Numerical Analysis

Jonas Tölle[1]

Lecture notes for MS-C1650 Numerical Analysis, Aalto University

Last updated: 30.5.2024

Largely based the lecture transcript by Harri Hakula, 2021.

> **Intended learning outcomes.** After the course, the student will be able to...
>
> - explain the fundamental concepts of numerical analysis, like condition number, stability, and convergence rate;
> - construct the floating point numbers;
> - discuss and employ basic numerical algorithms like Newton's method;
> - use the Monte-Carlo method in basic problems in analysis and geometry;
> - apply different methods of interpolation polynomials and numerical quadrature rules;
> - understand the Euler scheme and linear multi-step methods for solving ordinary differential equations.

Floating-point numbers

> The set of real numbers $\mathbb{R}$ is infinite in two ways: it is unbounded and continuous. In most practical computing, the second kind of infiniteness is more consequential than the first kind, so we turn our attention there first.

Instead of $\mathbb{R}$, we shall introduce the set of *floating-point numbers* (*floats*) $\mathbb{F}$. They come with different bases, precisions and exponent ranges, and other features. The basic representation is

$$x = \pm(d_0.d_1d_2\dots d_p)_k \cdot k^e.$$

$k \in \mathbb{N} \setminus \{1\}$ is called *base* or *radix*, $p \in \mathbb{N}_0 := \mathbb{N} \cup \{0\}$ is called the precision, d_i, $i \in \{0, \dots, p\}$, and the sequence of numbers

$$(d_0.d_1d_2\dots d_p)_k := \sum_{i=0}^{p} d_i k^{-i}$$

is called *mantissa* or *significand*. The exponent $e \in \mathbb{Z}$ is bounded $m \leq e \leq M$, where $m, M \in \mathbb{Z}$.
If $k = 10$, we can read the usual decimal commas from the mantissa:

$$(1.01)_{10} = 1 \cdot 10^0 + 0 \cdot 10^{-1} + 1 \cdot 10^{-2} = 1.01.$$

If $k = 2$, we have binary floats. In the binary case, we observe that we can always choose $d_0 = 1$, hence saving one bit, which can be expressed by e. We refer to this as *normalization*. The mantissa is always contained in the interval $[1, 2)$.

> **Example.** (Toy floating point system). Binary floats of the type
>
> $$(1.b_1b_2)_2$$
>
> with exponents $e = -1, 0, 1$.
> Hence $(1.00)_2 = 1$, $(1.01)_2 = \frac{5}{4}$, $(1.10)_2 = \frac{3}{2}$, and >$(1.11)_2 = \frac{7}{4}$. By multiplying with the exponents $2^{-1} = \frac{1}{2}$, $2^0 = 1$, $2^1 = 2$, we get the whole set:

e		
1	$\frac{5}{4}$	$\frac{3}{2}$
2	$\frac{5}{2}$	3
$\frac{1}{2}$	$\frac{5}{8}$	$\frac{3}{4}$

Important quantity: $(1.01)_2 - 1 = \frac{1}{4}$, the so-called *machine epsilon*.

Define the machine epsilon by $\varepsilon := 2^{-p} = (1.00\ldots01)_2 - 1$.

Rounding

For the rounding function $\mathrm{round} : \mathbb{R} \to \mathbb{F}$, we have 5 alternative definitions:

- Rounding to nearest (*default*)
- Rounding to $+\infty$
- Rounding to $-\infty$
- Rounding to 0
- Rounding away from 0

It holds that $\mathrm{round}(x) = x(1+\delta)$, where $|\delta| < \frac{\varepsilon}{2}$, where ε denotes the machine epsilon. Note that usually δ depends on x. There is a way to define the standard arithmetic operations on $\mathbb{F}$ such that

$$a \oplus b = \mathrm{round}(a+b) = (a+b)(1+\delta_1),$$

$$a \ominus b = \mathrm{round}(a-b) = (a-b)(1+\delta_2),$$

$$a \otimes b = \mathrm{round}(ab) = ab(1+\delta_3),$$

$$a \oslash b = \mathrm{round}\left(\frac{a}{b}\right) = \frac{a}{b}(1+\delta_4), \quad b \neq 0.$$

Here, generally $\delta_i \neq \delta_j, i \neq j$.

IEEE 754 “Double precision”

$k = 2$, 64 bits, where:

- The sign: 1 bit;
- The exponent field $0 \leq \mathrm{ex} \leq 2047$: 11 bits, where $e = \mathrm{ex} - 1023$, and $-1022 \leq e \leq 1023$, where $\mathrm{ex} = 0$ and $\mathrm{ex} = 2047$ are special cases.
- The mantissa 52 bits, precision $p = 52$.

Exponent field	Number	Type of number
$00\ldots00 = 0$	$\pm(0.b_1b_2\ldots b_{52})_2 \cdot 2^{-1022}$	0 or *subnormal*
$00\ldots01 = 1$	$\pm(1.b_1b_2\ldots b_{52})_2 \cdot 2^{-1022}$	
$00\ldots10 = 2$	$\pm(1.b_1b_2\ldots b_{52})_2 \cdot 2^{-1021}$	

$\dots$	$\dots$	
$01\dots11 = 1023$	$\pm(1.b_1b_2\dots b_{52})_2 \cdot 2^0$	
$\dots$	$\dots$	
$11\dots10 = 2046$	$\pm(1.b_1b_2\dots b_{52})_2 \cdot 2^{1023}$	
$11\dots11 = 2047$	$\pm\infty$ if $b_1 = b_2 = \dots = b_{52} = 0$, otherwise *NaN*	*exception*

Thus, there are two zeros, two infinities and NaN which denotes "*not a number*". The smallest positive normalized number is:

$$(1.0)_2 \cdot 2^{-1022} \approx 2.2 \cdot 10^{-308}.$$

The largest positive number is:

$$(1.1\dots1)_2 \cdot 2^{1023} \approx 1.8 \cdot 10^{308}$$

The machine epsilon is:

$$2^{-52} \approx 2.22 \cdot 10^{-16}.$$

Here's an easy-to-follow video explaining floating point numbers (and a specific version of Newton's algorithm).

Condition number and stability

Conditioning of problems

Assume that $f : \mathbb{R} \to \mathbb{R}$ "solution map" of the problem, input numbers x, $\hat{x}$, close in value, e.g. $\hat{x} = \mathrm{round}(x)$. Set $y := f(x)$, $\hat{y} := f(\hat{x})$.

Definition. The *absolute condition number* $C(x)$ is defined by the relation

$$|y - \hat{y}| \approx C(x)|x - \hat{x}|.$$

The *relative condition number* $K(x)$ is defined by the relation

$$\left|\frac{y - \hat{y}}{y}\right| \approx K(x) \left|\frac{x - \hat{x}}{x}\right|$$

By the normalization, we guarantee that

$$(\text{relative error in the output}) \approx K(x) \times (\text{relative error in the input}).$$

Now,

$$y - \hat{y} = f(x) - f(\hat{x}) = \underbrace{\frac{f(x) - f(\hat{x})}{x - \hat{x}}}_{\to f'(x) \text{ as } \hat{x} \to x}(x - \hat{x})$$

Thus, $C(x) = |f'(x)|$.
Furthermore,

$$\frac{y-\hat{y}}{y} = \frac{f(x)-f(\hat{x})}{f(x)} = \underbrace{\frac{f(x)-f(\hat{x})}{x-\hat{x}}}_{\to f'(x) \text{ as } \hat{x}\to x} \frac{x-\hat{x}}{x} \frac{x}{f(x)}$$

Thus, $K(x) = \left|\frac{xf'(x)}{f(x)}\right|$.

> **Example.** $f(x) = 2x$, $f'(x) = 2$. Thus, $C(x) = 2$, $K(x) = \left|\frac{2x}{2x}\right| = 1$. This is a well-conditioned problem.

> **Example.** $g(x) = \sqrt{x}$, $g'(x) = \frac{1}{2\sqrt{x}}$. Thus, $C(x)$ is becomes unbounded for $x > 0$ close to zero, e.g. $x \approx 10^{-8}$ yields $C(x) \approx 10^4$. On the other hand, $K(x) = \left|\frac{x}{2\sqrt{x}\sqrt{x}}\right| = \frac{1}{2}$.

Stability of algorithms

Definition. An algorithm or numerical process is called *stable* if small changes in the input produce small changes in the output. It is called *unstable* if large changes in the output are produced by small changes in the input.

An algorithm is stable, if every step is well-conditioned (i.e. has a uniformly bounded condition number). It is unstable if any step is ill-conditioned (i.e. the condition number may become arbitrarily large).

Forward error analysis (FEA) is asking:
"How far are we from the true solution?"

Backward error analysis (BEA) is asking:
"Given the answer, what was the problem?"

> **Example.**
> Set:
>
> $$\mathrm{fl}(x+y) := \mathrm{round}(x) \oplus \mathrm{round}(y) = ((x(1+\delta_1) + y(1+\delta_2))(1+\delta_3),$$
>
> where $|\delta_i| < \frac{\varepsilon}{2}, i = 1, 2, 3.$
> FEA:
>
> $$\mathrm{fl}(x+y) = x + y + x(\delta_1 + \delta_3 + \delta_1\delta_3) + y(\delta_2 + \delta_3 + \delta_2\delta_3).$$
>
> The absolute error is
>
> $$|\mathrm{fl}(x+y) - (x+y)| \leq (|x| + |y|)\left(\varepsilon + \frac{\varepsilon^2}{4}\right).$$
>
> The relative error is:
>
> $$\left|\frac{\mathrm{fl}(x+y) - (x+y)}{x+y}\right| \leq \frac{(|x| + |y|)\left(\varepsilon + \frac{\varepsilon^2}{4}\right)}{|x+y|}.$$
>
> BEA:
>
> $$\mathrm{fl}(x+y) = x(1+\delta_1)(1+\delta_3) + y(1+\delta_2)(1+\delta_3).$$
>
> Thus the relative error for each term is less or equal to $\varepsilon + \frac{\varepsilon^2}{4}$.
> Hence the sum of two floating point numbers is backwards stable.

Well-conditioned problems may have unstable algorithms. For stability, each step has to be well conditioned. Some ill-conditioned steps produce an unstable algorithm. Ill-conditioned problems cannot be reliably solved with a stable algorithm.

> **Example.** Consider evaluating $f(x) = \sqrt{1+x} - 1$ for x close to zero. The relative condition number is:
>
> $$K(x) = \left| \frac{x f'(x)}{f(x)} \right| = \frac{x}{2\sqrt{1+x}(\sqrt{1+x} - 1)}$$
>
> $$= \frac{x(\sqrt{1+x}+1)}{2\sqrt{1+x}(\sqrt{1+x}-1)(\sqrt{1+x}+1)} = \frac{\sqrt{1+x}+1}{2\sqrt{1+x}},$$
>
> and $K(0) = 1$.
> Consider the following 3 steps;
>
> 1. $t_1 := 1 + x$, well-conditioned, x close to 0.
> 2. $t_2 := \sqrt{t_1}$, relatively well-conditioned, also absolutely well conditioned, because t_1 is close to 1.
> 3. $t_3 := t_2 - 1$, ill-conditioned, relative condition number of this step: $K_3(t_2) = \left| \frac{t_2}{t_2 - 1} \right|$, which becomes unbounded for t_2 close to 1!
>
> On the other hand, the problem is well-conditioned. Solve it by writing:
>
> $$f(x) = \sqrt{1+x} - 1 = \frac{(\sqrt{1+x}+1)(\sqrt{1+x}-1)}{\sqrt{1+x}+1} = \frac{x}{\sqrt{1+x}+1},$$
>
> which can be evaluated directly close to zero.

Numerical differentiation

Recall Taylor's theorem, for a twice differentiable function $f : \mathbb{R} \to \mathbb{R}$,

$$f(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2}(x - x_0)^2 f''(\xi),$$

for any $x_0, x \in \mathbb{R}$, where $\xi \in [x_0, x]$.

> What is that ξ (= "xi")? Under certain assumptions elementary functions have their series expansions. If the series is truncated, we have the Taylor polynomial. However, the residual has an explicit expression but due to application of an intermediate value theorem, the exact location of the point ξ is not known, i.e. "generic".

By setting $x := z + h$, $x_0 := z$, we obtain the useful equivalent formula

$$f(z + h) = f(z) + f'(z)h + \frac{1}{2} f''(\xi) h^2,$$

for every $z, h \in \mathbb{R}$, $\xi \in [z, z + h]$.

Rate of convergence (Q-convergence)

Let (x_k) be an infinite sequence of real numbers. Let $s_k := \sup_{l \geq k} x_l$, $k \in \mathbb{N}$, be the *supremum* (i.e., the lowest upper bound) of the *tail* (that is, large indices $l \geq k$) of (x_k). Define the $\limsup$ (*limes superior*) as

$$\limsup_{k \to \infty} x_k := \lim_{k \to \infty} s_k \in [-\infty, +\infty].$$

Other than a limit, it always exists, but can be $\pm\infty$. If (x_k) is bounded, the $\limsup$ is the largest limit of a converging subsequence. If $\lim_{k\to\infty} x_k \in (-\infty, \infty)$ exists, then $\lim_{k\to\infty} x_k = \limsup_{k\to\infty} x_k$. The opposite is not true.

> **Examples**. (1) $x_k := (-1)^k$, then $s_k = 1$, and $\limsup_{k\to\infty} x_k = 1$.
> (2) $x_k = \sin(k)$, then $s_k = 1$, and $\limsup_{k\to\infty} x_k = 1$.

Assume that $\lim_{k\to\infty} x_k = x$ and that there exists some large index $M \in \mathbb{N}$ such that $x_k \neq x$ for all $k \geq M$. Then we define the following quantity for $p \geq 0$

$$C(p) := \limsup_{k\to\infty} \frac{|x_{k+1} - x|}{|x_k - x|^p}.$$

We observe that $C(p^*) < \infty$ for some $p^* > 0$ implies $C(p) = 0$ for every $0 \leq p < p^*$. If $C(p^*) > 0$ for some $p^* > 0$ then $C(p) = \infty$ for any $p > p^*$.

> **Proof**. By the submultiplicative property of $\limsup$,
>
> $$C(p) = \limsup_{k\to\infty} \frac{|x_{k+1} - x|}{|x_k - x|^p} = \limsup_{k\to\infty} \left[\frac{|x_{k+1} - x|}{|x_k - x|^{p^*}} |x_k - x|^{p^* - p} \right]$$
>
> $$\leq \left(\limsup_{k\to\infty} \frac{|x_{k+1} - x|}{|x_k - x|^{p^*}} \right) \cdot \left(\limsup_{k\to\infty} |x_k - x|^{p^* - p} \right) = C(p^*) \cdot \begin{cases} 0 & \text{if } p < p^*, \\ \infty & \text{if } p > p^*. \end{cases}$$
>
> □

Thus, there exists a (possibly infinite) p^* such that

$$C(p) = \begin{cases} 0 & \text{if } 0 \leq p < p^*, \\ C(p^*) & \text{if } p = p^*, \\ \infty & \text{if } p > p^*. \end{cases}$$

The number p^* is called *order of convergence* for the sequence (x_k) and determines the *rate of convergence* as follows:

- If $p^* = 1$ and $C(1) = 1$ then we say the convergence is *sublinear*.
- If $p^* = 1$ and $1 > C(1) > 1$ then we say the convergence is *linear*.
- If $p^* > 1$ or $C(1) = 0$ then we say the convergence is *superlinear*.
- If $p^* = 2$ then we say the convergence is *quadratic*.
- If $p^* = 3$ then we say the convergence is *cubic*, etc.

When working with convergence estimates it is often useful to use the following approximation:

$$|x_{k+1} - x| \approx C|x_k - x|^{p^*}$$

for some constant $C > 0$, not necessarily $C(p^*)$.
Here, it is useful to look at the logarithmic behavior:

$$\log(|x_{k+1} - x|) \approx \log\left(C|x_k - x|^{p^*}\right) = \log(C) + \log\left(|x_k - x|^{p^*}\right) = \log(C) + p^* \log(|x_k - x|).$$

> The *rate of convergence* can be used interchangeably with the *order of convergence*. However, there is some caution necessary, as different authors use different terminology here. Usually, the order of convergence always refers to the same thing, namely, the p^*-exponent in the denominator of the limit defining the order of convergence. Most confusingly, some authors call the order of convergence "rate of convergence", as e.g. here. The English Wikipedia article calls it the order of

> convergence, whereas here the rate of convergence is the constant in the definition, which also determines the speed of convergence, together with the order of convergence. So, please always check the context, as the use of the terminology should be clear from it. If there is no definition, try to figure out what is meant in each text. As a rule of thumb: The "order of convergence" is a unique terminology in numerical analysis. The "rate of convergence" can mean at least two different things. I will use both words for the same thing, but will try to make clear what I mean from case to case. In any case, to be sure, use "order of convergence". My PhD advisor usually said that in mathematics "it's all hollow words" (meaning that one should check the definition).

Landau's little o- and big O-notation

The Landau notation is an amazing tool applicable in all of real analysis. The reason it is so convenient and widely used is because it underlines a key principle of real analysis, namely ''estimation". Loosely speaking, the Landau notation introduces two operators which can be called the "order of magnitude" operators, which essentially compare the magnitude of two given functions.

The "little-o"

The "little-o" provides a function that is of lower order of magnitude than a given function, that is the function $o(g(x))$ is of a lower order than the function $g(x)$. Formally,

Definition.
Let $A \subseteq \mathbb{R}$ and let $c \in \mathbb{R}$.
Let $f, g : A \to \mathbb{R}$.

If $\lim_{x \to c} \frac{f(x)}{g(x)} = 0$ then we say that

"As $x \to c$, $f(x) = o(g(x))$"

Examples.

- As $x \to \infty$, (and $m < n$), $x^m = o(x^n)$;
- As $x \to \infty$, (and $n \in \mathbb{N}$), $\log x = o(x^n)$;
- As $x \to 0$, $\sin x = o(1)$.

The "Big-O"

The "Big-O" provides a function that is at most the same order as that of a given function, that is the function $O(g(x))$ is at most the same order as the function $g(x)$. Formally,

Definition.
Let $A \subseteq \mathbb{R}$ and let $c \in \mathbb{R}$

Let $f, g : A \to \mathbb{R}$

If there exists $M > 0$ such that $\lim_{x \to c} \left| \frac{f(x)}{g(x)} \right| < M$ then we say that

"As $x \to c$, $f(x) = O(g(x))$"

Examples.

- As $x \to 0$, $\sin x = O(x)$;
- As $x \to \frac{\pi}{2}$, $\sin x = O(1)$.

Applications

We will now consider few examples which demonstrate the power of this notation.

Differentiability

Let $f : \mathcal{U} \subseteq \mathbb{R} \to \mathbb{R}$ and $x_0 \in \mathcal{U}$.

Then f is differentiable at x_0 if and only if

There exists a $\lambda \in \mathbb{R}$ such that as $x \to x_0$, $f(x) = f(x_0) + \lambda(x - x_0) + o\left(|x - x_0|\right)$.

Mean Value Theorem

Let $f : [a, x] \to \mathbb{R}$ be differentiable on $[a, b]$. Then,

As $x \to a$, $f(x) = f(a) + O(x - a)$.

Taylor's Theorem

Let $f : [a, x] \to \mathbb{R}$ be n-times differentiable on $[a, b]$. Then,

As $x \to a$, $f(x) = f(a) + \frac{(x-a)f'(a)}{1!} + \frac{(x-a)^2 f''(a)}{2!} + \ldots + \frac{(x-a)^{n-1} f^{(n-1)}(a)}{(n-1)!} + O\left((x - a)^n\right)$.

Finding roots of functions and fixed points

Let $f : \mathbb{R} \to \mathbb{R}$ be continuous. We are interested in methods for finding *zeros*, that is, *roots* of f, in other words, $x \in \mathbb{R}$, such that $f(x) = 0$.

Definition. If $f : \mathbb{R}^n \to \mathbb{R}$, $n \in \mathbb{N}$ is k-times continuously differentiable, then we write $f \in C^k(\mathbb{R}^n)$ or just $f \in C^k$. For $k = 0$, $f \in C^0$ or $f \in C(\mathbb{R}^n)$ or $f \in C([a, b])$ just means that f is assumed to be continuous.

Bisection method

The intermediate value theorem for continuous functions implies that $x_1 < x < x_2$ with $f(x) = 0$ exists if $f(x_1)f(x_2) < 0$, i.e., there is a sign change. The *bisection method* is based on halving the interval such that the sign condition is preserved. Note that, in principle, we have to look for intervals $[x_1, x_2]$.

Let us analyze the convergence rate. Let $[a, b]$ be an interval. After k steps the interval of analysis has length $\frac{b-a}{2^k}$ which converges to zero for $k \to \infty$. Let us look in a neighborhood of radius $\delta > 0$, so that

$$\frac{b-a}{2^k} \leq 2\delta \quad \Leftrightarrow \quad 2^{k+1} \geq \frac{b-a}{\delta} \quad \Leftrightarrow \quad k \geq \log_2\left(\frac{b-a}{\delta}\right) - 1.$$

Every step reduces the error by factor $\frac{1}{2}$. The convergence rate is thus linear.

Newton's method

Assume that $f \in C^1$. For an initial value x_0, consider the iteration

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

> **Heuristics.** If $f(x_*) = 0$ and $f \in C^2$, by the Taylor's expansion,
>
> $$0 = f(x_*) = f(x_0) + (x_* - x_0)f'(x_0) + \frac{(x_* - x_0)^2}{2} f''(\xi)$$
>
> for $\xi \in [x_0, x_*]$, and upon neglecting the 2nd order term,
>
> $$x_* \approx x_0 - \frac{f(x_0)}{f'(x_0)}.$$

Theorem. If $f \in C^2$ and x_0 is sufficiently good (i.e. close to the root x_*) and if $f'(x_*) \neq 0$, then Newton's method converges quadratically.

Proof. By Taylor's expansion, it follows that

$$x_* = x_k - \frac{f(x_k)}{f'(x_k)} - \frac{(x_* - x_k)^2}{2} \frac{f''(\xi_k)}{f'(x_k)}$$

for some $\xi_k \in [x_*, x_k]$. Take x_{k+1} from the method and subtract,

$$x_{k+1} - x_* = (x_* - x_k)^2 \underbrace{\frac{f''(\xi_k)}{2f'(x_k)}}_{\leq D}.$$

In other words,

$$|x_{k+1} - x_*| \leq D|x_k - x^*|^2,$$

as $k \to \infty$ and thus $x_k \to x_*$.
Hence the method is quadratic. Note that $f'(x_k)$ does not vanish by continuity if x_k is close to x_*. $\square$

What happens if $f'(x_*) = 0$?

$$x_{k+1} - x_* = (x_* - x_k)^2 \underbrace{\frac{f''(\xi_k)}{2f'(x_k)}}_{\to 0}$$

as $k \to \infty$.
By Taylor's expansion (η="eta"):

$$f'(x_k) = \underbrace{f'(x_*)}_{=0} + (x_k - x_*)f''(\eta_k) = (x_k - x_*)f''(\eta_k)$$

for some $\eta_k \in [x_*, x_k]$, and hence

$$x_{k+1} - x_* = f''(\eta_k) = (x_k - x_*)f''(\eta_k)(x_k - x_*).$$

The method has degenerated to a linear method!

> **Example.** $f(x) = x^2$, $f'(x) = 2x$. Newton:

$$x_{k+1} = x_k - \frac{x_k^2}{2x_k} = \frac{1}{2}x_k.$$

Secant method

Sometimes it can be difficult or computationally expensive to compute the derivative $f'(x_k)$. Newton 's method can be adapted by approximating the derivative by the differential quotient. The secant method is the following two-step recursive algorithm.

$$x_{k+1} = x_k - \frac{f(x_k)(x_k - x_{k-1})}{f(x_k) - f(x_{k-1})}, \quad k = 1, 2, \dots$$

with two distinct starting points $x_0 \neq x_1$. The convergence rate is $\frac{1+\sqrt{5}}{2} \approx 1.62$, the *golden ratio*.

Fixed point iteration

Definition. A point $x \in \mathbb{R}$ is called a fixed point of $\varphi : \mathbb{R} \to \mathbb{R}$ if $\varphi(x) = x$.

We could for instance use Newton's method to find fixed points by setting $f(x) := \varphi(x) - x$.

Banach's Fixed Point Theorem. Suppose that φ is a *contraction*, that is, there exists a constant $L < 1$ such that

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

for all $x, y \in \mathbb{R}$. Then there exists a unique fixed point $x_* \in \mathbb{R}$ of φ, i.e., $\varphi(x_*) = x_*$, and the fixed point iteration $\varphi_n := \underbrace{\varphi \circ \ldots \circ \varphi}_{n\text{-times}}$ satisfies $\lim_{n\to\infty} \varphi_n(x_0) = x_*$ for any starting point $x_0 \in \mathbb{R}$. The convergence rate is at least linear.

Proof. We prove that the sequence $\{\varphi_k(x_0)\}_{k=0}^{\infty} = \{x_k\}_{k=0}^{\infty}$ is a Cauchy sequence. Let $k > j$. Then, by the triangle inequality,

$$|x_k - x_j| \leq \underbrace{|x_k - x_{k-1}| + |x_{k-1} - x_{k-2}| + \ldots + |x_{j+1} - x_j|}_{(k-j)\text{-summands}}.$$

Furthermore,

$$|x_m - x_{m-1}| = |\varphi(x_{m-1}) - \varphi(x_{m-2})| \leq L|x_{m-1} - x_{m-2}| \leq L^{m-1}|x_1 - x_0|.$$

Hence, by the geometric series,

$$|x_k - x_j| \leq L^j \frac{1 - L^{k-j}}{1 - L}|x_1 - x_0|.$$

If $k > N, j > N$, then

$$|x_k - x_j| \leq L^N \frac{1}{1 - L}|x_1 - x_0| \to 0$$

as $N \to \infty$, which proves that $\{x_k\}$ is a Cauchy sequence. The linear convergence rate follows also from this estimate.
The existence of a fixed point follows from the continuity of φ (as contractions are uniformly continuous, in fact, even Lipschitz continuous) as follows.

$$x_* = \lim_{k\to\infty} x_k = \lim_{k\to\infty} x_{k+1} = \lim_{k\to\infty} \varphi(x_k) = \varphi(\lim_{k\to\infty} x_k) = \varphi(x_*).$$

□

Theorem. Assume that $\varphi \in C^p$ for $p \geq 1$. Furthermore, assume that has a fixed point x_* and assume that

$$\varphi'(x_*) = \varphi''(x_*) = \ldots = \varphi^{(p-1)}(x_*) = 0$$

for $p \geq 2$ and

$$G'(x_*) < 1$$

if $p = 1$. Then the fixed point sequence $\{\varphi_k(x_0)\}$ converges to x_* at least with rate p, provided that the starting point x_0 is sufficiently close to x_*. If, in addition, $\varphi^{(p)}(x_*) \neq 0$, then the rate of convergence is precisely p.

Proof. First note that by Banach's fixed point theorem the limit indeed converges to x_* for suitable starting points x_0. By the Taylor expansion,

$$x_{k+1} - x_* = \varphi(x_k) - \varphi(x_*) = \sum_{l=1}^{p-1} \frac{\varphi^{(l)}(x_*)}{l!}(x_k - x_*)^l + \frac{G^{(p)}(\xi_k)}{p!}(x_k - x_*)^p$$

for some ξ_k between x_* and x_k. The sum will be left empty for the case $p = 1$. Since $\varphi^{(l)}(x_*) = 0$ for $1 \leq l \leq p-1$, we get that

$$|x_{k+1} - x_*| = \frac{|\varphi^{(p)}(\xi_k)|}{p!}|x_k - x_*|^p$$

.
By continuity, there exists $C > 0$ (with $C < 1$ for $p = 1$) such that

$$\frac{|\varphi^{(p)}(\xi_k)|}{p!} \leq C$$

for ξ_k sufficiently close to x_* (that is, for sufficiently large k). Thus,

$$|x_{k+1} - x_*| \leq C|x_k - x_*|^p$$

for large k, and thus the rate of convergence is at least p. Note that for $p = 1$, this also proves convergence by

$$|x_{k+1} - x_*| < \underbrace{|\varphi(\xi_k)|}_{<1}|x_k - x_*|.$$

If $\varphi^{(p)} \neq 0$, then by continuity, there exists $K > 0$ such that

$$\frac{|\varphi^{(p)}(\xi_k)|}{p!} \geq K$$

for ξ_K sufficiently close to x_*. Thus

$$|x_{k+1} - x_*| \geq K|x_k - x_*|^p$$

which implies that the rate of convergence cannot be higher than p. Thus the rate of convergence is precisely p. □

> **Note.** From the above proof, we expect that close to the fixed point x_*
>
> $$|x_{k+1} - x_*| \approx \frac{|\varphi^{(p)}(x_*)|}{p!}|x_k - x_*|^p,$$

> when
>
> $$\varphi'(x_*) = \varphi''(x_*) = \ldots = \varphi^{(p-1)}(x_*) = 0,$$
>
> but $\varphi^{(p)}(x_*) \neq 0$.

Polynomial interpolation

Idea. Approximate a function $f : \mathbb{R} \to \mathbb{R}$ over $[a, b]$ by a polynomial p such that in **distinct** data points (x_i, y_i), $i = 0, 1, \ldots, n$, the approximation is *exact*, that is,

$$f(x_i) = y_i = p(x_i), \quad \text{for all} \quad i = 0, 1, \ldots, n.$$

We may call x_i *node* and y_i *value*.
We need at least 2 data points. We usually just assume that $x_i \neq x_j$ for $i \neq j$.

> **Note.** Interpolation polynomials are not per se unique, for instance the data $\{(-1, 1), (1, 1)\}$ can be interpolated by $p(x) = 1$, $q(x) = x^2$, or $r(x) = x^4 - x^2 + 1$. However, we will see later that p is the unique interpolation polynomial with $\deg p \leq 1 = n$.

Example. $(1, 2)$, $(2, 3)$, $(3, 6)$, as data set $\{(x_i, y_i) \ : \ i = 0, 1, 2\}$ on the interval $[1, 3]$.
We are looking for a polynomial $p_2(x) = \sum_{j=0}^{2} c_j x^j$, which is chosen to be 2nd order, because we have 3 data points and 3 unknown coefficients.
We can formulate the problem in matrix form:

$$\begin{pmatrix} 1 & x_0 & x_0^2 \\ 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \end{pmatrix} \cdot \begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \end{pmatrix},$$

which is a so-called *Vandermonde matrix* which has determinant

$$\det \begin{pmatrix} 1 & x_0 & x_0^2 \\ 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \end{pmatrix} = \prod_{i<j} (x_j - x_i) \neq 0,$$

and is thus invertible. Here,

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \end{pmatrix} \cdot \begin{pmatrix} c_0 \\ c_1 \\ c_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \\ 6 \end{pmatrix}.$$

As a result, $c_0 = 3$, $c_1 = -2$, and $c_2 = 1$, and thus,

$$p_2(x) = x^2 - 2x + 3.$$

The computational complexity of solving the linear system is $O(n^3)$. We used the natural basis for the polynomials.
What would be the ideal basis?
Definition. (Lagrange basis polynomials) Suppose that $x_i \neq x_j$ if $i \neq j$. We call

$$\phi_i(x) := \prod_{\substack{j=0 \\ i \neq j}}^{n} \frac{x - x_j}{x_i - x_j}$$

the ith *Lagrange basis polynomial*.
The *Lagrange interpolation polynomial* is given by

$$p(x) := \sum_{i=0}^{n} y_i \varphi_i(x).$$

Clearly,

$$\varphi_i(x_j) = \delta_{i,j} := \begin{cases} 1 \text{ if } i = j, \\ 0 \text{ if } i \neq j. \end{cases}$$

> **Example.** $(1,2), (2,3), (3,6)$:
>
> $$\varphi_0(x) = \frac{(x-2)(x-3)}{(1-2)(1-3)}$$
>
> $$\varphi_1(x) = \frac{(x-1)(x-3)}{(2-1)(2-3)}$$
>
> $$\varphi_2(x) = \frac{(x-1)(x-2)}{(3-1)(3-2)}$$
>
> $$p_2(x) = 2\varphi_0(x) + 3\varphi_1(x) + 6\varphi_2(x) = x^2 - 2x + 3.$$

Evaluating the Lagrange polynomials has the computational complexity $O(n^2)$.

Newton's interpolation

Idea. Extend the natural basis:

$$1, \quad x - x_0, \quad (x-x_0)(x-x_1), \quad \ldots, \quad \prod_{j=0}^{n-1}(x - x_j).$$

Definition. Define Newton's interpolation polynomials by

$$p_n(x) = a_0 + a_1(x - x_0) + \ldots + a_n \prod_{j=0}^{n-1}(x - x_j)$$

in such a way that $p_n(x_i) = y_i$.
Clearly,

$$p(x_0) = y_0 \quad \Rightarrow \quad a_0 = y_0,$$

and

$$p(x_1) = a_0 + a_1(x_1 - x_0) = y_1 \quad \Rightarrow \quad a_1 = \frac{y_1 - a_0}{x_1 - x_0}.$$

More generally, we have the lower triangular linear system

$$\begin{pmatrix} 1 & 0 & \cdots & & \\ 1 & x_1 - x_0 & 0 & \cdots & \\ 1 & x_1 - x_0 & (x_2 - x_0)(x_2 - x_1) & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \\ 1 & x_1 - x_0 & (x_2 - x_0)(x_2 - x_1) & \cdots & \prod_{j=0}^{n-1}(x_n - x_j) \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ \vdots \\ a_n \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}.$$

> **Example.**
>
> $$p_2(x) = a_0 + a_1(x-1) + a_2(x-1)(x-2)$$
>
> with the system
>
> $$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 2 & 2 \end{pmatrix} \cdot \begin{pmatrix} a_0 \\ a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} 2 \\ 3 \\ 6 \end{pmatrix}$$
>
> and hence $a_0 = 2$, $a_1 = 1$, and $a_2 = 1$ which yields $p_2(x) = x^2 - 2x + 3$.

Uniqueness

Theorem. Interpolation polynomials with $n+1$ nodes x_i, $i = 0, 1, \ldots, n$ are unique in the class of polynomials q with $\deg q \leq n$.

Proof. (Idea). p_n has at most n roots. Let p_n and q_n be two interpolating polynomials for the same set of data. Then

$$p_n(x_j) = q_n(x_j) = 0, \quad \text{for any} \quad j = 0, 1, \ldots, n.$$

Hence $p_n - q_n$ has $n+1$ distinct roots. As $\deg(p_n - q_n) \leq \max(\deg p_n, \deg q_n) = n$, the only polynomial with $n+1$ roots is the polynomial which is constantly zero. Hence,

$$p_n = q_n.$$

We have used the corollary to the fundamental theorem of algebra which states that every non-constant real polynomial of degree m has at most m zeros. □

Divided differences

Let p be a Newton interpolation polynomial

$$p(x) = a_0 + a_1(x_1 - x_0) + a_2(x - x_2)(x - x_1) + \ldots + a_n \prod_{j=0}^{n=1}(x - x_j).$$

Definition. The *divided difference of order* k, denoted by $f[x_0, x_1, \ldots, x_k]$, is defined as the a_k-coefficient of the Newton interpolation polynomial with data $y_i = f(x_i)$, in other words,

$$f[x_0, x_1, \ldots, x_k] := a_k.$$

Theorem.

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

> **Note.** The recursion terminates because $f[x_i] = y_i$.

> **Example.** ** $(1, 2), (2, 3), (3, 6)$, $p_2(x) = x^2 - 2x + 3$, Newton: $a_0 = 2$, $a_1 = 1$, $a_2 = 1$.

> $f[x_0] = 2 = a_0, f[x_1] = 3, f[x_2] = 6, f[x_0, x_1] = \frac{3-2}{2-1} = 1 = a_1, f[x_1, x_2] = \frac{6-3}{3-2} = 3, f[x_0, x_1, x_2] = \frac{3-1}{3-1} = 1 = a_2.$

Why does this work?

One point: $f[x_j] = f_j = y_j$.
Two points: $f[x_i, x_j] = \frac{f[x_j]-f[x_i]}{x_j-x_i}$ which is the line spanned by the two points (x_i, y_i) and (x_j, y_j), i.e.,

$$y - y_i = \frac{y_j - y_i}{x_j - x_i}(x - x_i).$$

Proof. (Idea). We have three interpolation polynomials p, q, r, where $\deg p = k$, $\deg q = \deg r = k - 1$. p interpolates at $x_0, x_1, \ldots, x_k$, q interpolates at $x_0, x_1, \ldots, x_{k-1}$, and r interpolates at $x_1, \ldots, x_k$.
Claim.

$$p(x) = q(x) + \frac{x - x_0}{x_k - x_0}\underbrace{(r(x) - q(x))}_{=0 \text{ for } x_i}.$$

x_0: $p(x_0) = q(x_0) = f_0$.
$x_1, \ldots, x_{k-1}$: $p(x_i) = q(x_i)$.
x_k: $p(x_k) = q(x_k) + \underbrace{\frac{x_k - x_0}{x_k - x_0}}_{=1}(r(x_k) - q(x_k) = r(x_k)$.
The highest order term has the coefficient

$$\frac{p^{(k)}(x)}{k!} = \frac{r_{k-1} - q_{k-1}}{x_k - x_0},$$

where $r_{k-1} = f[x_1, x_2, \ldots, x_k] = \frac{r^{(k-1)}}{(k-1)!}$ and $q_{k-1} = f[x_0, x_2, \ldots, x_{k-1}] = \frac{q^{(k-1)}}{(k-1)!}$,
which can be proved by the general Leibniz rule. □

Interpolation error

Assume that $f \in C^{n+1}$. We are interested in the local (pointwise) error (*residual*)

$$R(x) := f(x) - p(x),$$

where p is the interpolation polynomial with $\deg p = n$.
Fix data (x_i, y_i), $y_i = f(x_i)$, $i = 0, 1, \ldots, n$, $x_i \neq x_j$, $i \neq j$. Let x' be an distinct extra point.
Define an auxiliary function:

$$h(x) = f(x) - p(x) - cw(x),$$

where

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

and

$$c = \frac{f(x') - p(x')}{w(x')}.$$

We find that

$$h(x_i) = 0 \quad \text{for} \quad i = 0, 1, \dots n.$$

Furthermore,

$$h(x') = f(x') - p(x') - \frac{f(x') - p(x')}{w(x')} w(x') = 0.$$

Hence h has $n + 2$ distinct zeros. By Rolle's theorem (see Differential and Integral Calculus 1), $h^{(n+1)}$ will have at least one zero. Let's call this point ξ.

$$h^{(n+1)}(x) = f^{(n+1)} - \underbrace{p^{(n+1)}(x)}_{=0} - cw^{(n+1)}(x) = f^{(n+1)}(x) - c(n+1)!$$

Hence

$$h^{(n+1)}(\xi) = f^{(n+1)}(\xi) - c(n+1)! = 0$$

and thus

$$c = \frac{f^{(n+1)}(\xi)}{(n+1)!}.$$

We have proved that:

Theorem. If $f \in C^{n+1}$, the residual $R = f - p$ at x has the form

$$R(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!} \prod_{j=0}^{n} (x - x_j).$$

Notice that, in general, R is not a polynomial, as $\xi = \xi(x)$ depends nonlinearly on x.

Note. The constant c is a divided difference:

$$f[x_0, x_1, \dots, x_n, x] = \frac{1}{(n+1)!} f^{(n+1)}(\xi(x)),$$

which follows from the formula for R, $R^{(n+1)} = f^{(n+1)}$ and $R(x_i) = f(x_i)$ for $i = 0, 1, \dots, n$.

Piecewise interpolation

Setup. Fix a bounded interval $[a, b]$ and a step size / mesh

$$h := \frac{b - a}{n}$$

for some $n \in \mathbb{N}$, where n is the number of subintervals.

Idea. Approximate the function on each subinterval using some *low order* interpolation polynomial such that the interpolation function is exact at the nodes.

Piecewise linear case:

$$\ell_i(x) = f(x_{i-1})\frac{x - x_i}{x_{i-1} - x_i} + f(x_i)\frac{x - x_{i-1}}{x_i - x_{i-1}}, \quad x \in [x_{i-1}, x_i].$$

By the residual formula on each subinterval $R(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\prod_{j=0}^{n}(x - x_j)$, we get the interpolation error

$$f(x) - \ell_i(x) = \frac{f''(\xi)}{2!}(x - x_{i-1})(x - x_i),$$

which simplifies if $|f''(x)| \leq M$ by maximization as follows

$$|f(x) - \ell_i(x)| \leq M\frac{h^2}{8}, \quad x \in [x_{i-1}, x_i].$$

> **Note.** If f'' is bounded over the whole interval $[a, b]$ then the error is the same over the whole interval.

Hermite interpolation

Piecewise interpolation by degree 3 polynomials $p_3(x) = \sum_{j=0}^{3} c_j x^j$. As we have 4 coefficients, we need 4 constraints. We demand that not only the function but also the derivatives are exact at the nodes. Let p be a cubic interpolation polynomial on $[x_{i-1}, x_i]$. Then p' is a quadratic polynomial. Recall that $h = x_i - x_{i-1}$.
We have the conditions:

1. $p(x_{i-1}) = f(x_{i-1})$,
2. $p(x_i) = f(x_i)$,
3. $p'(x_{i-1}) = f'(x_{i-1})$,
4. $p'(x_i) = f'(x_i)$.

Set

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

Integrating yields

$$p(x) = -\frac{f'(x_{i-1})}{h}\int_{x_{i-1}}^{x}(t - x_i)\,dt + \frac{f'(x_i)}{h}\int_{x_{i-1}}^{x}(t - x_{i-1})\,dt + \alpha\int_{x_{i-1}}^{x}(t - x_{i-1})(t - x_i)\,dt + C.$$

Plugging in x_{i-1} for x yields that $C = f(x_{i-1})$. Plugging in x_i for x and integrating yields

$$\alpha = \frac{3}{h^3}(f'(x_{i-1}) + f'(x_i)) + \frac{6}{h^3}(f(x_{i-1}) - f(x_i)).$$

Splines

Let us construct a global piecewise interpolation function $s \in C^2$ such that:

1. We do not impose exactness for derivatives.
2. We get a piecewise polynomial construction of cubic interpolation polynomials which is exact and has continuous 1st and 2nd derivatives.

This requires a *global setup*. All coefficients are defined first, only evaluation is piecewise.
Setup. Let $h = x_i - x_{i-1}$ be constant. Define

$$z_i := s''(x_i), \quad i = 1, \ldots, n-1.$$

Now,

$$s''(x) = \frac{1}{h} z_{i-1}(x_i - x) + \frac{1}{h} z_i (x - x_{i-1}).$$

Denote s on the interval $[x_{i-1}, x_i]$ by s_i. Integrating twice yields

$$s_i(x) = \frac{1}{h} z_{i-1} \frac{(x_i - x)^3}{6} + \frac{1}{h} z_i \frac{(x - x_{i-1})^3}{6} + C_i(x - x_{i-1}) + D_i,$$

where

$$s_i'(x) = -\frac{1}{h} z_{i-1} \frac{(x_i - x)^2}{2} + \frac{1}{h} z_i \frac{(x - x_{i-1})^2}{2} + C_i.$$

Set $f_i := f(x_i)$. We get that

$$D_i = f_{i-1} - \frac{h^2}{6} z_{i-1}$$

and

$$C_i = \frac{1}{h} \left[f_i - f_{i-1} + \frac{h^2}{6}(z_{i-1} - z_i) \right].$$

Now s has been defined over all subintervals. However, the z_i are still unknown!
Using the condition for continuity of the derivatives $s_i'(x_i) = s_{i+1}'(x_i)$ for all i yields

$$\frac{h}{2} z_i + \frac{1}{h} \left[(f_i - f_{i-1}) + \frac{h^2}{6}(z_{i-1} - z_i) \right] = -\frac{h}{2} z_i + \frac{1}{h} \left[(f_{i+1} - f_i) + \frac{h^2}{6}(z_i - z_{i+1}) \right],$$

for $i = 1, \ldots, n-1$.
In fact, this constitutes a triangular system:

$$\frac{2h}{3} z_i + \frac{h}{6} z_{i-1} + \frac{h}{6} z_{i+1} = \frac{1}{h}(f_{i+1} - 2f_i + f_{i-1}) =: b_i.$$

The values z_0 and z_n at the interval boundary have to be moved to the right hand side, and thus:

$$b_1 := \frac{1}{h}(f_2 - 2f_1 + f_0) - \frac{h}{6} z_0$$

and

$$b_{n-1} := \frac{1}{h}(f_n - 2f_{n-1} + f_{n-2}) - \frac{h}{6} z_n.$$

z_0 and z_n can be chosen freely, for example to force that the 1st derivative of the spline is exact at the interval boundary points. If $z_0 = z_n = 0$, s is called a *natural spline*.

Bézier curves

Bézier curves are parametrized curves in $\mathbb{R}^2$, that is, $\mathbf{r}(t) = x(t)\mathbf{i} + y(t)\mathbf{j}$,

where $\mathbf{i} := \begin{pmatrix}1\\0\end{pmatrix}, \mathbf{j} := \begin{pmatrix}0\\1\end{pmatrix}$ and $x, y : [0,1] \to \mathbb{R}$.

Bernstein polynomials

Define the *Bernstein polynomial* $B_k^n(t), t \in [0,1], n \in \mathbb{N} \cup \{0\}, k = 0, \ldots, n$, by

$$B_k^n(t) := \binom{n}{k} t^k (1-t)^{n-k}.$$

Properties:

1. $\sum_{k=0}^n B_k^n(t) = 1 \, (= (t + 1 - t)^n)$,
2. $0 \leq B_k^n(t) \leq 1$,
3. $B_0^n(0) = B_n^n(1) = 1$, otherwise, if $k \neq 0$, $B_k^n(0) = 0$ and if $k \neq n$, $B_k^n(1) = 0$.

We have the combinatorial rule:

$$B_k^n(t) = (1-t)B_k^{n-1}(t) + tB_{k-1}^{n-1}(t).$$

Bézier curves

Fix a finite set $X = \{x_0, x_1 \ldots, x_k\}$ of control points $x_i \in \mathbb{R}^n$.

Definition. The *convex hull* of X is defined by

$$\mathrm{conv}(X) = \left\{ y \in \mathbb{R}^n \, : \, y = \sum_{k=0}^k \lambda_i x_i, \; \lambda_i \in [0,1], \; \sum_{k=0}^k \lambda_i = 1 \right\}.$$

Definition. The Bézier curve β^n is defined,

$$\beta^n(t) = \sum_{k=0}^n x_k B_k^n(t).$$

Sanity check: $t = 0$, $B_k^n(0) = 0$, except $B_0^n(0) = 1$.
$\Rightarrow \beta^n(0) = x_0$,
$\Rightarrow \beta^n(1) = x_n$.
We get closed curves if $x_0 = x_n$.

What about the continuous tangents?
Recall that $\binom{n}{k} = \frac{n!}{k!(n-k)!}$.

$$\frac{d}{dt} B_k^n(t) = \binom{n}{k} \left(k t^{k-1} (1-t)^{n-k} - (n-k) t^k (1-t)^{n-k-1} \right)$$

$$= n \left[\frac{(n-1)!}{(k-1)!(n-k)!} t^{k-1}(1-t)^{n-k} - \frac{(n-1)!}{(k)!(n-k-1)!} t^k (1-t)^{n-k-1} \right]$$

$$= n \left(B_{k-1}^{n-1}(t) - B_k^{n-1}(t) \right).$$

Therefore,

$$\frac{d}{dt}\beta^n(t) = n\sum_{k=0}^{n}\left(B_{k-1}^{n-1}(t) - B_k^{n-1}(t)\right)x_k$$

$$= n\left[\sum_{k=1}^{n} B_{k-1}^{n-1}(t)x_k - \sum_{k=0}^{n-1} B_k^{n-1}(t)x_k\right]$$

$$= n\left[\sum_{k=0}^{n-1} B_k^{n-1}(t)x_{k+1} - \sum_{k=0}^{n-1} B_k^{n-1}(t)x_k\right]$$

$$= \underbrace{n\sum_{k=0}^{n-1}(x_{k+1} - x_k)B_k^{n-1}(t)}_{\text{Bézier}}.$$

Hence, for the closed curves:

$$\begin{cases} \frac{d}{dt}\beta^n(0) = n(x_1 - x_0), \\ \frac{d}{dt}\beta^n(1) = n(x_n - x_{n-1}). \end{cases}$$

For smoothness, we need that $(x_1 - x_0) \parallel (x_n - x_{n-1})$, i.e., $x_1 - x_0$ and $x_n - x_{n-1}$ are parallel.

Lifting

Control points define the curve but the converse is not true.
Consider:

$$\beta^n(t) = \sum_{k=0}^{n} B_k^n(t)x_k = \sum_{k=0}^{n+1} B_k^{n+1}(t)y_k = \alpha^{n+1}(t).$$

Let us use the convention $x_{-1} = x_{n+1} = 0$. We get the condition

$$y_k = \left(1 - \frac{k}{n+1}\right)x_k + \frac{k}{n+1}x_{k-1}.$$

De Casteljau's algorithm

For control points $x_0, x_1, \ldots, x_n$ the algorithm of De Casteljau is as follows:

1. Define constant curves $\beta_i^0(t) = x_i$.
2. Set

$$\beta_i^r(t) = (1-t)\beta_i^{r-1}(t) + t\beta_{i+1}^{r-1}(t), \quad r = 1, \ldots, n, \quad i = 0, \ldots, n-r.$$

Th

The algorithm terminates at $\beta_0^n(t)$ and has $\binom{n}{2}$ operations.

There is also a reverse algorithm for splitting Bézier curves.

Numerical integration

Integration schemes are called quadratures. Therefore, numerical integration methods are simply called numerical quadratures.

> **Note.** There are no simple integration schemes in higher dimensions. Already 2D-cases are complicated.

Newton-Cotes quadrature rules

Let $f : [a, b] \to \mathbb{R}$.
Idea. Approximate

$$\int_a^b f(x)\,dx =: I$$

by the integral of an interpolation polynomial

$$I \approx \int_a^b p_k(x)\,dx =: Q(p_k),$$

where p_k is an interpolant of f over $[a, b]$.

Lagrange:

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n} f(x_i) \int_a^b \left(\prod_{\substack{j=0 \\ i \neq j}} \frac{x - x_j}{x_i - x_j} \right) dx.$$

Let $n = 1$:

$$p_1(x) = f(a)\frac{x-b}{a-b} + f(b)\frac{x-a}{b-a},$$

so

$$\int_a^b f(x)\,dx \approx \int_a^b p_1(x)\,dx = \frac{b-a}{2}[f(a) + f(b)].$$

$\Rightarrow$ Trapezoidal rule!
Error formulation:

$$\int_a^b f(x)\,dx - \int_a^b p_1(x)\,dx = \frac{1}{2}\int_a^b f''(\xi)(x-a)(x-b)\,dx.$$

Now, $(x-a)(x-b) < 0$ for $x \in (a, b)$.
Therefore, by the mean value theory of integration,

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

$$= -\frac{1}{12}(b-a)^3 f''(\eta).$$

Composite rule: $h = \frac{b-a}{n}$, $x_i = a + ih$, $i = 0, \ldots, n$.

$$\int_a^b f(x)\,dx \approx \frac{h}{2}\left[f(x_0) + 2f(x_1) + \ldots + 2f(x_{n-1}) + f(x_n)\right].$$

Total error: $O(h^2) \sim O(\frac{1}{n^2})$. We say that the method is quadratic.

Let $n = 2$. When is a method *exact* for degree 2 (or lower)?

> **Note.** In this context, exactness means, that the integral and the method give the exact same result for polynomials of certain order.

$$\int_a^b f(x)\,dx = A_1 f(a) + A_2 f\left(\frac{a+b}{2}\right) + A_3 f(b),$$

where we call the A_i *weights*.

$$\int_a^b 1\,dx = b - a \quad \Rightarrow A_1 + A_2 + A_3 = b - a.$$

$$\int_a^b x\,dx = \frac{b^2 - a^2}{2} \quad \Rightarrow A_1 a + A_2\left(\frac{a+b}{2}\right) + A_3 b = \frac{b^2 - a^2}{2}.$$

$$\int_a^b x^2\,dx = \frac{1}{3}(b^3 - a^3) \quad \Rightarrow A_1 a^2 + A_2\left(\frac{a+b}{2}\right)^2 + A_3 b^2 = \frac{1}{3}(b^3 - a^3).$$

Thus,

$$A_1 = A_3 = \frac{b-a}{6}$$

and

$$A_2 = \frac{4(b-a)}{6}.$$

As integrals and the methods are linear, this extends to all polynomials of $\deg \leq 2$.

This is the so-called *Simpson's rule*:

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

The associated *composite rule* becomes:

$$\int_a^b f(x)\,dx \approx \frac{h}{6}\left[f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \ldots + 4f(x_{n-1}) + f(x_n)\right].$$

Error for $n = 2$:

$$\frac{1}{4!5!}(b-a)^5 f^{(4)}(\eta).$$

Error for the composite: $O(h^4)$. The method is exact for cubic polynomials!

Orthogonal polynomials

Define the *inner product* of two real-valued polynomials on $[a, b]$ (depends on a and b!) by:

$$\langle p, q \rangle = \int_a^b p(x) q(x)\, dx.$$

The associated *norm* on $[a, b]$ is given by

$$\|q\| := \left(\int_a^b |q(x)|^2\, dx \right)^{1/2}.$$

Definition. Two non-zero polynomials are said to be *orthogonal* on $[a, b]$ if their inner product is zero. They are said to be *orthonormal* if they are orthogonal and have both norm 1.
In other words, orthogonality: $\langle p, q \rangle = 0$, then we write $p \perp q$.
Orthonormality: $p \perp q$ and $\langle p, p \rangle = 1 = \langle q, q \rangle$.

Gram-Schmidt (GS) procedure.
Idea. Transform a basis to an orthogonal one:

$$\{1, x, x^2, \ldots, x^k, \ldots\} \longrightarrow \{q_0, q_1, \ldots, q_k, \ldots\}, \quad \text{orthonormal.}$$

> **Note.** The GS procedure depends on the inner product, and thus, here, on a and b.

The elements of the orthonormal basis are called *orthogonal polynomials*.

1. $$q_0 = \frac{1}{\|1\|} = \frac{1}{\left(\int_a^b 1^2\, dx \right)^{1/2}} = \frac{1}{\sqrt{b-a}}.$$

2. For $j = 1, 2, \ldots$,

$$\tilde{q}_j(x) = x q_{j-1}(x) - \sum_{i=0}^{j-1} \langle x q_{j-1}, q_i \rangle q_i(x),$$

and

$$q_j(x) := \frac{\tilde{q}_j}{\|\tilde{q}_j\|}.$$

The new basis is (pairwise) orthonormal!
Above, as usually, we denote the polynomial $p(x) = x$ with the symbol x.

Observation. By bilinearity, q_{j-1} is orthogonal to all polynomials of $\deg \leq j - 2$.
Thus,

$$\langle x q_{j-1}, q_i \rangle = \langle q_{j-1}, x q_i \rangle = 0, \quad i \leq j - 3.$$

As a consequence, the GS procedure reduces to

$$\tilde{q}_j(x) = x q_{j-1}(x) - \langle x q_{j-1}, q_{j-1} \rangle q_{j-1}(x) - \langle x q_{j-1}, q_{j-2} \rangle q_{j-2}(x)$$

which is a three-term recurrence rule!

> Note. The trick $\langle x q_{j-1}, q_i \rangle = \langle q_{j-1}, x q_i \rangle$ relies heavily on the fact that the inner product is defined by an integral and that we are dealing with polynomials. The GS procedure works generally in pre-Hilbert spaces, however, then we do not

> expect this kind of simplification.

Claim. The GS procedure works.
Proof.

$$\langle \tilde{q}_j, q_{j-1} \rangle = \langle x q_{j-1}, q_{j-1} \rangle - \sum_{i=0}^{j-1} \langle x q_{j-1}, q_i \rangle \underbrace{\langle q_i, q_{j-1} \rangle}_{=0 \text{ except when } i=j-1, \text{ then it is } =1}$$

$$= \langle x q_{j-1}, q_{j-1} \rangle - \langle x q_{j-1}, q_{j-1} \rangle = 0.$$

□

Gauss quadrature

Idea. Choose the nodes and the weights simultaneously.
One interval:

$$\int_a^b f(x)\,dx = A_0 f(x_1) + A_1 f(x_1),$$

with *weights* A_0, A_1, and *nodes* x_0, x_1, for $n = 1$, this is a $(n+1) = 2$-rule.
The coefficients are determined by the usual process:

$$\int_a^b 1\,dx = b - a = A_0 + A_1.$$

$$\int_a^b x\,dx = \frac{b^2 - a^2}{2} = A_0 x_0 + A_1 x_1.$$

$$\int_a^b x^2\,dx = \frac{1}{3}(b^3 - a^3) = A_0 x_0^2 + A_1 x_1^2.$$

The resulting system is nonlinear!

Let us use the orthogonal polynomials in the following way.

Theorem. Let $x_0, x_1, \ldots, x_n$ be the roots of an orthogonal polynomial q_{n+1} on $[a, b]$ of degree n.
Then

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n} A_i f(x_i),$$

where

$$A_i := \int_a^b \varphi_i(x)\,dx, \quad \varphi_i(x) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{x - x_j}{x_i - x_j},$$

is exact for all polynomials of degree $2n+1$ or less.

Proof. Let f be a polynomial with $\deg f = 2n+1$. By the polynomial division algorithm,

$$f = q_{n+1}p_n + r_n,$$

where $\deg p_n \leq n$ and $\deg r_n \leq n$. Then,

$$f(x_i) = \underbrace{q_{n+1}(x_i)}_{=0}p_n(x_i) + r_n(x_i) = r_n(x_i).$$

Integrate,

$$\int_a^b f(x)\,dx = \underbrace{\int_a^b q_{n+1}(x)p_n(x)\,dx}_{=\langle q_{n+1},p_n\rangle=0} + \int_a^b r_n(x)\,dx$$

$$= \int_a^b r_n(x)\,dx = \sum_{i=0}^{n} A_i r_n(x_i) = \sum_{i=0}^{n} A_i f(x_i).$$

Because r_n can be interpolated exactly with $n+1$ nodes. The last equality follows from the reasoning before. □

We can extend the notion of orthogonal polynomials to so-called *weighted orthogonal polynomials* with respect to the inner product

$$\langle p, q\rangle_w = \int_a^b p(x)q(x)w(x)\,dx,$$

where w is a positive *weight function*.

One (mathematical) advantage: Works also on $\mathbb{R} = (-\infty, \infty)$.

> **Example.** If $w(x) = e^{-x}$, we get the so-called *Laguerre polynomials*. If $w(x) = e^{-\frac{x^2}{2}}$, we get the so-called *Hermite polynomials*, which are meaningful in probability theory (the weight is the density of the Gaussian normal distribution up to multiplication by a normalization constant).

Theorem. The previous theorem holds a $\langle\cdot,\cdot\rangle_w$-orthogonal polynomial q_{n+1} with

$$A_i := \int_a^b \varphi_i(x)w(x)\,dx, \quad \varphi_i(x) = \prod_{\substack{j=0\\ j\neq i}}^{n} \frac{x - x_j}{x_i - x_j}.$$

Error formula. $(n+1)$-point rule with nodes $x_0, x_1, \ldots, x_n$:

$$\text{error} = \frac{f^{(2(n+1))}(\xi(x))}{(2(n+1))!}\prod_{\substack{j=0\\ j\neq i}}^{n}(x - x_j)^2.$$

Where does the square come from?
We assume that the derivatives of f are continuous, therefore Hermite interpolation is the natural choice.

Example. Gauss rule on $[-1, 1]$, $n = 1$. Notice, since we only want the roots, there is no need to normalize $\tilde{q}_i$, $i = 0, 1, 2$.
GS: $\tilde{q}_0 = 1$.

$$\tilde{q}_1 = x \cdot 1 - \frac{\langle x, 1 \rangle}{\langle 1, 1 \rangle} \cdot 1 = x - \frac{\int_{-1}^{1} x \, dx}{\int_{-1}^{1} 1 \, dx} \cdot 1 = x,$$

where $\langle 1, 1 \rangle = 2$, and $\langle x, 1 \rangle = 0$.

$$\tilde{q}_2 = x \cdot x - \frac{\langle x^2, 1 \rangle}{\langle 1, 1 \rangle} \cdot 1 - \frac{\langle x^2, x \rangle}{\langle x, x \rangle} \cdot x = x^2 - \frac{1}{3},$$

where $\langle x, x \rangle = \langle x^2, 1 \rangle = \frac{2}{3}$ and $\langle x^2, x \rangle = 0$.
The resulting orthogonal polynomials on $[-1, 1]$ are called *Legendre polynomials*.
$\tilde{q}_2$ (and q_2) has the roots $x = \pm \frac{1}{\sqrt{3}}$.
The associated Newton quadrature rule is:

$$\int_{-1}^{1} f(x) \, dx = A_0 f\left(-\frac{1}{\sqrt{3}}\right)) + A_1 f\left(\frac{1}{\sqrt{3}}\right).$$

Let us check exactness:

$$\int_{-1}^{1} 1 \, dx = 2 = A_0 + A_1.$$

$$\int_{-1}^{1} x \, dx = 0 = \frac{-A_0}{\sqrt{3}} + \frac{A_1}{\sqrt{3}}.$$

From this, we obtain easily that $A_0 = A_1 = 1$. This weights could of course also been determined by integrating the Lagrange polynomials over $[-1, 1]$.

$$\int_a^b x^2 \, dx = \frac{2}{3} = 1 \cdot \left(-\frac{1}{\sqrt{3}}\right)^2 + 1 \cdot \left(\frac{1}{\sqrt{3}}\right)^2.$$

$$\int_a^b x^3 \, dx = 0 = 1 \cdot \left(-\frac{1}{\sqrt{3}}\right)^3 + 1 \cdot \left(\frac{1}{\sqrt{3}}\right)^3.$$

Thus, the Newton quadrature is indeed exact up to degree $2n + 1 = 3$!

Probabilistic examples

Monte Carlo integration

Let X_i, $i \in \mathbb{N}$, be i.i.d. (independent, identically distributed) random variables with *mean* μ and *variance* σ^2. Then for the arithmetic mean (also called *Césaro sum*)

$$A_N := \frac{1}{N} \sum_{i=1}^{N} X_i.$$

By the law of large numbers, we have almost surely

$$\lim_{N\to\infty} A_N = \mu.$$

We have that

$$\operatorname{var}(A_N) = \frac{1}{N^2}\sum_{i=1}^{N}\operatorname{var}(X_i) = \frac{\sigma^2}{N}.$$

In order to get the right unit, we have to consider the standard deviation

$$\sigma(A_N) = \frac{\sigma}{\sqrt{N}}.$$

Consequence:
If our integration problem can be cast into an averaging problem, the convergence rate will be $O(\frac{1}{\sqrt{N}})$.

> **Note.** The rate is independent of the spatial dimension.

Example. Estimating the value of π. The area of a circle is $A = \pi r^2$. Set $r = 1$. Consider the box $V = [-1, 1] \times [-1, 1]$ with volume $|V| = 4$. The ratio of the areas of circle enclosed by the box and the enclosing box is $\frac{\pi}{4}$. Let

$$g_i = \begin{cases} 1, & \text{if } p \text{ is inside } A, \\ 0, & \text{otherwise.} \end{cases}$$

Idea: Let us sample the points p_i uniformly from V. In the limit, the number of "hits" over all samples tends to the ratio of the areas!

Buffon's needle

Suppose we doing a random experiment with a large number of needles of same length L that we throw on the floor, which has parallel strips drawn on it which have all the same distance D to their neighboring strip.

What is the probability that a dropped needle intersects with a line?

Let y be the distance from the center of the needle to the closest line and let θ be the acute angle of the intersection point.

We assume, for simplicity, $L = D = 1$.

Both y and θ are random variables with distribution

$$y \sim \operatorname{Unif}\left(\left[0, \frac{1}{2}\right]\right),$$

where $y = 0$ means that the needle is centered on a line and $y = \frac{1}{2}$ means that y is perfectly centered between two lines.

$$\theta \sim \operatorname{Unif}\left(\left[0, \frac{\pi}{2}\right]\right),$$

where $\theta = 0$ means that the needle is parallel to the lines and $\theta = \frac{\pi}{2}$ means that the needle is perpendicular to the lines.

We may assume that y and θ are independent random variables (*why?*).

By the law of sines, the condition for intersection with a line is

$$2y \leq \sin\theta.$$

The joint probability distribution of two independent random variables is the product of the respective distributions on $[0, \frac{\pi}{2}] \times [0, \frac{1}{2}]$. Determining the probability requires calculation of the ratio of the area of the condition of intersection in relation to the total area $\frac{\pi}{4}$.

The condition is fulfilled by

$$\frac{1}{2}\int_0^{\frac{\pi}{2}} \sin\theta \, d\theta = \frac{1}{2}.$$

Thus the probability of intersection is

$$\frac{1}{2}\Big/\frac{\pi}{4} = \frac{2}{\pi}.$$

By the law of large numbers, the ratio of needles intersecting the lines with all needles converges to $\frac{2}{\pi} \approx 0.6366$. Hence, we have found yet another probabilistic algorithm to determine the value of π.

Initial value problems (IVPs)

Problem. (Not necessarily linear) ordinary differential equation (ODE), with initial value y_0 at initial time t_0 up to a finite time horizon $T > t_0$:

$$\begin{cases} y'(t) = f(t, y(t)), \\ y(t_0) = y_0. \end{cases}$$

Assumptions. Existence and uniqueness of the solutions are understood (by e.g. Picard iteration).
Let us assume that f is continuous as a function from $[t_0, T] \times \mathbb{R} \to \mathbb{R}$ and Lipschitz continuous in the following sense: There exists $L > 0$ such that for every $y, z \in \mathbb{R}$, $t \in [t_0, T]$,

$$|f(t, y) - f(t, z)| \leq L|y - z|.$$

Euler's method

Fix a constant step size $h > 0$.

1. $y_0 := y(t_0)$.
2. $t_k := t_{k-1} + h$ and $y_{k+1} = y_k + hf(t_k, y_k)$, $k = 0, 1, 2 \ldots,$.

Applying Taylor's formula yields:

$$y(t_{k+1}) = y(t_k) + hy'(t_k) + \frac{h^2}{2}y''(\xi_k) = y(t_k) + hf(t_k, y(t_k)) + \frac{h^2}{2}y''(\xi_k),$$

with $\xi_k \in [a, b]$.

We shall deal with two types of error:

(A) truncation error (local),
(B) global error.

Notation. $y(t_k)$ denotes the exact solution to the IVP at $t = t_k$, whereas y_k denotes the result of the method at step k.

For Euler, we get that

$$\frac{y_{k+1}-y_k}{h} = f(t_k, y_k) + \underbrace{\frac{h}{2}y''(\xi_k)}_{\text{local error } O(h)}.$$

Hence the Euler method is locally (in each point) or order 1.
The method is consistent:

$$\lim_{h\to 0}\frac{y_{k+1}-y_k}{h} = y'(t_k) = f(t_k, y(t_k)).$$

> Note that k depends on h, which we omit in the notation.

What about the global error, that is, uniform convergence on $[t_0, T]$?

$$\max|y(t_k) - y_k| \to 0$$

as $h \to 0$?

Theorem. Assume the following:

1. f is Lipschitz in the second component.
2. $\max|y''(t_k)| \leq M$ for some global constant $M > 0$.
3. $y_0 \to y(t_0)$ as $h \to 0$.

Then Euler's method is uniformly convergent to the exact solution on $[t_0, T]$ and the global error is $O(h)$.

Proof. Set $d_j := y(t_j) - y_j$.
By Taylor and Euler:

$$d_{k+1} = d_k + h[f(t_k, y(t_k)) - f(t_k, y_k)] + \frac{h^2}{2}y''(\xi_k).$$

We get that

$$|d_{k+1}| \leq |d_k| + hL|d_k| + \frac{h^2}{2}M = (1+hL)|d_k| + \frac{h^2}{2}M.$$

We shall need a lemma on recursive inequalities.

Lemma. If for $\alpha, \beta > 0$,

$$\gamma_{k+1} \leq (1+\alpha)\gamma_k + \beta,$$

then

$$\gamma_n \leq e^{n\alpha}\gamma_0 + \frac{e^{n\alpha}-1}{\alpha}\beta.$$

Proof. Iterating the inequality yields

$$\gamma_n \leq (1+\alpha)^2\gamma_{n-2} + [(1+\alpha)+1]\beta \leq (1+\alpha)^n\gamma_0 + \left[\sum_{j=0}^{n-1}(1+\alpha)^j\right]\beta.$$

Note that by the Taylor formula,

$$e^\alpha = e^0 + e^0\alpha + \frac{\alpha^2}{2}e^\xi = 1 + \alpha + \frac{\alpha^2}{2}e^\xi$$

with $\xi \in [0, \alpha]$.
Hence

$$1 + \alpha \leq 1 + \alpha + \underbrace{\frac{\alpha^2}{2} e^{\xi}}_{>0} = e^{\alpha}.$$

And thus,

$$\gamma_n \leq e^{n\alpha}\gamma_0 + \frac{1 - (1+\alpha)^n}{1 - (1-\alpha)}\beta \leq e^{n\alpha}\gamma_0 + \frac{e^{n\alpha} - 1}{\alpha}\beta.$$

□

Returning to the proof of the theorem, we get that

$$|d_k| \leq e^{khL}|d_0| + \frac{e^{khL} - 1}{Lh}\frac{h^2}{2}M.$$

Now for $kh \leq T - t_0$,

$$\max_k |d_k| \leq e^{L(T-t_0)}|d_0| + \frac{e^{L(T-t_0)} - 1}{L}\frac{h}{2}M.$$

$|d_0| \to 0$ as $h \to 0$ by the 3rd assumption. Hence, the method converges uniformly with linear convergence rate. □

Heun's method

Idea. Predictor –corrector.

$$\tilde{y}_{k+1} = y_k + hf(t_k, y_k) \quad \text{(prediction)}$$

$$y_{k+1} = y_k + \frac{h}{2}[f(t_k, y_k) + f(t_{k+1}, \tilde{y}_{k+1})] \quad \text{(correction)}$$

Explicit vs. Implicit

Quadrature. Integral formulation of the IVP:

$$y(t+h) = y(t) + \int_t^{t+h} f(s, y(s))\, ds,$$

apply your favorite quadrature rule, for instance:

$$\frac{h}{2}[f(t, y(t)) + f(t+h, y(t+h))] + O(h^3).$$

Combined, we get:

$$y_{k+1} = y_k + \frac{h}{2}[f(t_k, y_k) + f(t_{k+1}, y_{k+1})].$$

This method is *implicit*. Every step requires a solution of a nonlinear (fixed point) problem.

Heun's method and Euler's method are *explicit*.

Linear multistep methods

$$y(t_{k+1}) = y(t_k) + \int_{t_k}^{t_{k+1}} f(s, y(s))\, ds.$$

Adams-Bashforth (explicit)

Interpolation nodes $t_k, t_{k-1}, \ldots, t_{k-m+1}$. Polynomial p_{m-1}.

$$y_{k+1} = y_k + \int_{t_k}^{t_{k+1}} p_{m-1}(s)\, ds = y_k + h \sum_{l=0}^{m-1} b_l f(t_{k-l}, y_{k-l}),$$

where

$$b_l = \frac{1}{h} \int_{t_k}^{t_{k+1}} \left(\prod_{\substack{j=0 \\ j \neq l}}^{m-1} \frac{s - t_{k-j}}{t_{k-l} - t_{k-j}} \right) ds.$$

The truncation error is $O(h^m)$.

What methods can be recovered?
For $m = 1, l = 0$, we get $b_0 = 1$ and

$$y_{k+1} = y_k + h f(t_k, y_k),$$

and thus Euler's method!

Adams-Moulton (implicit)

Add t_{k+1} as an interpolation node. Interpolation polynomial q_m.

$$y_{k+1} = y_k + \int_{t_k}^{t_{k+1}} q_m(s)\, ds = y_k + h \sum_{l=0}^{m} c_l f(t_{k+1-l}, y_{k+1-l}),$$

where

$$c_l = \frac{1}{h} \int_{t_k}^{t_{k+1}} \left(\prod_{\substack{j=0 \\ j \neq l}}^{m} \frac{s - t_{k+1-j}}{t_{k+1-l} - t_{k+1-j}} \right) ds.$$

The truncation error is $O(h^{m+1})$.

For $m = 0, l = 0$, we get $c_0 = 1$ and

$$y_{k+1} = y_k + h f(t_{k+1}, y_{k+1}),$$

which is the so-called *backward Euler method* (also called *implicit Euler method*)!

Why do we need multistep methods?

Bad Example.

$$y' = -15y, \quad y(0) = 1.$$

Exact solution $y(t) = e^{-15t}$. For $h = \frac{1}{4}$ Euler's method oscillates about zero.
Adams-Moulton (trapezoidal method) works!

General form.

The general form of a *linear multistep method* is given for $s \in \mathbb{N}$ by

$$\sum_{j=0}^{s} a_k y_{n+j} = h \sum_{j=0}^{s} b_j f(t_{n+j}, y_{n+j}),$$

where $a_s = 1$ (*normalization*) and the coefficients $a_0, \ldots, a_{s-1}$ and $b_0, \ldots, b_s$ determine the method.

The method is called *explicit* if $b_s = 0$, and *implicit* otherwise.

We call the multistep method *consistent* if the truncation error is $O(h)$ or better.

Theorem. The linear multistep method is consistent if and only if

$$\sum_{k=0}^{s-1} a_k = -1$$

and

$$\sum_{k=0}^{s} b_k = s + \sum_{k=0}^{s-1} k a_k.$$

If, moreover,

$$q \sum_{k=0}^{s} k^{q-1} b_k = s^q + \sum_{k=0}^{s-1} k^q a_k,$$

for every $q = 1, \ldots, p$ then the truncation error is $O(h^{1+p})$.
(Here, we follow the non-standard convention that $k^0 = 0$ if $k = 0$).

See [Ernst Hairer, Gerhard Wanner, Syvert P. Nørsett. Solving Ordinary Differential Equations I: Nonstiff Problems. Springer, 2nd ed., 1993] for a proof.

The stability of multistep methods depends on the convergence of the initial values $y_1, \ldots, y_{s-1}$ to y_0 as $h \to 0$. The second condition yields a global error $O(h^p)$.

Example. $m = 1$, $a_0 + a_1 = 0$, $0 \cdot a_0 + 1 \cdot a_1 = b_1$, and by the normalization assumption, $a_1 = 1$, $a_0 = -1$, $b_1 = 1$, and thus we get,

$$y_{k+1} = y_k + h f(t_{k+1}, y_{k+1})$$

backward Euler!

Example. (Good bad example)

$$y_{k+2} - 3y_{k+1} + 2y_k = h\left[\frac{13}{12}f(t_{k+2}, y_{k+2}) - \frac{5}{3}f(t_{k+1}, y_{k+1}) - \frac{5}{12}f(t_k, y_k)\right].$$

This method satisfies the first condition. For the second condition, $p = q = 1$, we get that $-1 \neq \frac{13}{12}$, so the second condition is not satisfied and the method is not stable. Indeed,
for

$$y' = 0, \quad y(0) = 1,$$

which has the explicit solution $y(t) = 1$, we consider a small perturbation of the initial value, $\delta > 0$,
so that

$$y_0 = 1, \quad y_1 = 1 + \delta,$$

$$y_2 = 3y_1 - 2y_0 = 1 + 3\delta,$$

$$\dots$$

$$y_k = 3y_{k-1} - 2y_{k-2} = 1 + (2^k - 1)\delta.$$

Hence, for $\delta \sim 2^{-53}$, and $k = 100$, we get the error $\sim 2^{47}$.
We note, however, that the method is consistent and the exact differential equation is stable (in the mathematical sense), and the perturbation $y_\delta(t) = 1 + \delta$ converges uniformly to the exact solution $y(t) = 1$ as $\delta \to 0$.

Example. (Effect of rounding error)

Returning to the proof of convergence for Euler, for the rounding error $\delta > 0$,

$$|d_{k+1}| \leq (1 + hL)|d_k| + \delta,$$

we get,

$$|d_{k+1}| \leq e^{L(T-t_0)} \underbrace{|d_0|}_{\text{initial error or uncertainty}} + \underbrace{\frac{e^{L(T-t_0)} - 1}{Lh}\delta}_{\text{dominant term, if } h \text{ is sufficiently small}}$$

Gradient descent

The following algorithm is widely used in machine learning, together with its probabilistic counterpart, the stochastic gradient decent (SDG).

The goal is to find the minima of a function

$$f : D \to \mathbb{R}, \quad D \subset \mathbb{R}^d,$$

which is assumed suitably regular, e.g. $f \in C^1(D \setminus \partial D)$.

Gradient descent algorithm.

For simplicity, assume that $0 \in D$.

For $k = 0, \dots, N$, where $N \in \mathbb{N}$, iterate:

1. Fix initial point $w_0 := 0 \in D$.
2. w_{k+1} is obtained by moving away from w_k in the opposite direction of the gradient of f at w_k, with step size $\eta_{k+1} > 0$, more precisely,

$$w_{k+1} := w_k - \eta_{k+1} \nabla f(w_k), \quad k = 0, 1, \dots, N.$$

The constants η_k are called *learning rates*.

3. After the Nth step, we may choose different outputs, as e.g. just $\bar{w}_N := w_N$ or

$$\bar{w}_N := \arg\min_{k=0,\dots,N} f(w_k).$$

   Less obviously, one may also choose

$$\bar{w}_N := \frac{1}{N+1} \sum_{k=0}^{N} w_k,$$

   which is particularly useful for the SDG.

Definition. $f : \mathbb{R}^d \to \mathbb{R}$ is called convex, if for every $\lambda \in [0,1]$, $x, y \in \mathbb{R}^d$,

$$f(\lambda x + (1 - \lambda y) \leq \lambda f(x) + (1 - \lambda) f(y).$$

> Note that if f is convex,
>
> $$f\left(\sum_{i=0}^{N} \lambda_i x_i \right) \leq \sum_{i=0}^{N} \lambda_i f(x_i),$$
>
> for every $x_0, x_1, \dots, x_N \in \mathbb{R}^d$, whenever $\lambda_i \in [0,1]$ satisfy $\sum_{i=0}^{N} \lambda_i = 1$.

Continuously differentiable convex functions $f : \mathbb{R}^d \to \mathbb{R}$ enjoy the so-called *subgradient property*, i.e.

$$f(x) - f(y) \leq \langle \nabla f(x), x - y \rangle, \quad x, y \in \mathbb{R}^d,$$

where $\langle \cdot, \cdot \rangle$ denotes the Euclidean scalar product.

Theorem. Let $f : \mathbb{R}^d \to \mathbb{R}$ be convex, continuously differentiable and L-Lipschitz continuous, i.e.,

$$|f(x) - f(y)| \leq L|x - y|, \quad x, y \in \mathbb{R}^d.$$

Let $R > 0$, $N \in \mathbb{N}$. Set

$$m := \min_{|w| \leq R} f(w), \quad \eta_k := \eta := \frac{R}{L\sqrt{N+1}}.$$

Then for

$$\bar{w}_N := \frac{1}{N+1} \sum_{k=0}^{N} w_k,$$

we have that

$$f(\bar{w}_N) - m \leq \frac{RL}{\sqrt{N+1}}.$$

> **Note.** The point $\bar{w}_N$ is doubly dependent on N; not only through the number of steps, but also through the choice of η.

Remark.

1. Assume that f has a global minimum in $w_* \in \mathbb{R}^d$. Then, the above result ensures the convergence of $f(\bar{w}_N)$ to the minimum $f(w_*)$, provided that $R \geq |w_*|$. Indeed, the claimed estimate, together with

   $$f(\bar{w}_N) - f(w_*) \geq 0$$

   yields

   $$|f(\bar{w}_N) - f(w_*)| \leq \frac{RL}{\sqrt{N+1}}.$$

2. It is not guaranteed that $\{\bar{w}_N\}_{N\in\mathbb{N}}$ converges to w_* unless w_* is the unique minimizer (e.g. if f is so-called *strictly convex* (i.e., the inequality in the definition of convexity is strict for $\lambda \in (0,1)$).
3. The convergence rate is sublinear unless f is so-called *strongly convex* (i.e., $f - \delta|\cdot|^2$ is convex for some $\delta > 0$), which gives a linear convergence rate.

We start by proving an auxiliary result.

Lemma. Let $v_1, v_2, \ldots, v_{k+1}, w_*$ be a sequence of vectors in $\mathbb{R}^d$, and let $\eta > 0$. Setting $w_0 = 0$ and

$$w_k := w_{k-1} - \eta v_k \quad k \in \mathbb{N},$$

we get that

$$\sum_{k=0}^{N} \langle v_{k+1}, w_k - w_* \rangle \leq \frac{|w_*|^2}{2\eta} + \frac{\eta}{2}\sum_{k=0}^{N} |v_{k+1}|^2.$$

In particular, we have that

$$\frac{1}{N+1}\sum_{k=0}^{N} \langle v_{k+1}, w_k - w_* \rangle \leq \frac{RL}{\sqrt{N+1}},$$

for any $R, L > 0$ such that

$$\eta = \frac{R}{L\sqrt{N+1}},$$

and

$$|w_*| \leq R, \quad |v_k| \leq L, \quad k = 1, \ldots, N+1.$$

Proof. A direct computation shows (polarization identity)

$$\langle v_{k+1}, w_k - w_* \rangle = \frac{1}{2\eta}\left(|w_k - w_*|^2 + \eta^2|v_{k+1}|^2 - |w_k - w_* - \eta v_{k+1}|^2\right) = \frac{1}{2\eta}\left(|w_k - w_*|^2 - |w_{k+1} - w_*|^2\right) + \frac{\eta}{2}|v_{k+1}|^2.$$

Adding up with respect to k yields

$$\sum_{k=0}^{N}\langle v_{k+1}, w_k - w_*\rangle = \frac{1}{2\eta}\sum_{k=0}^{N}\left(|w_k - w_*|^2 - |w_{k+1} - w_*|^2\right) + \frac{\eta}{2}\sum_{k=0}^{N}|v_{k+1}|^2.$$

The first term is a telescoping sum and $w_0 = 0$, so that we get

$$\sum_{k=0}^{N}\langle v_{k+1}, w_k - w_*\rangle = \frac{1}{2\eta}\left(|w_*|^2 - |w_N - w_*|^2\right) + \frac{\eta}{2}\sum_{k=0}^{N}|v_{k+1}|^2,$$

which proves the first assertion.
For the second assertion it is enough to observe that under our conditions

$$\frac{|w_*|^2}{2\eta} + \frac{\eta}{2}\sum_{k=0}^{N}|v_{k+1}|^2 \leq \frac{R^2}{2\eta} + \frac{\eta(N+1)L^2}{2} = RL\sqrt{N+1}$$

□

Proof of the Theorem. Recalling that f is convex, we get that

$$f(\bar{w}_N) = f\left(\frac{1}{N+1}\sum_{k=0}^{N} w_k\right) \leq \frac{1}{N+1}\sum_{k=0}^{N} f(w_k).$$

Therefore, for any $w_* \in \arg\min_{|w|\leq R} f(w)$, we obtain by the Lemma,

$$f(\bar{w}_N) - m = f(\bar{w}_N) - f(w_*) \leq \frac{1}{N+1}\sum_{k=0}^{M}(f(w_k) - f(w_*)) \leq \frac{1}{N+1}\sum_{k=0}^{N}\langle \underbrace{\nabla f(w_k)}_{=:v_{k+1}}, w_k - w_*\rangle \leq \frac{RL}{\sqrt{N+1}}.$$

□

Literature

1. Anne Greenbaum and Tim P. Chartier. Numerical Methods: Design, Analysis, and Computer Implementation of Algorithms, Princeton University Press, 2012.
2. L. Ridgway Scott. Numerical Analysis, Princeton University Press, 2011.
3. Qingkai Kong, Timmy Siauw, and Alexandre Bayen. Python Programming and Numerical Methods. A Guide for Engineers and Scientists. Academic Press, 2020.
4. Tobin A. Driscoll and Richard J. Braun, Fundamentals of Numerical Computation, SIAM, 2017.
5. Ernst Hairer, Gerhard Wanner, Syvert P. Nørsett. Solving Ordinary Differential Equations I: Nonstiff Problems. Springer, 2nd ed., 1993.
6. Real Analysis, Wikibooks, Creative Commons BY-SA 4.0.
7. Stefano Pagliarani. An introduction to discrete-time stochastic processes and their applications. Lecture notes, Alma Mater Studiorum - Università di Bologna, 2024.
8. Harri Hakula. Numerical Analysis. Lecture transcript, Aalto University, 2021.

---

1. jonas.tolle@aalto.fi ↩