functions, we can conclude that g has at least one zero in $[a,b]$, i.e. $\exists \alpha \in [a,b]$ such that

$$0 = g(\alpha) = \phi(\alpha) - \alpha \quad \iff \quad \phi(\alpha) = \alpha$$

so ϕ has at least one fixed point in $[a,b]$.

2. Indeed, should two different fixed points α_1 and α_2 exist, then

$$|\alpha_1 - \alpha_2| = |\phi(\alpha_1) - \phi(\alpha_2)| \leq L|\alpha_1 - \alpha_2| < |\alpha_1 - \alpha_2|$$

(since, in order, α_i is a fixed point of ϕ, ϕ is Lipschitz continuous and $L < 1$) which cannot be. So there exists a unique fixed point $\alpha \in [a,b]$ of ϕ.

Let $x^{(0)} \in [a,b]$ and $x^{(k+1)} = \phi(x^{(k)})$. We have

$$0 \leq |x^{(k+1)} - \alpha| = |\phi(x^{(k)}) - \phi(\alpha)| \leq L|x^{(k)} - \alpha| \leq \ldots \leq L^{k+1}|x^{(0)} - \alpha|,$$

i.e. $\forall k \geq 0$:

$$\frac{|x^{(k)} - \alpha|}{|x^{(0)} - \alpha|} \leq L^k.$$

For the convergence analysis and because $L < 1$, for $k \to \infty$, we obtain

$$\lim_{k\to\infty} |x^{(k)} - \alpha| \leq \lim_{k\to\infty} L^k |x^{(0)} - \alpha| = 0.$$

So, $\forall x^{(0)} \in [a,b]$, the sequence $\{x^{(k)}\}$ defined by $x^{(k+1)} = \phi(x^{(k)}), k \geq 0$ converges to α when $k \to \infty$.

□

The Proposition 1 ensures the convergence of the sequence $\{x^{(k)}\}$ at the root α for *any* choice of the initial guess $x^{(0)} \in [a,b]$. So it is a result of **global convergence**.

Remark. *If $\phi(x)$ is differentiable in $[a,b]$ and $\exists K < 1$ such that $|\phi'(x)| \leq K \; \forall x \in [a,b]$, then the condition (2) of the Proposition 1 is satisfied. This assumption is stronger, but is more often used in practice because it is easier to check.*

Definition. For a sequence of real numbers $\{x^{(k)}\}$ that converges, $x^{(k)} \to \alpha$, we say that **the convergence to α is linear** if exists a constant $C < 1$ such that, for k that is large enough

$$\boxed{|x^{(k+1)} - \alpha| \leq C|x^{(k)} - \alpha|}.$$

If exists a constant $C > 0$ such that the inequality

$$\boxed{|x^{(k+1)} - \alpha| \leq C|x^{(k)} - \alpha|^2}$$

is satisfied, we say that **the convergence is quadratic**.
In general, **the convergence is with order p**, with $p \geq 1$, if exists a constant $C > 0$ (with $C < 1$ when $p = 1$) such that the following inequality is satisfied

$$|x^{(k+1)} - \alpha| \leq C|x^{(k)} - \alpha|^p.$$

Proposition 2 (Ostrowski's theorem: local convergence)**.** *Let ϕ be a continuous and differentiable function on $[a, b]$ and α be a fixed point of ϕ. If $|\phi'(\alpha)| < 1$, then there exists $\delta > 0$ such that, for all $x^{(0)}$, $|x^{(0)} - \alpha| \leq \delta$, the sequence $\{x^{(k)}\}$ defined by $x^{(k+1)} = \phi(x^{(k)})$ converges to α when $k \to \infty$. Moreover, it holds*

$$\boxed{\lim_{k\to\infty} \frac{x^{(k+1)} - \alpha}{x^{(k)} - \alpha} = \phi'(\alpha)}. \tag{1}$$

Proof. The first parts follow directly from the previous theorem, the Proposition 1. Let's prove the property (1). For the Taylor series expansion (or for the Lagrange theorem), for any $k \geq 0$ there exists a value $\eta^{(k)}$ between α and $x^{(k)}$ such that

$$x^{(k+1)} - \alpha = \phi(x^{(k)}) - \phi(\alpha) = \phi(\eta^{(k)})(x^{(k)} - \alpha).$$

Since $\eta^{(k)}$ is included between α and $x^{(k)}$, we have that $\lim_{k\to\infty} \eta^{(k)} = \alpha$, and remembering that ϕ' in continuous in a neighbourhood of α, this implies that

$$\lim_{k\to\infty} \frac{x^{(k+1)} - \alpha}{x^{(k)} - \alpha} = \lim_{k\to\infty} \phi'(\eta^{(k)}) = \phi'(\alpha),$$

that is (1).

□

Note that, if $0 < |\phi'(\alpha)| < 1$, then for any constant C such that $|\phi'(\alpha)| < C < 1$, if k is large enough, we have:

$$|x^{(k+1)} - \alpha| \leq C|x^{(k)} - \alpha|.$$

This is a way to compute C locally.

Proposition 3 (Quadratic convergence)**.** *Let ϕ be a twice differentiable function on $[a, b]$, so $\phi \in C^2([a, b])$, and α be a fixed point of ϕ. Let us consider that $x^{(0)}$ converges locally. If $\phi'(\alpha) = 0$ and $\phi''(\alpha) \neq 0$, then the fixed point iterations converges with order 2 and*

$$\boxed{\lim_{k\to\infty} \frac{x^{(k+1)} - \alpha}{(x^{(k)} - \alpha)^2} = \frac{\phi''(\alpha)}{2}}.$$

Proof. Using the Taylor series for ϕ with $x = \alpha$, we have

$$x^{(k+1)} - \alpha = \phi(x^{(k)}) - \phi(\alpha) = \phi'(\alpha)(x^{(k)} - \alpha) + \frac{\phi''(\eta^{(k)})}{2}(x^{(k)} - \alpha)^2$$

where $\eta^{(k)}$ is between $x^{(k)}$ and α. So, since $\phi'(\alpha) = 0$ and $\lim_{k\to\infty} \eta^{(k)} = \alpha$, we have

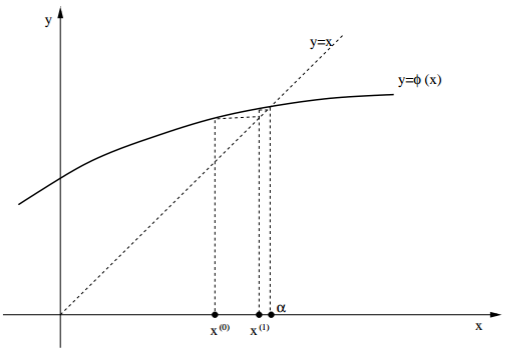
$$\lim_{k\to\infty} \frac{x^{(k+1)} - \alpha}{(x^{(k)} - \alpha)^2} = \lim_{k\to\infty} \frac{\phi''(\eta^{(k)})}{2} = \frac{\phi''(\alpha)}{2}.$$

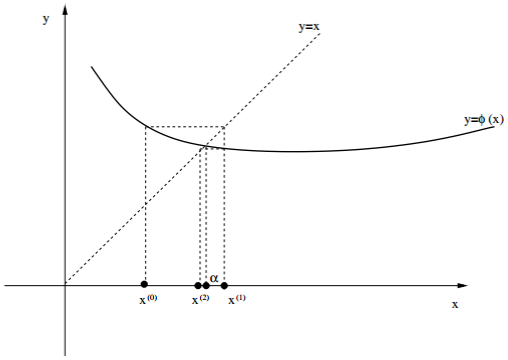
□

The value $|\phi'(\alpha)|$ influences the convergence: if $|\phi'(\alpha)| < 1$ we have that the method diverges. If $|\phi'(\alpha)| > 1$ then for (1) we have that $|x^{(k+1)} - \alpha| > |x^{(k)} - \alpha|$, thus no convergence is possible. While if $|\phi'(\alpha)| = 1$ no general conclusion can be stated since both convergence and non-convergence may be possible, depending on the problem at hand.

Convergent cases:

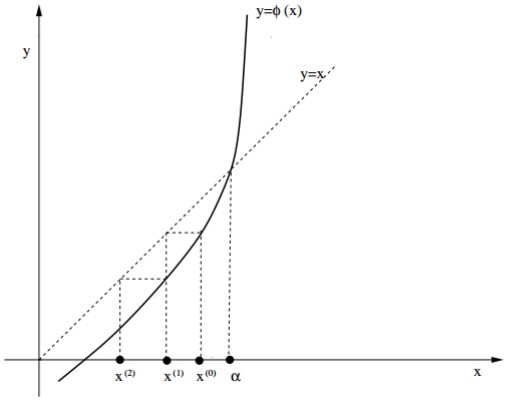
$$0 < \phi'(\alpha) < 1, \qquad -1 < \phi'(\alpha) < 0.$$

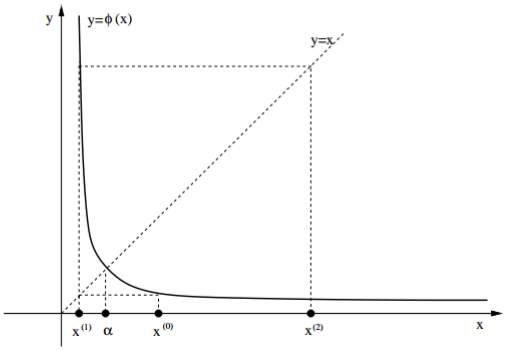




Divergent cases:

$$\phi'(\alpha) > 1, \qquad \phi'(\alpha) < -1.$$





More about the Newton method

Theorem 1. *If f is twice differentiable ($f \in C^2([a,b])$), $f(\alpha) = 0$ and $f'(\alpha) \neq 0$, then there exists $\delta > 0$ such that, if $|x^{(0)} - \alpha| \leq \delta$, the sequence defined by the Newton method converges to α. Moreover, the convergence is quadratic; more precisely*

$$\boxed{\lim_{k\to\infty} \frac{x^{(k+1)} - \alpha}{(x^{(k)} - \alpha)^2} = \frac{f''(\alpha)}{2f'(\alpha)}}.$$

Proof. The property of convergence comes from the Proposition 2, while the quadratic convergence is a consequence of the Proposition 3, because $\phi'(\alpha) = 0$ and $\frac{\phi''(\alpha)}{2} = \frac{f''(\alpha)}{2f'(\alpha)}$.

□

Definition. Let α be a zero of f. Then α is said to have **multiplicity** m, with $m \in \mathbb{N}$, if

$$f(\alpha) = f'(\alpha) = \ldots = f^{(m-1)}(\alpha) = 0 \text{ and } f^{(m)}(\alpha) \neq 0.$$

A zero that has multiplicity $m = 1$ is called **simple zero**.

Remark. If the root α has multiplicity $m > 1$, that means $f'(\alpha) = 0$, the convergence of the Newton method is linear, not quadratic. To restore the quadratic convergence we can use the **modified Newton method**:

$$\boxed{x^{(k+1)} = x^{(k)} - m\frac{f(x^{(k)})}{f'(x^{(k)})}}, \quad k = 0, 1, 2, \dots$$

where m is the multiplicity of α.

If the multiplicity m of α is unknown, there are other methods, the **adaptive Newton methods**, which can recover the quadratic order of convergence.

Stopping criteria

Suppose that $\{x^{(k)}\}$ is a sequence converging to a zero α of the function f. We provide some stopping criteria for terminating the iterative process that approximates α. In general, for all discussed methods, we can use two different stopping criteria: the iterations is completed when

$$|x^{(k+1)} - x^{(k)}| < \varepsilon \quad \textbf{(control of the increment)},$$

or

$$|f(x^{(k)})| < \varepsilon \quad \textbf{(control of the residual)},$$

where ε is a fixed tolerance.

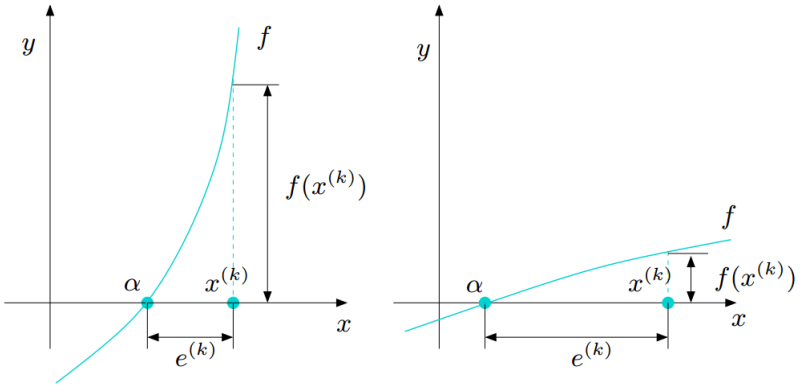
Below, ε is a fixed tolerance on the approximate calculation of α and $e^{(k)} = \alpha - x^{(k)}$ denotes the **absolute error** of the iteration k. We shall moreover assume that f is continuously differentiable in a suitable neighborhood of the root.

Control of the residual: *the iterative process terminates at the first step k such that $|f(x^{(k)})| < \varepsilon$.*

In the case of simple roots, the error is bound to the residual by the factor $1/|f'(\alpha)|$ so that the following conclusions can be drawn:

1. if $|f'(\alpha)| \simeq 1$, then $|e^{(k)}| \simeq \varepsilon$; therefore, the test provides a satisfactory indication of the error;
2. if $|f'(\alpha)| \ll 1$, the test is not reliable since $|e^{(k)}|$ could be quite large with respect to ε (the test is too weak);
3. if, finally, $|f'(\alpha)| \gg 1$, we get $|e^{(k)}| \ll \varepsilon$ and the test is too restrictive (the test is too strong).

Two cases where the residual is a bad estimator of the error: $|f'(x)| \gg 1$ (left), and $|f'(x)| \ll 1$ (right)) with x near to α:



Control of the increment: *the iterative process terminates as soon as $|x^{(k+1)} - x^{(k)}| < \varepsilon$.*

Using fixed point iterations, where $x^{(k+1)} = \phi(x^{(k)})$, we obtain the following estimation of the absolute error $e^{(k)}$ using the mean value theorem:

$$e^{(k+1)} = \alpha - x^{(k+1)} = \phi(\alpha) - \phi(x^{(k)}) = \phi'(\xi^{(k)})(\alpha - x^{(k)}) = \phi'(\xi^{(k)})e^{(k)},$$

where $\xi^{(k)}$ lies between $x^{(k)}$ and α. Then,

$$x^{(k+1)} - x^{(k)} = x^{(k+1)} - \alpha + \alpha - x^{(k)} = e^{(k)} - e^{(k+1)} = \left(1 - \phi'(\xi^{(k)})\right) e^{(k)}$$

so that, assuming that if k is large enough, we have $\phi'(\xi^{(k)}) \approx \phi'(\alpha)$, so it follows that

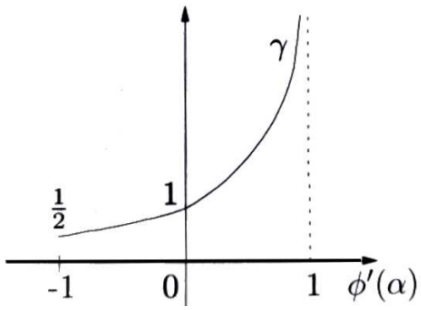
$$e^{(k)} \approx \frac{1}{(1 - \phi'(\alpha))}(x^{(k+1)} - x^{(k)}).$$

We can plot the graph of $\frac{1}{(1-\phi'(\alpha))}$ and comment on the relevance of the stopping criteria based on the increment:

- if $\phi'(\alpha)$ is close to 1 the test is unsatisfactory
- for methods of order 2, for which $\phi'(\alpha) = 0$, the criteria is optimal: it provides an optimal balancing between increment and error. This is the case of the Newton method, for which if α is a simple zero, we have the estimation

$$|x^{(k+1)} - x^{(k)}| \approx |(1 - \phi'(\alpha))e^{(k)}| = |e^{(k)}|.$$

- if $-1 < \phi'(\alpha) < 0$ the criteria is still satisfactory.



Aitken's method

Consider a fixed-point iteration that is linearly converging to a zero α of a given function f. Denote by λ an approximation of $\phi'(\alpha)$ to be suitably determined.

If ϕ converges linearly to α, since for Proposition 2 it holds $\lim_{k\to\infty} \frac{x^{(k+1)}-\alpha}{x^{(k)}-\alpha} = \phi'(\alpha)$, there must be a λ s.t.

$$\phi(x^{(k)}) - \alpha = x^{(k+1)} - \alpha = \phi'(\alpha)(x^{(k)} - \alpha) \simeq \lambda(x^{(k)} - \alpha), \quad \text{for } k \geq 1.$$

This allows us to obtain a better estimate of $x^{(k+1)}$ than $\phi(x^{(k)})$. Thus

$$\alpha \simeq x^{(k)} + \frac{\phi(x^{(k)}) - x^{(k)}}{1 - \lambda}$$

Let us consider, for $k \geq 2$, the following ratio

$$\lambda^{(k)} = \frac{\phi(\phi(x^{(k)})) - \phi(x^{(k)})}{\phi(x^{(k)}) - x^{(k)}},$$

and check that (see book)

$$\lim_{k\to\infty} \lambda^{(k)} = \phi'(\alpha).$$

We have that

$$x^{(k+1)} = x^{(k)} - \frac{(\phi(x^{(k)}) - x^{(k)})^2}{\phi(\phi(x^{(k)})) - 2\phi(x^{(k)}) + x^{(k)}}, \quad k \geq 0$$

that is the **Aitken's extrapolation formula** or the **Steffenson's method**.

The derived function $\phi_\Delta(x)$ has the same α as $\phi(x)$, but converges faster:

- *linear* $\phi \longrightarrow$ *quadratic* ϕ_Δ
- $p \geq 2 \longrightarrow 2p - 1\ \phi_\Delta$
- *linearly* with $m \geq 2\ \phi \longrightarrow$ *linearly* with $L = 1 - 1/m\ \phi_\Delta$

It may converge even if normal fixed-point iteration diverges.

The rope method

This method is obtained by replacing $f'(x^{(k)})$ by a fixed q in the Newton method:

$$x^{(k+1)} = x^{(k)} - \frac{1}{q} f(x^{(k)}), \quad k = 0, 1, 2, \ldots$$

We can take, for example, $q = f'(x^{(0)})$ or $q = \frac{f(b)-f(a)}{b-a}$, in the case when we search a zero in the interval $[a, b]$.

Remark. The rope method is also a fixed point method for

$$\phi(x) = x - \frac{1}{q} f(x).$$

So, we have $\phi'(x) = 1 - \frac{1}{q} f'(x)$ and thanks to the Proposition 2, we obtain that the method converges if the following condition is satisfied:

$$|\phi'(\alpha)| = \left|1 - \frac{1}{q} f'(\alpha)\right| < 1.$$

Interpolation

This section is addressed to the approximation of a function which is known through its values at a given number of points.

Precisely, given $n+1$ distinct pairs $\{q_i = (x_i, y_i)\}_{i=0}^n$ on an interval I, the problem consists of finding a function $\Phi = \Phi(x)$ such that $\Phi(x_i) = y_i$ for $i = 0, \dots, n$, the y_i being some given values, and say that Φ *interpolates* $\{y_i\}$ at the *nodes* $\{x_i\}$. We speak about **polynomial interpolation** if Φ is an algebraic polynomial, **trigonometric approximation** if Φ is a trigonometric polynomial or **piecewise polynomial interpolation** (or **spline interpolation**) if Φ is only locally a polynomial. The interpolants can be *polynomial*, *trigonometric*, *rational*, etc.

The numbers y_i may represent the values attained at the nodes x_i by a function f that is known in closed form, as well as experimental data. In the former case, the approximation process aims at replacing f with a simpler function to deal with, in particular in view of its numerical integration or derivation. In the latter case, the primary goal of approximation is to provide a compact representation of the available data, whose number is often quite large. So interpolation is a form of approximation that could be used both to simplify a complex function in order to make it easier to derive, or to understand the data distribution.

Lagrange interpolation

The **Lagrange interpolation** is a type of interpolation where the function Φ is polynomial. Given $n+1$ couples $\{x_i, y_i\}, i = 0, \dots, n$ with x_i as nodes, we want to find a polynomial $\Pi_m \in \mathbb{P}^m$, where $\mathbb{P}^m := \text{span}\{x^i\}_{i=0}^m$, called **interpolating polynomial**, such that

$$\Pi_n(x_i) = a_n x_i^n + \dots + a_1 x_i + a_0 = y_i, \quad i = 0, \dots, n. \quad (1)$$

The points x_i are called **interpolation nodes**. If $n \neq m$ the problem is over or under-determined, while if $n = m$, the following result holds:

Theorem 1. *Given $n+1$ distinct nodes $x_0, \dots, x_n$ and $n+1$ corresponding values $y_0, \dots, y_n$, there exists a unique polynomial $\Pi_n \in \mathbb{P}^n$ such that $\Pi_n(x_i) = y_i$ for $i = 0, \dots, n$.*

Proof. To prove existence, let us use a constructive approach, providing an expression for Π_n. Denoting by $\{l_i\}_{i=0}^n$ a basis for $\mathbb{P}^n$, then Π_n admits a representation on such a basis of the form $\Pi_n(x) = \sum_{i=0}^n b_i l_i(x)$ with the property that

$$\Pi_n(x_i) = \sum_{j=0}^n b_j l_j(x_i) = y_i, \quad i = 0, \dots, n. \quad (2)$$

If we define

$$l_i \in \mathbb{P}_n : l_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^n \frac{x - x_j}{x_i - x_j}, \quad i = 0, \dots, n; \qquad l_0 = 1, \quad (3)$$

then $l_i(x_j) = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$ and we immediately get from (2) that $b_i = y_i$.

It can be proven that the polynomials $\{l_i, i = 0, \dots, n\}$ form a basis for $\mathbb{P}^n$ (they define the whole space $\mathbb{P}^n$), called **Lagrange basis**. As a consequence, the interpolating polynomial exists and has the following form (called *Lagrange* form)

$$\Pi_n(x) = \sum_{i=0}^{n} y_i l_i(x); \qquad \Pi_0(x) = y_0. \tag{4}$$

To prove uniqueness, suppose that another interpolating polynomial Ψ_m of degree $m \leq n$ exists, such that $\Psi_m(x_i) = y_i$ for $i = 0, \ldots, n$. Then, the difference polynomial $\Pi_n - \Psi_m$ vanishes at $n+1$ distinct points x_i and thus coincides with the null polynomial: $\Pi_n - \Psi_m \equiv 0$. Therefore, $\Psi_m = \Pi_n$.

□

It can be checked that

$$\Pi_n(x) = \sum_{i=0}^{n} \frac{\omega_{n+1}(x)}{(x - x_i)\omega'_{n+1}(x_i)} y_i, \tag{5}$$

where ω_{n+1} is the nodal polynomial of degree $n+1$ defined as

$$\omega_{n+1}(x) = \prod_{i=0}^{n} (x - x_i). \tag{6}$$

Example. *Linear interpolation*: we have that if $n = 1$, given two nodes (x_i, y_i), with $i = 0, 1$, the interpolating polynomial of degree 1 is

$$\Pi_1(x) = y_0 \frac{x - x_1}{x_0 - x_1} + y_1 \frac{x - x_0}{x_1 - x_0}.$$

Quadratic interpolation: if $n = 2$, given three nodes (x_i, y_i), with $i = 0, 1, 2$, the interpolating polynomial of degree 2 is

$$\Pi_2(x) = y_0 \frac{x - x_1}{x_0 - x_1} \frac{x - x_2}{x_0 - x_2} + y_1 \frac{x - x_0}{x_1 - x_0} \frac{x - x_2}{x_1 - x_2} + y_2 \frac{x - x_0}{x_2 - x_0} \frac{x - x_1}{x_2 - x_0}.$$

◇

Formula (4) is called the **Lagrange form** of the interpolating polynomial, while the polynomials $l_i(x)$ are the **(Lagrange) characteristic polynomials**. In the figure we show the Lagrange characteristic polynomials $l_2(x)$, $l_3(x)$ and $l_4(x)$, in the case of degree $n = 6$, on the interval $[-1, 1]$ where equally spaced nodes are taken, including the end points.
Notice that $|l_i(x)|$ can be greater than 1 within the interpolation interval.

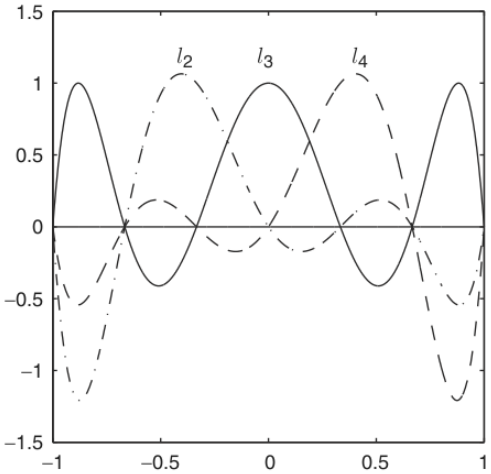


Fig. 8.1. Lagrange characteristic polynomials

If y_i represents the values of a continuous function $f \in C^0([a, b])$, so $y_i = f(x_i)$ for $i = 0, \ldots, n$, we want that $\Pi_n(x_i) = y_i = f(x_i)$. We say that interpolating polynomial $\Pi_n(x)$ is the interpolant of f, and will be denoted by $\Pi_n f(x)$.

Another formulation

Given $n+1$ distinct points $\{q_i\}_{i=0}^n$ in the interval $[0,1]$, we define the **Lagrange interpolation** operator $\mathcal{L}^n$ the operator

$$\mathcal{L}^n : C^0([0,1]) \mapsto \mathcal{P}^n$$

which satisfies

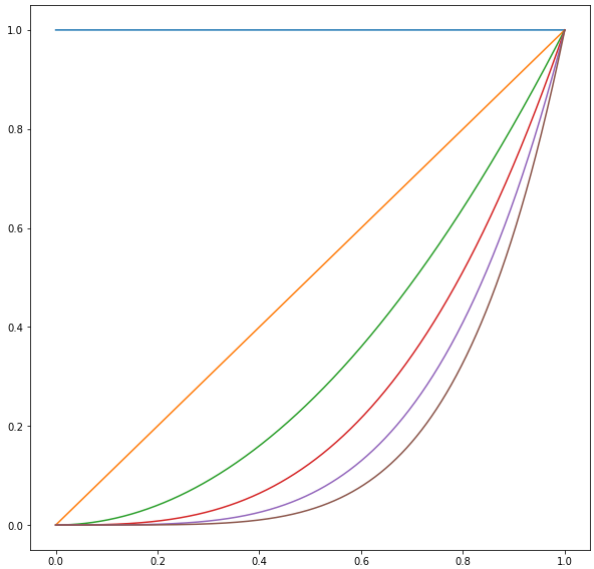
$$(\mathcal{L}^n f)(q_i) = f(q_i), \qquad i = 0, \dots, n.$$

This operator is used to approximate the infinitely dimensional space $C^0([0,1])$ with a finite dimensional one, $\mathcal{P}^n$, which is the space of polynomials of order n.

Such a space has dimension $n+1$, and can be constructed using linear combinations of monomials of order $\leq n$:

$$\mathbb{P}^n = \text{span}\{p_i := x^i\}_{i=0}^n \tag{7}$$

If we want to construct the Lagrange interpolation of a given function on $n+1$ equispaced points in $[0,1]$, then we are actively looking for an element of $\mathcal{P}^n$ that coincides with the function at these given points.



Given a basis $\{v_i\}_{i=0}^n$, any element of $\mathbb{P}^n$ can be written as a linear combination of the basis, i.e.,

$$\forall u \in \mathcal{P}^n, \quad \exists! \{u^i\}_{i=0}^n \quad | \quad u(x) = \sum_{i=0}^n u^i v_i(x) \tag{8}$$

in what follows, we'll use Einstein summation convention, and call u both the function of $\mathbb{P}^n$, or the $\mathbb{R}^{n+1}$ vector representing its coefficients.

Remark on the notation (advanced topic. Ignore if you don't understand it)

We use upper indexes to indicate both "contravariant" coefficients and the *canonical basis of the dual space*, i.e., the linear functionals in $(\mathbb{P}^n)^*$ such that

$$(\mathbb{P}^n)^* := \text{span}\{v^i\}_{i=0}^n \qquad | \qquad v^i(v_j) = \delta^i_j \qquad i,j = 0, \dots, n$$

With this notation, we have that the coefficients of a polynomial are uniquely determined by

$$u^i = v^i(u)$$

where the u on the right hand side is an element of $\mathbb{P}^n$ (not its coefficients).

If we want to solve the interpolation problem above, then we need to find the coefficients u^i of the polynomial u that interpolates f at the points q_i:

$$v_j(q_i)u^j = f(q_i)$$

(Remember Einstein summation convention)

This can be written as a linear problem $Au = F$, with system matrix $A_{ij} := v_j(q_i)$ and right hand side $F_i = f(q_i)$.

Condition number of interpolation

What is the condition number of this problem?

Given a set of $n+1$ (distinct) points $\{q_i\}_{i=0}^n$, and a basis $\{p_i\}_{i=0}^n$ for the polynomial space $\mathbb{P}^n([0,1])$ (that has dimension $n+1$), we would like to estimate the condition number of the interpolation problem, defined as

> Given a function $g \in C^0([0,1])$, find the polynomial $p \in \mathbb{P}^n([0,1])$ such that $p(q_i) = g(q_i)$ for all $i = 0, \dots, n$.

Parameters of the problem: the points q_i

Input of the problem: the function g

Output of the problem: the polynomial p in $\mathbb{P}^n$, where $n = len(q) - 1$

If we have a basis v_i for $\mathbb{P}^n$, any polynomial in $\mathbb{P}^n$ can be written as

$$p(x) = p^i v_i(x)$$

A possible algorithm for the interpolation problem then can obtained as:

- Define a basis of $\mathbb{P}^n$ (basis = set of $n+1$ linearly independent functions, whose linear combination covers the entire space $\mathbb{P}^n$). For example: $p_i = x^i$
- Evaluate the function we want to interpolate in q_i. Call the resulting vector $g_i : g(q_i)$ for $i = 0, \dots, n$
- Write $p(x) = p^i v_i$ (sum is implied for repeated indexes), and impose that $p(q_i) = g(q_i)$, i.e.: $p^j v_j(q_i) = g(q_i)$ or:
  - Construct the **interpolation matrix**: $A_{ij} = v_j(q_i)$
  - Solve the linear system $A_{ij}p^j = g(q_i)$

NOTATION: We indicate the coefficients of the inverse of the matrix with coefficients A_{ij} using the following notation: $A^{ij} = (A^{-1})_{ij}$, that is, we define

$$A^{ij}A_{jk} = \delta^i_k$$

Where δ^i_k is one if $i = j$ and zero otherwise (the identity, or Kronecker delta).

How do we estimate the absolute condition number of the problem?

Given a perturbation function δg, the interpolation of $g + \delta g$ results in a perturbed polynomial $p + \delta p$, where, by linearity, δp interpolates δg, i.e., $\delta p^i = A^{ij}\delta g(q_j)$.

We would like to estimate

$$K_{abs} := \sup_{\delta g \in C^0([0,1])} \frac{\|\delta p\|_\infty}{\|\delta g\|_\infty}$$

where $\|v\|_\infty := \max_{x\in[0,1]} |v(x)|$ is the L^∞ norm of the function v.

We start by estimating the numerator:

$$\begin{aligned}
\|\delta p\|_\infty = \|\delta p^i v_i\|_\infty &\leq \max_i |\delta p^i| \quad \left\| \sum_i |v_i| \right\|_\infty \\
&\leq \max_i |A^{ij} \delta g^j| \quad \left\| \sum_i |v_i| \right\|_\infty \\
&\leq C\|A\| \quad \max_i |\delta g^j| \quad \left\| \sum_i |v_i| \right\|_\infty
\end{aligned}$$

Now we observe that the second term is always bounded by $\max_{x\in[0,1]} |g(x)|$, i.e.,

$$\sup_{\delta g \in C^0([0,1])} \frac{\max_i |\delta g^j|}{\|\delta g\|_\infty} \leq 1$$

and therefore:

$$K_{abs} := \sup_{\delta g \in C^0([0,1])} \frac{\|\delta p\|_\infty}{\|\delta g\|_\infty} \leq C\|A\| \quad \left\| \sum_i |v_i| \right\|_\infty .$$

The condition number depends on three parts:

- the constant C, depending on the norm type we chose for A and on n
- the norm of the matrix A
- the quantity $\left\| \sum_i |v_i| \right\|_\infty$

Let's start by evaluating the norm of A for the monomial basis, and the norm of the last term when we increase n:

As we see, the condition number of this matrix explodes as n increases. Since the interpolation problem reduces to solving the matrix constructed as $A_{ij} := p_j(x_i)$, one way to ensure a good condition number (of the matrix!) is to choose the basis such that A is the identity matrix, i.e., to choose the basis such that $v_j(x_i) = \delta_{ij}$. Such a basis is called the **Lagrange basis**, and it is constructed explicitly as:

$$l_i^n(x) := \prod_{j=0, j\neq i}^{n} \frac{(x - x_j)}{(x_i - x_j)} \qquad i = 0, \ldots, n \qquad (9)$$

With this basis, no matrix inversion is required, and we can simply write the **Lagrange interpolant** as

$$\mathcal{L}^n f := \sum_{i=0}^{n} f(x_i) l_i^n(x), \qquad (10)$$

Theorem. *Given a set of $(n+1)$ distinct points $\{x_i\}_{i=0}^n$, there exist a unique Lagrange interpolation of order n.*

The *Lagrange basis* is especially fit for good approximations, since other polynomial sets may be ill-conditioned for the approximation task. The definition of $\mathbb{P}^n$ as in (7) generates the **Vandermonde matrix**

$$B = (q_i)^j = \begin{bmatrix} 1 & q_0 & q_0^2 & \cdots & q_0^n \\ 1 & q_1 & q_1^2 & \cdots & q_1^n \\ \vdots & & & \ddots & \vdots \\ 1 & q_n & q_n^2 & \cdots & q_n^n \end{bmatrix},$$

which is ill-conditioned since for n large, the rightmost columns will be very similar, inducing the matrix hardly invertible.

Two dimensional Lagrange interpolation

To extend to the two dimensional case, we start from two sets of distinct points, say $(n+1)$ points in the x direction, and $m+1$ points in the y direction, in the interval $[0,1]$.

A two dimensional version of the the Lagrange interpolation is then used to construct polynomial approximations of functions of two dimensions. A polynomial from $\Omega := [0,1] \times [0,1]$ to R is defined as:

$$\mathcal{P}^{n,m} : \text{span}\{p_i(x)p_j(y)\}_{i,j=0}^{n,m}$$

and each *multi-index* (i,j) represents a polynomial of order $i+j$, i along x, and j along y. For convenience, we define

$$p_{i,j}(x,y) := p_i(x)p_j(y) \qquad i = 0,\dots,n \qquad j = 0,\dots,m$$

Alternatively we can construct a basis starting from the Lagrange polynomials:

$$l_{i,j}(x,y) := l_i(x)l_j(y) \qquad i = 0,\dots,n \qquad j = 0,\dots,m$$

where we use the same symbol for the polynomials along the two directions for notational convenience, even though they are constructed from two different sets of points.

We define the *Lagrange interpolation* operator $\mathcal{L}^{n,m}$ the operator

$$\mathcal{L}^{n,m} : C^0([0,1] \times [0,1]) \mapsto \mathcal{P}^{n,m}$$

which satisfies

$$(\mathcal{L}^{n,m} f)(q_{i,j}) = f(q_{i,j}), \qquad i = 0,\dots,n, \qquad q_{i,j} := (x_i, y_j)$$

The interpolation error

In this section we estimate the **interpolation error** that is made when replacing a given function f with its interpolating polynomial $\Pi_n f$ at the nodes $x_0, x_1, \dots, x_n$.

Theorem 2. *Let $x_0, x_1, \dots, x_n$ be $n+1$ distinct nodes and let x be a point belonging to the domain of a given function f. Assume that $f \in C^{n+1}(I_x)$, where I_x is the smallest interval containing the nodes $x_0, x_1, \dots, x_n$ and x.*
Then the interpolation error at the point x is given by

$$E_n(x) = f(x) - \Pi_n f(x) = \frac{1}{(n+1)!} f^{(n+1)}(\xi)\omega_{n+1}(x), \quad (11)$$

where $\xi \in I_x$ and ω_{n+1} is the nodal polynomial of degree $n+1$, defined in (6).

Proof. The result is obviously true if x coincides with any of the interpolation nodes (since $f(x_i) = \Pi_n f(x_i)$ and also $\omega_{n+1}(x_i) = 0$ by definition). Otherwise, define, for any $t \in I_x$, the function

$$G(t) = E_n(t) - \omega_{n+1}(t)E_n(x)/\omega_{n+1}(x)$$

(or $G(t) = E_n(t)\omega(x) - E_n(x)\omega(t) = (f(t) - \Pi_n f(t))\omega(x) - (f(x) - \Pi_n f(x))\omega(t)$). Since $f \in C^{(n+1)}(I_x)$ and ω_{n+1} is a polynomial (so $\omega_{n+1} \in C^\infty(I_x)$), then $G \in C^{(n+1)}(I_x)$ and it has at least $n+2$ distinct zeros in I_x, since:

$$\begin{aligned} G(x_i) &= E_n(x_i) - \omega_{n+1}(x_i)E_n(x)/\omega_{n+1}(x) = 0, \quad i = 0, \dots, n, \\ G(x) &= E_n(x) - \omega_{n+1}(x)E_n(x)/\omega_{n+1}(x) = 0. \end{aligned}$$

Then, thanks to the mean value theorem, G' has at least $n+1$ distinct zeros and, by recursion, $G^{(j)}$ admits at least $n+2-j$ distinct zeros. As a consequence, $G^{(n+1)}$ has at least one zero, which we denote by ξ (Rolle). On the other hand, since $E_n^{(n+1)}(t) = f^{(n+1)}(t)$ and $\omega_{n+1}^{(n+1)}(t) = (n+1)!$ we get

$$G^{(n+1)}(t) = E_n^{(n+1)}(t) - \frac{\omega_{n+1}^{(n+1)}(t)}{\omega_{n+1}(x)}E_n(x) = f^{(n+1)}(t) - \frac{(n+1)!}{\omega_{n+1}(x)}E_n(x),$$

which, evaluated at $t = \xi$, gives the desired expression for $E_n(x)$.

□

Since the norm $||\cdot||_\infty$ represents the highest value (the $\sup$) of a function, and we have that $f(x) - \Pi_n f(x) = \frac{1}{(n+1)!}f^{(n+1)}(\xi)\omega_{n+1}(x)$, we can bound the error as

$$||f(x) - \Pi_n f(x)||_\infty \leq \frac{1}{(n+1)!}||f^{(n+1)}||_\infty ||\omega_{n+1}(x)||$$

Theorem. *If f is analytically extendible in an oval $O(a,b,R)$ with $R > 0$ and the x_i are $n+1$ distinct points in $[a,b]$, we have that*

$$|f^{(n+1)}(\xi)| \leq \frac{(n+1)!}{R^{n+1}}||f||_{\infty,\bar{B}(\xi,R)}$$

that is $||f - \Pi_n f||_{\infty,[a,b]} = ||f - \mathcal{L}^n f||_{\infty,[a,b]} \leq ||f||_{\infty,O(a,b,R)}\left(\frac{b-a}{R}\right)^{n+1}$.

This means that increasing the degree n of the interpolation *does not guarantee* a better approximation of f. Indeed, we may have that

$$\lim_{n\to\infty} ||f - \Pi_n f||_\infty = ||f - \mathcal{L}^n f||_\infty = \infty$$

Drawbacks of polynomial interpolation on equally spaced nodes and Runge's counterexample

In this section we analyze the behavior of the interpolation error (11) as n tends to infinity. For this purpose, for any function $f \in C^0([a,b])$, define its **maximum norm**

$$||f||_\infty = \max_{x\in[a,b]} |f(x)|. \tag{12}$$

Then, let us introduce a lower triangular matrix X of infinite size, called the **interpolation matrix** on $[a,b]$, whose entries x_{ij}, for $i,j = 0,1,\dots$, represent the interpolation points of $[a,b]$, with the assumption that on each row the entries are all distinct. So X is an infinite triangular matrix of interpolation points and is like

$$X = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ x_0 & 0 & 0 & \cdots & 0 \\ x_0 & x_1 & 0 & \cdots & 0 \\ \vdots & & & \ddots & \vdots \\ x_0 & x_1 & x_2 & \cdots & x_n \end{bmatrix}$$

Thus, for any $n \geq 0$, the $n+1$-th row of X contains $n+1$ distinct values that we can identify as nodes, so that, for a given function f, we can uniquely define an interpolating polynomial $\Pi_n f$ of degree n at those nodes (any polynomial $\Pi_n f$ depends on X, as well as on f).

Having fixed f and an interpolation matrix X, let us define the interpolation error

$$E_{n,\infty}(X) = ||f - \Pi_n f||_\infty, \quad n = 0, 1, \ldots \tag{13}$$

Next, denote by $p_n^* \in \mathbb{P}_n$ the **best approximation polynomial**, for which

$$E_n^* = ||f - p_n^*||_\infty \leq ||f - q_n||_\infty \quad \forall q_n \in \mathbb{P}_n.$$

The following comparison result holds:

Property 1. *Let $f \in C^0([a,b])$ and X be an interpolation matrix on $[a,b]$. Then*

$$E_{n,\infty}(X) \leq E_n^*(1 + \Lambda_n(X)), \quad n = 0, 1, \ldots, \tag{14}$$

*where $\Lambda_n(X)$ denotes the **Lebesgue constant** of X, defined as*

$$\Lambda_n(X) = \left\| \sum_{j=0}^{n} |l_j^{(n)}| \right\|_\infty, \tag{15}$$

and where $l_j^{(n)} \in \mathbb{P}^n$ is the j-th characteristic polynomial associated with the $n+1$-th row of X, that is, satisfying $l_j(x_{nk}) = \delta_{jk},\ j,k = 0, 1, \ldots$.

Proof. How distant are we from the best approximation (B.A.)? Let us define

$$||\mathcal{L}^n|| := \sup_{\substack{f \in C(I) \\ ||f||_\infty < 1}} ||\mathcal{L}^n f||_\infty = \sup_{\substack{f \in C(I) \\ ||f||_\infty < 1}} \left\| \sum_i l_i^n f(x_i) \right\|_\infty \tag{16}$$

Then, since $\forall q \in \mathbb{P}^n$ we have that $\mathcal{L}^n q = q$, we have

$$\begin{aligned} E_{n,\infty}(X) &= ||f - \mathcal{L}^n f|| = ||f - \mathcal{L}^n f + \mathcal{L}^n q - q|| \\ &= ||f - q - \mathcal{L}^n(f - q)|| \leq ||f - q|| + ||\mathcal{L}^n(f - q)|| \\ &\leq ||f - q|| - ||\mathcal{L}^n||\ ||f - q|| = (1 - ||\mathcal{L}^n||)||f - q|| \\ &\leq (1 - ||\mathcal{L}^n||)||f - p_n^*|| \end{aligned}$$

since it is valid $\forall q \in \mathbb{P}^n$ and so also for the polynomial $p_n^* = BA(f)$, the best approximation of f. □

Since E_n^* does not depend on X, all the information concerning the effects of X on $E_{n,\infty}(X)$ must be looked for in $\Lambda_n(X)$. Although there exists an interpolation matrix X^* such that $\Lambda_n(X)$ is minimized, it is not in general a simple task to determine its entries explicitly. We shall see that the zeros of the Chebyshev polynomials provide on the interval $[-1,1]$ an interpolation matrix with a very small value of the Lebesgue constant.

Besides, we have that

$$||\mathcal{L}^n f|| := \sup_{\substack{f \in C(I) \\ ||f||_\infty < 1}} ||\sum_i l_i^n f(x_i)||_\infty \leq ||\sum_i |l_i^n(x)|\ ||_\infty =: \Lambda_n(X)$$

On the other hand, for any possible choice of X, there exists a constant $C > 0$ such that

$$\Lambda_n(X) > \frac{2}{\pi}\log(n+1) - C, \quad n = 0, 1, \dots$$

Erdos' theorem. *For each infinite triangular matrix of interpolation points X, $\exists C > 0$ such that*

$$||\mathcal{L}^n|| \geq \frac{2}{\pi}\log(n+1) - C.$$

This property shows that $\Lambda_n(X) \to \infty$ as $n \to \infty$. This fact has important consequences: in particular, it can be proved that, given an interpolation matrix X on an interval $[a, b]$, there always exists a continuous function f in $[a, b]$, such that $\Pi_n f$ does not converge uniformly (that is, in the maximum norm) to f.

Faber's theorem. *For all X, $\exists f \in C^0([a, b])$ such that $||f - \mathcal{L}^n f||_\infty \nrightarrow 0$.*

Thus, polynomial interpolation does not allow for approximating any continuous function, as demonstrated by the following example.

Runge's counterexample. Suppose we approximate the following function

$$f(x) = \frac{1}{1 + x^2}, \quad x \in I = [-5, 5], \tag{17}$$

using Lagrange interpolation on equally spaced nodes $x_i = -5 + \frac{10i}{n}$. It can be checked that some points x exist within the interpolation interval such that

$$\lim_{n \to \infty} |f(x) - \Pi_n f(x)| \neq 0.$$

In particular, Lagrange interpolation diverges for $|x| > 3.63\dots$. This phenomenon is particularly evident in the neighborhood of the end points of the interpolation interval, as shown in the following figure, and is due to the choice of equally spaced nodes.

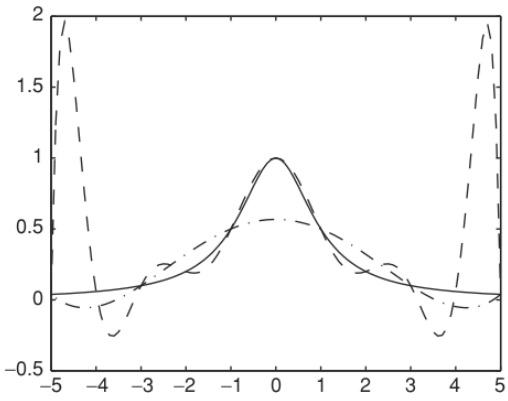


Fig. 8.2. Lagrange interpolation on equally spaced nodes for the function $f(x) = 1/(1 + x^2)$: the interpolating polynomials $\Pi_5 f$ and $\Pi_{10} f$ are shown in dotted and dashed line, respectively

The presence of sever oscillations of $\Pi_n f$ w.r.t. f, especially near the endpoints, indicates a lack of convergence. This is also called **Runge's phenomenon**.

We shall see that resorting to suitably chosen nodes will allow for uniform convergence of the interpolating polynomial to the function f to hold.

■

Stability of Polynomial Interpolation

Let $f \in C^0([a,b])$ and X be an interpolation matrix on $[a,b]$. Let us consider a set of function values $\{\tilde{f}(x_i)\}$ which is a perturbation of the data $f(x_i)$ relative to the nodes x_i, with $i = 0, \ldots, n$, in an interval $[a,b]$. The perturbation may be due, for instance, to the effect of rounding errors, or may be caused by an error in the experimental measure of the data.

We want to estimate the impact of the perturbed values $\tilde{f}(x_i)$ on the interpolant $\Pi_n f = \mathcal{L}^n f$. Denoting by $\Pi_n \tilde{f}$ the interpolating polynomial on the set of values $\tilde{f}(x_i)$, we have

$$\begin{aligned} ||\Pi_n f - \Pi_n \tilde{f}||_\infty &= \max_{j=0} \left| \sum_{a \le x \le b}^{n} (f(x_j) - \tilde{f}(x_j)) l_j(x) \right| \\ &\le \Lambda_n(X) \max_{i=0,\ldots,n} |f(x_i) - \tilde{f}(x_i)|, \end{aligned}$$

where $\Lambda_n(X)$ denotes the **Lebesgue constant** of X, depending on the interpolation nodes, defined in (15).

As a consequence, small changes on the data give rise to small changes on the interpolating polynomial only if the Lebesgue constant is small. This constant plays the role of the **condition number** for the interpolation problem.

As previously noticed, Λ_n grows as $n \to \infty$ and in particular, in the case of Lagrange interpolation on equally spaced nodes, it can be proved that

$$\Lambda_n(X) \simeq \frac{2^{n+1}}{en \log n}, \tag{18}$$

where $e \simeq 2.7183$ is the Euler's number. This shows that, for n large, this form of interpolation can become unstable. Notice also that so far we have completely neglected the errors generated by the interpolation process in constructing $\Pi_n f$. However, it can be shown that the effect of such errors is generally negligible.

Example 2. On the interval $[-1,1]$ let us interpolate the function $f(x) = \sin(2\pi x)$ at 22 equally spaced nodes x_i. Next, we generate a perturbed set of values $\tilde{f}(x_i)$ of the function evaluations $f(x_i) = \sin(2\pi x_i)$ with $\max_{i=0,\ldots,21} |f(x_i) - \tilde{f}(x_i)| \simeq 9.5 \cdot 10^{-4}$. In the following figure we compare the polynomials $\Pi_{21} f$ and $\Pi_{21} \tilde{f}$: notice how the difference between the two interpolating polynomials, around the end points of the interpolation interval, is much larger than the impressed perturbation (actually, $||\Pi_{21} f - \Pi_{21} \tilde{f}||_\infty \simeq 1.5926$).

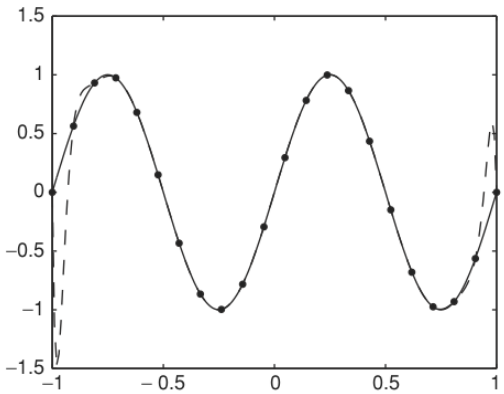
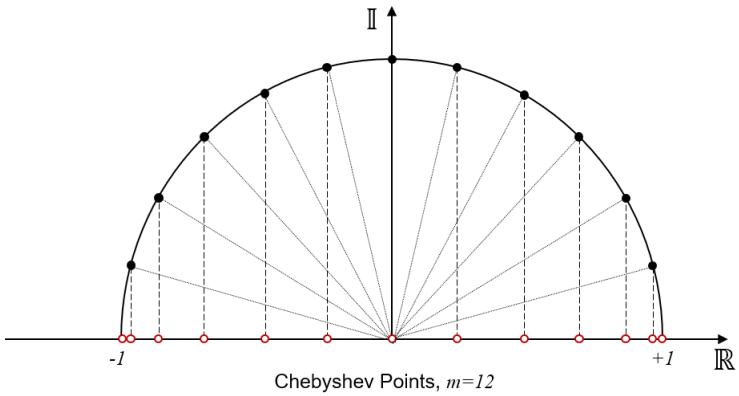


Fig. 8.3. Instability of Lagrange interpolation. In solid line $\Pi_{21} f$, on unperturbed data, in dashed line $\Pi_{21} \tilde{f}$, on perturbed data, for Example 8.2

Chebyshev nodes

In order to minimize the Lebesgue constant $\Lambda_n(X)$ and thus avoid Runge's phenomenon, we can use the **Chebyshev nodes** on the interval $[a, b]$, defined as

$$x_i = \frac{a+b}{2} + \frac{b-a}{2}\hat{x}_i, \quad \text{where } \hat{x}_i = \cos\left(\frac{(2i+1)}{2(n+1)}\pi\right), \qquad i = 0, \dots, n$$

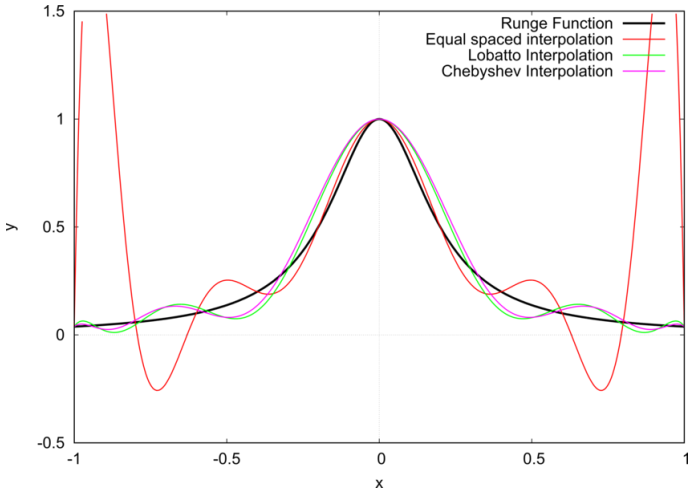


Chebyshev Points, *m=12*

The nodes are equispaced on the semi-circumference of diameter $[a, b]$ while they are clustered toward the endpoints of the interval. For the Chebyshev nodes, if f is continuously differentiable in $[a, b]$ then $\mathcal{L}^n f$ converges to f as $n \to \infty$.

The Gauss-Lobatto nodes instead are

$$x_i = \frac{a+b}{2} + \frac{b-a}{2}\hat{x}_i, \quad \text{where } \hat{x}_i = \cos\left(\frac{\pi i}{n}\right), \qquad i = 0, \dots, n$$



For the Chebyshev nodes we have that $||\mathcal{L}^n||_\infty \leq \frac{2}{\pi}\log(n+1) + 1$, so for the Erdos' theorem we have that $\exists c$ such that

$$\frac{2}{\pi}\log(n+1) - c \leq ||\mathcal{L}^n||_\infty \leq \frac{2}{\pi}\log(n+1) + 1$$

Instead, for the Lagrange equispaced nodes we have that, as seen in (18),

$$\Lambda_n(X) \leq \frac{2^{n+1}}{en\log n},$$

The Faber's theorem proves that, even on Chebyshev nodes not all continuous functions will converge when used for interpolation.

So we have that interpolation does not work well... Is all lost? NO! Let's give one positive result:

Weierstrass approximation theorem. *Suppose that $f \in C^0([a,b])$. Then, $\forall \varepsilon > 0$ there exist n and a polynomial $p \in \mathbb{P}^n$ such that $||f-p||_\infty \leq \varepsilon$, $\forall x \in [a,b]$.*

It shows that polynomial functions are dense in $C^0([a,b])$, and each polynomial can be uniformly approximated by one with rational coefficients.

Bernstein polynomials

The $n+1$ **Bernstein (basis) polynomials** of degree n over the interval $[0,1]$ are defined as

$$b_{n,k}(x) = \binom{n}{k} x^k (1-x)^{n-k}, \qquad n = 0, 1, \dots, \quad k = 0, \dots, n, \quad (19)$$

They can be obtained by the following recursive formula

$$\begin{cases} b_{n,0}(t) = (1-t)^n, \\ b_{n,k}(t) = (1-t) b_{n-1,k}(t) + t b_{n-1,k-1}(t), \quad k = 1, \dots, n, \ t \in [0,1]. \end{cases}$$

It is easily seen that $\{b_{n,k}, k = 0, \dots, n\}$ provides a basis for $\mathbb{P}^n$. A linear combination of $b_{n,k}$ is called a **Bernstein polynomial** or **polynomial in Bernstein form** of degree n based on function f:

$$B_n f(x) = \sum_{k=0}^{n} f\left(\frac{k}{n}\right) b_{n,k}(x) = \sum_{k=0}^{n} f\left(\frac{k}{n}\right) \binom{n}{k} x^k (1-x)^{n-k}. \quad (20)$$

We have that $\forall x$, $B_n f(x)$ is a weighted average of the $n+1$ values $f\left(\frac{k}{n}\right)$, called **Bernstein coefficients**.

Properties:

- $\sum_{k=0}^{n} b_{n,k}(x) = \sum_{k=0}^{n} \binom{n}{k} x^k (1-x)^{n-k} = (x+1-x)^n = 1^n = 1 \ \forall x \in [a,b]$, thanks to the *binomial theorem*.
- $b_{n,k} \geq 0, \ \forall x \in [0,1]$ and $b_{n,k} \in \mathbb{P}^n$ for $k = 0, \dots, n$.
- B_n is a **linear positive operator**: $B_n f \geq 0$ if $f \geq 0$.
- $b_{n,k}(\frac{j}{n}) \neq \delta_{kj}$, so they are different from the Lagrange basis!
- $B_n f\left(\frac{k}{n}\right) \neq f\left(\frac{k}{n}\right)$ and if $f \in C^0([0,1])$ we have that $B_n f(x) \to f(x)$ as $n \to \infty$.
- $\forall p \in \mathbb{P}^2, B_n p \to p$, so we have that $\mathcal{L}^n p \equiv p$ for $n < 2$.

We have then that the convergence is **not pointwise** as in the interpolation, but it is **uniform**:

$$\lim_{n \to \infty} ||f(x) - B_n f(x)||_\infty = 0 \quad \text{with } 0 \leq x \leq 1$$

Direct computations: $\forall f_0 \in \mathbb{P}^0, f_1 \in \mathbb{P}^1, f_2 \in \mathbb{P}^2, f_i = x^i$ we have that:

$$\begin{aligned} n \geq 2 \quad & B_n f_0 = f_0, \qquad B_n f_1 = f_1, \qquad B_n f_2 = \frac{n-1}{n} f_2 \\ & B_n 1 = 1, \qquad B_n x = x, \qquad B_n x^2 = \frac{n-1}{n} x^2 + \frac{1}{n} x \neq x^2 \end{aligned}$$

Theorem. *Let B_n be a sequence of linear positive operators such that $B_n f$ converges uniformly to f, $\forall f \in \mathbb{P}^2$. Then $B_n f$ converges uniformly to f, $\forall f \in C(I)$.*

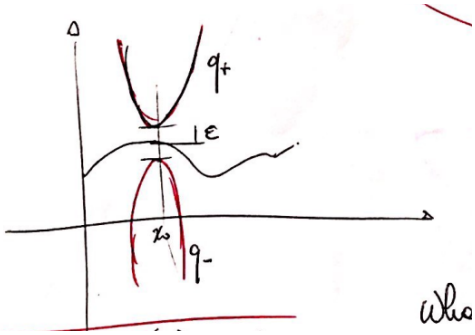
Idea of proof. We have that $\forall f \in C(I), \forall x_0 \in I$, we can find a quadratic function $q > f$ but such that $q(x_0)$ is close to $f(x_0)$: $|q(x_0) - f(x_0)| < \varepsilon$. The same can be done with $q < f$.
For $n > \bar{n}$, $|B_n q(x_0) - q(x_0)| < \varepsilon$, that is $|B_n q(x_0) - f(x_0)| < \varepsilon$.

Qualitative proof of Weierstrass theorem. Let $f \in C(I)$, with I compact. Then we have that f is uniformly continuous, so $x_1, x_2 \in I$

$$\forall \varepsilon > 0, \exists \delta > 0 \text{ s.t. if } |x_1 - x_2| < \delta \text{ then } |f(x_1) - f(x_2)| \leq \varepsilon$$

For any x_0, let's set

$$q_{\pm}(x) = f(x_0) \pm \left(\varepsilon + \frac{2||f||_\infty}{\delta^2}(x - x_0)^2 \right)$$



We have that $q_+(x) \geq f(x)\ \forall x$, while $q_-(x) \leq f(x)\ \forall x$.

We have that $q(x) = a + bx + cx^2$, with $|a|, |b|, |c| \leq M$, where M depends on $||f||, \varepsilon, \delta$ but *not* on x_0.

Then we choose n large enough such that $||B_n x^i - x^i||_\infty \leq \frac{\varepsilon}{M}$ for $i = 0, 1, 2$. So with the triangle inequality we get $||B_n q_\pm - q_\pm||_\infty \leq 3\varepsilon$. Then we have that

$$B_n f(x_0) \leq B_n q_+(x_0) \leq q_+(x_0) + 3\varepsilon = f(x_0) + 4\varepsilon$$
$$B_n f(x_0) \geq B_n q_-(x_0) \geq q_-(x_0) - 3\varepsilon = f(x_0) - 4\varepsilon$$

that is

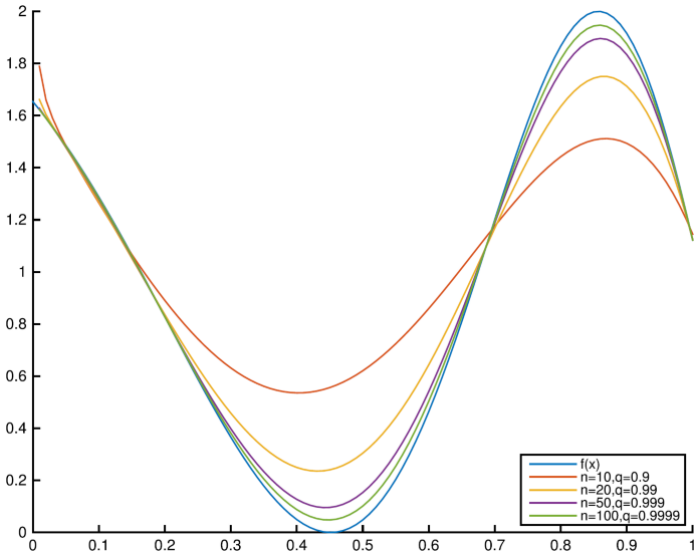
$$-4\varepsilon \leq B_n f(x_0) - f(x_0) \leq 4\varepsilon \quad \Longrightarrow \quad ||B_n f - f||_\infty \leq 4\varepsilon$$

□

This is very robust but very slow.

$$||f - B_n f||_\infty = O\left(\frac{1}{n}\right)$$

This hold also for all C^2 functions! $||f - B_n f||_\infty \leq \frac{1}{f_n}||f''||_\infty$.



Piecewise Lagrange Interpolation

We can build a piecewise linear interpolant of f to avoid Runge's effect when the number of nodes increases. f is a piecewise linear continuous function also called **finite element interpolant**.

We have outlined the fact that, for equally spaced interpolating nodes, uniform convergence of $\Pi_n f$ to f is not guaranteed as $n \to \infty$. On the other hand, using equally spaced nodes is clearly computationally convenient and, moreover, Lagrange interpolation of low degree is sufficiently accurate, provided sufficiently small interpolation intervals are considered.

Therefore, it is natural to introduce a partition $\mathcal{T}_h$ of $[a,b]$ into K subintervals $I_j = [x_j, x_{j+1}]$ of length h_j, with $h = \max_{0 \le j \le K-1} h_j$, such that $[a,b] = \bigcup_{j=0}^{K-1} I_j$ and then to employ Lagrange interpolation on each I_j using $k+1$ equally spaced nodes $\left\{ x_j^{(i)}, 0 \le i \le k \right\}$ with a small k.

For $k \ge 1$, we introduce on $\mathcal{T}_h$ the piecewise polynomial space

$$X_h^k = \left\{ v \in C^0([a,b]) : v|_{I_j} \in \mathbb{P}^k(I_j) \; \forall I_j \in \mathcal{T}_h \right\}$$

which is the space of the continuous functions over $[a,b]$ whose restrictions on each I_j are polynomials of degree $\le k$. Then, for any continuous function f in $[a,b]$, the **piecewise interpolation polynomial** $\Pi_h^k f$ coincides on each I_j with the interpolating polynomial of $f|_{I_j}$ at the $k+1$ nodes $\left\{ x_j^{(i)}, 0 \le i \le k \right\}$. As a consequence, if $f \in C^{k+1}([a,b])$, using (11) within each interval we obtain the following error estimate

$$||f - \Pi_h^k f||_\infty \le C h^{k+1} ||f^{(k+1)}||_\infty. \tag{21}$$

Note that a small interpolation error can be obtained even for low k provided that h is sufficiently "small".

Example. We have that if $n = 1$, then the piecewise interpolation polynomial is

$$\Pi_h^1 f(x) = f(x_{j-1}) \frac{x - x_j}{x_{j-1} - x_j} + f(x_j) \frac{x - x_{j-1}}{x_j - x_{j-1}} \quad \text{if } x \in I_j.$$

Besides estimate (21), convergence results in integral norms exist. For this purpose, we introduce the following space

$$L^2(a,b) = \left\{ f : (a,b) \to \mathbb{R}, \int_a^b |f(x)|^2 dx < +\infty \right\}. \tag{22}$$

with

$$||f||_{L^2(a,b)} = \left(\int_a^b |f(x)|^2 dx \right)^{1/2}. \tag{23}$$

Formula (23) defines a norm for $L^2(a,b)$. We warn the reader that the integral of the function $|f|^2$ in (22) has to be intended in the Lebesgue sense. In particular, f needs not be continuous everywhere.

Theorem. *Let $0 \le m \le k+1$, with $k \ge 1$ and assume that $f^{(m)} \in L^2(a,b)$ for $0 \le m \le k+1$; then there exists a positive constant C, independent of h, such that*

$$||(f - \Pi_h^k f)^{(m)}||_{L^2(a,b)} \le C h^{k+1-m} ||f^{(k+1)}||_{L^2(a,b)}.$$

In particular, for $k = 1$, and $m = 0$ or $m = 1$, we obtain

$$
\begin{aligned}
||f - \Pi_h^1 f||_{L^2(a,b)} &\leq C_1 h^2 ||f''||_{L^2(a,b)}, \\
||(f - \Pi_h^1 f)''||_{L^2(a,b)} &\leq C_2 h ||f''||_{L2(a,b)},
\end{aligned}
$$

for two suitable positive constants C_1 and C_2.

More on interpolation

- We can perform interpolation by cubic splines, which are piecewise cubic functions $f \in C^2$.
- While the **minmax approximation** we used so far is based on the norm $||\cdot||_\infty$, the **least squares approximation** uses the Euclidean norm $||\cdot||_2$ to minimize the $MSE = \sum_{i=0}^{n} (y_i - \Pi_n f(x_i))^2$.
- Piecewise linear and splines are well suited to approximate data and functions in several dimensions.
- Trigonometric interpolation is well suited to approximate periodic functions. We have that $\Pi_n f$ is a linear combination of $\sin$ and $\cos$ functions. The FFT algorithm (Fast Fourier transform) and the IFFT algorithm (Inverse Fast Fourier transform) allow for an efficient computation of Fourier coefficients for a trigonometric interpolant from node values.

Best approximation

Approximation

Approximating a set of data or a function in $[a, b]$ consists in finding a suitable function f that represents them with enough accuracy.

We can use Taylor polynomials to approximate complex functions, but they require many computations and have unpredictable behaviors on the side of the domain.

Definition. If X is a Banach space and $M \subseteq X$ is a subset, we say that $p \in M$ is the **best approximation** of a function $f \in M$ when

$$||f - p|| = E(f) := \inf_{q \in M} ||f - q||$$

Theorem (Existence theorem) **.** *If M is a finite-dimensional subspace of X, so if $\exists \{v_n\}$ s.t. $M = \text{span}\{v_n\}$, then $\exists p$ the best approximation of f in M.*

Definition. A vector space X is **strictly convex** if $f \neq g$ with $||f|| = ||g|| = 1$ and $0 < \theta < 1$, then $||\theta f + (1 - \theta)g|| < 1$. Geometrically, this means that if any x, y on the unit sphere ∂B are joined by a segment that touches ∂B only in x and y.

Theorem (Uniqueness theorem) **.** *If X is **strictly convex**, then the best approximation p is **unique**.*

Proof. We should prove the existence and the in Hilbert spaces uniqueness. Let's prove the uniqueness: let $p_1 \neq p_2$, we have that

$$\begin{aligned} E(f) &= ||f - p_1|| = ||f - p_2|| \\ &\leq \left|\left| f - \frac{1}{2}(p_1 + p_2) \right|\right| = \left|\left| \frac{1}{2}(f - p_1) + \frac{1}{2}(f - p_2) \right|\right| \\ &< \frac{1}{2}||f - p_1|| + \frac{1}{2}||f - p_2|| = E(f) \end{aligned}$$

(the first inequality holds because $\frac{1}{2}(p_1 + p_2)$ is any function while p_1 and p_2 are the best approximations) that is impossible, so it must be $p_1 = p_2$.

□

Best approximation in Hilbert Spaces

Recall that an Hilbert space is a Banach space plus a scalar product $(\cdot, \cdot)$ with norm defined by $||u||^2 := (u, u)$ and that p is B.A. of f w.r.t. a chosen norm if we have that $||f - p|| \leq ||f - q||$ $\forall q \in \mathbb{P}^n$.

Theorem (Best approximation theorem)**.** *Let H be a Hilbert space. Given a function $f \in H$, then p is best approximation of f in H (from $V \subset H$) if and only if*

$$(f, q) = (p, q) \; \forall q \in V.$$

Proof. ($\Rightarrow$) If p is B.A. of f in H, then $||f - p||^2 = E(f)^2 := \inf_{q \in V} ||f - q||^2$ and so we have that for any perturbation $p + tq$ of p, it holds

$$||f - p||^2 \leq ||f - p + tq||^2 \quad \forall t > 0, \forall q \in V$$

Consider that $||a + b||^2 - ||a - b||^2 = 4(a, b)$, then we have that, if $a = f - p + \frac{t}{2}q$ and $b = \frac{t}{2}q$,

$$0 \leq ||f-p+tq||^2 - ||f-p||^2 = \left|\left|f-p+\frac{t}{2}q+\frac{t}{2}q\right|\right|^2 - \left|\left|f-p+\frac{t}{2}q-\frac{t}{2}q\right|\right|^2$$
$$= 4\left(f-p+\frac{t}{2}q, \frac{t}{2}q\right) = 4\left(f-p, \frac{t}{2}q\right) + 4\left(\frac{t}{2}q, \frac{t}{2}q\right) = 2t\,(f-p,q) + t^2||q||^2$$

so we have that $(f-p,q) \geq -\frac{t}{2}||q||^2$.

By adding instead of tq the term $-tq$, with the same reasoning we have that $(f-p,q) \leq \frac{t}{2}||q||^2$, so we have that $\forall t > 0, \forall q \in V$

$$-\frac{t}{2}||q||^2 \leq (f-p,q) \leq \frac{t}{2}||q||^2$$

which implies that $(f-p,q) = 0$ since a t can be chosen to bound it on both sides. Then we have that

$$(f-p,q) = 0 \iff (f,q) - (p,q) = 0 \iff (f,q) = (p,q)$$

for each $q \in V$, so we have our thesis.

($\Leftarrow$) If $(f,q) = (p,q)\ \forall q \in V \iff (f-p,q) = 0\ \forall q \in V$, then we have that

$$||f-q||^2 = ||f-p+p-q||^2 = ||f-p||^2 + ||p-q||^2 + 2(f-p, p-q)$$

where $2(f-p,p-q) = 0$ since $p-q \in V$ (both $p, q \in V$).

So we have that $||f-q||^2 = ||f-p||^2 + ||p-q||^2\ \forall q \in V$, which implies that $||f-p||^2 \leq ||f-q||^2$ and so $||f-p|| \leq ||f-q||\ \forall q \in V$, and this means that p is B.A. of f in H. □

Now consider $L^2(0,1)$, that is a Hilbert space where the scalar product between vectors is $(a,b) := \int_0^1 ab\,ds$ for $a, b \in L^2(0,1)$ and the norm is $||a|| := \sqrt{\int_0^1 |a|^2 ds}$, and take the space $V = \text{span}\{v_i\}_{i=1}^n$.

The best approximation in L^2. We want the best approximation (in Hilbert Spaces) of the function f, on the space $V = \text{span}\{v_i\}$. Then we have seen that $p \in V$ is best approximation of f if and only if:

$$(f,v) = (p,v), \quad \forall v \in V.$$

In particular, for every basis functions $v_i \in V$ we have $(p,v_i) = (f,v_i)$, for $i = 1, \dots, n$. Since $p \in V$, we have that it can be expressed as a linear combination of the basis functions v_i and is uniquely defined by the coefficients p_j: $p = \sum_{j=1}^n p_j v_j$, so we have that

$$(p,v_i) = (\sum_j p_j v_j, v_i) = \sum_j p_j (v_j, v_i).$$

Collecting this informations together we get:

$$\sum_{j=1}^n p_j (v_j, v_i) = (f, v_i),\ \forall v_i \in V, i = 1, \dots, n \quad \iff \quad Mp = F$$

where M and F are matrices such that $M_{ij} := (v_j, v_i) = \int_0^1 v_j v_i\,ds$ and $F_i := (f, v_i) = \int_0^1 f v_i\,ds$.

If we set $V^n := \text{span}\{x^i\}_{i=0}^{n-1}$ we will then have that $v_i = x^i$, then

$$M_{ij} := \int_0^1 x^j x^i\,dx = \frac{x^{j+i+1}}{j+i+1}\bigg|_0^1 = \frac{1}{j+i+1}.$$

that is called the $n \times n$ **Hilbert matrix** H, which is invertible but it is very ill conditioned, so it is difficult to invert, because of collinear lines. Its condition number is

$$K(H) = ||H||\,||H^{-1}|| \sim O\left(\left(1+\sqrt{2}\right)^{4n}/\sqrt{n}\right).$$

When n increases K explodes, which is very bad. We would like to have $H = I$ the identity, which means $M_{ij} = \delta_{ij}$, so we use the **Legendre basis function**, which are orthonormal basis, to make it orthonormal (perpendicular, a.k.a. diagonal) w.r.t. L^2.

We want $v_i \in \mathbb{P}^n = V^n = \text{span}\{x^i\}$ s.t. $M_{ij} = (v_i, v_j) = \delta_{ij}$. To build it, we use the **Gram-Schmidt process**:

$$\begin{cases} v_0 = 1 & f \text{ s.t. } \int_0^1 f = 1 \\ v_{i+1} = \frac{k_{i+1}}{||k_{i+1}||} & \text{where } k_{i+1} = [x\,v_i - \sum_i (x\,v_i, v_i) v_i] \end{cases}$$

or, alternatively

$$\begin{cases} p_0(x) = 1 \\ p_k(x) = x^k - \displaystyle\sum_{j=0}^{k-1} \frac{(x^k, p_j(x))}{(p_j(x), p_j(x))} \\ \qquad\quad = x\,p_{k-1}(x) - \displaystyle\sum_{j=0}^{k-1} \frac{(x p_{k-1}(x), p_j(x))}{(p_j(x), p_j(x))} \end{cases}$$

The set of additive basis having unity as first element. This ensures orthogonality between basis functions.

As the degree i increases, we have that $v_{i+1} = \frac{k_{i+1}}{||k_{i+1}||} \to \infty$ since $x^{i+1} \to \infty$ and thus $k^{i+1} \to 0$.
We can avoid instability by using $v_{i+1} = \frac{k_{i+1}}{k_{i+1}(0)}$ instead.

The points created with Gram-Schmidt represent the **Legendre basis**. The make the best approximation p easy to compute, since M becomes easy to invert and we have a diagonal matrix formed by orthogonal basis $p = M^{-1}F$.

Integration

Let f be a real integrable function over the interval $[a,b]$: $f:[a,b]\to\mathbb{R}$. Computing explicitly the definite integral

$$I(f)=\int_a^b f(x)dx \tag{1}$$

may be difficult or even impossible. Integration is very expensive from a numerical point of view if f is complicated. Our purpose is to make it simpler. Any explicit formula that is suitable for providing an approximation of $I(f)$ is said to be a **quadrature formula** or **numerical integration formula**.

An example can be obtained by replacing f with an approximation f_n, depending on the integer $n\geq 0$, then computing $I(f_n)$ instead of $I(f)$. Letting $I_n(f)=I(f_n)$, we have

$$I_n(f)=\int_a^b f_n(x)dx,\quad n\geq 0.$$

The dependence on the end points a,b is always understood, so we write $I_n(f)$ instead of $I_n(f;a,b)$.

If $f\in C^0([a,b])$, the **quadrature error** $E_n(f)=I(f)-I_n(f)$ satisfies

$$\begin{aligned}|E_n(f)|=|I(f)-I_n(f)|&=\left|\int_a^b f(x)dx-\int_a^b f_n(x)dx\right|\\&\leq\int_a^b|f(x)-f_n(x)|dx\leq(b-a)||f-f_n||_\infty.\end{aligned}$$

Therefore, if for some n, $||f-f_n||_\infty<\varepsilon$, then $|E_n(f)|\leq\varepsilon(b-a)$.

The approximant f_n must be easily integrable, which is the case if, for example, $f_n\in\mathbb{P}^n$. In this respect, a natural approach consists of using $f_n=\Pi_n f$, the interpolating Lagrange polynomial of f over a set of $n+1$ distinct nodes $\{x_i\}$, with $i=0,\dots,n$. By doing so, from (1) it follows that

$$I_n(f)=\sum_{i=0}^n f(x_i)\int_a^b l_i(x)dx, \tag{2}$$

where l_i is the characteristic Lagrange polynomial of degree n associated with node x_i (see the Interpolation section). We notice that (2) is a special instance of the following quadrature formula

$$I_n(f)=\sum_{i=0}^n\alpha_i f(x_i), \tag{3}$$

where the coefficients α_i of the linear combination are given by $\int_a^b l_i(x)dx$.

Formula (3) is a weighted sum of the values of f at the points x_i, for $i=0,\dots,n$. These points are said to be the **nodes** of the quadrature formula, while the numbers $\alpha_i\in\mathbb{R}$ are its **coefficients** or **weights**. Both weights and nodes depend in general on n; again, for notational simplicity, this dependence is always understood.

The formula (2), called **Lagrange quadrature formula**, can be generalized to the case where also the values of the derivative of f are available. This leads to the **Hermite quadrature formula**

$$I_n(f) = \sum_{k=0}^{1} \sum_{i=0}^{n} \alpha_{ik} f^{(k)}(x_i), \tag{4}$$

where the weights are now denoted by α_{ik}.

Both (2) and (4) are **interpolatory quadrature formula**, since the function f has been replaced by its interpolating polynomial (Lagrange and Hermite polynomials, respectively). We define the **degree of exactness** of a quadrature formula as the maximum integer $r \geq 0$ for which

$$I_n(f) = I(f), \quad \forall f \in \mathbb{P}^r.$$

Any interpolatory quadrature formula that makes use of $n+1$ distinct nodes has degree of exactness equal to at least n. Indeed, if $f \in \mathbb{P}^n$, then $\Pi_n f = f$ and thus $I_n(\Pi_n f) = I(\Pi_n f) = I(f)$. The converse statement is also true, that is, a quadrature formula using $n+1$ distinct nodes and having degree of exactness equal at least to n is necessarily of interpolatory type.

As we will see, the degree of exactness of a Lagrange quadrature formula can be as large as $2n+1$ in the case of the so-called Gaussian quadrature formulae.

The Midpoint or Rectangle Formula

This formula is obtained by replacing f over $[a,b]$ with the constant function equal to the value attained by f at the midpoint of $[a,b]$.
This yields

$$I_0(f) = \int_a^b \Pi_0(x)dx = (b-a)\, f\left(\frac{a+b}{2}\right), \tag{5}$$

with weight $\alpha_0 = b-a$ and node $x_0 = (a+b)/2$. If $f \in C^2([a,b])$, the quadrature error is

$$E_0(f) = \frac{(b-a)^3}{24} f''(\xi) = \frac{h^3}{3} f''(\xi), \quad h = \frac{b-a}{2}, \tag{6}$$

where ξ lies within the interval (a,b).

Indeed, expanding f in a Taylor's series around $c = (a+b)/2$ and truncating at the second-order, we get

$$f(x) = f(c) + f'(c)(x-c) + f''(\xi(x,c))(x-c)^2/2,$$

from which, integrating on (a,b) and using the mean-value theorem we have that:

$$\begin{aligned}
\int_a^b f(x)dx &= \int_a^b [f(c) + f'(c)(x-c) + f''(\xi(x,c))(x-c)^2/2]dx \\
&= f(c)(b-a) + f'(c)\frac{(x-c)^2}{2}\bigg|_a^b + f''(\xi)\frac{(x-c)^3}{6}\bigg|_a^b \\
&= f(c)(b-a) + f'(c)\left[\frac{(b-c)^2}{2} - \frac{(a-c)^2}{2}\right] + f''(\xi)\left[\frac{(b-c)^3}{6} - \frac{(a-c)^3}{6}\right]
\end{aligned}$$

We have that $b - c = b - \frac{a+b}{2} = \frac{2b-a-b}{2} = \frac{b-a}{2}$ and $a - c = a - \frac{a+b}{2} = \frac{2a-a-b}{2} = \frac{a-b}{2}$, so

$$\begin{aligned}
&= f(c)(b-a) + f'(c)\left[\frac{((b-a)/2)^2}{2} - \frac{((a-b)/2)^2}{2}\right] + f''(\xi)\left[\frac{((b-a)/2)^3}{6} - \frac{((a-b)/2)^3}{6}\right] \\
&= f(c)(b-a) + f''(\xi)\left[\frac{(b-a)^3}{6\cdot 8} - \frac{(a-b)^3}{6\cdot 8}\right] = f(c)(b-a) + f''(\xi)\left[\frac{(b-a)^3}{48} + \frac{(b-a)^3}{48}\right] \\
&= f(c)(b-a) + f''(\xi)\frac{2(b-a)^3}{48} = f(c)(b-a) + \frac{(b-a)^3}{24} f''(\xi).
\end{aligned}$$

Also, we have that $\int_a^b f'(c)(x-c)dx = 0$ since $f'(c)(x-c)$ is the line passing through $(c,0)$ with inclination $f'(c)$, with c the midpoint of $[a,b]$, so there are two equal regions one above the x-axis and one below that cancels out.

So (6) follows form the following passages:

$$\begin{aligned} E_0(f) = I(f) - I_0(f) &= \int_a^b f(x)dx - (b-a)f(c) \\ &= f(c)(b-a) + \frac{(b-a)^3}{24}f''(\xi) - (b-a)f(c) \\ &= \frac{(b-a)^3}{24}f''(\xi) = \frac{h^3}{3}f''(\xi), \quad h = \frac{b-a}{2} \end{aligned}$$

From this, it turns out that (5) is exact for constant and affine functions (since in both cases $f''(\xi) = 0$ for any $\xi \in (a,b)$), so that the midpoint rule has degree of exactness equal to 1.

It is worth noting that if the width of the integration interval $[a,b]$ is not sufficiently small, the quadrature error (6) can be quite large. This drawback is common to all the numerical integration formulae that will be described in the two forthcoming sections and can be overcome by resorting to their composite counterparts.

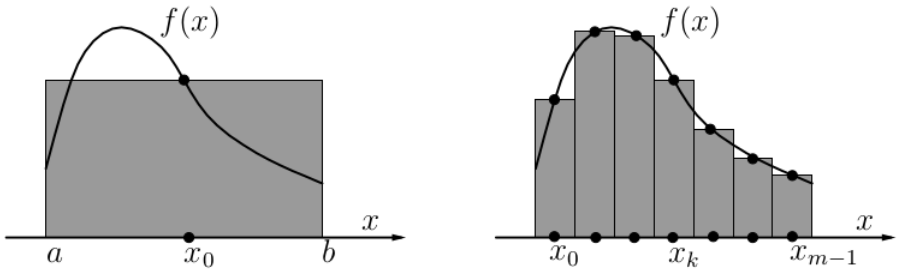


Fig. 9.1. The midpoint formula (*left*); the composite midpoint formula (*right*)

Suppose now that we approximate the integral $I(f)$ by replacing f over $[a,b]$ with its composite interpolating polynomial of degree zero, constructed on m subintervals of width $H = (b-a)/m$, for $m \geq 1$ (see Figure 9.1, right). Introducing the quadrature nodes $x_k = a + (2k+1)H/2$, for $k = 0, \ldots, m-1$, we get the composite midpoint formula

$$I_{0,m}(f) = \sum_{k=0}^{m-1} \int_{x_j}^{x_{j+1}} \Pi_H^0(x)dx = H\sum_{k=0}^{m-1} f(x_k), \quad m \geq 1. \tag{7}$$

The quadrature error $E_{0,m}(f) = I(f) - I_{0,m}(f)$ is given by

$$E_{0,m}(f) = \frac{b-a}{24}H^2 f''(\xi), \quad H = \frac{b-a}{m} \tag{8}$$

provided that $f \in C^2([a,b])$ and where $\xi \in (a,b)$. From (8) we conclude that (7) has degree of exactness equal to 1; (8) can be proved by recalling (6) and using the additivity of integrals. Indeed, for $k = 0, \ldots, m-1$ and $\xi_k \in (a+kH, a+(k+1)H)$,

$$E_{0,m}(f) = \sum_{k=0}^{m-1} f''(\xi_k)(H/2)^3/3 = \sum_{k=0}^{m-1} f''(\xi_k)\frac{H^2}{24}\frac{b-a}{m} = \frac{b-a}{24}H^2 f''(\xi).$$

The last equality is a consequence of the following theorem, that is applied letting $u = f''$ and $\delta_j = 1$ for $j = 0, \ldots, m-1$.

Theorem 1 (Discrete mean-value theorem). *Let $u \in C^0([a,b])$ and let x_j be $s+1$ points in $[a,b]$ and δ_j be $s+1$ constants, all having the same sign. Then there exists $\eta \in [a,b]$ such that*

$$\sum_{j=0}^{s} \delta_j u(x_j) = u(\eta)\sum_{j=0}^{s} \delta_j. \tag{9}$$

Proof. Let $u_m = \min_{x\in[a,b]} u(x) = u(\bar{x})$ and $u_M = \max_{x\in[a,b]} u(x) = u(\bar{\bar{x}})$, where $\bar{x}$ and $\bar{\bar{x}}$ are two points in (a, b). Then

$$u_m \sum_{j=0}^{s} \delta_j \le \sum_{j=0}^{s} \delta_j u(x_j) \le u_M \sum_{j=0}^{s} \delta_j. \tag{10}$$

Let $\sigma_s = \sum_{j=0}^{s} \delta_j u(x_j)$ and consider the continuous function $U(x) = u(x) \sum_{j=0}^{s} \delta_j$. Thanks to (10), $U(\bar{x}) \le \sigma_s \le U(\bar{\bar{x}})$. Applying the mean-value theorem, there exists a point η between a and b such that $U(\eta) = \sigma_s$, which is (9). A similar proof can be carried out if the coefficients δ_j are negative.

□

The Trapezoidal Formula

This formula is obtained by replacing f with $\Pi_1 f$, its Lagrange interpolating polynomial of degree 1, relative to the nodes $x_0 = a$ and $x_1 = b$ (see Figure 9.2, left). The resulting quadrature, having nodes $x_0 = a$, $x_1 = b$ and weights $\alpha_0 = \alpha_1 = (b - a)/2$, is

$$I_1(f) = \int_a^b \Pi_1(x)dx = \frac{b-a}{2}[f(a) + f(b)]. \tag{11}$$

In fact, as seen in the Interpolation chapter, we have that

$$\Pi_1(x) = y_0 \frac{x - x_1}{x_0 - x_1} + y_1 \frac{x - x_0}{x_1 - x_0} = f(a)\frac{x-b}{a-b} + f(b)\frac{x-a}{b-a}$$

so

$$\begin{aligned}
I_1(f) &= \int_a^b \Pi_1(x)dx = \int_a^b \left(f(a)\frac{x-b}{a-b} + f(b)\frac{x-a}{b-a} \right) dx \\
&= \frac{f(a)}{a-b}\int_a^b (x-b)dx + \frac{f(b)}{b-a}\int_a^b (x-a)dx \\
&= \frac{f(a)}{a-b}\left[\frac{x^2}{2} - bx\right]_a^b + \frac{f(b)}{b-a}\left[\frac{x^2}{2} - ax\right]_a^b \\
&= \frac{f(a)}{a-b}\left[\frac{b^2}{2} - b^2 - \frac{a^2}{2} + ab\right] + \frac{f(b)}{b-a}\left[\frac{b^2}{2} - ab - \frac{a^2}{2} + a^2\right] \\
&= \frac{f(a)}{2(b-a)}\left[b^2 + a^2 - 2ab\right] + \frac{f(b)}{2(b-a)}\left[b^2 - 2ab + a^2\right] \\
&= \frac{f(a)}{2(b-a)}(b-a)^2 + \frac{f(b)}{2(b-a)}(b-a)^2 = \frac{b-a}{2}[f(a) + f(b)]
\end{aligned}$$

If $f \in C^2([a, b])$, the quadrature error is given by

$$E_1(f) = I(f) - I_1(f) = -\frac{h^3}{12} f''(\xi), \quad h = b - a, \tag{12}$$

where ξ is a point within the integration interval.

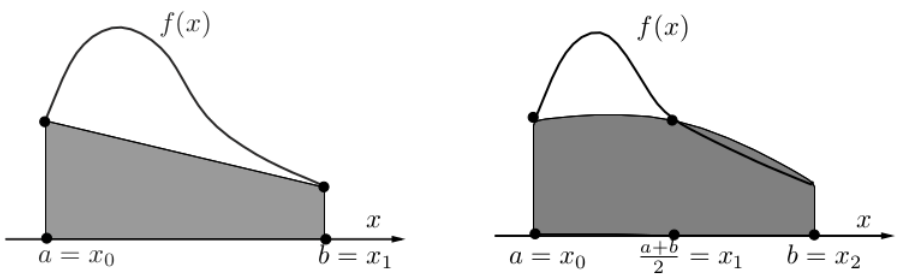


Fig. 9.2. Trapezoidal formula (*left*) and Cavalieri-Simpson formula (*right*)

Indeed, from the expression of the interpolation error one gets

$$E_1(f) = \int_a^b (f(x) - \Pi_1 f(x))dx = -\frac{1}{2}\int_a^b f''(\xi(x))(x-a)(b-x)dx.$$

Since $\omega_2(x) = (x-a)(x-b) < 0$ in (a,b), the mean-value theorem yields

$$E_1(f) = (1/2)f''(\xi)\int_a^b \omega_2(x)dx = -f''(\xi)(b-a)^3/12,$$

for some $\xi \in (a,b)$, which is (12). The trapezoidal quadrature therefore has degree of exactness equal to 1, as is the case with the midpoint rule.

To obtain the composite trapezoidal formula, we proceed as in the case where $n = 0$, by replacing f over $[a,b]$ with its composite Lagrange polynomial of degree 1 on m subintervals, with $m \geq 1$. Introduce the quadrature nodes $x_k = a + kH$, for $k = 0, \ldots, m$ and $H = (b-a)/m$, getting

$$I_{1,m}(f) = \sum_{k=0}^{m-1}\int_{x_j}^{x_{j+1}} \Pi_H^1(x)dx = \frac{H}{2}\sum_{k=0}^{m-1}(f(x_k) + f(x_{k+1})), \quad m \geq 1. \tag{13}$$

Each term in (13) is counted twice, except the first and the last one, so that the formula can be written as

$$I_{1,m}(f) = H\left[\frac{1}{2}f(x_0) + f(x_1) + \ldots + f(x_{m-1}) + \frac{1}{2}f(x_m)\right]. \tag{14}$$

As was done for (8), it can be shown that the quadrature error associated with (14) is

$$E_{1,m}(f) = -\frac{b-a}{12}H^2 f''(\xi),$$

provided that $f \in C^2([a,b])$, where $\xi \in (a,b)$. The degree of exactness is again equal to 1.

The Cavalieri-Simpson Formula

The Cavalieri-Simpson formula can be obtained by replacing f over $[a,b]$ with its interpolating polynomial of degree 2 at the nodes $x_0 = a$, $x_1 = (a+b)/2$ and $x_2 = b$ (see Figure 9.2, right). The weights are given by $\alpha_0 = \alpha_2 = (b-a)/6$ and $\alpha_1 = 4(b-a)/6$, and the resulting formula reads

$$I_2(f) = \int_a^b \Pi_2(x)dx = \frac{b-a}{6}\left[f(a) + 4f\left(\frac{a+b}{2}\right) + f(b).\right] \tag{15}$$

It can be shown that the quadrature error is

$$E_2(f) = I(f) - I_2(f) = -\frac{h^5}{90}f^{(4)}(\xi), \quad h = \frac{b-a}{2}, \tag{16}$$

provided that $f \in C^4([a,b])$, and where ξ lies within (a,b). From (16) it turns out that (15) has degree of exactness equal to 3.

Replacing f with its composite polynomial of degree 2 over $[a,b]$ yields the composite formula corresponding to (15). Introducing the quadrature nodes $x_k = a + kH/2$, for $k = 0, \ldots, 2m$ and letting $H = (b-a)/m$, with $m \geq 1$ gives

$$I_{2,m} = \sum_{k=0}^{m-1}\int_{x_j}^{x_{j+1}} \Pi_H^2(x)dx = \frac{H}{6}\left[f(x_0) + 2\sum_{r=1}^{m-1} f(x_{2r}) + 4\sum_{s=0}^{m-1} f(x_{2s+1}) + f(x_{2m})\right]. \tag{17}$$

The quadrature error associated with (17) is

$$E_{2,m}(f) = -\frac{b-a}{180}(H/2)^4 f^{(4)}(\xi), \tag{18}$$

provided that $f \in C^4([a,b])$ and where $\xi \in (a,b)$; the degree of exactness of the formula is 3.

Integration in Hilbert spaces

Take $p \in \mathbb{P}^n$, make sure you integrate p exactly, then use $L^n f$. We define

$$I(f) := \int_I f \quad \text{and} \quad I_n(f) := \int_I L^n f.$$

Then we have

$$|I_n(f) - I(f)| \leq \int_I |f - L^n f| \leq (b-a)E_n(f)$$

We can use $L^n f \in \mathbb{P}^n$ and given $X$of quadrature points / nodes $\{q_i\}$, using $f_n(x) = L^n f(x)$ we define the **interpolatory quadrature formula** as

$$I_n(f) = \int_a^b f_n(x)dx = \int_I L^n f = \int_I l_i^n(x) f(q_i)dx = f(q_i)w_i$$

where $w_i = \int_I l_i^n(x)dx$.

More generally, the **weights** w_i and the quadrature points q_i define a generic quadrature formula I_n. If I_n is derived from L^n, then it is interpolatory.

Definition. The **degree of accuracy** q of a quadrature formula is the integer $n \in \mathbb{N}$ s.t. the quadrature using $\mathbb{P}^n$ doesn't produce errors on $I(p)$, for $p \in \mathbb{P}^n$:

$$\max_{n \in \mathbb{N}} \text{ s.t. } I_n(p) - I(p) = 0 \iff I_n(p) = \int_I p$$

Theorem. *If I_n is interpolatory, then it has degree at least n on $n+1$ points.*

The composite formulae has the same order of accuracy and have one order less of infinitesimal in the error. We define q the order of accuracy or precision (exactness for polynomials of order q) and r the order of convergence (for composite formulas). Composite formulas with low precision $\rightarrow$ more robust.

We can prove that the maximum degree of accuracy cannot be $2n+2$, because if we pick $f(x) = \omega_{n+1}^2(x) = \prod_{i=0}^n (x - x_i^2)$ we have that

$$I_n(f) = \sum_{i=0}^n f(x_i)\alpha_i = 0 \quad \text{since} \quad f(x_i) = 0 \ \forall i = 0, \dots, n$$

while of course $I(f) > 0$ because the function $f(x)$ is always positive (it is squared), so we have $I_n(f) \neq I(f)$.

Legendre polynomials and max accuracy

Let $m \in \mathbb{N}$, $m > 0$, be the number of quadrature points. Can m raise the degree of accuracy, keeping n constant? Up to what value?

Theorem. *Let $f \in \mathbb{P}^{n+m}$ with $m \leq n+1$. Then $I_n(f) = I(f)$, which means that the quadrature formula $\sum_{i=0}^n \bar{\alpha}_i f(\bar{q}_i)$ has degree of accuracy $n+m$, if and only if it makes use of interpolation and the nodal polynomial $\omega_{n+1}(x) = \prod_{i=0}^n (x - \bar{q}_i)$ associated to nodes $\{\bar{q}_i\}$ is s.t.*

$$\int_a^b \omega_{n+1}(x)p(x)dx = 0 \quad \forall p \in \mathbb{P}^{m-1}$$

Proof. ($\Leftarrow$) Assume that $\int_a^b \omega_{n+1}(x)p(x)dx = 0, \forall p \in \mathbb{P}^{m-1}$. Given $f \in \mathbb{P}^{m+n}$, by applying the quotient theorem for $\mathbb{P}$ it can be written as

$$f(x) = \omega_{n+1}(x)p(x) + q(x)$$

with $\deg(q) < \deg(\omega)$, where $\omega_{n+1}(x) \in \mathbb{P}^{n+1}$, $p(x) \in \mathbb{P}^{m-1}$ and $q(x) \in \mathbb{P}^n$. So by integrating over $[a, b]$ we have that

$$\int_a^b f(x)dx = \int_a^b \omega_{n+1}(x)p(x)dx + \int_a^b q(x)dx$$

and thanks to the hypothesis the central term is zero, so we have that $\int_a^b f(x)dx = \int_a^b q(x)dx$, which means that $I(f) = \int_a^b f(x)dx = \int_a^b q(x)dx = I(q)$. Given that $q \in \mathbb{P}^n$ its quadrature is exact since we take $n+1$ nodes, so $I_n(q) = I(q) = I(f)$, so we have that we can think of q as the approximation f_n, so $I_n(f) = I(f_n) = I(q_n) = I_n(q) = I(f)$. (or $I(f) = \int q = I_n(q)$ because we can interpolate exactly polynomials of degree $n+1$)

($\Rightarrow$) Assume that the quadrature formula has degree of accuracy $n+m$, so $I_n(f) = I(f) = \int f,\ \forall f \in \mathbb{P}^{n+m}$. If we take $\omega_{n+1}(x)p(x) \in \mathbb{P}^{n+m}$ with $\omega_{n+1}(x) = \prod_{i=0}^n (x - \bar{q}_i)$, we have that $\int_a^b \omega_{n+1}(x)p(x) = I_n(\omega_{n+1}p) = 0$ since on the nodes $\bar{q}_i$ we have that $\omega_{n+1}(\bar{q}_i) = \prod_{i=0}^n (\bar{q}_i - \bar{q}_i) = 0,\ \forall i$.

Theorem. *Le q be nonzero polynomial of degree $n+1$ and $\omega(x)$ a positive weight function, s. t.:*

$$\int_a^b x^k q(x)\,\omega(x) = 0, \quad k = 0, \ldots, n \tag{19}$$

If x_i are zeros of $q(x)$, then

$$\int_a^b f(x)\,\omega(x) \approx \sum_{i=0}^n w_i\, f(x_i) \tag{20}$$

with

$$w_i = \int_a^b l_i(x)\,\omega(x) \tag{21}$$

is exact for all polynomials of degree at most $2n+1$. Here $l_i(x)$ are the usual Lagrange interpolation polynomials.

Proof. Assume $f(x)$ is a polynomial of degree at most $2n+1$ and show:

$$\int_a^b f(x)\,\omega(x) = \sum_{i=0}^n w_i\, f(x_i).$$

Using the polynomial division we have:

$$\underbrace{f(x)}_{2n+1} = \underbrace{q(x)}_{n+1}\underbrace{p(x)}_{n} + \underbrace{r(x)}_{n}.$$

By taking x_i as zeros of $q(x)$ we have that $f(x_i) = r(x_i)$. Now:

$$\begin{aligned}\int_a^b f(x)\,\omega(x) &= \int_a^b [q(x)\,p(x) + r(x)]\,\omega(x) \\ &= \underbrace{\int_a^b q(x)\,p(x)\,\omega(x)}_{=0} + \int_a^b r(x)\,\omega(x)\end{aligned}$$

Since $r(x)$ is a polynomial of order n this is exact:

$$\int_a^b f(x)\,\omega(x) = \int_a^b r(x)\,\omega(x) = \sum_{i=0}^{n} w_i\, r(x_i)$$

But since we chose x_i such that $f(x_i) = r(x_i)$, we have:

$$\int_a^b f(x)\,\omega(x) = \int_a^b r(x)\,\omega(x) = \sum_{i=0}^{n} w_i\, f(x_i)$$

This completes the proof.

□

Legendre Polynomials

We use the orthogonal polynomials we obtained with Gram-Schmidt in finding the best approximation.

Two term recursion, to obtain the same orthogonal polynomials above (defined between $[-1, 1]$), normalized to be one in $x = 1$:

$$(n+1)p^{n+1}(x) = (2n+1)\,x\,p^n(x) - n\,p^{n-1}(x) \tag{22}$$

In our proof we selected to evaluate x_i at the zeros of the Legendre polynomials, this is why we need to evaluate the zeros of the polynomials.

To prove that m is bounded at $n+1$, we could replace $p \in \mathbb{P}^{m-1}$ with $\omega_{n+1}(x)$, obtaining

$$\int_a^b \omega_{n+1}(x)\omega_{n+1}(x)dx = 0 \quad \text{for } m \geq n+2$$
$$\Rightarrow \omega_{n+1}(x) = 0$$

which is *false* because based on false assumption.

□

The maximum value for m is $n+1$, achieved when ω_{n+1} is proportional to $L_{n+1}(x)$, the Legendre polynomial of degree $n+1$. Legendre polynomials can be computed recursively as

$$\begin{cases} L_0(x) = 1 \\ L_1(x) = x \\ L_{k+1}(x) = \frac{2k+1}{k+1} x L_k(x) - \frac{k}{k+1} L_{k-1}(x) \end{cases}$$

Since L_{n+1} is orthogonal to every $L_0, L_1, \ldots, L_n$ (because we have that $\int_a^b L_{n+1}(x)L_j(x)dx = 0\ \forall j < n+1$) we can see why m is bounded at $n+1$. Thus, the highest degree of accuracy is $2n+1$, obtained using the **Gauss-Legendre formula**

$$I_{\text{GL}} = \begin{cases} \bar{q}_i = \text{roots of } L_{n+1}(x) \\ \bar{\alpha}_i = \frac{2}{(1-q_i^2)(L'_{n+1}(q_i))^2} \end{cases} \qquad i = 0, \ldots, n$$

The related **Gauss-Legendre-Lobatto formula** includes interval bounds among quadrature points, and has a degree of accuracy of $2n-1$.

The interval used for I_{GL} is $[-1, 1]$, thus the bar notation of $\bar{q}_i, \bar{\alpha}_i$. To reconvert to original values for (a, b), use the **Chebyshev formula**:

$$q_i = \frac{a+b}{2} + \frac{b-a}{2}\bar{q}_i, \quad \alpha_i = \frac{b-a}{2}\bar{\alpha}_i$$

Quadrature rules: methods of undetermined coefficients

Given a quadrature rule $I(f) := \sum_{i=0}^{n} f(x_i)w_i$, we want to determine w_i *and* x_i such that the degree of accuracy is as high as possible / desired.

We impose it to be exact for polynomials of order q. We have $2(n+1)$ unknowns ($n+1$ points x_i and $n+1$ weights w_i). We can impose it on all monomials of order $q = 2n+1$:

$$\sum_{i=0}^{n}(x_i)^j w_i = \int x^j \qquad \text{for } j = 0, \dots, 2n+1,$$

which are $2n+2$ conditions on $2n+2$ unknowns. So this is a non-linear system of equations: if you can solve it, you win!

Peano integration kernel theorem

We've seen how to estimate the errors for the quadrature formulae: we use the Taylor's expansion and the mean-value theorem (see the calculation for the mid-point rule). Can we generalize this? Yes.

We define $x_+^n = \begin{cases} x^n & \text{for } x > 0 \\ 0 & \text{for } x < 0 \end{cases}$.

The **Peano kernel** represents the error we make when integrating a function $g(x) = (x-t)_+^k$ for a given $t \in [a, b]$. An explicit expression of it is:

$$k(t) = E_x[(x-t)_+^k] = \int_a^b (x-t)_+^k dx - I_n((x-t)_+^k)$$

We have that $\int_a^b (x-t)_+^k dx = \frac{(x-t)^{k+1}}{k+1}\Big|_{x=b} - \frac{(x-t)^{k+1}}{k+1}\Big|_{x=a} = \frac{(b-t)^{k+1}}{k+1}$, since $a \leq t$, so $a - t \leq 0$ which means that $\frac{(a-t)^{k+1}}{k+1} = 0$, and so $k(t)$ it doesn't depend on a.

Theorem (Taylor's theorem)**.** *Given a quadrature formula of degree d, and an integer $0 \leq k \leq d$, let $f \in C^{k+1}([a, b])$. Then we have that*

$$f(x) = p(x) + r(x) = p(x) + \frac{1}{k!}\int_a^x f^{(k+1)}(t)(x-t)^k dt,$$

with $p(x) \in \mathbb{P}^k$ the Taylor Expansion to order k of f around a, i.e.

$$p(x) = \sum_{i=0}^{k} \frac{f^{(i)}(a)}{i!}(x-a)^i,$$

and using the integral form of the remainder.

See the Wikipedia page for the derivation for the integral form of the remainder.

The reminder can be written as:

$$r(x) = \frac{1}{k!}\int_a^b f^{(k+1)}(t)(x-t)_+^k dt,$$

(note that now the integral is from a to b, and *not* from a to x, and we integrate the function $f^{(k+1)}(t)(x-t)_+^k$) so we have the following theorem.

Theorem (Peano kernel theorem)**.** *Given a quadrature formula of degree d, and an integer $0 \leq k \leq d$, let $f \in C^{k+1}([a, b])$. Then*

$$E(f) = \frac{1}{k!}\int_a^b f^{(k+1)}(t)k(t)dt.$$

Proof. From the previous theorem we have that

$$\begin{aligned} f(x) = p(x) + r(x) &= \sum_{i=0}^{k} \frac{f^{(i)}(a)}{i!}(x-a)^i + \frac{1}{k!}\int_a^x f^{(k+1)}(t)(x-t)^k dt \\ &= \sum_{i=0}^{k} \frac{f^{(i)}(a)}{i!}(x-a)^i + \frac{1}{k!}\int_a^b f^{(k+1)}(t)(x-t)_+^k dt \end{aligned}$$

so it follows that $I(f) = I(p+r) = I(p) + I(r)$, so (remember that $I(f) = \int_a^b f(x)dx$)

$$\begin{aligned} E(f) &= \int_a^b f(x)dx - I_n(f) = \int_a^b p(x)dx - I_n(p) + \int_a^b r(x)dx - I_n(r) = \int_a^b r(x)dx - I_n(r) \\ &= \int_a^b \left(\frac{1}{k!}\int_a^b f^{(k+1)}(t)(x-t)_+^k dt\right)dx - I_n\left(\frac{1}{k!}\int_a^b f^{(k+1)}(t)(x-t)_+^k dt\right) \\ &= \int_a^b \left(\frac{1}{k!}\int_a^b f^{(k+1)}(t)(x-t)_+^k dt\right)dx - \frac{1}{k!}\int_a^b I_n\left(f^{(k+1)}(t)(x-t)_+^k\right)dt \\ &= \frac{1}{k!}\int_a^b \left(\int_a^b f^{(k+1)}(t)(x-t)_+^k dx - I_n\left(f^{(k+1)}(t)(x-t)_+^k\right)\right)dt \\ &= \frac{1}{k!}\int_a^b \left(\int_a^b f^{(k+1)}(t)(x-t)_+^k dx - f^{(k+1)}(t)I_n\left((x-t)_+^k\right)\right)dt \\ &= \frac{1}{k!}\int_a^b f^{(k+1)}(t)\left(\int_a^b (x-t)_+^k dx - I_n\left((x-t)_+^k\right)\right)dt \\ &= \frac{1}{k!}\int_a^b f^{(k+1)}(t)E_x[(x-t)_+^k]dt \\ &= \frac{1}{k!}\int_a^b f^{(k+1)}(t)k(t)dt \end{aligned}$$

where $\int_a^b p(x)dx - I_n(p) = 0$ since $p \in \mathbb{P}^k$ and we have a quadrature formula of degree $d \geq k$, and $I_n\left(f^{(k+1)}(t)(x-t)_+^k\right) = f^{(k+1)}(t)I_n\left((x-t)_+^k\right)$ since $f^{(k+1)}(t)$ doesn't depend on the variable x on which the integral I_n is computed.

□

From this, we get the error bound:

$$|E(f)| \leq \frac{1}{k!}||k||_2||f^{(k+1)}||_2.$$

Other norm combinations are $1-\infty$ or $\infty - 1$.

More on numerical integration:

- **Simpson adaptive formula** uses different steplengths to compute the composite interpolant on the integral, reducing the nodes needed.
- **Monte Carlo methods** approximate the integral of f as a function statistical mean. They usually lead to poor results.

Least Squares

Approximation by the least square method

Suppose we have $n+1$ points $x_0, x_1, \dots, x_n$ and $n+1$ values $y_0, y_1, \dots, y_n$. We have seen that if n is large, the interpolating polynomial may show large oscillations.

One solution could be to break the interpolation domain in pieces and then perform a multiple interpolation. Or, instead of interpolating the values, it is possible to define a polynomial of degree $m < n$ that approximates the data "at best".

Definition 1. *We call* ***least squares polynomial approximation of degree*** m *the polynomial* $\tilde{f}_m(x)$ *of degree* m *such that*

$$\boxed{\sum_{i=0}^{n} |y_i - \tilde{f}_m(x_i)|^2 \leq \sum_{i=0}^{n} |y_i - p_m(x_i)|^2} \quad \forall p_m(x) \in \mathbb{P}^m$$

Remark 1. *If* $y_i = f(x_i)$ *with* f *a continuous function, then* $\tilde{f}_m$ *is called the* ***approximation of*** f ***in the least squares sense****, or* ***least squares approximation of*** f*.*

In other words, the least squares polynomial approximation is the polynomial of degree m that minimizes the distance from the data points.
Let note the polynomial $\tilde{f}_m(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^m$ with the $m+1$ coefficients a_i unknown, and define the **loss function** as

$$\Phi(a_0, a_1, \dots, a_m) = \sum_{i=0}^{n} |y_i - \tilde{f}_m(x_i)|^2 = \sum_{i=0}^{n} |y_i - (a_0 + a_1 x_i + a_2 x_i^2 + \ldots + a_m x_i^m)|^2$$

Since we want to minimize the loss function Φ, we put the derivative w.r.t. the coefficients to zero. Then the coefficients of $\tilde{f}_m$ can be determined by the relation

$$\frac{\partial \Phi}{\partial a_k} = 0, \quad k = 0, \dots, m, \tag{1}$$

i.e., $m+1$ linear equations with $m+1$ unknowns a_k, $k = 0, \dots, m$, which means that the problem admits an unique solution, so it is well posed.

Ideally, for $\tilde{f}_m(x) = a_0 + a_1 x + a_2 x^2 + \ldots + a_m x^m$ we would like to impose $\tilde{f}_m(x_i) = y_i$ for $i = 0, \dots, n$. This can be written as a linear system with basis $1, x, x^2, \dots, x^m$ and unknowns $a_k, k = 0, \dots, m$: $B\mathbf{a} = y$, where B is a matrix of dimension $(n+1) \times (m+1)$, called **Vandermonde matrix**:

$$B = \begin{pmatrix} 1 & x_0 & \dots & x_0^m \\ 1 & x_1 & \dots & x_1^m \\ \vdots & & & \vdots \\ 1 & x_n & \dots & x_n^m \end{pmatrix}$$

Since $m < n$, the system is oversized. The solution to (1) is equivalent to the square system (**system of normal equations**)

$$\boxed{B^T B\mathbf{a} = B^T y}$$

While $\tilde{f}_m$ is a polynomial, we can generalize the formula for functions of a space V_m obtained by linearly combining $m+1$ independent functions $\{\psi_j, j=0,1,\dots,m\}$. The choice of ψ is dictated by the conjectured behaviour of the function underlying the current data distribution. So we have that $\tilde{f}(x) = \sum_{j=0}^{m} a_j \psi_j(x)$ and the unknown coefficients $a = (a_0, a_1, \dots, a_m)$ can be obtaining solving the system $B^T B a = B^T y$ where in this case $B = b_{ij} = \psi_j(x_i)$ and y are the data.

Generalization

We would like to approximate a function evaluated on a (large) number of data points, using a finite dimensional space V_h of dimension n, defined as the *span* of a set of basis functions v_i: any function in V_h can be expressed as a linear combination of the basis v_i:

$$v_h(x) = v^i v_i(x)$$

where summation is implied on i (Einstein notation).

Assume we'd like to approximate the function $f : \Omega \mapsto \mathbb{R}$ and that the only thing we have at our disposal is N pairs (x_i, y_i), i.e., N points $x_i \in \Omega$ in which we know the values $f(x_i) = y_i$.

Given *any* finite dimensional space V_h of dimension n (i.e., any collection of n *linearly independent* functions $v_i : \Omega \mapsto \mathbb{R}$), we define the **basis collocation matrix** B as the rectangular matrix

$$B_{ij} = v_j(x_i), \quad i = 1, \dots, N, \; j = 1, \dots, n.$$

An element of V_h evaluated in all points x_i can be computed easily by the matrix vector product between B and the vector of coefficients v:

$$v_h(x_i) = (Bv)_i = B_{ij} v^j = v^j v_j(x_i)$$

Computing the **least square approximation** of f in V_h is equivalent to finding the element of V_h that minimizes the following functional:

$$E(v_h) := \frac{1}{2N} \sum_{i=1}^{N} |v_h(x_i) - y_i|^2 \tag{2}$$

where $E(v_h)$ is the **mean squared error (MSE)** or **mean squared deviation (MSD)** of the approximation v_h, i.e., **the average of the squares of the errors**—that is, the **average squared difference between the approximated values and the actual value**.

Expressing $v_h(x_i)$ with the matrix product, $E(v_h)$ can be written as

$$E(v_h) := \frac{1}{2N} (Bv - y)^T (Bv - y) \tag{3}$$

If we want to minimize E, we can take its derivative w.r.t. the coefficients v^i and set it to zero, i.e.:

$$\frac{\partial E}{\partial v^i} = \frac{1}{2N} \frac{\partial [(Bv - y)^T (Bv - y)]}{\partial v^i} = \frac{1}{N} (B^T B v - B^T y) = 0$$

which admits a unique solution if the following linear system has a solution:

$$B^T B v = B^T y. \tag{5}$$

Linear systems: Direct Methods

We call **linear system of order** n (n positive integer), an expression of the form

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

where $A = (a_{ij})$ is a *given matrix* of size $n \times n$, $\mathbf{b} = (b_j)$ is a *given vector* and $\mathbf{x} = (x_j)$ is the *unknown vector* of the system. The previous relation is equivalent to the n equations

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = 1, \ldots, n.$$

The matrix A is called non-singular if $\det(A) \neq 0$; the solution $\mathbf{x}$ will be unique (for any given vector $\mathbf{b}$) if and only if the matrix associated to the linear system is non-singular.

In theory, if A is non-singular, the solution is given by the Cramer's rule:

$$x_i = \frac{\det(B_i)}{\det(A)}, \quad i = 1, \ldots, n,$$

where B_i is the matrix by substituting the i-th column of A by the vector $\mathbf{b}$:

$$B_i = \begin{bmatrix} a_{11} & \ldots & b_1 & \ldots & a_{1n} \\ a_{21} & \ldots & b_2 & \ldots & a_{2n} \\ \vdots & & \vdots & & \vdots \\ 0a_{n1} & \ldots & b_n & \ldots & a_{nn} \end{bmatrix} \begin{matrix} \\ \\ \\ \\ \uparrow \\ i \end{matrix}$$

Unfortunately, the application of this rule is unacceptable for the practical solution of systems because the computational cost is of the order of $(n+1)!$ floating point operations per second (flops). In fact, every determinant requires $n!$ flops.

Triangular systems

A matrix $U = (u_{ij})$ is **upper triangular** if

$$u_{ij} = 0 \ \forall i, j : 1 \leq j < i \leq n$$

and a matrix $L = (l_{ij})$ is **lower triangular** if

$$l_{ij} = 0 \ \forall i, j : 1 \leq i < j \leq n.$$

A diagonal matrix is a special triangular matrix. Respectively, the system to be solved is called **upper or lower triangular system**.

Remark: If a matrix A in non-singular and triangular, knowing that

$$\det(A) = \prod_{i=1}^{n} \lambda_i(A) = \prod_{i=1}^{n} a_{ii}$$

($\lambda_i(A)$ being the i-th eigenvalue of A), we can deduce that $a_{ii} \neq 0$, for all $i = 1, \ldots, n$.

If L is lower triangular and non-singular, the linear system $L\mathbf{y} = \mathbf{b}$ corresponds to

$$\begin{cases} l_{11}y_1 & = b_1 \\ l_{21}y_1 + l_{22}y_2 & = b_2 \\ \vdots & \\ l_{n1}y_1 + l_{n2}y_2 + \ldots + l_{nn}y_n & = b_n \end{cases}$$

Thus:

$$\boxed{y_1 = \frac{b_1}{l_{11}}}, \quad [1 \text{ operation}]$$

and for $i = 2, 3, \ldots, n$

$$\boxed{y_i = \frac{1}{l_{ii}}\left(b_i - \sum_{j=1}^{i-1} l_{ij}y_j\right)}. \quad [1 + 2(i-1) \text{ operations}]$$

This algorithm is called **forward substitutions algorithm**.

The forward substitutions algorithm requires n^2 operations, where n is the size of the system, since

$$1 + \sum_{i=2}^{n}(1 + 2(i-1)) = 1 + \sum_{i=1}^{n}(2i-1) - 1 = n^2.$$

If U is upper triangular and non-singular, the system $U\mathbf{x} = \mathbf{y}$ is:

$$\begin{cases} u_{11}x_1 + \ldots + u_{1,n-1}x_{n-1} + u_{1n}x_n & = y_1 \\ \vdots & \\ u_{n-1,n-1}x_{n-1} + u_{n-1,n}x_n & = y_{n-1} \\ u_{nn}x_n & = y_n \end{cases}$$

Thus:

$$\boxed{x_n = \frac{y_n}{u_{nn}}},$$

and for $i = n-1, n-2, \ldots, 2, 1$

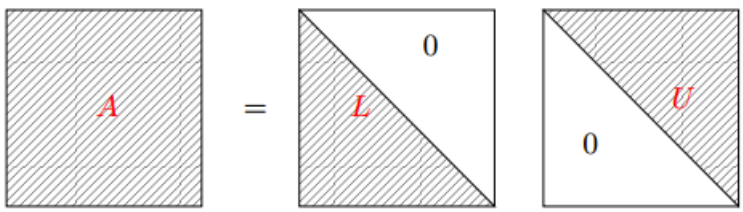
$$\boxed{x_i = \frac{1}{u_{ii}}\left(y_i - \sum_{j=i+1}^{n} u_{ij}x_j\right)}.$$

This algorithm is called **backward substitutions algorithm**. The cost is, once again, n^2 operations.

The LU factorization method

Let $A = (a_{ij})$ be a non-singular $n \times n$ matrix. Assume that there exist a matrix $U = (u_{ij})$, **upper triangular** and a matrix $L = (l_{ij})$, **lower triangular** such that

$$A = LU. \tag{1}$$



We call (1) a **factorization / decomposition LU of A**.

If we know the factorization LU of A, solving the system $A\mathbf{x} = \mathbf{b}$ is equivalent to solving two systems defined by triangular matrices. Indeed,

$$A\mathbf{x} = \mathbf{b} \iff LU\mathbf{x} = \mathbf{b} \iff \begin{cases} L\mathbf{y} = \mathbf{b}\,, \\ U\mathbf{x} = \mathbf{y}\,. \end{cases}$$

We can easily calculate the solutions of both systems:

- first, we use the forward substitutions algorithm to solve $L\mathbf{y} = \mathbf{b}$ (order n^2 flops);
- then, we use the backward substitutions algorithm to solve $U\mathbf{x} = \mathbf{y}$ (order n^2 flops).

It is required to find first (if possible) the matrices L and U, which requires a number of operations of the order $\frac{2n^3}{3}$ flops, that is better than $(n+1)!$.

Example. Lets try to find a factorization LU in the case case where the size of the matrix A is $n = 2$. We can write the equation (1) as

$$\begin{bmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 \\ l_{21} & l_{22} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} \\ 0 & u_{22} \end{bmatrix},$$

Or equivalently:

$$\begin{aligned} &(a) \quad l_{11}u_{11} = a_{11}\,, \qquad (b) \quad l_{11}u_{12} = a_{12}\,, \\ &(c) \quad l_{21}u_{11} = a_{21}\,, \qquad (d) \quad l_{21}u_{12} + l_{22}u_{22} = a_{22}\,. \end{aligned}$$

We have then a system (non-linear) with 4 equations and 6 unknowns; in order to have the same number of equations and unknowns, we fix the diagonal of L by taking $l_{11} = l_{22} = 1$. Consequently, from (a) and (b) we have $u_{11} = a_{11}$ and $u_{12} = a_{12}$; finally, if we assume $a_{11} \neq 0$, we obtain $l_{21} = \frac{a_{21}}{a_{11}}$ and $u_{22} = a_{22} - l_{21}u_{12} = a_{22} - \frac{a_{21}a_{12}}{a_{11}}$ using the equations (c) and (d). ■

To determine a factorization LU of the matrix A of any size n, we apply the following method.

1. The elements of L and U satisfy the non-linear system

$$\sum_{r=1}^{\min(i,j)} l_{ir}u_{rj} = a_{ij}, \quad i,j = 1,\dots,n; \tag{2}$$

2. The system (2) has n^2 equations and $n^2 + n$ unknowns, so it is undetermined. Consequently, the LU factorization is not unique. We can wipe out n unknowns if we set the n diagonal elements of L equal to 1:

$$l_{ii} = 1, \quad i = 1,\dots,n.$$

   We will see that in this case there exists an algorithm (*Gauss factorization*) allowing us to efficiently compute the factors L and U.

Any $n \times n$ matrix A admits a LU factorization with partial pivoting (LUP)[1] [we will see it later]. It turns out that all square matrices can be factorized in this form, and the factorization is numerically stable in practice. Instead, if A is invertible, then it admits an LU factorization if and only if all its leading principal minors[2] are nonzero (like for the GEM).

The Gauss elimination method

The **Gauss elimination method** (GEM) transforms the system

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

with $A \in \mathbb{R}^{n\times n}$, in an equivalent system (i.e. with the same solution) of the form:

$$U\mathbf{x} = \hat{\mathbf{b}},$$

where U is an upper triangular matrix and $\hat{\mathbf{b}}$ is a properly modified second member. This system can be solved by a backward substitutions method.

In the transformation, we essentially use the property that says that we do not change the solution of the system if we add to a given equation a linear combination of other equations.

Let us consider an invertible matrix $A \in \mathbb{R}^{n\times n}$ in which the diagonal element a_{11} is assumed to be non-zero. we set $A^{(1)} = A$ and $\mathbf{b}^{(1)} = \mathbf{b}$. We introduce the **multiplier**

$$l_{i1} = \frac{a_{i1}^{(1)}}{a_{11}^{(1)}},\ i = 2, 3, \ldots, n, \qquad A^{(1)} = \begin{bmatrix} a_{11}^{(1)} & \cdots & a_{1j}^{(1)} & \cdots & a_{1n}^{(1)} \\ \vdots & & \vdots & & \vdots \\ a_{i1}^{(1)} & \cdots & a_{ij}^{(1)} & \cdots & a_{in}^{(1)} \\ \vdots & & \vdots & & \vdots \\ a_{n1}^{(1)} & \cdots & a_{nj}^{(1)} & \cdots & a_{nn}^{(1)} \end{bmatrix}$$

where the $a_{ij}^{(1)}$ represent the elements of $A^{(1)}$. This multiplier will yield 0 as coefficient for the unknown x_1 in the lines $i \geq 2$ when combined with the first line of the matrix. In this way the unknown x_1 can be removed from the rows $i = 2, \ldots, n$ by subtracting l_{i1} times the first row and doing the same at the right-hand side.

Let us define

$$\begin{aligned} a_{ij}^{(2)} &= a_{ij}^{(1)} - l_{i1}a_{1j}^{(1)}, \quad & i, j = 2, \ldots, n, \\ b_{i}^{(2)} &= b_{i}^{(1)} - l_{i1}b_{1}^{(1)}, & i = 2, \ldots, n, \end{aligned}$$

where the $b_i^{(1)}$ are the components of $\mathbf{b}^{(1)}$. The coefficients l_{i1} will set to 0 all the elements a_{ij} below the pivot, and $\mathbf{b}^{(2)}$ will be the adjusted $\mathbf{b}$ accordingly to the changes made to the matrix A. We get a new system of the form

$$\begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \ldots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & \ldots & a_{2n}^{(2)} \\ \vdots & \vdots & & \vdots \\ 0 & a_{n2}^{(2)} & \ldots & a_{nn}^{(2)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ b_n^{(2)} \end{bmatrix},$$

which will be written as $A^{(2)}\mathbf{x} = \mathbf{b}^{(2)}$ and that is equivalent to the system we had at the beginning.

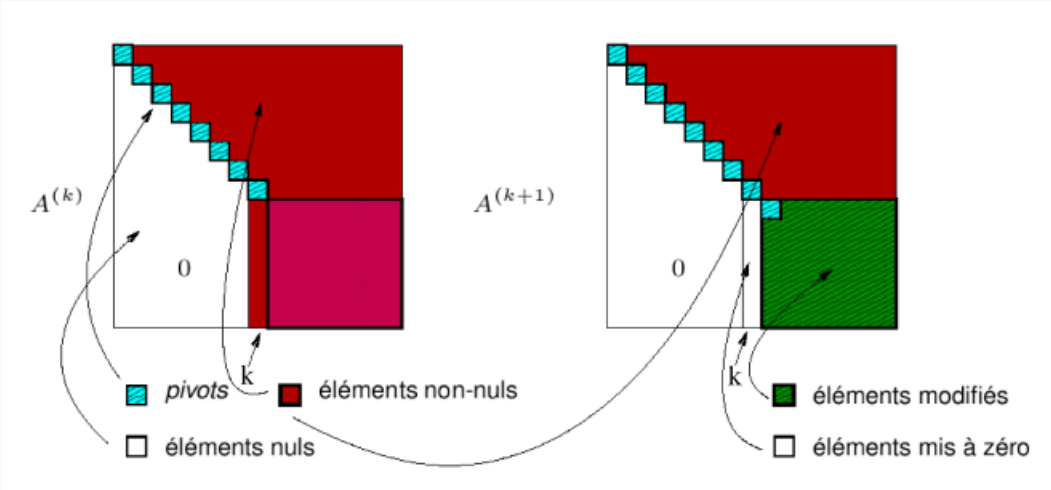
Once again we can transform this system by removing the unknown x_2 from the rows $3, \ldots, n$. By repeating this step we obtain a finite series of systems

$$A^{(k)}\mathbf{x} = \mathbf{b}^{(k)},\ 1 \leq k \leq n,$$

where, for $k \geq 2$, the matrix $A^{(k)}$ is of the form

$$A^{(k)} = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & \cdots & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & & & & a_{2n}^{(2)} \\ \vdots & & \ddots & & & \vdots \\ 0 & \cdots & 0 & a_{kk}^{(k)} & \cdots & a_{kn}^{(k)} \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \cdots & 0 & a_{nk}^{(k)} & \cdots & a_{nn}^{(k)} \end{bmatrix},$$

where we assume $a_{ii}^{(i)} \neq 0$ for $i = 1, \ldots, k-1$.



GEM: diagram showing how the matrix $A^{(k+1)}$ is obtained from the matrix $A^{(k)}$.

It is clear that for $k = n$ we obtain the following upper triangular system $A^{(n)}\mathbf{x} = \mathbf{b}^{(n)}$:

$$\begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \cdots & \cdots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & & & a_{2n}^{(2)} \\ \vdots & & \ddots & & \vdots \\ 0 & & & \ddots & \vdots \\ 0 & & & & a_{nn}^{(n)} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1^{(1)} \\ b_2^{(2)} \\ \vdots \\ \vdots \\ b_n^{(n)} \end{bmatrix}.$$

To be consistent with the previous notation, we write as U the upper triangular matrix $A^{(n)}$. The elements on the main diagonals, the $a_{kk}^{(k)}$, are called **pivots** and have to be non-zero for $k = 1, \ldots, n-1$.

In order to make explicit the formula to get from the k-th system to the $(k+1)$-th, for $k = 1, \ldots, n-1$, we assume that $a_{kk}^{(k)} \neq 0$ and we define the **multiplier**

$$\boxed{l_{ik} = \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}}, \quad i = k+1, \ldots, n, \qquad [(n-k) \text{ operations}] \qquad (3)$$

we set then

$$\begin{aligned} &\boxed{a_{ij}^{(k+1)} = a_{ij}^{(k)} - l_{ik} a_{kj}^{(k)}}, \quad i, j = k+1, \ldots, n, \quad [2(n-k)^2 \text{ operations}] \\ &\boxed{b_i^{(k+1)} = b_i^{(k)} - l_{ik} b_k^{(k)}}, \quad i = k+1, \ldots, n. \qquad [2(n-k) \text{ operations}] \end{aligned} \qquad (38)$$

Remark. *To perform the Gauss elimination we require $2(n-k)^2$ operations to update A, $2(n-k)$ operations to update b and $(n-k)$ operations to update l, so in total*

$$\begin{aligned} 2\sum_{k=1}^{n-1}(n-k)^2 + 3\sum_{k=1}^{n-1}(n-k) &= 2\sum_{p=1}^{n-1}p^2 + 3\sum_{p=1}^{n-1}p \\ &= 2\frac{(n-1)n(2n-1)}{6} + 3\frac{n(n-1)}{2} = \frac{n(n-1)(2n+3)}{3} \end{aligned}$$

operations are required, plus n^2 operations for the resolution with the backward substitutions method of the triangular system $U\mathbf{x} = \mathbf{b}^{(n)}$. By keeping only the dominant elements (of order n^3), we can say that the Gauss elimination method has a cost of around

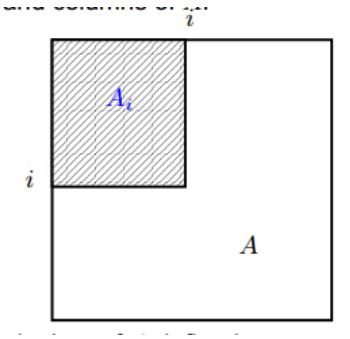
$$\frac{2}{3}n^3 \ operations.$$

The Gauss method is only properly defined if the pivots $a_{kk}^{(k)}$ are non-zero for $k = 1, \ldots, n-1$. Unfortunately, knowing that the diagonal elements of A are not zero is not enough to avoid null pivots during the elimination phase. For example, the following matrix A in (4) is invertible and its diagonal elements are non-zero

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 7 & 8 & 9 \end{bmatrix}, \quad \text{but we find} \quad A^{(2)} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & \boxed{0} & -1 \\ 0 & -6 & -12 \end{bmatrix}. \quad (4)$$

Nevertheless, we have to stop the Gauss method at the second step, because $a_{22}^{(2)} = 0$.

Let A_i be the i-th main submatrix of A ($i = 1, \ldots, n-1$), i.e. the submatrix made of the i first rows and columns of A:



and let d_i be the principal minor of A defined as $d_i = \det(A_i)$. We have the following result.

Proposition 1. *For a given matrix $A \in \mathbb{R}^{n \times n}$, its Gauss factorization exists and is unique iff the principal submatrices A_i ($i = 1, \ldots, n-1$) are non-singular (i.e. the principal minors d_i are non-zero: $d_i \neq 0$).*

Remark: If $d_i \neq 0$ ($i = 1, \ldots, n-1$), then the pivots $a_{ii}^{(i)}$ are also non-zero.
The matrix of the previous example does not satisfy this condition because $d_1 = \det[1] = 1$ but $d_2 = \det \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix} = 0$.

There are some categories of matrices for which the hypothesis of the Proposition 1 are fulfilled. In particular, we mention:

1. **(Strictly $>$) diagonal dominant by row matrices.** A matrix A is said diagonal dominant by row if
$$|a_{ii}| \geq \sum_{j=1,\ldots,n; j \neq i} |a_{ij}|, \quad i = 1, \ldots, n.$$
2. **(Strictly $>$) diagonal dominant by column matrices.** A matrix A is said diagonal dominant by column if
$$|a_{jj}| \geq \sum_{i=1,\ldots,n; i \neq j} |a_{ij}|, \quad j = 1, \ldots, n.$$
3. **Symmetric positive definite matrices.** A matrix A is symmetric if $A = A^T$; it is positive definite if all its eigenvalues are positive, i.e.:
$$\lambda_i(A) > 0, \quad i = 1, \ldots, n.$$

Example 3: The matrix $\begin{bmatrix} -4 & 1 & 2 \\ 2 & 5 & 0 \\ -2 & 1 & 7 \end{bmatrix}$ is diagonal dominant by row and by column, whereas $\begin{bmatrix} -3 & 1 & 2 \\ 2 & 5 & 0 \\ -2 & 1 & 7 \end{bmatrix}$ is only diagonal dominant by row (in the first column we have $|-3| < |2| + |-2|$).

Gauss ~ LU

We can show that the Gauss method is equivalent to the factorization $A = LU$ of the matrix A, with L = multiplier matrix and $U = A^{(n)}$.

More exactly:

$$A = \underbrace{\begin{bmatrix} 1 & 0 & \dots & \dots & 0 \\ l_{21} & 1 & & & 0 \\ \vdots & l_{32} & \ddots & & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ l_{n1} & & & l_{n,n-1} & 1 \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \dots & \dots & a_{1n}^{(1)} \\ 0 & a_{22}^{(2)} & & & a_{2n}^{(2)} \\ \vdots & & \ddots & & \vdots \\ 0 & & & \ddots & \vdots \\ 0 & & & & a_{nn}^{(n)} \end{bmatrix}}_{U}.$$

The matrices L and U only depend on A (and not on $\mathbf{b}$), so the same factorization can be reused for solving several linear systems that share the same matrix A but **different vectors** $\mathbf{b}$.

The number of operations is then considerably reduced, since most of the computational weight, around $\frac{2}{3}n^3$ flops, is due to the Gaussian elimination process. Indeed, let us consider the M linear systems:

$$A\mathbf{x}_m = \mathbf{b}_m \quad m = 1, \dots, M.$$

Therefore:

- the cost of the factorization $A = LU$ is $\frac{2}{3}n^3$ flops;
- the cost of the resolution of both triangular systems, $L\mathbf{y}_m = \mathbf{b}_m$ and $U\mathbf{x}_m = \mathbf{y}_m$ ($m = 1, \dots, M$) is $2Mn^2$ flops,

for a total of $\frac{2}{3}n^3 + 2Mn^2$ flops which is much smaller than $\frac{2}{3}Mn^3$ flops required to solve all the systems with the Gauss elimination method.

The pivoting technique

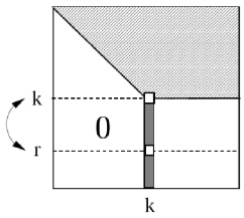
It has been already noted that the Gauss method fails if a pivot becomes zero. In that case, we can use a technique called **pivoting** that consists in exchanging the rows (or the columns) of the system in such a way that no pivot is zero.

Example. Let us go back to the matrix (4) for which the Gauss method gives a null pivot at the second step. By just exchanging the second and the third rows, we get a non-zero pivot and can execute one step further. Indeed,

$$A^{(2)} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 0 & -1 \\ 0 & -6 & -12 \end{bmatrix} \implies P_2 A^{(2)} = \begin{bmatrix} 1 & 2 & 3 \\ 0 & -6 & -12 \\ 0 & 0 & -1 \end{bmatrix},$$

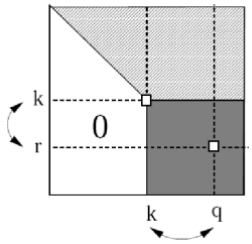
where $P_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$ is called **permutation matrix**.

The pivoting strategy used for the previous example can be generalized by finding, at every step k of the elimination, a non-zero pivot among the elements of the subcolumn $A^{(k)}(k:n,k)$. This is called a **partial pivot change** (by row).



From (3) we know that a big value of l_{ik} (coming for instance from a small $a_{kk}^{(k)}$) can amplify the rounding errors affecting the elements $a_{jk}^{(k)}$. Consequently, in order to ensure a better stability, we choose as pivot the biggest element in module of the column $A^{(k)}(k:n,k)$, and the partial pivoting is performed at every step, even if it is not strictly necessary (i.e. even if the pivot is non-zero). In this way a_{ij} and b_j are under control.

An alternative method consists looking for the pivot in the whole submatrix $A^{(k)}(k:n,k:n)$, performing what is called **complete pivoting**.



Remark that, whereas partial pivoting requires just an additional cost of n^2 tests, complete pivoting needs some $2n^3/3$, what considerably increases the cost of the Gauss method.

In general, if at the step k we have to exchange the rows k and r, we will have to multiply $A^{(k)}$ by the following **permutation matrix** P_k before continuing:

$$
\begin{array}{c}
\\
k \rightarrow \\
r \rightarrow \\
\\
\\
\\
\\
\\
\end{array}
\begin{pmatrix}
1 & & & & & & \\
& \ddots & & & & & \\
& & 0 & \dots & 1 & & \\
& & & \vdots & & & \\
& & 1 & \dots & 0 & & \\
& & & & & \ddots & \\
& & & & & & 1
\end{pmatrix}
= P_k
$$

$$
\begin{array}{cc}
\uparrow & \uparrow \\
r & k
\end{array}
$$

This means we will consider $P_k A^{(k)}$ instead of $A^{(k)}$.

Setting $\mathbf{l}_k = [0, \dots, 0, l_{k+1,k}, \dots, l_{n,k}]^T \in \mathbb{R}^n$, and defining

$$
M_k = \begin{bmatrix}
1 & \cdots & 0 & 0 & \cdots & 0 \\
\vdots & \ddots & \vdots & \vdots & & \vdots \\
0 & \cdots & 1 & a_{kk}^{(k)} & \cdots & a_{kn}^{(k)} \\
0 & \cdots & -l_{k+1,k} & 1 & \cdots & 0 \\
\vdots & & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & -l_{n,k} & 0 & \cdots & 1
\end{bmatrix} = I_n - \mathbf{l}_k \mathbf{e}_k^T
$$

as the **k-th Gaussian transformation matrix**, one finds out that

$$(M_k)_{ip} = \delta_{ip} - (\mathbf{l}_k \mathbf{e}_k^T)_{ip} = \delta_{ip} - l_{ik}\delta_{kp}, \qquad i,p = 1,\dots,n.$$

Let us analyze how partial pivoting affects the LU factorization induced by GEM. At the first step of GEM with partial pivoting, after finding out the entry a_{r1} of maximum module in the first column, the elementary permutation matrix P_1 which exchanges the first row with the r-th row is constructed (if $r = 1$, P_1 is the identity matrix). Next, the first Gaussian transformation matrix M_1 is generated and we set $A^{(2)} = M_1 P_1 A^{(1)}$. A similar approach is now taken on $A^{(2)}$, searching for a new permutation matrix P_2 and a new matrix M_2 such that

$$A^{(3)} = M_2 P_2 A^{(2)} = M_2 P_2 M_1 P_1 A^{(1)}.$$

Executing all the elimination steps, the resulting upper triangular matrix U is now given by

$$U = A^{(n)} = M_{n-1} P_{n-1} \dots M_1 P_1 A^{(1)}. \tag{5}$$

Letting $M = M_{n-1} P_{n-1} \dots M_1 P_1$ and $P = P_{n-1} \dots P_1$, we obtain that $U = MA$ and, thus, $U = (MP^{-1})PA$. It can be checked that the matrix $L = PM^{-1}$ is unit lower triangular, so that the LU factorization reads

$$\boxed{PA = LU}, \tag{6}$$

being $P = P_{n-1} P_{n-2} \dots P_2 P_1$ the **global permutation matrix**, L the **multiplier matrix** (the new ones!) and $U = A^{(n)}$.

Once the matrices L, U and P have been calculated, the resolution of the initial system is transformed into the resolution of the triangular systems

$$A\mathbf{x} = b \quad \Longrightarrow \quad PA\mathbf{x} = P\mathbf{b} \quad \Longrightarrow \quad LU\mathbf{x} = P\mathbf{b} \quad \Longrightarrow \quad \begin{cases} L\mathbf{y} = P\mathbf{b}\ , \\ U\mathbf{x} = \mathbf{y}\ . \end{cases}$$

Remark that the coefficients of the matrix L have the same values as the multipliers calculated by a factorization LU of the matrix PA without pivoting.

If complete pivoting is performed, at the first step of the process, once the element a_{qr} of largest module in submatrix $A(1:n, 1:n)$ has been found, we must exchange the first row and column with the q-th row and the r-th column. This generates the matrix $P_1 A^{(1)} Q_1$, where P_1 and Q_1 are permutation matrices by rows and by columns, respectively. As a consequence, the action of matrix M_1 is now such that $A^{(2)} = M_1 P_1 A_{(1)} Q_1$. Repeating the process, at the last step, instead of (5) we obtain

$$U = A^{(n)} = M_{n-1} P_{n-1} \dots M_1 P_1 A^{(1)} Q_1 \dots Q_{n-1}.$$

In the case of complete pivoting the LU factorization becomes

$$\boxed{PAQ = LU} \tag{7}$$

where $P = P_{n-1} \dots P_1$ is a permutation matrix that takes into account all permutations by row, and $Q = Q_1 \dots Q_{n-1}$ is a permutation matrix that takes into account all permutations by column. By construction, the matrix L is still lower triangular, and its elements have a module lower or equal to 1. As for the partial pivoting, the elements of L are the multipliers generated by the factorization LU of the matrix PAQ with no pivoting.

Once the matrices L, U, P and Q have been calculated, for solving the linear system we notice that we can write

$$A\mathbf{x} = \mathbf{b} \iff \underbrace{PAQ}_{LU}\underbrace{Q^{-1}\mathbf{x}}_{\mathbf{x}^*} = P\mathbf{b} \iff LU\mathbf{x}^* = P\mathbf{b}.$$

Which brings us to the resolution of two triangular systems and an equation

$$\begin{cases} L\mathbf{y} = P\mathbf{b}\ , \\ U\mathbf{x}^* = \mathbf{y}\ , \\ \mathbf{x} = Q\mathbf{x}^*. \end{cases}$$

Remark 1. *The matrix P is a permutation matrix. In the case where the matrix P is the identity, the matrices L and U are the matrices we are looking for (such that $LU = A$). Otherwise, we have $LU = PA$.*

Remark 2. *Using the LU factorization, obtained by fixing the value 1 for the n diagonal elements of L, we can calculate the determinant of a square matrix with $O(n^3)$ operations, thanks to the Binet theorem:*

$$\det(A) = \det(L)\det(U) = \det(U) = \prod_{k=1}^{n} u_{kk}; \tag{8}$$

indeed, the determinant of a triangular matrix is is the product of the diagonal elements.

The inverse matrix

If A is a $n \times n$ *non-singular* matrix, let us call $x^{(1)}, \ldots, x^{(n)}$ the columns of its inverse matrix A^{-1}, i.e. $A^{-1} = (x^{(1)}, \ldots, x^{(n)})$. The relation $AA^{-1} = I$ can be expressed by the following n systems : for $1 \leq k \leq n$,

$$A\mathbf{x}^{(k)} = \mathbf{e}^{(k)}, \tag{9}$$

where $\mathbf{e}^{(k)} = (0, 0, \ldots, 0, 1, 0, \ldots, 0)$ is the column vector with all the elements equal to 0 except the one corresponding to the k-th row, which equals 1 (it corresponds to the k-th column of I). Once we know the matrices L and U that factorizes A, solving the n systems (9) defined by the same matrix A requires $2n \cdot n^2 = 2n^3$ operations.

The Cholesky factorization

In the case where the $n \times n$ matrix A is symmetric and positive definite, there exists a unique upper triangular matrix R with positive diagonal elements such that

$$\boxed{A = R^T R}.$$

This factorization is called **Cholesky factorization**.

The elements r_{ij} of R can be calculated using the expressions

$$r_{11} = \sqrt{a_{11}} \tag{10}$$

and for $i = 2, \ldots, n$:

$$r_{ji} = \frac{1}{r_{jj}}\left(a_{ij} - \sum_{k=1}^{j-1} r_{ki}r_{kj}\right), \quad j = 1, \ldots, i-1, \tag{11}$$

$$r_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} r_{ki}^2} \tag{12}$$

The Cholesky factorization needs around $\frac{n^3}{3}$ operations (half the operations for a LU factorization, so it uses half of the computational time!).

Memory space limitations

A square matrix of order n is called **sparse** if the number of nonzero entries is of order n (on n^2 total entries).

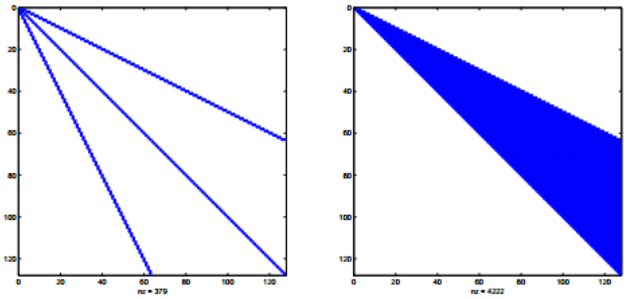
The **pattern** is the 2D representation of nonzero entries positions:

- lower band p_1: $a_{ij} = 0$ when $i > j + p_1$
- upper band p_2: $a_{ij} = 0$ when $j > i + p_2$

The maximum between p_1 and p_2 is called **matrix bandwidth**.

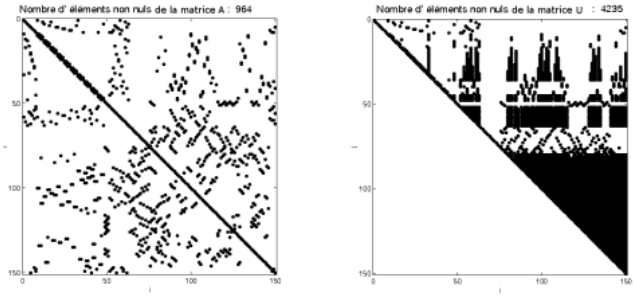
The **fill-in phenomenon** occurs when, after a LU decomposition, L and U present less sparsity then the original matrix A, leading to a bigger memory usage. To reduce the phenomenon we can apply row and column permutations to **reorder** A before performing the factorization (**pivoting**).

Example 1. Let A be a matrix of size 127×127, symmetric and positive definite. The number of non-null entries of A is 379 and thus much smaller than $(127)^2 = 16129$. It is a sparse matrix. The figure on the left shows the disposition of the non-null entries of A, whereas the one on the right shows the non-null entries of the matrix R. Even if also R is a sparse matrix, we need one order of magnitude more to allocate this matrix!



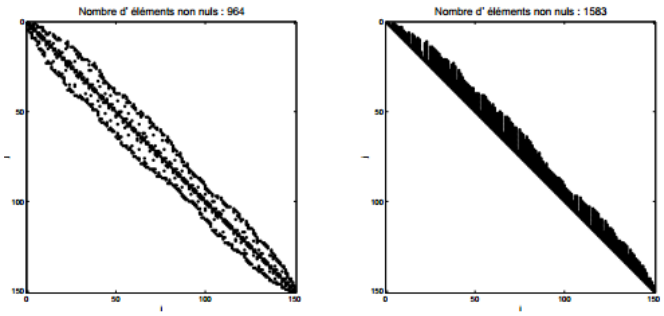
Example 2. Let us consider the problem of calculating the deformations in a structure subject to a given set of forces. The discretization using the finite elements method generates a matrix A of size 150×150. (The same matrix would have been produced by the approximation of an electric potential field.) This matrix is symmetric positive definite. The number of non-null entries of A is 964, and thus much smaller than $(150)^2 = 22500$. It is a *sparse* matrix.

The figure on the left shows the disposition of the non-null entries of A, whereas the one on the right shows the non-null entries of the matrix R.



We notice that the number of non-null entries of R is much bigger than those of A (**fill-in phenomenon**, due to the change of the structure of the matrix during the factorization). This leads to a bigger memory usage, and it's not suggested to use the sparse structure anymore. To reduce the fill-in phenomenon, we can re-order rows and columns of A in a particular fashion; this is called **re-ordering** of the matrix. There are several algorithms that allow us to do this.

For example, the following figure shows, on the left, one possibility of reordering A, while the one on the right shows the disposition of the non-null entries of the Cholesky factorization of the reordered matrix A. With the new techniques this will be much more easy and less costly.



Precision limitations

Example 3. Rounding errors can induce important differences between the calculated solution using the Gauss elimination method (GEM) and the exact solution. This happens when the *conditioning (number)* of the matrix of the system (representing how numerically stable this matrix can be) is very big.

The Hilbert matrix of size $n \times n$ is a symmetric matrix defined by:

$$A_{ij} = \frac{1}{i+j-1}, \quad i,j = 1,\dots,n$$

For example, for $n = 4$, we get:

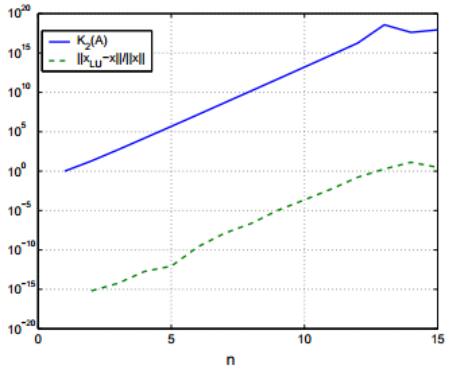
$$A = \begin{bmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \end{bmatrix}$$

This matrices are often used as benchmark to estimate the performance of the numerical method.

We consider the linear systems $A_n \mathbf{x}_n = \mathbf{b}_n$ where A_n is the Hilbert matrix of size n with $n = 4, 6, 8, 10, 12, \dots$, whereas $\mathbf{b}_n$ is chosen such that the exact solution is $\mathbf{x}_n = (1, 1, \dots, 1)^T$.

The residual are always computable so they can be used as estimation of the error when you cannot compute it.

For every n, we calculate the conditioning of the matrix, we solve the linear system by LU factorization and we get $\mathbf{x}_n^{LU}$ as the found solution. The obtained conditioning as well as the (relative) error $||\mathbf{x}_n - \mathbf{x}_n^{LU}||/||\mathbf{x}_n||$ (where $||\cdot||$ is the Euclidean norm of a vector, $||\mathbf{x}|| = \sqrt{\mathbf{x}^T \cdot \mathbf{x}}$) are shown in the figure below.



We have that the relative error is greater than 1 → 120% of error! IT'S IMPOSSIBLE!!!

Considerations on the precision

The methods we have seen until now allow us to find the solution of a linear system in a finite number of operations. That is why they are called **direct methods**. However, there are cases where these methods are not satisfactory.

Total pivoting is more stable than partial pivoting.

Definition. *We call **conditioning** of a matrix M, symmetric positive definite, the ratio between the maximum and minimum of its eigenvalues, i.e.*

$$K(M) = \frac{\lambda_{\max}(M)}{\lambda_{\min}(M)}$$

*It is also called the **spectral condition number** of M.*

It can be shown that, the bigger the conditioning of a matrix, the worse the solution obtained by a direct method.

For example, let us consider a linear system $A\mathbf{x} = \mathbf{b}$. If we solve this system with a computer, due to rounding errors, we will not find the exact solution $\mathbf{x}$ but an approximate solution $\hat{\mathbf{x}}$. The following relationship can be shown:

$$\frac{||\mathbf{x} - \hat{\mathbf{x}}||}{||\mathbf{x}||} \leq K(A)\frac{||r||}{||b||} \tag{13}$$

where $\mathbf{r}$ is the residual $\mathbf{r} = \mathbf{b} - A\hat{\mathbf{x}}$; we write as $||\mathbf{v}|| = \left(\sum_{k=1}^{n} v_k^2\right)^{1/2}$ the Euclidean norm of a vector $\mathbf{v}$.

Remark that, if the conditioning of A is big, the distance $||\mathbf{x} - \hat{\mathbf{x}}||$ between the exact solution and the numerically computed solution can be very big even if the residual is very small. So the bigger $K(A)$, the worse the solution provided by a direct method. If $K \approx 1$, the matrix is **well conditioned**. We have that $\mathbf{r}$ is an estimation of the error $||x - \hat{x}||$: if $K(A)$ is small, then the error is small when $||\mathbf{r}||$ is small; vice versa, if $K(A)$ is large we can't use $||\mathbf{r}||$ as measure for the error.

Proof for (13): Let A be a symmetric positive definite matrix, we can consider the n eigenvalues $\lambda_i > 0$ and the associated unitary eigenvectors $\{\mathbf{v}_i\}$, $i = 1, \ldots, n$: $A\mathbf{v}_i = \lambda_i \mathbf{v}_i$, $i = 1, \ldots, n$. These vectors form an orthonormal base of $\mathbb{R}^n$, which means $\mathbf{v}_i^T \mathbf{v}_j = \delta_{ij}$ for $i, j = 1, \ldots, n$. For any $\mathbf{w} \in \mathbb{R}^n$, if we write it as

$$\mathbf{w} = \sum_{i=1}^{n} w_i \mathbf{v}_i,$$

we have

$$\begin{aligned} ||A\mathbf{w}||^2 &= (A\mathbf{w})^T (A\mathbf{w}) \\ &= (\lambda_1 w_1 \mathbf{v}_1^T + \ldots + \lambda_n w_n \mathbf{v}_n^T)(\lambda_1 w_1 \mathbf{v}_1 + \ldots + \lambda_n w_n \mathbf{v}_n) \\ &= \sum_{i,j=1}^{n} \lambda_i \lambda_j w_i w_j \mathbf{v}_i^T \mathbf{v}_j = \sum_{i,j=1}^{n} \lambda_i \lambda_j w_i w_j \delta_{ij} = \sum_{i=1}^{n} \lambda_i^2 w_i^2 \end{aligned}$$

And yet, as $||\mathbf{w}||^2 = \sum_{i=1}^{n} w_i^2$, we get $||A\mathbf{w}||^2 \leq \lambda_{\max}^2 ||\mathbf{w}||^2$, i.e. $||A\mathbf{w}|| \leq \lambda_{\max} ||\mathbf{w}||$ where $\lambda_{\max}$ is the biggest eigenvalue of A.

As the eigenvalues of A^{-1} are $1/\lambda_i$, we also get $||A^{-1}\mathbf{w}|| \leq \frac{1}{\lambda_{\min}} ||\mathbf{w}|| \ \forall \mathbf{w} \in \mathbb{R}^n$, where $\lambda_{\min}$ is the smallest eigenvalue of A.

Thus, we have

$$||\mathbf{x} - \hat{\mathbf{x}}|| = ||A^{-1}\mathbf{r}|| \leq \frac{1}{\lambda_{\min}}||\mathbf{r}||,$$
$$||\mathbf{b}|| = ||A\mathbf{x}|| \leq \lambda_{\max}||\mathbf{x}||,$$

from where we directly find the inequality (13):

$$\frac{||\mathbf{x} - \hat{\mathbf{x}}||}{\lambda_{\max}||\mathbf{x}||} \leq \frac{\lambda_{\min}^{-1}||\mathbf{r}||}{||\mathbf{b}||} \iff \frac{||\mathbf{x} - \hat{\mathbf{x}}||}{||\mathbf{x}||} \leq \frac{\lambda_{\max}}{\lambda_{\min}}\frac{||\mathbf{r}||}{||\mathbf{b}||} \iff \frac{||\mathbf{x} - \hat{\mathbf{x}}||}{||\mathbf{x}||} \leq K(A)\frac{||\mathbf{r}||}{||\mathbf{b}||}$$

□

Other direct methods

- The **Thomas algorithm** is used to perform an optimized LU factorization of a tridiagonal matrix in n operations.
- The solution of an overdetermined system $A\mathbf{x} = \mathbf{b}$ with $A \in \mathbb{R}^{m\times n}$, with $m > n$, can be computed using the QR factorization or the singular value decomposition.

Linear Systems: Iterative Methods

Solve the linear system $A\mathbf{x} = \mathbf{b}$ using an **iterative method** consists in building a series of vectors $\mathbf{x}^{(k)}$, $k \geq 0$, in $\mathbb{R}^n$ that converge at the exact solution $\mathbf{x}$, i.e.:

$$\lim_{k\to\infty} \mathbf{x}^{(k)} = \mathbf{x}$$

for any initial vector $\mathbf{x}^{(0)} \in \mathbb{R}^n$.
We can consider the following recurrence relation:

$$\mathbf{x}^{(k+1)} = B\mathbf{x}^{(k)} + \mathbf{g}, \quad k \geq 0 \tag{1}$$

where B is a well chosen matrix (depending on A) and $\mathbf{g}$ is a vector (that depends on A and $\mathbf{b}$, satisfying the relation (of consistence)

$$\mathbf{x} = B\mathbf{x} + \mathbf{g}. \tag{2}$$

Given $\mathbf{x} = A^{-1}\mathbf{b}$, we get $\mathbf{g} = (I - B)A^{-1}\mathbf{b}$; the iterative method is therefore completely defined by the matrix B, known as **iteration matrix**.
By defining the error at step k as

$$\boxed{\mathbf{e}^{(k)} = \mathbf{x} - \mathbf{x}^{(k)}},$$

we obtain the following recurrence relation:

$$\begin{aligned} e^{(k+1)} &= \mathbf{x} - \mathbf{x}^{(k+)} = A^{-1}\mathbf{b} - B\mathbf{x}^{(k)} - \mathbf{g} = A^{-1}\mathbf{b} - B\mathbf{x}^{(k)} - (I - B)A^{-1}\mathbf{b} \\ &= A^{-1}\mathbf{b} - B\mathbf{x}^{(k)} - A^{-1}\mathbf{b} + BA^{-1}\mathbf{b} = -B\mathbf{x}^{(k)} + B\mathbf{x} = B(\mathbf{x} - \mathbf{x}^{(k)}) \end{aligned}$$

so

$$\mathbf{e}^{(k+1)} = B\mathbf{e}^{(k)} \quad \text{and thus} \quad \mathbf{e}^{(k+1)} = B^{k+1}\mathbf{e}^{(0)}, \quad k = 0, 1, \ldots$$

Theorem. *We have that* $\lim_{k\to\infty} \mathbf{e}^{(k)} = 0$ *for all* $\mathbf{e}^{(0)}$ *(and thus for all* $\mathbf{x}^{(0)}$*) if and only if*

$$\boxed{\rho(B) = \max |\lambda_i(B)| < 1},$$

where $\rho(B)$ *is the spectral radius of the matrix* B*, and* $\lambda_i(B)$ *are the eigenvalues of the matrix* B*.*

The smaller the value of $\rho(B)$, the less iterations are needed to reduce the initial error $e^{(0)}$ of a given factor or under a threshold ε; we would require at least $k_{\min}$ iterations, where

$$\min(k_{\min}) \quad \text{s.t.} \quad \rho(B)^{k_{\min}} \leq \varepsilon. \tag{3}$$

Construction of an iterative method

A general way of setting up an iterative method is based on the decomposition of the matrix A:

$$A = P - (P - A)$$

where P in an invertible matrix called **preconditioner** of A.
Hence,

$$A\mathbf{x} = \mathbf{b} \quad \Longleftrightarrow \quad P\mathbf{x} = (P - A)\mathbf{x} + \mathbf{b}$$

which is of the form (2) leaving

$$B = P^{-1}(P - A) = I - P^{-1}A \quad \text{and} \quad \mathbf{g} = P^{-1}\mathbf{b}.$$

We can define the corresponding iterative method

$$P(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = \mathbf{r}^{(k)} \quad k \geq 0$$

where $\mathbf{r}^{(k)}$ represents the **residual** at the iteration k: $\boxed{\mathbf{r}^{(k)} = \mathbf{b} - A\mathbf{x}^{(k)}}$.

We can generalize this method as follows:

$$\boxed{P(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = \alpha_k \mathbf{r}^{(k)}} \quad k \geq 0 \tag{4}$$

where $\alpha_k \neq 0$ is a parameter that improves the convergence of the series $\mathbf{x}^{(k)}$, called **acceleration parameter**. The method (4) is called **Richardson's method**.

This is equal to solve the linear system

$$P\mathbf{z}^{(k)} = \mathbf{r}^{(k)}, \quad \text{with } \mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{z}^{(k)} \iff P\left(\frac{\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}}{\alpha_k}\right) = \mathbf{r}^{(k)}$$

where $\mathbf{z}^{(k)} = \frac{\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}}{\alpha_k}$ is called the **preconditioned residual** at step k.

The matrix P has to be chosen in such a way that renders the cost of solving (4) small enough. For example a diagonal or triangular P matrix would comply with this criterion.

Jacobi method

If the elements of the diagonal of $A = (a_{ij}) \in \mathbb{R}^{n \times n}$ are non-zero, so $a_{ii} \neq 0, \forall i = 0, \ldots, n$, we can write

$$P = D = \text{diag}(a_{11}, a_{22}, \ldots, a_{nn})$$

D with the diagonal part of A being:

$$D_{ij} = \begin{cases} 0 & \text{if } i \neq j \\ a_{ij} & \text{if } i = j. \end{cases}$$

The Jacobi method corresponds to this choice with $\alpha_k = 1$ for all k. We deduce:

$$D\mathbf{x}^{(k+1)} = \mathbf{b} - (A - D)\mathbf{x}^{(k)} \quad k \geq 0.$$

By components:

$$\boxed{x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1, j \neq i}^{n} a_{ij} x_j^{(k)}\right)}, \quad i = 1, \ldots, n. \tag{5}$$

The Jacobi method can be written under the general form

$$\boxed{\mathbf{x}^{(k+1)} = B\mathbf{x}^{(k)} + \mathbf{g}},$$

with

$$\boxed{B = B_J = D^{-1}(D - A) = I - D^{-1}A}, \quad \mathbf{g} = \mathbf{g}_J = D^{-1}\mathbf{b}.$$

Gauss-Seidel method

In order to obtain a faster convergence, we can include the newly computed components of vector $x_j^{(k+1)}$, $j = 1, \ldots, i-1$ to the previous $x_j^{(k)}$, $j \geq i$, to compute $x_i^{(k+1)}$. This method is defined as follows:

$$\boxed{\mathbf{x}_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right)}, \quad i = 1, \ldots, n.$$

In this case the update is **sequential** rather than **simultaneous**, but leads to faster convergence. This method corresponds to (1) with $P = D - E$ and $\alpha_k = 1$ ($\forall k \geq 0$), where E is the lower triangular matrix

$$\begin{cases} E_{ij} = -a_{ij} & \text{if } i > j \\ E_{ij} = 0 & \text{if } i \leq j \end{cases}$$

(lower triangular part of A without the diagonal and with its elements' sign inverted).

We can write this method under the form (4), with the iteration matrix $B = B_{GS}$ given by

$$\boxed{B_{GS} = (D - E)^{-1}(D - E - A)}$$

and

$$\boxed{\mathbf{g}_{GS} = (D - E)^{-1}\mathbf{b}}.$$

Convergence

We have the following convergence results:

- **Proposition 1.** If the matrix A is *strictly diagonally dominant by row*, i.e.,
$$|a_{ii}| > \sum_{j=1,\ldots,n; j \neq i} |a_{ij}|, \quad i = 1, \ldots, n,$$
then the Jacobi and the Gauss-Seidel methods converge.
- If A is *symmetric positive definite*, then the Gauss-Seidel method converges (Jacobi maybe not).
- **Proposition 2.** Let A be a *tridiagonal non-singular matrix* whose diagonal elements are all non-null. Then the Jacobi and the Gauss-Seidel methods are either *both divergent or both convergent*. In the latter case, $\rho(B_{GS}) = \rho(B_J)^2$.

Richardson method

Let consider the iterative method (4):

$$P(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = \alpha_k \mathbf{r}^{(k)}, \quad k \geq 0.$$

If the acceleration parameter is $\alpha_k = \alpha$ (a constant) this method is called **stationary preconditioned Richardson method**; otherwise is called **dynamic preconditioned Richardson method** when α_k varies during the iterations.

Recall that the matrix P is called **preconditioner** of A.

If A and P are symmetric positive definite, then there are two optimal criteria to choose α_k:

1. **Stationary case:**

$$\alpha_k = \alpha_{opt} = \frac{2}{\lambda_{\min} + \lambda_{\max}}, \quad k \geq 0,$$

where $\lambda_{\min}$ and $\lambda_{\max}$ represent the smaller and the larger eigenvalue of the matrix $P^{-1}A$.

2. **Dynamic case:**

$$\alpha_k = \frac{(\mathbf{z}^{(k)})^T \mathbf{r}^{(k)}}{(\mathbf{z}^{(k)})^T A \mathbf{z}^{(k)}}, \quad k \geq 0,$$

where $\mathbf{z}^{(k)} = P^{-1}\mathbf{r}^{(k)}$ is the *preconditioned residual*. This method is also called **preconditioned gradient method**.

If $P = I$ and A is symmetric definite positive, we get the following methods:

- the **Stationary Richardson** if $B = I - \alpha_k A$, $\mathbf{g} = \mathbf{b}$ and we choose:

$$\alpha_k = \alpha_{opt} = \frac{2}{\lambda_{\min}(A) + \lambda_{\max}(A)}. \tag{6}$$

- the **Gradient method** if $B = I - \alpha_k A$, $\mathbf{g} = \mathbf{b}$ and:

$$\alpha_k = \frac{(\mathbf{r}^{(k)})^T \mathbf{r}^{(k)}}{(\mathbf{r}^{(k)})^T A \mathbf{r}^{(k)}}, \quad k \geq 0, \tag{7}$$

(we have that $\mathbf{z}^{(k)} = P^{-1}\mathbf{r}^{(k)} = I^{-1}\mathbf{r}^{(k)} = \mathbf{r}^{(k)}$).

Methodology. The gradient method can be written as:

Let $\mathbf{x}^{(0)}$ be given, set $\mathbf{r}^{(0)} = \mathbf{b} - A\mathbf{x}^{(0)}$, then for $k \geq 0$,

$$\begin{aligned}
&\text{solve the linear system} && P\mathbf{z}^{(k)} = \mathbf{r}^{(k)} \\
&\text{compute the acceleration parameter} && \alpha_k = \frac{(\mathbf{z}^{(k)})^T \mathbf{r}^{(k)}}{(\mathbf{z}^{(k)})^T A \mathbf{z}^{(k)}} \\
&\text{update the solution} && \mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{z}^{(k)} \\
&\text{update the residual} && \mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - \alpha_k A \mathbf{z}^{(k)}.
\end{aligned}$$

We can compute the preconditioned residual $\mathbf{z}^{(k)}$ in any way we want: with the LU decomposition, the Gauss method, etc. We have to apply once A and inverse P at each iteration. P should then be such that the resolution of the associated system results easy (i.e. it requires a reasonable amount of computing cost). For example, we can choose a diagonal P (like in the gradient or the stationary Richardson cases) or triangular.

Convergence of Richardson method

When A and P are symmetric positive definite (s.p.d.) and with the two optimal choices for α, we can show that the series $\{x^{(k)}\}$ given by the preconditioned Richardson method (stationary and dynamic) converges to $\mathbf{x}$ when $k \to \infty$, and that

$$||\mathbf{x}^{(k)} - \mathbf{x}||_A \leq \left(\frac{K(P^{-1}A) - 1}{K(P^{-1}A) + 1}\right)^k ||\mathbf{x}^{(0)} - \mathbf{x}||_A, \quad k \geq 0, \tag{8}$$

where $||\mathbf{v}||_A = \sqrt{\mathbf{v}^T A \mathbf{v}}$ is the energy norm of A and $K(P^{-1}A)$ is the condition number of $P^{-1}A$.

Remark 1. In the case of the gradient method or the Richardson stationary method the error estimation becomes

$$||\mathbf{x}^{(k)} - \mathbf{x}||_A \leq \left(\frac{K(A) - 1}{K(A) + 1}\right)^k ||\mathbf{x}^{(0)} - \mathbf{x}||_A, \quad k \geq 0.$$

The gradient method converges faster, followed by the Gauss-Seidel method and the Jacobi method. If A is a generic matrix, keeping low both K and the number of operations is hard.

Remark 2. If A and P are s.p.d., we have that

$$K(P^{-1}A) = \frac{\lambda_{\max}(P^{-1}A)}{\lambda_{\min}(P^{-1}A)}.$$

Theorem. Assume that P is a nonsingular matrix and that $P^{-1}A$ has positive real eigenvalues, ordered in such a way that $\lambda_1 = \lambda_{\max} \geq \lambda_2 \geq \ldots \geq \lambda_n = \lambda_{\min} > 0$. Then, the stationary Richardson method (4) with $\alpha_k = \alpha$ is convergent iff $0 < \alpha < 2/\lambda_{\max}$. Moreover, letting

$$\alpha_{\text{opt}} = \frac{2}{\lambda_{\max} + \lambda_{\min}},$$

the spectral radius of the iteration matrix R_α is minimum if $\alpha = \alpha_{\text{opt}}$, with

$$\rho_{\text{opt}} = \min_{\alpha}[\rho(R_\alpha)] = \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}}.$$

Proof. The iteration matrix of the method is given by $R_\alpha = I - \alpha P^{-1}A$, where the eigenvalues of R_α are of the form $\lambda_i(R_\alpha) = 1 - \alpha\lambda_i$ (where λ_i is the eigenvalue of the matrix $P^{-1}A$). The method is convergent if and only if $|\lambda_i(R_\alpha)| = |1 - \alpha\lambda_i| < 1$ for $i = 1, \ldots, n$, therefore $-1 < 1 - \alpha\lambda_i < 1$ for $i = 1, \ldots, n$. As $\alpha > 0$, this is the equivalent to $-1 < 1 - \alpha\lambda_{\max}$, from where the necessary and sufficient condition for convergence remains $\alpha < 2/\lambda_{\max}$. Consequently, $\rho(R_\alpha)$ is minimum if $1 - \alpha\lambda_{\min} = \alpha\lambda_{\max} - 1$, i.e., for $\alpha_{opt} = 2/(\lambda_{\min} + \lambda_{\max})$. By substitution, we obtain

$$\rho_{opt} = \rho(R_{opt}) = 1 - \alpha_{opt}\lambda_{\min} = 1 - \frac{2\lambda_{\min}}{\lambda_{\min} + \lambda_{\max}} = \frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\min} + \lambda_{\max}}$$

which allows us to complete the proof.

□

In the dynamic case, we get a result that allows us to optimally choose the iteration parameters at each step, if the matrix A is symmetric definite positive:

Theorem 1 (Dynamic case)**.** *If A is symmetric definite positive, the optimal choice for α_k is given by*

$$\alpha_k = \frac{(\mathbf{r}^{(k)}, \mathbf{z}^{(k)})}{(A\mathbf{z}^{(k)}, \mathbf{z}^{(k)})}, \quad k \geq 0 \tag{9}$$

where

$$\mathbf{z}^{(k)} = P^{-1}\mathbf{r}^{(k)}. \tag{10}$$

Proof. On the one hand we have

$$\mathbf{r}^{(k)} = \mathbf{b} - A\mathbf{x}^{(k)} = A(\mathbf{x} - \mathbf{x}^{(k)}) = -A\mathbf{e}^{(k)},$$

and thus, using (10),

$$P^{-1}A\mathbf{e}^{(k)} = -\mathbf{z}^{(k)},$$

where $\mathbf{e}^{(k)}$ represents the error at the step k. On the other hand

$$\mathbf{e}^{(k+1)} = \mathbf{e}^{(k+1)}(\alpha) = \underbrace{(I - \alpha P^{-1}A)}_{R_\alpha}\mathbf{e}^{(k)}.$$

We notice that, in order to update the residual, we have the relation

$$\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - \alpha A\mathbf{z}^{(k)} = \mathbf{r}^{(k)} - \alpha AP^{-1}\mathbf{r}^{(k)}.$$

Thus, expressing as $||\cdot||_A$ the vector norm associated to the scalar product $(\mathbf{x},\mathbf{y})_A = (A\mathbf{x},\mathbf{y})$, which means, $||\mathbf{x}||_A = (A\mathbf{x},\mathbf{x})^{1/2}$ we can write

$$\begin{aligned}||\mathbf{e}^{(k+1)}||_A^2 &= (A\mathbf{e}^{(k+1)},\mathbf{e}^{(k+1)})\\ &= -(\mathbf{r}^{(k+1)},\mathbf{e}^{(k+1)})\\ &= -(\mathbf{r}^{(k)}-\alpha AP^{-1}\mathbf{r}^{(k)},\mathbf{e}^{(k)}-\alpha P^{-1}A\mathbf{e}^{(k)})\\ &= -(\mathbf{r}^{(k)},\mathbf{e}^{(k)})+\alpha[(\mathbf{r}^{(k)},P^{-1}A\mathbf{e}^{(k)})+(A\mathbf{z}^{(k)},\mathbf{e}^{(k)})]\\ &\quad -\alpha^2(A\mathbf{z}^{(k)},P^{-1}A\mathbf{e}^{(k)})\end{aligned}$$

Now we choose α as the α_k that minimizes $||\mathbf{e}^{(k+1)}(\alpha)||_A$:

$$\frac{d}{d\alpha}\left\|\mathbf{e}^{(k+1)}(\alpha)\right\|_A\Big|_{\alpha=\alpha_k} = 0$$

We then obtain

$$\alpha_k = \frac{1}{2}\frac{(\mathbf{r}^{(k)},P^{-1}A\mathbf{e}^{(k)})+(A\mathbf{z}^{(k)},\mathbf{e}^{(k)})}{(A\mathbf{z}^{(k)},P^{-1}A\mathbf{e}^{(k)})} = \frac{1}{2}\frac{-(\mathbf{r}^{(k)},\mathbf{z}^{(k)})+(A\mathbf{z}^{(k)},\mathbf{e}^{(k)})}{-(A\mathbf{z}^{(k)},\mathbf{z}^{(k)})}$$

and using the equality $(A\mathbf{z}^{(k)},\mathbf{e}^{(k)}) = (\mathbf{z}^{(k)},A\mathbf{e}^{(k)})$ knowing that A is symmetric definite positive, and noting that $A\mathbf{e}^{(k)} = -\mathbf{r}^{(k)}$, we find

$$\alpha_k = \frac{(\mathbf{r}^{(k)},\mathbf{z}^{(k)})}{(A\mathbf{z}^{(k)},\mathbf{z}^{(k)})}$$

□

The conjugate gradient method

When A and P are s.p.d., there exists a very efficient and effective method to iteratively solve the system, which converges even faster then the gradient method (at most n steps): the **conjugate gradient method**.

Let $\mathbf{x}^{(0)}$ be given; we compute $\mathbf{r}^{(0)} = \mathbf{b}-A\mathbf{x}^{(0)}$, $\mathbf{z}^{(0)} = P^{-1}\mathbf{r}^{(0)}$, $\mathbf{p}^{(0)} = \mathbf{z}^{(0)}$, then for $k \geq 0$,

$$\boxed{\begin{aligned}\alpha_k &= \frac{(\mathbf{p}^{(k)})^T\mathbf{r}^{(k)}}{(\mathbf{p}^{(k)})^TA\mathbf{p}^{(k)}}\\ \mathbf{x}^{(k+1)} &= \mathbf{x}^{(k)}+\alpha_k\mathbf{p}^{(k)}\\ \mathbf{r}^{(k+1)} &= \mathbf{r}^{(k)}-\alpha_kA\mathbf{p}^{(k)}\\ P\mathbf{z}^{(k+1)} &= \mathbf{r}^{(k+1)}\\ \beta_k &= \frac{(A\mathbf{p}^{(k)})^T\mathbf{z}^{(k+1)}}{(A\mathbf{p}^{(k)})^T\mathbf{p}^{(k)}}\\ \mathbf{p}^{(k+1)} &= \mathbf{z}^{(k+1)}-\beta_k\mathbf{p}^{(k)}.\end{aligned}}$$

The error estimate is given by

$$||\mathbf{e}^{(k)}||_A = ||\mathbf{x}^{(k)}-\mathbf{x}||_A \leq \frac{2c^k}{1+c^{2k}}||\mathbf{x}^{(0)}-\mathbf{x}||_A,\ k\geq 0 \ \text{ where } \ c = \frac{\sqrt{K_2(P^{-1}A)}-1}{\sqrt{K_2(P^{-1}A)}+1}.$$

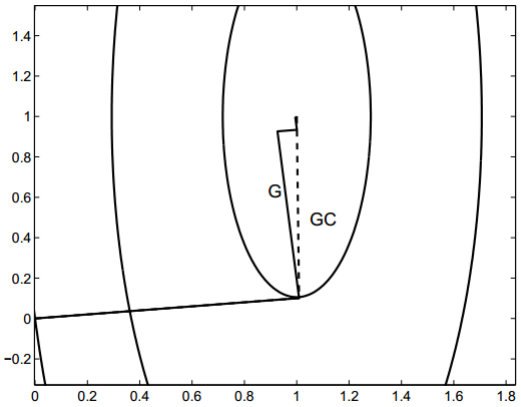


Fig. 4.7. Directions for the conjugate gradient method (denoted by CG, dashed line) and the gradient method (denoted by G, solid line). Notice that the CG method reaches the solution after two iterations.

Convergence criteria

As for direct methods, we have the following error bound:

If A is s.p.d, then

$$\boxed{\frac{||\mathbf{x}^{(k)} - \mathbf{x}||}{||\mathbf{x}||} \leq K(A)\frac{||\mathbf{r}^{(k)}||}{||\mathbf{b}||}} \qquad (11)$$

The relative error at the iteration k is bounded by the condition number of A times the residual scaled with the right hand side.
We can also use another relation in case of a preconditioned system:

$$\boxed{\frac{||\mathbf{x}^{(k)} - \mathbf{x}||}{||\mathbf{x}||} \leq K(P^{-1}A)\frac{||P^{-1}\mathbf{r}^{(k)}||}{||P^{-1}\mathbf{b}||}}$$

Some observations

Stopping conditions

As for nonlinear systems, we can choose to control the residual $\mathbf{r}$ or the increment:

- $||\mathbf{r}^{(k_{\min})}|| \leq \varepsilon||\mathbf{b}|| \implies ||\mathbf{e}^{(k_{\min})}||/||\mathbf{x}|| \leq \varepsilon K(A)$, meaningful only if $K(A)$ is reasonably small
- $\delta^{(k)} = \mathbf{x}^{(k+1)} - \mathbf{x}^{(k)} \implies ||\delta^{(k_{\min})}|| \leq \varepsilon$ → latter if $\rho(B) \gg 1$

Choosing the method

The choice of the method is particularly important for large A, and depends largely on context (A properties, resources, etc). Direct methods are usually more effective in absence of a good P, but more sensitive to ill-conditioning and require large storage.

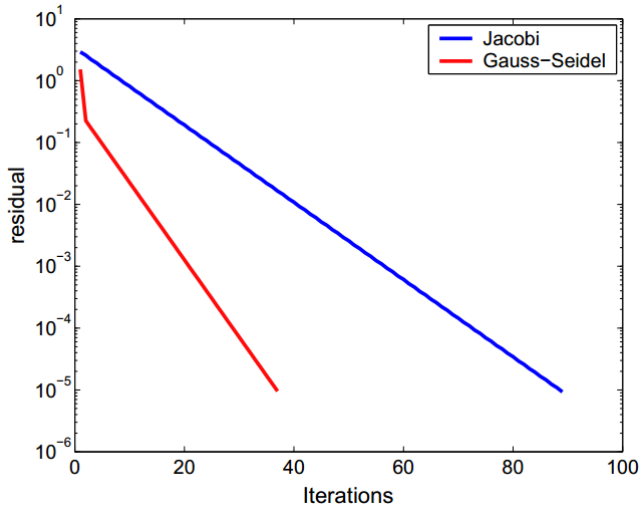
Memory and computational costs

Computational cost (*flops*) and used memory (*bytes*): we consider the Cholesky an the conjugate gradient methods for sparse matrices of size n (originated in the approximation of solutions to the Poisson equation in the finite elements method) with m non zero elements.

		Cholesky		Conjugate gradient		flops(Chol.)/	Mem(Chol.)/
n	m/n^2	flops	Memory	flops	Memory	flops(GC)	Mem(GC)
47	0.12	8.05e+03	464	1.26e+04	228	0.64	2.04
83	0.07	3.96e+04	1406	3.03e+04	533	1.31	2.64
150	0.04	2.01e+05	4235	8.86e+04	1245	2.26	3.4
225	0.03	6.39e+05	9260	1.95e+05	2073	3.27	4.47
329	0.02	1.74e+06	17974	3.39e+05	3330	5.15	5.39
424	0.02	3.78e+06	30815	5.49e+05	4513	6.88	6.83
530	0.01	8.31e+06	50785	8.61e+05	5981	9.65	8.49
661	0.01	1.19e+07	68468	1.11e+06	7421	10.66	9.23

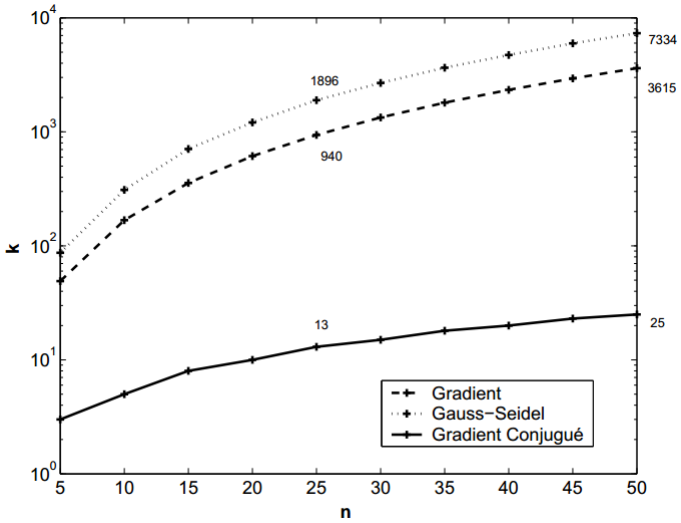
Iterative methods: Jacobi, Gauss-Seidel

Behavior of the error for a well conditioned matrix ($K = 20$). Gauss-Seidel is performing better than Jacobi, as we expect.



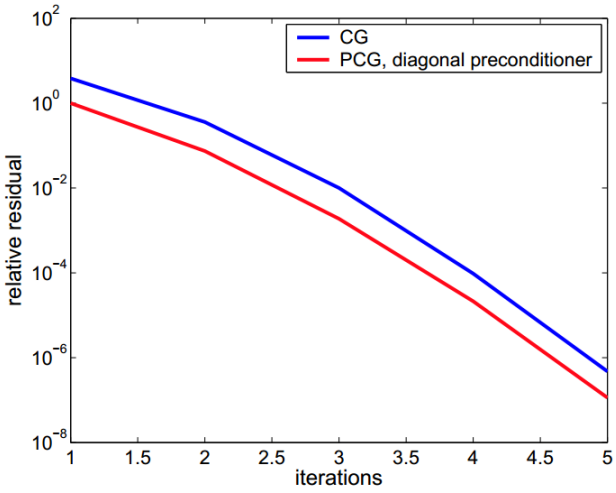
Iterative methods: G-S, Gradient, Gradient Conjugate

Number of iterations needed to reach a solution as function of the size n of a stiffness matrix, tolerance 10^{-6}. We can notice that the Gradient Conjugate is better!

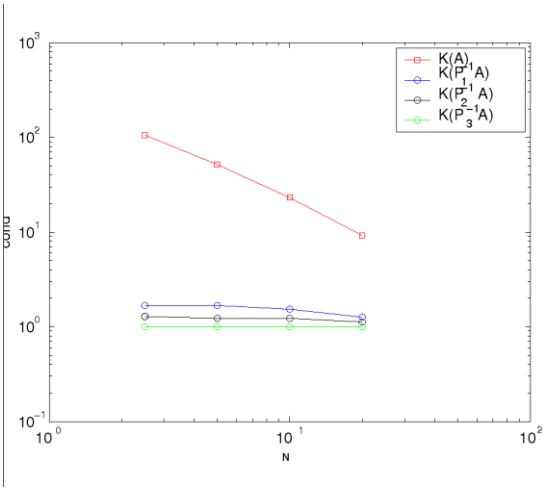


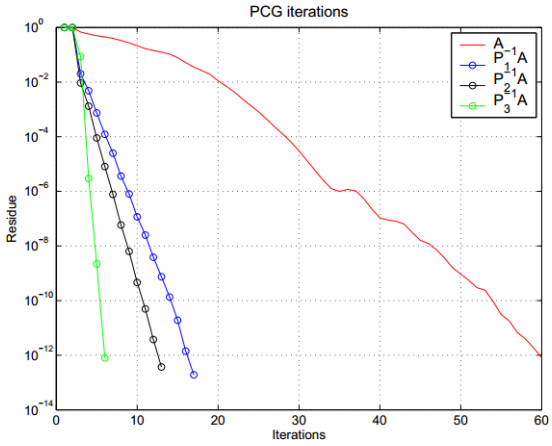
Preconditioning

The convergence conjugate gradient and the preconditioned conjugate gradient methods with a diagonal preconditioner ($K = 4 \cdot 10^8$).



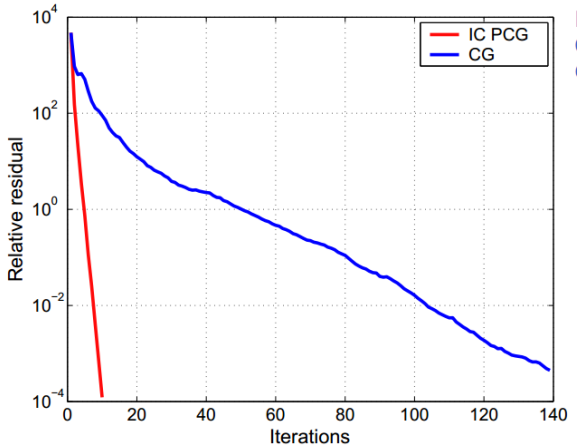
Role of the preconditioning over the condition number of a matrix (approximation by the finite element method of the Laplacian operator)





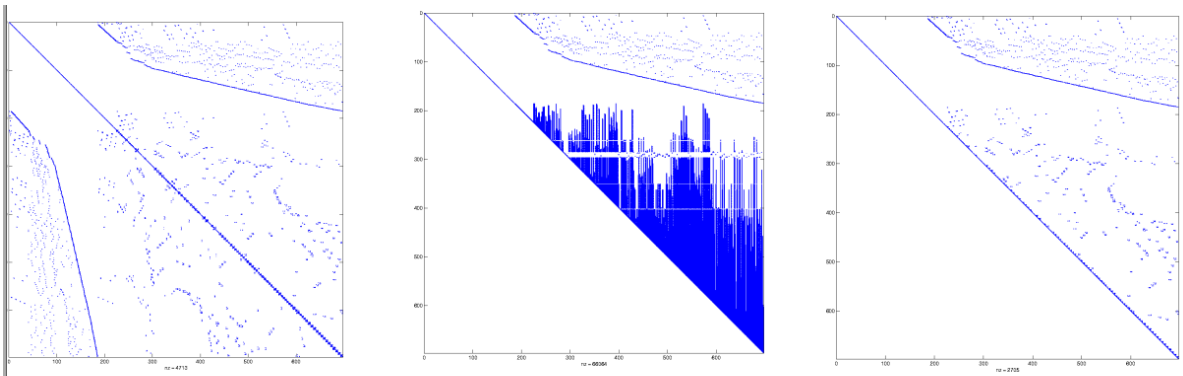
Sometimes the condition number doesn't depend on the dimension of the matrix (green line), and that's the ideal condition!

The convergence of the conjugate gradient method and the preconditioned conjugate gradient method with a preconditioner based on an Cholesky incomplete decomposition for a sparse matrix originated from the finite element method ($K = 1.5 \cdot 10^3$).



Incomplete Cholesky Preconditioned Conjugate Gradient vs. Conjugate Gradient

Comparison between the non zero elements of the sparse matrix A from the previous example, its Cholesky factor R and the matrix $\tilde{R}$ obtained by the Cholesky incomplete decomposition:



A R $\tilde{R}$

Non-linear systems

We want to generalize the *Newton method* for the case of non-linear systems. To do this, we define the *Jacobian* matrix of the vector $\mathbf{f}$:

$$J_f(\mathbf{x} = (x_1, x_2)) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix}.$$

If $J_{\mathbf{f}}(\mathbf{x}^{(k)})$ is invertible, the Newton method for non linear systems is written: *Let* $\mathbf{x}^{(0)} = (x_1^{(0)}, x_2^{(0)})$*, we compute for* $k = 0, 1, 2, \ldots$

$$\boxed{\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - [J_{\mathbf{f}}(\mathbf{x}^{(k)})]^{-1}\mathbf{f}(\mathbf{x}^{(k)})}, \quad k = 0, 1, 2 \ldots \qquad (12)$$

We can write (12) as

$$\boxed{[J_{\mathbf{f}}(\mathbf{x}^{(k)})](\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = -\mathbf{f}(\mathbf{x}^{(k)})}, \quad k = 0, 1, 2 \ldots \qquad (14)$$

Eigenvalues and Eigenvectors

Let $A \in \mathbb{C}^{n\times n}$, the **eigenvalue problem** consists in finding a scalar λ (real or complex) and a nonnull vector $\mathbf{x}$ such that

$$A\mathbf{x} = \lambda\mathbf{x} \qquad (1)$$

Any such λ is called an **eigenvalue** of A, while $\mathbf{x}$ is the associated **eigenvector**. The latter is not unique; indeed all multiples of $\mathbf{x}$, $\alpha\mathbf{x}$ with $\alpha \neq 0$, are also eigenvectors associated with λ.

Should $\mathbf{x}$ be known, λ can be recovered by using the **Rayleigh quotient**

$$\mathbf{x}^H A\mathbf{x}/||\mathbf{x}||^2 = \bar{\mathbf{x}}^T A\mathbf{x}/||\mathbf{x}||^2.$$

The eigenvalues of A are the roots of the **characteristic polynomial** of A:

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

A $n \times n$ matrix has exactly n eigenvalues (real or complex), not necessarily distinct. A matrix $A \in \mathbb{C}^{n\times n}$ is said to be **diagonalizable** if there exists a nonsingular matrix $U \in \mathbb{C}^{n\times n}$ such that

$$U^{-1}AU = \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n). \qquad (2)$$

The columns of U are the eigenvectors of A. If A is diagonal or triangular, the λ's are its diagonal entries; otherwise, if A is a general large matrix, seeking the zeros of p_A is hard.

The power method

Let $A \in \mathbb{R}^{n\times n}$, so with real entries, and assume that its eigenvalues are ordered

$$|\lambda_1| = |\lambda_{\max}| > |\lambda_2| \geq |\lambda_3| \geq \ldots \geq |\lambda_n| = |\lambda_{\min}|.$$

The **power method** can find the greater eigenvalue $\lambda_{\max} = \lambda_1$ of a non-singular matrix A.

Let $\mathbf{x}^{(0)} \in \mathbb{C}^n$ be given and set $\mathbf{y}^{(0)} = \mathbf{x}^{(0)}/||\mathbf{x}^{(0)}||$, for $k = 1, 2, \ldots$ compute

$$\boxed{\begin{aligned} \mathbf{x}^{(k)} &= A\mathbf{y}^{(k-1)} \\ \mathbf{y}^{(k)} &= \frac{\mathbf{x}^{(k)}}{||\mathbf{x}^{(k)}||} \\ \lambda^{(k)} &= (\mathbf{y}^{(k)})^H A\mathbf{y}^{(k)} \end{aligned}}$$

until $|\lambda^{(k)} - \lambda^{(k-1)}| < \varepsilon|\lambda^{(k)}|$ (stopping condition), where ε is the desired tolerance.

By induction on k one can check that

$$\mathbf{y}^{(k)} = \frac{\mathbf{x}^{(k)}}{||\mathbf{x}^{(k)}||} = \frac{A\mathbf{y}^{(k-1)}}{||A\mathbf{y}^{(k-1)}||} = \ldots = \frac{A^k\mathbf{y}^{(0)}}{||A^k\mathbf{y}^{(0)}||}, \quad k \geq 1.$$

This relation explains the role played by the powers of A in the method: the term $\mathbf{y}^{(k)}$ can also be expressed as a power, thus the name of the method.

This method generates a sequence of unitary vectors $\{\mathbf{y}^{(k)}\}$ such that for $k \to \infty$ they align in the direction of the eigenvector $\mathbf{x}_1$. In all cases, we have that $\lambda^{(k)} \to \lambda_{\max} = \lambda_1$ for $k \to \infty$.

Convergence

Since the eigenvectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$ of A are linearly independent, these eigenvectors form a basis for $\mathbb{C}^n$. Thus the vectors $\mathbf{x}^{(0)}$ and $\mathbf{y}^{(0)}$ can be written as

$$\mathbf{x}^{(0)} = \sum_{i=1}^{n} \alpha_i \mathbf{x}_i, \qquad \mathbf{y}^{(0)} = \beta^{(0)} \sum_{i=1}^{n} \alpha_i \mathbf{x}_i, \qquad \text{with } \beta^{(0)} = 1/||\mathbf{x}^{(0)}|| \text{ and } \alpha_i \in \mathbb{C}.$$

At the first step the power method gives

$$\mathbf{x}^{(1)} = A\mathbf{y}^{(0)} = \beta^{(0)} A \sum_{i=1}^{n} \alpha_i \mathbf{x}_i = \beta^{(0)} \sum_{i=1}^{n} \alpha_i \lambda_i \mathbf{x}_i \quad \text{and, similarly,}$$

$$\mathbf{y}^{(1)} = \beta^{(1)} \sum_{i=1}^{n} \alpha_i \lambda_i \mathbf{x}_i, \quad \beta^{(1)} = \frac{1}{||\mathbf{x}^{(0)}|| \cdot ||\mathbf{x}^{(1)}||}.$$

At a given step k we will have

$$\mathbf{y}^{(k)} = \beta^{(k)} \sum_{i=1}^{n} \alpha_i \lambda_i^k \mathbf{x}_i, \qquad \beta^{(k)} = \frac{1}{||\mathbf{x}^{(0)}|| \cdots ||\mathbf{x}^{(k)}||} = \frac{1}{\prod_{i=0}^{k} ||\mathbf{x}^{(i)}||}$$

And therefore

$$\mathbf{y}^{(k)} = \lambda_1^k \beta^{(k)} \left(\alpha_1 \mathbf{x}_1 + \sum_{i=2}^{n} \alpha_i \frac{\lambda_i^k}{\lambda_1^k} \mathbf{x}_i \right).$$

Since $|\lambda_i/\lambda_1| < 1$ for $i = 2, \ldots, n$, the vector $\mathbf{y}^{(k)}$ tends to align along the same direction as the eigenvector $\mathbf{x}_1$ when k tends to $+\infty$, provided $\alpha_1 \neq 0$.

The inverse power method

The **inverse power method** can find the smaller eigenvalue of a non-singular matrix A. It is like the previous method, but if A is non-singular we can use A^{-1}, which eigenvalues are the reciprocal of those of A, to obtain the eigenvalue of A with minimum modulus.

Let $\mathbf{x}^{(0)} \in \mathbb{C}^n$ be given and set $\mathbf{y}^{(0)} = \mathbf{x}^{(0)}/||\mathbf{x}^{(0)}||$, for $k = 1, 2, \ldots$ compute

$$\boxed{\begin{aligned} \mathbf{x}^{(k)} &= A^{-1}\mathbf{y}^{(k-1)} \\ \mathbf{y}^{(k)} &= \frac{\mathbf{x}^{(k)}}{||\mathbf{x}^{(k)}||} \\ \mu^{(k)} &= (\mathbf{y}^{(k)})^H A^{-1} \mathbf{y}^{(k)} \end{aligned}}$$

until $|\mu^{(k)} - \mu^{(k-1)}| < \varepsilon |\mu^{(k)}|$, where ε is the desired tolerance.

If A admits n linearly independent eigenvectors, and if also the eigenvalue $\lambda_{\min} = \lambda_n$ of minimum modulus is distinct from the others, then

$$\lim_{k \to \infty} \mu^{(k)} = 1/\lambda_{\min},$$

At each step k we have to solve a linear system of the form $A\mathbf{x}^{(k)} = \mathbf{y}^{(k-1)}$, and we can use for example the LU factorization or the Cholesky factorization.

The power method with shift

The power method with shift can find the eigenvalue of A near to a given number μ.

Define $A_\mu = A - \mu I$, whose eigenvalues are $\lambda(A_\mu) = \lambda(A) - \mu$. In order to approximate λ_μ, we can at first approximate the eigenvalue of minimum length of A_μ with the inverse power method. Let $\mathbf{x}^{(0)} \in \mathbb{C}^n$ be given and set $\mathbf{y}^{(0)} = \mathbf{x}^{(0)}/||\mathbf{x}^{(0)}||$, for $k = 1, 2, \dots$ compute

$$\boxed{\begin{aligned} \mathbf{x}^{(k)} &= A_\mu^{-1}\mathbf{y}^{(k-1)} \\ \mathbf{y}^{(k)} &= \frac{\mathbf{x}^{(k)}}{||\mathbf{x}^{(k)}||} \\ \lambda_\mu^{(k)} &= \frac{1}{(\mathbf{y}^{(k)})^H A_\mu^{-1}\mathbf{y}^{(k)}} \end{aligned}}$$

until $|\lambda_\mu^{(k)} - \lambda_\mu^{(k-1)}| < \varepsilon|\lambda_\mu^{(k)}|$, where ε is the desired tolerance. The searched eigenvalue of A is approximated by $\lambda = \lambda_\mu + \mu$.

The value of the shift can be modified during the iterations, by setting $\mu = \lambda^{(k)}$. This yields a faster convergence; however the computational cost grows substantially since now at each iteration the matrix A_μ does change and the LU factorization has to be performed at each iteration.

How to compute the shift

We need to locate (more or less accurately) the eigenvalues of A in the complex plane.

Let $A \in \mathbb{C}^{n\times n}$ be a square matrix of dimension n. The **Gershgorin circles** $C_i^{(r)}$ and $C_i^{(c)}$ associated with its i-th row and i-th column are respectively defined as

$$\mathcal{R}_i = C_i^{(r)} = \left\{ z \in \mathbb{C} : |z - a_{ii}| \le \sum_{j=1, j\neq i}^{n} |a_{ij}| \right\},$$

$$\mathcal{C}_i = C_i^{(c)} = \left\{ z \in \mathbb{C} : |z - a_{ii}| \le \sum_{j=1, j\neq i}^{n} |a_{ji}| \right\}.$$

$\mathcal{R}_i$ is called the i-th **row circle** and $\mathcal{C}_i$ the i-th **column circle**.

First Gershgorin theorem. All the eigenvalues of a given matrix $A \in \mathbb{C}^{n\times n}$ belong to the region of the complex plane which is the intersection of the two regions formed respectively by the union of the row circles $\mathcal{S}_\mathcal{R} = \bigcup_{i=1}^n \mathcal{R}_i$ and the union of the column circles $\mathcal{S}_\mathcal{C} = \bigcup_{i=1}^n \mathcal{C}_i$:

$$\forall \lambda \in \sigma(A), \quad \lambda \in \mathcal{S}_\mathcal{R} \bigcap \mathcal{S}_\mathcal{C}.$$

Moreover, should m row circles (or column circles), with $1 \le m \le n$, be disconnected from the union of the remaining $n - m$ circles, then their union contains exactly m eigenvalues.

There is no guarantee that a circle should contain eigenvalues, unless it is isolated from the others. The information provided by Gershgorin circles are in general quite coarse, thus the previous result can provide only a preliminary guess of the shift.

Remark. Note that all the eigenvalues of a strictly diagonally dominant matrix are non-null.

The QR method

We will see some iterative techniques for simultaneously approximating all the eigenvalues of a given matrix A. The basic idea consists of reducing A, by means of suitable similarity transformations, into a form for which the calculation of the eigenvalues is easier than on the starting matrix.

If A and B are similar $P^{-1}AP = B$ then $\lambda_A = \lambda_B$

$$BP^{-1}\mathbf{x} = P^{-1}A\mathbf{x} = \lambda P^{-1}\mathbf{x}$$

A method to compute all the eigenvalues of A is transforming it in a similar diagonal / triangular matrix.

The QR method uses repeatedly the QR factorization to compute λ.

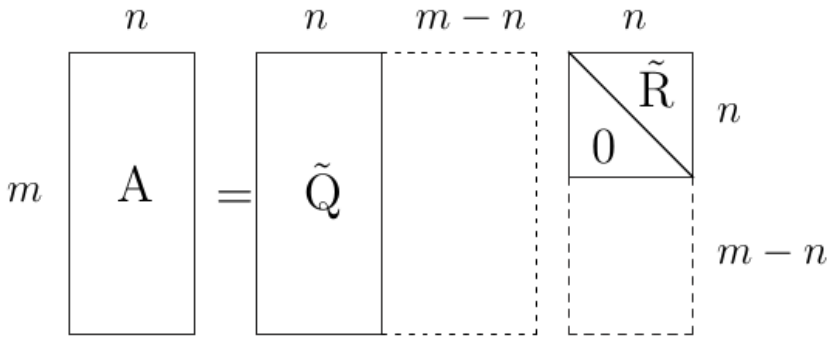


Fig. 3.1. The reduced factorization. The matrices of the QR factorization are drawn in dashed lines

$$\implies Q^{(k+1)}R^{(k+1)} = A^{(k)} \implies A^{(k+1)} = R^{(k+1)}Q^{(k+1)}$$

$A^{(k)}$ and $A^{(k+1)}$ are similar and the rate of decay to zero of lower triangular coefficients in $A^{(k)}$ depends on $\max_i |\lambda_{i+1}/\lambda_i|,\ \forall i$. If A is symmetric, $A^{(k)}$ for $k \to \infty$ is diagonal.

Ordinary differential equations (ODE)

A **differential equation** involves one or more derivatives of an unknown function. If those derivatives are taken w.r.t. a single variable, it is called **ordinary differential equation**, whereas it is a **partial differential equation** if partial derivatives are present. The ODE or PDE has order p, where p is the maximum order of differentiation.

Any equation of order $p > 1$ can always be reduced to a system of p equations of order 1.

An ODE has infinite solutions. We formulate a *Cauchy problem* by adding a **boundary condition** on initial data to the ODE, ensuring the uniqueness of the solution.

Consider a continuous function $f : \mathbb{R}_+ \times \mathbb{R} \to \mathbb{R}$. For given $y_0 \in \mathbb{R}$, we search $y : t \in I \subset \mathbb{R}_+ \to y(t) \in R$ that satisfies the following problem, called the **Cauchy problem**:

$$\begin{cases} y'(t) = f(t, y(t)) & \forall t \in I \\ y(t_0) = y_0 \end{cases} \tag{1}$$

where $y'(t) = \frac{dy(t)}{dt}$.

Theorem 1 (Cauchy-Lipschitz)**.** *If a function $f(t, y)$ is*

1. ***continuous** with respect to both its arguments t and y;*
2. ***Uniformly Lipschitz-continuous** with respect to its second argument, that is, there exists a positive constant L (named Lipschitz constant) such that*

$$|f(t, y_1) - f(t, y_2)| \leq L|y_1 - y2| \quad \forall y_1, y_2 \in \mathbb{R}, \forall t \in I,$$

*Then the solution $y = y(t)$ of the Cauchy problem (1) **exists**, is **unique** and belongs to $C^1(I)$.*

The Lipschitz continuity gives more regularity than normal continuity because incremental quotients are bounded (a.k.a. f cannot peak anywhere).

Solutions of the Cauchy problem are seldom explicit and often cannot be represented even in an implicit form. Numerical methods allows for the approximation of every ODE family for which solutions exists.

The common approach is to divide $I = [t_0, T]$ into N_h intervals of length $h = (T - t_0)/N_h$, where h is called the **time step** or **discretization step**. Each $t_n = t_0 + nh$ is a **node** on which we compute $u_n \approx y_n = y(t_n)$, and $\{u_0 = y_0, u_1, \dots, u_{N_h}\}$ is the **numerical solution** of the Cauchy problem.

Numerical differentiation

We aim to approximate, given a function $y : [a, b] \to \mathbb{R}$ continuously differentiable on $[a, b]$, its derivative at a generic $t_n \in [a, b]$.

Let $y : [a, b] \to \mathbb{R}$ be $C^1([a, b])$ and $t_n \in [a, b]$. The derivative $y'(t_n)$ is given by

$$\begin{aligned} y'(t_n) &= \lim_{h \to 0^+} \frac{y(t_n + h) - y(t_n)}{h}, \\ &= \lim_{h \to 0^+} \frac{y(t_n) - y(t_n - h)}{h}, \\ &= \lim_{h \to 0} \frac{y(t_n + h) - y(t_n - h)}{2h}. \end{aligned}$$

Let $t_0, t_1, \ldots, t_{N_h}$, be $N_h + 1$ equidistributed nodes at $[t_0, t_{N_h}]$. Let $h = (t_{N_h} - t_0)/N_h$ be the distance between two consecutive nodes. Let $(Dy)_n$ be an approximation of $y'(t_n)$. We say

- **Forward finite difference** if

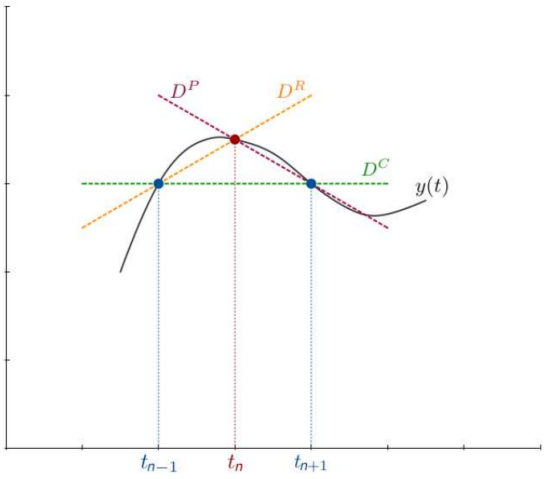
$$(Dy)_n^{FD} = \frac{y(t_{n+1}) - y(t_n)}{h}, \quad n = 0, \ldots, N_h - 1 \tag{2}$$

- **Backward finite difference** if

$$(Dy)_n^{BD} = \frac{y(t_n) - y(t_{n-1})}{h}, \quad n = 1, \ldots, N_h \tag{3}$$

- **Centered finite difference** if

$$(Dy)_n^{CD} = \frac{y(t_{n+1}) - y(t_{n-1})}{2h}, n = 1, \ldots, N_h - 1 \tag{4}$$



The error in the finite difference

If $y \in C^2(\mathbb{R})$ for all $t \in \mathbb{R}$, then there exists ξ_n between t_n and t such that (using the Taylor expansion)

$$y(t) = y(t_n) + y'(t_n)(t - t_n) + \frac{y''(\xi_n)}{2}(t - t_n)^2. \tag{5}$$

- For $t = t_{n+1}$ in (5), we obtain

$$y(t_{n+1}) - y(t_n) = y'(t_n)h + \frac{y''(\xi_n)}{2}h^2,$$

so the *forward finite difference* is given by

$$(Dy)_n^P = \frac{y(t_{n+1}) - y(t_n)}{h} = y'(t_n) + \frac{h}{2}y''(\xi_n).$$

In particular,

$$|y'(t_n) - (Dy)_n^P| \leq Ch, \quad \text{where } C = \frac{1}{2}\max_{t \in [t_n, t_{n+1}]} |y''(t)|.$$

- For $t = t_{n-1}$ in (5), we obtain

$$y(t_{n-1}) - y(t_n) = y'(t_n)(-h) + \frac{y''(\xi_n)}{2}(-h)^2,$$

so the *backward finite difference* is given by

$$(Dy)_n^R = \frac{y(t_n) - y(t_{n-1})}{h} = y'(t_n) - \frac{h}{2}y''(\xi_n).$$

In particular,

$$|y'(t_n) - (Dy)_n^R| \leq Ch, \quad \text{where } C = \frac{1}{2}\max_{t \in [t_{n-1}, t_n]} |y''(t)|.$$

- For $t = t_{n+1}$ and $t = t_{n-1}$ with expansion of order 2 (if $y \in C^3(\mathbb{R})$)

$$y(t_{n+1}) = y(t_n) + y'(t_n)h + \frac{y''(t_n)}{2}h^2 + \frac{y'''(\xi_{n_1})}{6}h^3,$$
$$y(t_{n-1}) = y(t_n) - y'(t_n)h + \frac{y''(t_n)}{2}h^2 - \frac{y'''(\xi_{n_2})}{6}h^3,$$

  and we obtain

$$y(t_{n+1}) - y(t_{n-1}) = 2y'(t_n)h + \frac{y'''(\xi_{n_1}) + y'''(\xi_{n_2})}{6}h^3,$$

  so the *centered finite difference* is given by

$$(Dy)_n^C = \frac{y(t_{n+1}) - y(t_{n-1})}{2h} = y'(t_n) + \frac{y'''(\xi_{n_1}) + y'''(\xi_{n_2})}{12}h^2.$$

  It has the following estimation

$$|y'(t_n) - (Dy)_n^C| \leq Ch^2, \quad \text{where } C = \frac{1}{6}\max_{t\in[t_{n-1},t_{n+1}]}|y'''(t)|.$$

Definition 1. *The difference $\tau_n(h) = |y'(t_n) - (Dy)_n^P|$ is called **truncation error in the point** t_n. We say that τ_n is of order $p > 0$ if*

$$\tau_n(h) \leq Ch^p,$$

for a positive constant C, where h is the time increment between two different timing t_n and t_{n+1}.

Thanks to the found estimation, the truncation error of the forward and the backward finite difference is of order 1; the truncation error of centered finite difference is of order 2.

The finite difference method

Let $0 = t_0 < t_1 < \ldots < t_n < t_{n+1} < \ldots$ be an equidistributed sequence of real numbers and $h = t_{n+1} - t_n$ be the time step. We denote by

$$u_n \quad \textbf{an approximation of} \quad y(t_n).$$

In the Cauchy problem (1), for $t = t_n$, we have

$$y'(t_n) = f(t_n, y(t_n)).$$

We want to approximate the derivative $y'(t_n)$ in the point t_n. We can use a **finite difference differentiation**.

Forward Euler

$$\begin{cases} \frac{u_{n+1}-u_n}{h} = f(t_n, u_n) & \text{for } n = 0, 1, 2, \ldots, N_h - 1 \\ u_0 = y_0 \end{cases} \tag{6}$$

so we have $u_{n+1} = u_n + hf(t_n, u_n)$.

Backward Euler

$$\begin{cases} \frac{u_n-u_{n-1}}{h} = f(t_n, u_n) & \text{for } n = 0, 1, 2, \ldots, N_h - 1 \\ u_0 = y_0 \end{cases} \tag{7}$$

so we have $u_n = u_{n-1} + hf(t_n, u_n) \;\Rightarrow\; u_{n+1} = u_n + hf(t_{n+1}, u_{n+1})$.

Centered scheme

$$\begin{cases} \frac{u_{n+1}-u_{n-1}}{2h} = f(t_n, u_n) & \text{for } n = 1, 2, \ldots, N_h - 1 \\ u_0 = y_0 \\ u_1 \text{ to determine} \end{cases} \tag{8}$$

so we have $u_{n+1} = u_{n-1} + 2hf(t_n, u_n)$.

Remark 1.

- *The forward Euler is **explicit** because u_{n+1} depends on u_n explicitly:*

$$\text{(forward Euler)} \quad u_{n+1} = u_n + hf(t_n, u_n).$$

- *The backward Euler is **implicit** because u_{n+1} is implicitly defined in terms of u_n:*

$$\text{(backward Euler)} \quad u_{n+1} = u_n + hf(t_{n+1}, u_{n+1}).$$

In general, for the forward Euler we have a simple computation, while for the backward Euler we have to solve a nonlinear equation at each time step. However, backward Euler is generally more stable. Since the centered Euler requires $\mathbf{u}_1$ to be applied, is generally preceded by a single pass of forward Euler to backward Euler.

Fixed point iterations: Note that backward Euler is equivalent to a fixed point problem with

$$u_{n+1} = \phi(u_{n+1}) = u_n + hf(t_{n+1}, u_{n+1})$$

We can solve this problem thanks to the following iterations

$$u_{n+1}^{(k+1)} = \phi(u_{n+1}^{(k)}), \quad k = 0, 1, 2, \dots \tag{9}$$

The Newton method: Starting from the equation:

$$F(u_{n+1}) \equiv u_{n+1} - \phi(u_{n+1}) = 0,$$

we use the following iterations:

$$u_{n+1}^{(k+1)} = u_{n+1}^{(k)} - \frac{F(u_{n+1}^{(k)})}{F'(u_{n+1}^{(k)})} = u_{n+1}^{(k)} - \frac{F(u_{n+1}^{(k)})}{1 - \phi'(u_{n+1}^{(k)})}, \quad k = 0, 1, 2, \dots \tag{10}$$

In both cases, we have $\lim_{k\to\infty} u_{n+1}^{(k)} = u_{n+1}$.

Stability conditions

The choice of time step h is not arbitrary. For forward Euler, we will see later that if h is not small enough then stability problems may arise.

For example, if we consider the problem

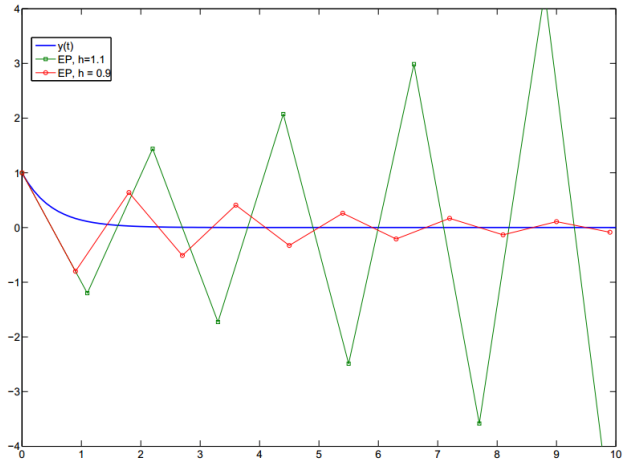
$$\begin{cases} y'(t) = -2y(t) & \text{for } t \in \mathbb{R}_+ \\ y(0) = 1, \end{cases} \tag{11}$$

then the solution is

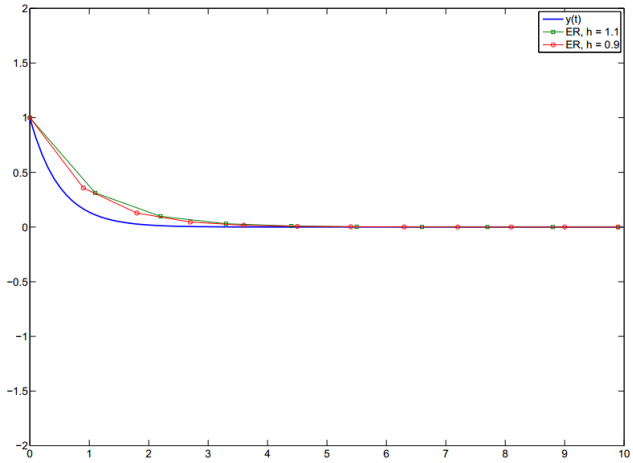
$$y(t) = e^{-2t},$$

We can observe that behavior with respect to h of forward and backward Euler methods are very different.

Forward Euler



Backward Euler



The (absolute) stability properties

For a given $\lambda < 0$, $\lambda \in \mathbb{R}$, we consider the model problem:

$$\begin{cases} y'(t) = \lambda y(t) & \text{for } t \in \mathbb{R}_+ \\ y(0) = 1 \end{cases} \tag{12}$$

The exact solution is

$$y(t) = e^{\lambda t}. \quad \text{In particular, } \lim_{t \to \infty} y(t) = 0.$$

We require $\lambda < 0$ in order to have a well-posed problem, since if $\lambda > 0$ then $\lim_{n\to\infty} u_n = 0$ but the real solution $y(t) = e^{\lambda t} \to \infty$ for $t \to \infty$, so the solution explodes and we are no longer interested in it.

Let $0 = t_0 < t_1 < \ldots < t_n < t_{n+1} < \ldots$ such that $t_n = nh$ and where the **time step** $h > 0$ is fixed.

We say that a numerical scheme associated to the model problem is **absolutely stable** if $\lim_{n\to\infty} u_n = 0$.

- For the **forward Euler**:

  $$u_{n+1} = (1 + \lambda h)u_n, \quad \text{where} \quad u_n = (1 + \lambda h)^n, \quad \forall n \geq 0. \tag{13}$$

  If $1 + \lambda h < -1$, then $|u_n| \to \infty$ when $n \to \infty$, therefore forward Euler is **unstable**.
  To ensure stability, we need to limit the time step h, by imposing the **stability condition**:

  $$\boxed{|1 + \lambda h| < 1 \quad \text{hence} \quad h < 2/|\lambda|.}$$

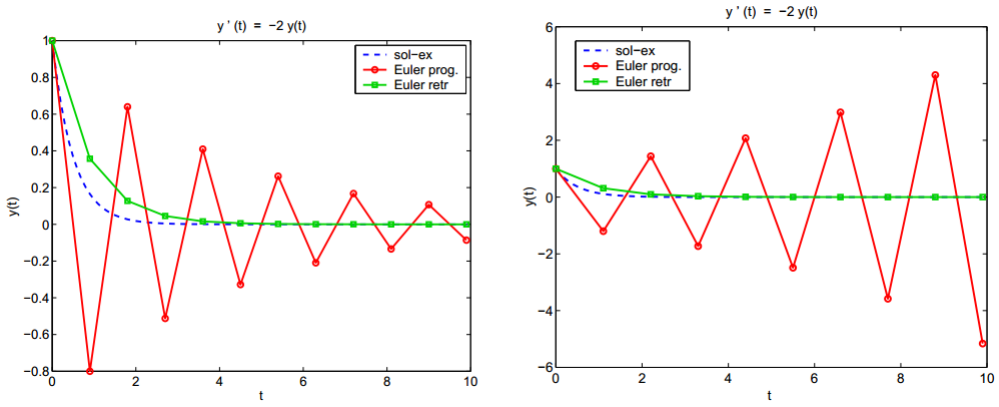
This condition is required on unbounded intervals since N_h (the number of t_n) may tend to infinity even if $h \nrightarrow 0$, in order to ensure stability.

- For the **backward Euler**:

$$u_{n+1} = \left(\frac{1}{1-\lambda h}\right) u_n \quad \text{and therefore} \quad u_n = \left(\frac{1}{1-\lambda h}\right)^n, \quad \forall n \geq 0.$$

Because $\lim_{n\to\infty} u_n = 0$, it is **unconditionally stable** (it is stable for any $h > 0$).

Example 4. Let solve the problem (12) for $\lambda = -2$ and $y_0 = 1$ at interval $[0, 10]$ using forward and backward Euler methods with $h = 0.9$ and $h = 1.1$. The following figure shows obtained solutions for $h = 0.9$ (on the left) and $h = 1.1$ (on the right) and the exact solution.



Comparison of solutions that we obtain by the forward and backward Euler methods for $h = 0.9$ (on the left, stable) and $h = 1.1$ (on the right, unstable) (stability condition for forward Euler: $|\lambda| = 2 \Rightarrow h < 2/|\lambda| = 1$).

■

Absolute stability controls perturbations

For a generic problem, it raises the question of **stability**, i.e. the property that **small perturbations on the data induce small perturbations on the approximation**.

We want to show the following property.

A numerical method which is absolutely stable on the model problem, guarantees that the perturbations are kept under control as t tends to infinity (is stable in the above sense).

Consider now the following generalized model problem on an unbounded interval:

$$\begin{cases} y'(t) = \lambda(t)y(t) + r(t), & t \in (0, +\infty), \\ y(0) = 1, \end{cases} \tag{14}$$

where λ and r are two continuous functions and $-\lambda_{\max} \leq \lambda(t) \leq -\lambda_{\min}$ with $0 < \lambda_{\min} \leq \lambda_{\max} < +\infty$. In this case the exact solution does not necessarily tend to zero as t tends to infinity.

For instance if both r and λ are constants we have

$$y(t) = \left(1 + \frac{\lambda}{r}\right) e^{\lambda t} - \frac{\lambda}{r}$$

whose limit when t tends to infinity is $-r/\lambda$. Thus, in general, it does not make sense to require a numerical method to be absolutely stable, i.e. to satisfy (14). However, it is possible to prove that a method which is absolutely stable on the original model problem keeps perturbations under control even when applied to the generalized model problem as $t \to \infty$.

For the sake of simplicity we will confine our analysis to the forward Euler method (14).

We have

$$\begin{cases} u_{n+1} = u_n + h(\lambda_n u_n + r_n), \quad n \geq 0, \\ u_0 = 1 \end{cases}$$

where $\lambda_n = \lambda(t_n)$ and $r_n = r(t_n)$.

Let us consider the following "perturbed" method:

$$\begin{cases} z_{n+1} = z_n + h(\lambda_n z_n + r_n + \rho_{n+1}), \quad n \geq 0, \\ z_0 = u_0 + \rho_0, \end{cases} \tag{15}$$

where $\rho_0, \rho_1, \ldots$ are given perturbations which are introduced at every time step.

This is a simple model in which ρ_0 and ρ_{n+1} represent truncation errors or numerical errors.

Question: Is the difference $z_n - u_n$ bounded for all $n = 0, 1, \ldots$ independently of n and h?

We will consider two cases:

1. Let $\lambda_n = \lambda$ and $\rho_n = \rho$ be two constants. We can write the schema for the **perturbation error** $e_n = z_n - u_n$ at step n:

   $$\begin{cases} e_{n+1} = e_n + h(\lambda e_n + \rho), \quad n \geq 0, \\ e_0 = \rho. \end{cases} \tag{16}$$

   that the solution is

   $$e_n = \rho(1 + h\lambda)^n + h\rho \sum_{k=0}^{n-1} (1 + h\lambda)^k = \rho\psi(h, \lambda), \tag{17}$$

   where

   $$\psi(h, \lambda) = \left((1 + h\lambda)^n \left(1 + \frac{1}{\lambda} \right) - \frac{1}{\lambda} \right)$$

   We use equation for the geometric sum

   $$\sum_{k=0}^{n-1} a^k = \frac{1 - a^n}{1 - a}. \tag{18}$$

   Suppose that $h < h_0(\lambda) = 2/|\lambda|$, i.e. h ensure the absolute stability of the forward Euler method applied to the problem (12).
   Therefore $(1 + h\lambda)^n < 1 \ \forall n$ and it follows that the error due to perturbations is bounded by

   $$|e_n| \leq \varphi(\lambda)|\rho|, \tag{19}$$

   where $\varphi(\lambda) = 1 + |2/\lambda|$. Moreover,

   $$\lim_{n \to \infty} |e_n| = \frac{|\rho|}{|\lambda|}.$$

   So, the error caused by perturbations is bounded by $|\rho|$ times a constant, that is independent of n and h. Obviously, if $h > h_0$, the perturbations amplifies when n increases because $(1 + h\lambda)^n \to \infty$ when $n \to \infty$.

2. In the general case where λ and r depends on t, we have

   $$z_n - u_n = \rho_0 \prod_{k=0}^{n-1} (1 + h\lambda_k) + h \sum_{k=0}^{n-1} \rho_{k+1} \prod_{j=k+1}^{n-1} (1 + h\lambda_j) \tag{20}$$

   We require the time step h to satisfy the restriction $h < h_0(\lambda)$, where $h_0(\lambda) = 2/\lambda_{\max}$. Then, $|1 + h\lambda_k| \leq \max(|1 - h\lambda_{\min}|, |1 - h\lambda| max|) < 1$. Let $\rho = \max |\rho_n|$ and λ such that $(1 + h\lambda) = \max(|1 - h\lambda_{\min}|, |1 - h\lambda_{\max}|)$.

   It holds:

$$|z_n - u_n| \leq |\rho_0| \prod_{k=0}^{n-1} |1 + h\lambda_k| + h \sum_{k=0}^{n-1} |\rho_{k+1}| \prod_{j=k+1}^{n-1} |1 + h\lambda_j|$$
$$\leq \rho \prod_{k=0}^{n-1} (1 + h\lambda) + h \sum_{k=0}^{n-1} \rho \prod_{j=k+1}^{n-1} (1 + h\lambda) = \rho\psi(h, \lambda)$$

So, even in this case, $e_n = z_n - u_n$ satisfies (19).

Remark 2. *Consider now the following generalized model problem*

$$\begin{cases} y'(t) = f(t, y(t)) & t > 0 \\ y(0) = y_0, \end{cases}$$

at unbounded interval. We can extend the control of perturbations to generalized model problem (14), in cases where exists $\lambda_{\min} > 0$ and $\lambda_{\max} < \infty$ such that

$$-\lambda_{\max} < \frac{\partial f}{\partial y}(t, y) < -\lambda_{\min}, \quad \forall t \geq 0, \forall y \in D_y, \tag{21}$$

where D_y is a set that contains the trajectory of $y(t)$ (possible values of u_n). This allows to get (20) and to obtain the same conclusions as in case (2) if $0 < h < 2/\lambda_{\max}$.
In this case, the steplength h should be chosen as function of $\frac{\partial f}{\partial y}$, depending on the case:

- *if h is constant:*

$$0 < h < 2/ \max_{t \in [t_0, T]} \left| \frac{\partial f}{\partial y}(t. y(t)) \right|$$

- *if h depends on the step:*

$$0 < h_n < 2 \frac{\alpha}{|f_y(t_n, u_n)|} \qquad \text{for } \alpha < 1$$

Convergence of the forward Euler

Definition 2. *Let $y(t)$ be the solution of the Cauchy problem (1) on the interval $[0, T]$; let u_n be an approximated solution at time $t_n = nh$, where $h = T/N_h$ ($N_h \in \mathbb{N}$) is the time step, found by a given numerical method.*
*The method is **convergent** if*

$$\forall n = 0, \ldots, N_h : \quad |u_n - y(t_n)| \leq C(h)$$

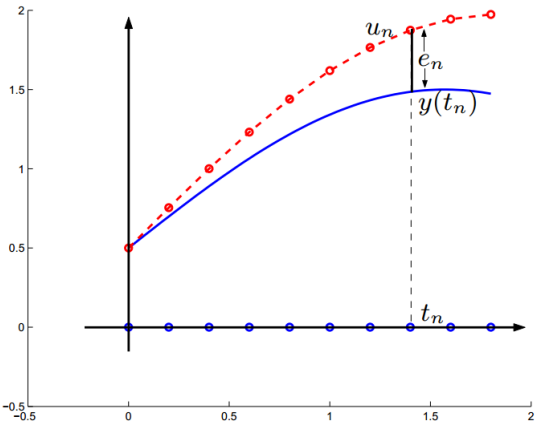
where $C(h) \to 0$ when $h \to 0$.

Moreover, if there exists $p > 0$ such that $C(h) = O(h^p)$ (so $\exists c > 0$ s.t. $C(h) \leq ch^p$ for $\max p$), we say that the method **converges with order p**.

In the following, we will analyze the convergence and the order of the forward Euler method.

Definition. *We define the error at time n as:*

$$e_n = u_n - y(t_n)$$



Definition. *The **local truncation error** (LTE) of a method represents the error that would be generated by forcing the exact solution to satisfy that specific numerical scheme. For the forward Euler method we define it as*

$$\tau_{n+1}(h) = \frac{y(t_{n+1}) - y(t_n)}{h} - y'(t_n). \tag{22}$$

*The **global truncation error** is*

$$\tau(h) = \max_n |\tau_n(h)|.$$

For the forward Euler method we define it as

$$\tau(h) = \frac{1}{2} \max_{t\in[t_0,T]} |y''(t)|h.$$

We will prove the following convergence result:

Theorem 2. *If $y \in \mathcal{C}^2([0,T])$ and f is uniform Lipschitz continuous on the second variable, L is the Lipschitz constant, then*

$$\forall n \geq 0, \quad |y(t_n) - u_n| \leq c(t_n)h, \tag{23}$$

where

$$c(t_n) = \frac{e^{Lt_n} - 1}{2L} \max_{t\in[0,T]} |y''(t)|.$$

In particular, the method converges with order $p = 1$ according to the previous definition, with

$$C(h) = c(T)h.$$

Proof. We know there exists $\xi_n \in (t_n, t_n + h)$ such that

$$\tau_{n+1}(h) = \frac{1}{2}y''(\xi_n)h.$$

So we have the following estimation for the global truncation error:

$$\tau(h) \leq \frac{1}{2} \max_{t\in[0,T]} |y''(t)|h.$$

The following equation for the numerical solution u_n

$$\begin{cases} \frac{u_{n+1}-u_n}{h} = f(t_n, u_n) & \text{for } n = 0, 1, 2, \ldots, N_h \\ u_0 = y_0. \end{cases}$$

and the equation (22) for a local truncation error

$$\tau_{n+1}(h) = \frac{y(t_{n+1}) - y(t_n)}{h} - y'(t_n) = \frac{y(t_{n+1}) - y(t_n)}{h} - f(t_n, y(t_n)),$$

we obtain:

$$\begin{cases} \frac{e_{n+1}-e_n}{h} = f(t_n, u_n) - f(t_n, y(t_n)) - \tau_{n+1}(h), \\ e_0 = 0 \end{cases} \tag{24}$$

Since the function f is Lipschitz, we have

$$|f(t_n, u_n) - f(t_n, y(t_n))| \leq L|u_n - y(t_n)| \leq L|e_n|.$$

Given this inequality, (24) can be written as:

$$|e_{n+1}| \leq (1 + Lh)|e_n| + h|\tau_{n+1}(h)|.$$

Let $E_j = |e_j|$. Then we have the following inequality:

$$\begin{aligned} E_{n+1} &\leq (1 + hL)E_n + h\tau(h) \\ &\leq (1 + hL)[(1 + hL)E_{n-1} + h\tau(h)] + h\tau(h) \\ &\leq \left[1 + (1 + hL) + (1 + hL)^2 + \ldots + (1 + hL)^n\right] h\tau(h) \\ &= \frac{(1 + hL)^{n+1} - 1}{hL} h\tau(h) \end{aligned}$$

But

$$1 + hL \leq e^{hL}, \quad \forall h > 0,$$

hence

$$(1 + hL)^n - 1 \leq e^{Lhn} - 1 = e^{Lt_n} - 1.$$

Therefore

$$E_n \leq \frac{e^{Ltn} - 1}{L}\tau(h) \leq \frac{e^{Lt_n} - 1}{L} h \cdot \frac{1}{2} \max_{t\in[0,T]} |y''(t)|,$$

i.e.

$$|u_n - y(t_n)| \leq \left[\frac{1}{2}\frac{e^{Lt_n} - 1}{L} \max_{t\in[0,T]} |y''(t)|\right] h \leq c(T)h,$$

where $c(T) = \frac{e^{LT}-1}{2L} \max_{t\in[0,T]} |y''(t)|$.

□

Remark 3. *The same type of results can be obtained for the backward Euler method.*

Remark 4. *The estimation of convergence (23) is obtained under the assumption that the function f is Lipschitz continuous. More precisely, the estimation*

$$|u_n - y(t_n)| \leq ht_n \frac{1}{2} \max_t |y''(t)|, \tag{25}$$

is true if f also satisfies the condition $\frac{\partial f}{\partial y}(t, y) \leq 0$ for all $t \in [0, T]$ and for all $y \in (-\infty, \infty)$.

We prove (25). Using the Lagrange theorem, there exists ξ_n such that

$$f(t_n, u_n) - f(t_n, y(t_n)) = \frac{\partial f(t, \xi_n)}{\partial y}(u_n - y(t_n)) = \frac{\partial f(t, \xi)}{\partial y} e_n.$$

So, using (24) we find

$$e_{n+1} = \left(1 + h\frac{\partial f}{\partial y}(t, \xi_n)\right) e_n - h\frac{h}{2}y''(\eta_n).$$

If $h < \frac{2}{\lambda_{\max}}$ then we have $1 + h\frac{\partial f}{\partial y}(t, \xi_n) \in (-1, 1)$ and therefore

$$|e_{n+1}| \leq |e_n| + h\tau(h).$$

Since $e_0 = 0$, we deduce

$$|e_n| \leq nh\tau(h) = t_n\tau(h),$$

which was to be demonstrated (25).

□

Consistency

Consistency is necessary in order to achieve convergence, since it fulfills the basic assumption that e_n is infinitesimal w.r.t. h. If violated, it would inhibit the global error tending to zero when $h \to 0$.

The error follows $O\left(\frac{1}{h}\right)$ when h approaches 0, so it can blow up due to roundoff errors if h is too small.

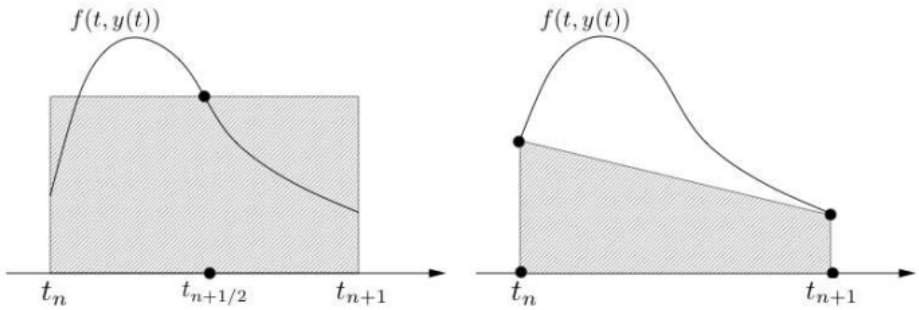
Runge-Kutta methods of order 2

The family of **Runge-Kutta methods** uses a single step h but evaluates $f(t, y)$ several times per interval $[t_n, t_{n+1}]$. The number of evaluations at each step is called the **order** w.r.t. h.

If we integrate the equation $y'(t) = f(t, y(t))$ between t_n and t_{n+1}, we obtain:

$$y(t_{n+1}) - y(t_n) = \int_{t_n}^{t_{n+1}} f(t, y(t))dt. \qquad (26)$$

Remark 5 (Numerical integration methods). *We want to approximate the integral of the function $f(t, y(t))$. If we use the midpoint formula, we approximate the area below the curve by the area of a rectangle that has as a basis h and as a height the value of the function at time $t_n + h/2$ (see figure on the left). If we use the trapezoidal formula, we approximate the area below the curve by the area of a trapezoid that has as basis both values of the function at times t_n and t_{n+1} and as a height h (see figure on the right).*



Using trapezoidal formula, we find the following implicit method, that is called **Crank-Nicolson** or **trapezoidal method**:

$$\boxed{u_{n+1} - u_n = \frac{h}{2}[f(t_n, u_n) + f(t_{n+1}, u_{n+1})]}, \quad \forall n \geq 0. \qquad (27)$$

This method is unconditionally stable when it is applied to the model problem (12), but we have an implicit part to handle.

If we modify the schema (27) (changing it to explicit) then we obtain the **Heun method**:

$$\boxed{u_{n+1} - u_n = \frac{h}{2}[f(t_n, u_n) + f(t_{n+1}, u_n + hf(t_n, u_n))]}. \tag{28}$$

Both methods (Crank-Nicolson and Heun) are of order 2 with respect to h.

If we use in (26) midpoint formula, we obtain

$$u_{n+1} - u_n = hf(t_{n+\frac{1}{2}}, u_{n+\frac{1}{2}}). \tag{29}$$

If we approximate $u_{n+\frac{1}{2}}$ by

$$u_{n+\frac{1}{2}} = u_n + \frac{h}{2} f(t_n, u_n),$$

we obtain the **improved Euler method**:

$$\boxed{u_{n+1} - u_n = hf\left(t_{n+\frac{1}{2}}, u_n + \frac{h}{2} f(t_n, u_n)\right)}. \tag{30}$$

Both Heun and improved Euler methods are particular cases of the Runge-Kutta method of order 2. When we apply them to the model problem (12), we have the same stability condition $h < 2/|\lambda|$ for both methods.

In the following table we summarize the characteristics of the methods:

Method	*Explicit/Implicit*	*Stability*	*w.r.to* h
Forward Euler	Explicit	Conditionally	1
Backward Euler	Implicit	Unconditionally	1
Crank–Nicolson	Implicit	Unconditionally	2
Heun	Explicit	Conditionally	2
Improved Euler	Explicit	Conditionally	2
Runge–Kutta	Explicit	Conditionally	4

There are more complicated methods, such as **Runge-Kutta method of order** 4, that is obtained by considering the integration of the Simpson method:

$$u_n \to \begin{cases} u_{n+1} = u_n + \frac{h}{6}(K_1 + 2K_2 + 2K_3 + K_4), \\ \text{where:} \\ K_1 = f(t_n, u_n), \\ K_2 = f(t_n + \frac{h}{2}, u_n + \frac{h}{2}K_1), \\ K_3 = f(t_n + \frac{h}{2}, u_n + \frac{h}{2}K_2), \\ K_4 = f(t_{n+1}, u_n + hK_3). \end{cases}$$

Systems of differential equations

Let us consider the following system of non-homogeneous ordinary differential equation with constant coefficients.

$$\begin{cases} \mathbf{y}'(t) = A\mathbf{y}(t) + \mathbf{b}(t) \quad t > 0, \\ \mathbf{y}(0) = \mathbf{y}_0, \end{cases} \tag{31}$$

where $A \in \mathbb{R}^{p \times p}$ and $\mathbf{b}(t) \in \mathbb{R}^p$, where we assume that A has got p distinct eigenvalues $\lambda_j, j = 1, \ldots, p$.

From the numerical point of view, the methods introduced in the scalar case can be extended to systems of differential equations. For example, the forward Euler method (6) becomes:

$$\begin{cases} \frac{\mathbf{u}_{n+1}-\mathbf{u}_n}{h} = A\mathbf{u}_n + \mathbf{b}_n \quad \text{for } n = 0, 1, 2, \dots, N_h - 1 \\ \mathbf{u}_0 = \mathbf{y}_0, \end{cases} \tag{32}$$

while the backward Euler method (7) becomes:

$$\begin{cases} \frac{\mathbf{u}_{n+1}-\mathbf{u}_n}{h} = A\mathbf{u}_{n+1} + \mathbf{b}_{n+1} \quad \text{for } n = 0, 1, 2, \dots, N_h - 1 \\ \mathbf{u}_0 = \mathbf{y}_0, \end{cases} \tag{33}$$

Regarding stability, if $\mathbf{b} \equiv 0$ and the eigenvalues λ_j ($j = 1, \dots, p$) of the matrix A are strictly negative: $\lambda_j < 0$, $j = 1, \dots, p$, then $\mathbf{y}(t) \to \mathbf{0}$ when $t \to +\infty$, and the forward Euler method is stable (i.e. $\mathbf{u}_n \to \mathbf{0}$ if $n \to +\infty$) if

$$h < \frac{2}{\max_{j=1,\dots,p} |\lambda_j|} = \frac{2}{\rho(A)}, \tag{34}$$

where $\rho(A)$ is the spectral radius of A, while the backward Euler method is unconditionally stable.

We could also consider the case of a nonlinear system of the form

$$\mathbf{y}'(t) = \mathbf{F}(t, \mathbf{y}(t)).$$

If $\frac{\partial \mathbf{F}}{\partial \mathbf{y}}$ is a matrix with real and negative eigenvalues ($\lambda_i < 0, \ \forall i$), then the backward Euler method is unconditionally stable, while the forward Euler method is stable under condition (34), where $A = \frac{\partial \mathbf{F}}{\partial \mathbf{y}}$.

Here is a summary of the stability:

Problem		Stability of the explicit methods
Model	$y' = \lambda y$	$h < 2/\lvert\lambda\rvert$
Cauchy	$y' = f(t, y(t))$	$h < 2/\max \left\lvert \frac{\partial f}{\partial y} \right\rvert$
System of linear eq.	$\mathbf{y}' = \mathbf{A}\mathbf{y} + \mathbf{b}$	$h < 2/\rho(\mathbf{A})$
System of nonlinear eq.	$\mathbf{y}' = \mathbf{F}(t, \mathbf{y}(t))$	$h < 2/\rho(\mathbf{J})$

for

- $\rho(\mathbf{A}) = \max_i |\lambda_i(\mathbf{A})|$, for a system of linear equations;
- $\rho(\mathbf{J}) = \max_i |\lambda_i(\mathbf{J})|$, for a system of nonlinear equations, for $\lambda_i(\mathbf{J}) < 0, \forall i$, where

  $$\mathbf{J}(t, \mathbf{y}) = \frac{\partial \mathbf{F}}{\partial \mathbf{y}} = \begin{bmatrix} \frac{\partial f_1}{\partial y_1} & \dots & \frac{\partial f_1}{\partial y_n} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial y_1} & \dots & \frac{\partial f_n}{\partial y_n} \end{bmatrix}$$

  is the Jacobian.

Other notions of ODEs

- The **Lotka-Volterra equations** are used to model predator-prey systems in population dynamics. Their form is:

  $$\begin{cases} y'_1(t) = C_1 y_1(t)[1 - b_1 y_1(t) - d_2 y_2(t)], \\ y'_2(t) = -C_2 y_2(t)[1 - b_2 y_2(t) - d_1 y_1(t)], \end{cases}$$

  where C_1 and C_2 represent the growth rates of the two populations. The coefficients d_1 and d_2 govern the type of interaction between the two populations, while b_1 and b_2 are related to the available quantity of nutrients.

- The **zero-stability** is the stability inside a bounded interval. For one-step methods, this derives from uniform Lipschitz continuity. The *Lax-Richtmyer equivalence theorem* says that any consistent method is convergent if and only if it is zero-stable.
- The **region of absolute stability** $\mathcal{A}$ is the set of $z \in \mathbb{C}, z = h\lambda$ for which a method is absolutely stable:

  $$\mathcal{A} = \{z = h\lambda \in \mathbb{C} : |u_n| \to 0 \text{ as } t_n \to +\infty\}$$

  Thus, $\mathcal{A}$ is the set of the values of the product $h\lambda$ for which the numerical method furnishes solutions that decay to zero as t_n tends to infinity. Methods that are unconditionally absolutely stable (for which $\mathcal{A} \cap \mathbb{C}^- = \mathbb{C}^-$) are called **$A$-stable**. The backward Euler and Crank-Nicolson methods are A-stable, while the forward Euler and Heun methods are conditionally stable.
- The **step adaptivity** allows to vary time-step h at each time level to match stability constraints and achieve the desired accuracy.
- The **multistep methods** achieve higher order of accuracy in general.
- The Heun method belongs to the **predictor-corrector methods** family, since it requires an explicit step (**predictor**) and an implicit one (**corrector**), which gives the order of accuracy. Being explicit, they are not adequate on unbounded intervals.

Finite elements and boundary-value problems

This chapter is devoted to the analysis of approximation methods for two-point boundary value problems for differential equations of elliptic type.

A **boundary-value problem** is a differential equation together with a set of additional constraints, called the **boundary conditions**. A solution to a boundary value problem is a solution to the differential equation which also satisfies the boundary conditions.

A model problem

To start with, let us consider the two-point boundary value problem, called **Poisson's equation in one dimension**

$$\begin{cases} -u''(x) = f(x), & 0 < x < 1, \\ u(0) = u(1) = 0. \end{cases} \tag{1}$$

From the fundamental theorem of calculus, if $u \in C^2([0,1])$ and satisfies the differential equation in (1) then

$$u(x) = c_1 + c2x - \int_0^x F(s)ds,$$

where c_1 and c_2 are arbitrary constants and $F(s) = \int_0^s f(t)dt$. Using integration by parts one has

$$\int_0^x F(s)ds = \Big[sF(s)\Big]_0^x - \int_0^x sF'(s)ds = \int_0^x (x-s)f(s)ds.$$

The constants c_1 and c_2 can be determined by enforcing the boundary conditions. The condition $u(0) = 0$ implies that $c_1 = 0$, and then $u(1) = 0$ yields $c_2 = \int_0^1 (1-s)f(s)ds$. Consequently, the *solution* of (1) can be written in the following form

$$u(x) = x\int_0^1 (1-s)f(s)ds - \int_0^x (x-s)f(s)ds$$

or, more compactly, the **analytic solution** is

$$u(x) = \int_0^1 G(x,s)f(s)ds, \tag{2}$$

where, for any fixed x, we have defined

$$G(x,s) = \begin{cases} s(1-x) & \text{if } 0 \le s \le x, \\ x(1-s) & \text{if } x \le s \le 1. \end{cases} \tag{3}$$

The function G is called **Green's function** for the boundary value problem (1). It is a piecewise linear function of x for fixed s, and vice versa. It is continuous, symmetric (i.e., $G(x,s) = G(s,x)$ for all $x, s \in [0,1]$), non-negative, null if x or s are equal to 0 or 1, and $\int_0^1 G(x,s)ds = \frac{1}{2}x(1-x)$. The function is plotted in Figure 12.1.

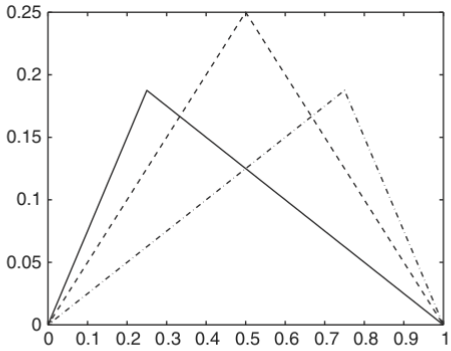


Fig. 12.1. Green's function for three different values of x: $x = 1/4$ (*solid line*), $x = 1/2$ (*dashed line*), $x = 3/4$ (*dash-dotted line*)

We can therefore conclude that for every $f \in C^0([0,1])$ there is a unique solution $u \in C^2([0,1])$ of the boundary value problem (1) which admits the representation (2). Further smoothness of u can be derived by the differential equation in (1); indeed, if $f \in C^m([0,1])$ for some $m \geq 0$ then $u \in C^{m+2}([0,1])$.

An interesting property of the solution u is that if $f \in C^0([0,1])$ is a non-negative function, then u is also nonnegative. This is referred to as the **monotonicity property**, and follows directly from (2), since $G(x,s) \geq 0$ for all $x, s \in [0,1]$. The next property is called the **maximum principle** and states that if $f \in C^0([0,1])$,

$$||u||_\infty \leq \frac{1}{8}||f||_\infty, \tag{4}$$

where $u = \max_{0 \leq x \leq 1} |u(x)|$ is the maximum norm. Indeed, since G is nonnegative,

$$|u(x)| \leq \int_0^1 G(x,s)|f(s)|ds \leq ||f||_\infty \int_0^1 G(x,s)ds = \frac{1}{2}x(1-x)||f||_\infty$$

from which the inequality (4) follows.

Finite difference approximation

By following the same approach we used to approximate $u'(x)$ through finite differences, to approximate $u''(x)$ we apply the Taylor expansion up to the third derivative of $u(x+h)$ and $u(x-h)$ (we have that x is x_0 in the formula, while $x \pm h$ is the x):

$$u(x+h) = u(x) + u'(x)h + u''(x)\frac{h^2}{2} + u'''(x)\frac{h^3}{6} + O(h^{(4)})$$
$$u(x-h) = u(x) - u'(x)h + u''(x)\frac{h^2}{2} - u'''(x)\frac{h^3}{6} + O(h^{(4)})$$

if we sum them we obtain

$$u(x+h) + u(x-h) = 2u(x) + u''(x)h^2 + O(h^{(4)})$$

So we have that

$$u''(x) = \frac{u(x+h) + u(x-h) - 2u(x)}{h^2}$$

This result is valid if $u : [0,1] \to \mathbb{R}$ is sufficiently smooth in a neighbourhood of $x \in [0,1]$: $u \in C^3([0,1])$.

We introduce on $[0,1]$ the grid points $\{x_j\}_{j=0}^n$ given by $x_j = jh$, where $n \geq 2$ is an integer and $h = 1/n$ is the grid spacing. Replacing $u''(x_j)$ by its second order **centered finite differences** the Poisson problem becomes

$$\begin{cases} -\frac{u_{j+1}-2u_j+u_{j-1}}{h^2} = f(x_j), \quad j = 1, \dots, n-1 \\ u_0 = u_n = 0 \end{cases} \tag{5}$$

where u_j is an approximation of $u(x_j) = u(x_0 + jh)$. So the approximation to the solution u is a finite sequence $\{u_j\}_{j=0}^n$ defined only at the grid points.

If we set $\mathbf{u}_h = (u_1, \dots, u_{n-1})^T$ (the unknowns), $\mathbf{f} = (f_1, f_2, \dots, f_{n-1})^T$ with $f_i = f(x_i)$, it is a simple matter to see that (5) can be written in a more compact form as a linear system

$$A_{\text{fd}}\,\mathbf{u}_h = \mathbf{f} \tag{6}$$

where A_{fd} is the symmetric $(n-1) \times (n-1)$ finite difference matrix defined as

$$A_{\text{fd}} = h^{-2}\text{tridiag}_{n-1}(-1, 2, -1) = \frac{1}{h^2}\begin{bmatrix} -2 & 1 & 0 & & \cdots & & 0 \\ 1 & -2 & 1 & 0 & \cdots & & 0 \\ 0 & 1 & -2 & 1 & 0 & \cdots & 0 \\ \vdots & & & & & & \vdots \\ 0 & & \cdots & & 0 & 1 & -2 \end{bmatrix}.$$

This matrix is diagonally dominant by rows; moreover, it is positive definite since for any vector $\mathbf{x} \in \mathbb{R}^{n-1}$

$$\mathbf{x}^T A_{\text{fd}}\mathbf{x} = h^{-2}\left[x_1^2 + x_{n-1}^2 + \sum_{i=2}^{n-1}(x_i - x_{i-1})^2\right] > 0.$$

So, A being symmetric and positive definite implies that (6) admits a unique solution, and can be solved with the Thomas algorithm.

For small h (large N), the matrix A is ill-conditioned, since $K(A) = \frac{\lambda_{\max}}{\lambda_{\min}} = Ch^{-2}$. Thus appropriate methods and precautions should be used.

In order to rewrite (5) in operator form, let V_h be a collection of discrete functions defined at the grid points x_j for $j = 0, \dots, n$. If $v_h \in V_h$, then $v_h(x_j)$ is defined for all j and we sometimes use the shorthand notation v_j instead of $v_h(x_j)$. Next, we let V_h^0 be the subset of V_h containing discrete functions that are zero at the endpoints x_0 and x_n. For a function w_h we define the operator L_h by

$$(L_h w_h)(x_j) = -\frac{w_{j+1} - 2w_j + w_{j-1}}{h^2}, \qquad j = 1, \dots, n-1, \tag{7}$$

and reformulate the finite difference problem (5) equivalently as: *find* $u_h \in V_h^0$ *such that*

$$(L_h u_h)(x_j) = f(x_j) \qquad \text{for } j = 1, \dots, n-1. \tag{8}$$

Notice that, in this formulation, the boundary conditions are taken care of by the requirement that $u_h \in V_h^0$.

Finite differences can be used to provide approximations of higher-order differential operators than the one considered in this section. An example is carrying out the finite difference centered discretization of the fourth-order derivative $-u^{(iv)}(x)$ by applying twice the discrete operator L_h. Again, extra care is needed to properly handle the boundary conditions.

In general, finite differences requires too much regularity ($u \in C^2$ and $f \in C^1$) so more flexible methods are used, like **finite elements**.

For two discrete functions $w_h, v_h \in V_h$ we define the **discrete inner product**

$$(w_h, v_h)_h = h \sum_{k=0}^{n} c_k w_k v_k,$$

with $c_0 = c_n = 1/2$ and $c_k = 1$ for $k = 1, \ldots, n-1$. This is nothing but the composite trapezoidal rule which is here used to evaluate the inner product $(w, v) = \int_0^1 w(x)v(x)dx$. Clearly,

$$||v_h||_h = (v_h, v_h)_h^{1/2}$$

is a norm on V_h.

Integral formulation of boundary value problems

Let's perform a reformulation of the Poisson problem.

We multiply both sides of the so-called **strong formulation** of the Poisson equation $-u''(x) = f(x),\ x \in (0,1)$, by $v \in C^1([0,1])$, hereafter called a "test function", and integrate over the interval $[0,1]$:

$$-\int_0^1 u''(x)v(x)dx = \int_0^1 f(x)v(x)dx, \quad \forall v \in V$$

where V is the test function space: it consists of all functions v that are continuous, vanish at $x = 0$ and $x = 1$ and whose first derivative is *piecewise continuous*, i.e., continuous everywhere except at a finite number of points in $[0,1]$ where the left and right limits v'_- and v'_+ exist but do not necessarily coincide. V is actually a vector space which is denoted by $H_0^1(0,1)$. Precisely,

$$\begin{aligned} H_0^1(0,1) &= \{v \in L^2(0,1) : v' \in L^2(0,1), v(0) = v(1) = 0\} \\ &= \{v : \int_0^1 v^2 < +\infty, \int_0^1 v' < +\infty, v(0) = v(1) = 0\}, \end{aligned} \tag{9}$$

where v' is the distributional derivative of v. It is an Hilbert space where the integral of the square is finite (L^2) (it is a Sobolev space).

By using integration by parts on the left integral ($f' = u''$ and $g = v$), we obtain

$$-\Big[u'(x)v(x)\Big]_0^1 + \int_0^1 u'(x)v'(x)dx = \int_0^1 f(x)v(x)dx$$

and assuming that v vanishes at endpoints $x = 0$ and $x = 1$, since $v \in V$ follows boundary conditions, we get:

$$\begin{cases} \int_0^1 u'(x)v'(x)dx = \int_0^1 f(x)v(x)dx \quad \forall v \in C^1([0,1]) \\ v(0) = v(1) = 0. \end{cases}$$

We have therefore shown that if a function $u \in C^2([0,1])$ satisfies (1), then u is also a solution of the following problem

$$\text{find } u \in V = H_0^1(0,1) :\ a(u,v) = (f,v) \ \text{ for all } v \in V = H_0^1(0,1), \tag{10}$$

where now $(f,v) = \int_0^1 fv\,dx$ denotes the scalar product of $L^2(0,1)$ and

$$a(u,v) = \int_0^1 u'v'dx \tag{11}$$

is a **bilinear form**, i.e. it is linear with respect to both arguments u and v. Problem (10) is called the **weak formulation** of problem (1). Since (10) contains only the first derivative of u it might cover cases in which a classical solution $u \in C^2([a,b])$ of (1) does not exist although the physical problem is well defined.

The Galerkin method

We now derive the Galerkin approximation of problem (1), which is the basic ingredient of the finite element method, widely employed in the numerical approximation of boundary value problems.

Unlike the finite difference method which stems directly from the differential (or *strong*) form (1), the Galerkin method is based on the weak formulation (10). If V_h is a finite dimensional vector subspace of V, $V_h \subseteq V$, the **Galerkin method** consists of approximating (10) by the problem

$$\text{find } u_h \in V_h : a(u_h, v_h) = (f, v_h) \quad \forall v_h \in V_h. \tag{12}$$

This is a finite dimensional problem. Actually, let $\{\phi_1, \dots, \phi_N\}$ denote a basis of V_h, i.e. a set of N linearly independent functions of V_h. Then we can write

$$u_h(x) = \sum_{j=1}^{N} u_j \varphi_j(x).$$

The integer N denotes the dimension of the vector space V_h. Taking $v_h = \varphi_i$ for $i = 1, \dots, N$ in (12), it turns out that the Galerkin problem (12) is equivalent to seeking N unknown coefficients $\{u_1, \dots, u_N\}$ such that

$$\sum_{j=1}^{N} u_j a(\varphi_j, \varphi_i) = (f, \varphi_i) \quad \forall i = 1, \dots, N, \tag{13}$$

where we have used the linearity of $a(\cdot, \cdot)$ with respect to its first argument, since

$$a(\sum_{j=1}^{N} u_j \varphi_j, \varphi_i) = \int_0^1 \left(\sum_{j=1}^{N} u_j \varphi_j \right)' \varphi_i' dx = \int_0^1 \sum_{j=1}^{N} u_j \varphi_j' \varphi_i' dx = \sum_{j=1}^{N} u_j \int_0^1 \varphi_j' \varphi_i' dx = \sum_{j=1}^{N} u_j a(\varphi_j, \varphi_i).$$

If we introduce the matrix $A_G = (a_{ij})$, $a_{ij} = a(\varphi_j, \varphi_i)$ (called the **stiffness matrix**), the unknown vector $\mathbf{u} = [u_1, \dots, u_N]^T$ and the right-hand side vector $\mathbf{f}_G = [f_1, \dots, f_N]^T$, with $f_i = (f, \varphi_i) = \int_0^1 f \varphi_i \, dx$, we see that (13) is equivalent to the linear system

$$A_G \mathbf{u} = \mathbf{f}_G. \tag{14}$$

The structure of A_G, as well as the degree of accuracy of u_h, depends on the form of the basis functions $\{\varphi_i\}$, and therefore on the choice of V_h.

There are two remarkable instances, the finite element method, where V_h is a space of piecewise polynomials over subintervals of $[0,1]$ of length not greater than h which are continuous and vanish at the endpoints $x = 0$ and 1, and the spectral method in which V_h is a space of algebraic polynomials still
vanishing at the endpoints $x = 0, 1$.

The finite element method

The **finite element method (FEM)** is an alternative to the finite differences for boundary-value problems, derived from a reformulation of the Poisson problem. It is a special technique for constructing a subspace V_h in (12) based on the piecewise polynomial interpolation.

With this aim, we introduce a partition $\mathcal{T}_h$ of $[0,1]$ into n subintervals $I_j = [x_j, x_{j+1}]$, $n \geq 2$, of width $h_j = x_{j+1} - x_j$, $j = 0, \ldots, n-1$, with

$$0 = x_0 < x_1 < \ldots < x_{n-1} < x_n = 1$$

and let $h = \max_{\mathcal{T}_h}(h_j)$. Since functions in $H_0^1(0,1)$ are continuous it makes sense to consider for $k \geq 1$ the family of piecewise polynomials X_h^k defined as

$$X_h^k = \{v \in C^0([0,1]) : v|_{I_j} \in \mathbb{P}^k(I_j), \forall I_j \in \mathcal{T}_h\} \tag{15}$$

Any function $v_h \in X_h^k$ is a continuous piecewise polynomial (finite elements) over $[0,1]$ and its restriction over each interval $I_j \in \mathcal{T}_h$ is a polynomial of degree $\leq k$. In the following we shall mainly deal with the cases $k = 1$ and $k = 2$.

Then, we set

$$V_h = X_h^{k,0} = \{v_h \in X_h^k : v_h(0) = v_h(1) = 0\}. \tag{16}$$

The dimension N of the finite element space V_h is equal to $nk - 1$.

Let us now focus on how to generate a suitable basis $\{\varphi_j\}$ for the finite element space X_h^k in the special cases $k = 1$. The basic point is to *choose* appropriately a set of degrees of freedom for each element I_j of the partition $\mathcal{T}_h$ (i.e., the parameters which permit uniquely identifying a function in X_h^k). The generic function v_h in X_h^k can therefore be written as

$$v_h(x) = \sum_{i=0}^{nk} v_i \varphi_i(x),$$

where $\{v_i\}$ denote the set of the degrees of freedom of v_h and the **basis functions** φ_i (which are also called **shape functions**) are assumed to satisfy the **Lagrange interpolation property** $\varphi_i(x_j) = \delta_{ij}$, $i, j = 0, \ldots, n$, where δ_{ij} is the Kronecker symbol.

The space X_h^1

This space consists of all continuous and piecewise linear functions over the partition $\mathcal{T}_h$. Since a unique straight line passes through two distinct nodes the number of degrees of freedom for v_h is equal to the number $n+1$ of nodes in the partition. As a consequence, $n+1$ shape functions $\varphi_i, i = 0, \ldots, n$, are needed to completely span the space X_h^1. The most **natural choice** for $\varphi_i, i = 1, \ldots, n-1$, is

$$\varphi_i(x) = \begin{cases} \frac{x - x_{i-1}}{x_i - x_{i-1}} & \text{for } x_{i-1} \leq x \leq x_i, \\ \frac{x_{i+1} - x}{x_{i+1} - x_i} & \text{for } x_i \leq x \leq x_{i+1}, \\ 0 & \text{elsewhere.} \end{cases} \tag{17}$$

The shape function φ_i is thus piecewise linear over $\mathcal{T}_h$, its value is 1 at the node x_i and 0 at all the other nodes of the partition. Its support (i.e., the subset of $[0,1]$ where φ_i is nonvanishing) consists of the union of the intervals I_{i-1} and I_i if $1 \leq i \leq n-1$ while it coincides with the interval I_0 (respectively I_{n-1}) if $i = 0$ (resp., $i = n$).

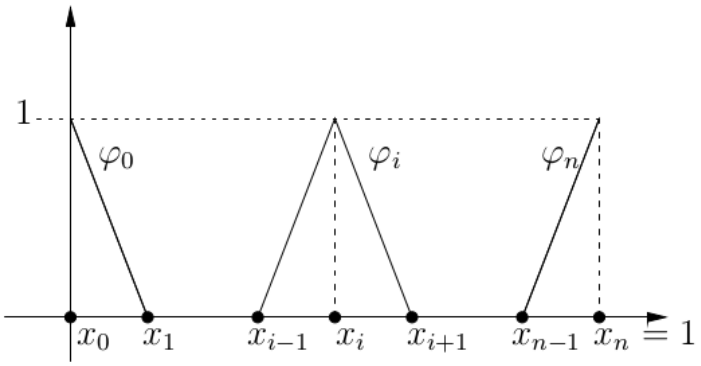


Fig. 12.3. Shape functions of X_h^1 associated with internal and boundary nodes

Let's take the weak formulation equation $\int_0^1 u_h'(x)v_h'(x)dx = \int_0^1 f(x)v_h(x)dx$.

We want to find $u_h \in V_h$ s.t. $u_h(0) = \alpha, u_h(1) = \beta$ and:

$$\forall v_h \in V_h^0 \qquad \sum_{i=0}^{N} \int_{x_i}^{x_{i+1}} u_h'(x)v_h'(x)dx = \int_0^1 f(x)v_h(x)dx$$

Since $|x_j - x_{j-1}| = h$ and the derivative of $\varphi_i(x)$ correspond to the shape of the line, we have that

$$\varphi_i'(x) = \begin{cases} \frac{1}{h} & x \in [x_{i-1}, x_i] \\ -\frac{1}{h} & x \in [x_i, x_{i+1}] \\ 0 & \text{otherwise} \end{cases}$$

So we have that $u_h(x) = \sum_{j=1}^N u_j\varphi_j(x)$, that leads to $u_h'(x) = \left(\sum_{j=1}^N u_j\varphi_j(x)\right)' = \sum_{j=1}^N u_j\varphi_j'(x)$, and we take $v_h(x) = \varphi_i(x),\ \forall i = 1, \dots, N$, that leads to $v_h'(x) = \varphi_i'(x)$, and we can rewrite as

$$\int_0^1 \sum_{j=1}^N u_j\varphi_j'(x) \cdot \varphi_i'(x)dx = \int_0^1 f(x)\varphi_i(x)dx$$
$$\sum_{j=1}^N u_j \int_0^1 \varphi_j'(x)\varphi_i'(x)dx = \int_0^1 f(x)\varphi_i(x)dx$$

for all $i = 1, \dots, N$, finding the system

$$A_{\text{fe}}\mathbf{u} = \mathbf{f}$$

where $\mathbf{u} = (u_1, \dots, u_N)$ is the vector of the unknowns coefficients u_j, $\mathbf{f} = (f_1, \dots, f_N)$ is the vector of $f_i = \int_0^1 f(x)\varphi_i(x)dx$ and $A_{\text{fe}} = (a_{ij})$ where $a_{ij} = a(\varphi_j, \varphi_i) = \int_0^1 \varphi_j'(x)\varphi_i'(x)dx$.

Then using the precedent formulation of the basis φ_i we have that

$$A_{\text{fe}} = \begin{cases} 0 & \text{when } |i-j| \geq 2 \\ \int_{x_{i-1}}^{x_{i+1}} \varphi_i'(x)^2dx = \frac{1}{h^2}\int_{x_{i-1}}^{x_{i+1}} dx = \frac{2h}{h^2} = \frac{2}{h} & \text{when } i = j \\ \int_{x_i}^{x_{i+1}} \varphi_i'(x)\varphi_{i-1}'(x)dx = -\frac{1}{h^2}\int_{x_i}^{x_{i+1}} dx = -\frac{h}{h^2} = -\frac{1}{h} & \text{when } |i-j| = 1 \end{cases}$$

so we have that

$$A_{\text{fe}} = \frac{1}{h}\begin{bmatrix} -2 & 1 & 0 & & \cdots & & 0 \\ 1 & -2 & 1 & 0 & \cdots & & 0 \\ 0 & 1 & -2 & 1 & 0 & \cdots & 0 \\ \vdots & & & & & & \vdots \\ 0 & & \cdots & & 0 & 1 & -2 \end{bmatrix}$$

The matrix is similar to the one of the finite difference, A_{fd}, but with $\frac{1}{h}$ instead of $\frac{1}{h^2}$.

The final system has different right-hand side and different solution than the FD one, but have the same accuracy w.r.t. h.

FD works (converges) for $f \in C^2([0,1])$ while FE converges if $\int_0^1 f^2(x)dx < \infty$. Using polynomials with $d > 1$ allows for greater convergence, and leads to different matrices.

Let us now examine the structure and the basic properties of the stiffness matrix associated with system (14) in the case of the finite element method ($A_G = A_{\text{fe}}$). Since the finite element basis functions for X_h^k have a local support, A_{fe} is sparse. In the particular case $k = 1$, the support of the shape function φ_i is the union of the intervals I_{i-1} and I_i if $1 \leq i \leq n-1$, and it coincides with the interval I_0 (respectively I_{n-1}) if $i = 0$ (resp., $i = n$). As a consequence, for a fixed $i = 1, \ldots, n-1$, only the shape functions φ_{i-1} and φ_{i+1} have a nonvanishing support intersection with that of φ_i, which implies that A_{fe} is tridiagonal since $a_{ij} = 0$ if $j \notin \{i-1, i, i+1\}$.

The condition number of A_{fe} is a function of the grid size h; indeed, it holds

$$K_2(A_{\text{fe}}) = ||A_{\text{fe}}||_2 \, ||A_{\text{fe}}^{-1}||_2 = O(h^{-2})$$

which demonstrates that the conditioning of the finite element system (14) grows rapidly as $h \to 0$. This is clearly conflicting with the need of increasing the accuracy of the approximation and, in multidimensional problems, demands suitable preconditioning techniques if iterative solvers are used.

Remark (Elliptic problems of higher order)**.** The Galerkin method in general, and the finite element method in particular, can also be applied to other type of elliptic equations, for instance to those of fourth order. In that case, the numerical solution (as well as the test functions) should be continuous together with their first derivative.

Analysis of the Galerkin method

We will state a couple of general results that hold for any Galerkin problem (12).

Theorem (Lax-Milgram Lemma)**.** *Let V be an Hilbert space, endowed with norm $||\cdot||_V$, and let $a(\cdot,\cdot) : V \times V \to \mathbb{R}$ be a map that satisfies the following properties:*

- ***bilinearity***: $\forall u, v \in V, a(u,\cdot)$ *and* $a(\cdot, v)$ *are linear, a.k.a. it is linear in both variables:* $\forall \alpha, \beta \in \mathbb{R}, \forall u, v, w \in V$
$$a(\alpha u + \beta v, w) = \alpha a(u,w) + \beta a(v,w)$$
$$a(u, \alpha v + \beta w) = \alpha a(u,v) + \beta a(u,w).$$
- **continuity**: $\exists M > 0$ s.t. $|a(u,v)| \leq M||u||_V||v||_V, \ \forall u, v \in V.$
- **coercivity**: $\exists \alpha_0 > 0$ s.t. $a(v,v) \geq \alpha_0 ||v||_V^2, \ \forall v \in V.$

Moreover, let the following inequality hold

$$|(f,v)| \leq K||v||_V \qquad \forall v \in V.$$

Then both problems (10) and (12) admit unique solutions that satisfies

$$||u||_V \leq \frac{M}{\alpha_0}, \qquad ||u_h||_V \leq \frac{M}{\alpha_0}.$$

So the Lax-Milgram theorem is used to prove the existence and uniqueness for both the weak formulation of Poisson problem and the Galerkin method.

Remark. For every $v_h \in V_h^0$, the grid function v_h whose grid values are $(v_{j+1} - v_j)/h$, $j = 0, \dots, n-1$, can be regarded as a discrete derivative of v_h.

Poincaré inequality. *For every interval $[a, b]$ there exists a constant $C_P > 0$ such that*

$$||v||_{L^2(a,b)} \leq C_P ||v^{(1)}||_{L^2(a,b)} \tag{18}$$

for all $v \in C^1([a, b])$ such that $v(a) = v(b) = 0$ and where $||\cdot||_{L^2(a,b)}$ is the norm in $L^2(a, b)$:

$$||f||_{L^2(a,b)} = \left(\int_a^b |f(x)|^2 dx \right)^{1/2},$$

and the space $L^2(a, b)$ is defined as

$$L^2(a, b) = \left\{ f : (a, b) \to \mathbb{R}, \int_a^b |f(x)|^2 dx < +\infty \right\}.$$

We endow the space $H_0^1(0, 1)$ with the following norm

$$|v|_{H^1}(0, 1) = \left\{ \int_0^1 |v'(x)|^2 dx \right\}^{1/2}.$$

Property: *Provided that $f \in L^2(0, 1)$, the norm of the Galerkin solution u_h remains bounded:*

$$|u_h|_{H^1(0,1)} = \frac{C_P}{\alpha_0} ||f||_{L^2(0,1)} \tag{19}$$

Proof. Taking $v_h = u_h$ in (12), we obtain

$$\alpha_0 |u_h|^2_{H^1(0,1)} \leq \int_0^1 u_h' u_h' dx = a(u_h, u_h) = (f, u_h) \leq ||f||_{L^2(0,1)} ||u_h||_{L^2(0,1)}$$

where we have used the *Cauchy-Schwartz inequality*

$$\left| \int_\alpha^\beta u(x) v(x) dx \right| \leq \left(\int_\alpha^\beta u^2(x) dx \right)^{1/2} \left(\int_\alpha^\beta v^2(x) dx \right)^{1/2}$$

to set the right-hand side inequality. Owing to the Poincaré inequality we have the inequality (19).

□

This represents a **stability** result for the solution of the Galerkin problem (12), since we have that it admits a unique solution depending continuously on the data.

Céa's Lemma. *The following error inequality holds*

$$||u - u_h||_V \leq \frac{M}{\alpha_0} \min_{w_h \in V_h} ||u - w_h||_V.$$

Proof. Subtracting (12) from (10) (where we use $v_h \in V_h \subset V$), owing to the bilinearity of the form $a(\cdot, \cdot)$, we obtain

$$a(u, v_h) - a(u_h, v_h) = (f, v_h) - (f, v_h) \quad \Longrightarrow \quad a(u - u_h, v_h) = 0 \qquad \forall v_h \in V_h \tag{20}$$

Besides, we have that

$$\begin{aligned}
\alpha_0 ||u - u_h||_V^2 &\leq a(u - u_h, u - u_h) && \text{[coercivity]} \\
&= a(u - u_h, u - w_h) + a(u - u_h, w_h - u_h) \quad \forall w_h \in V_h && \text{[linearity]} \\
&= a(u - u_h, u - w_h) + 0 \quad \forall w_h \in V_h && \text{[for (20) since } w_h - u_h \in V_h\text{]} \\
&\leq M ||u - u_h||_V \, ||u - w_h||_V \quad \forall w_h \in V_h && \text{[continuity]}
\end{aligned}$$

So we have that $||u - u_h||_V \leq \frac{M}{\alpha_0}||u - w_h||_V, \forall w_h \in V_h,$ which is also valid for the minimum, so $||u - u_h||_V \leq \frac{M}{\alpha_0} \min_{w_h \in V_h} ||u - w_h||_V$, which proves the theorem.

□

So this proves the **convergence** of the Galerkin problem, since $e(x) = u(x) - u_h(x)$ is the error.