Introduction to Scientific Computing

(Lecture 10: Numerical schemes for integration of ODEs)

Bojana Rosić

Institute of Scientific Computing

February 1, 2017

Solving ODE

In last lecture we studied the numerical evaluation of integral

$$\int_a^b f(t)dt = \sum_{n=1}^{N} f_n w_n$$

in which $f_n := f(t_n)$ is the value of function at given points t_n and w_n are corresponding weights. This knowledge further can be used to solve the first order differential equation

$$\dot{x}(t) = f(t, x(t)) \text{ for } t \in [0, T]$$

given initial condition

$$x_0 = x(0).$$

Solving ODE

The solution is given as

$$x(T) = x_0 + \int_0^T f(x,t)dt = x(0) + \sum_{i=0}^{N-1} \int_{t_i}^{t_{i+1}} f(x,t)dt$$

Having that $t_n := t_0 + n \cdot h$ one may further write

$$x(t_1) = x(0) + \int_{t_0}^{t_1} f(x,t)dt, \quad x(t_2) = x(t_1) + \int_{t_1}^{t_2} f(x,t)dt, \quad \dots$$

This then delivers the recursive formula

$$x(t_{n+1}) = x(t_n) + \int_{t_n}^{t_{n+1}} f(x,t)dt \approx x(t_n) + \int_{t_n}^{t_{n+1}} P_m(t)dt$$

Solving ODE

Depending on choice of polynomial degree m and interpolation points $t_i, \quad i = 1, ..., m+1$, the previous recursive scheme will obtain different form. Thus, we may distinguish:

- one step methods
  - explicit: for interpolation is used one known point (initial value or value from the previous time step)
  - implicit: for interpolation is used unknown point or both known and unknown points (state at the begining and at the end of the current step)
- multistep methods: use known points from the presvious steps (explicit) or use the unknown points (implicit) for the interpolation

Explicit (forward) Euler method

This method is based on the piecewise constant interpolation in which

$$P_0(t) = f(t), \quad a_0 = f(t)$$

To determine a_0 we use one interpolation point which denotes the beginning of time step, i.e.

$$a_0 = f(x_n, t_n) := f_n$$

which gives us recursive formula (left-rectungle rule)

$$x_{n+1} = x_n + \int_{t_n}^{t_{n+1}} a_0 dt = x_n + f_n h$$

with $h := (t_{n+1} - t_n)$.

Mathematical formulation

The idea of explicit Euler method emerges from the Taylor expansion of the solution $x(t+h)$ around $x(t)$:

$$x(t+h) \approx \sum_{n=0}^{m} \frac{h^i}{i!} x^{(i)}(t).$$

where only the first order terms are taken into consideration such that

$$x(t+h) \approx x(t) + \dot{x}|_t (t+h-t) = x(t) + f(x,t)h$$

i.e. if $t = nh$

$$x_{n+1} \approx x_n + f(x_n, t_n)h$$

holds.

Mathematical formulation

Note that in

$$x(t+h) \approx x(t) + \dot{x}(t)(t+h-t) = x(t) + f(x,t)h$$

one may write

$$\dot{x}(t) = f(x,t) \approx \frac{x(t+h) - x(t)}{t+h-t}$$

which is nothing else but approximation of slope at point t, i.e. forward difference. Thus, the name **forward Euler method**.

Pros & Cons

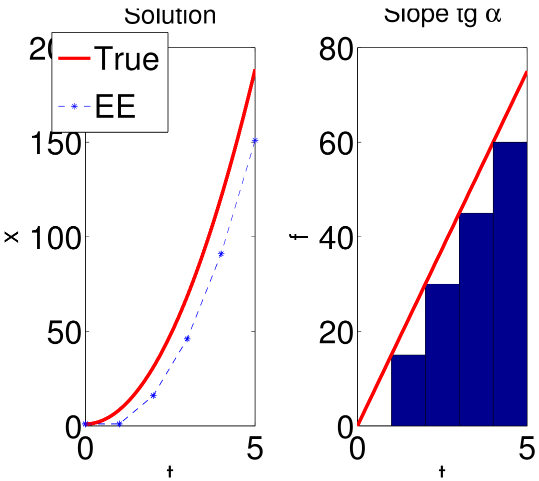
Pros:

- simple numerical scheme
- easy implementation

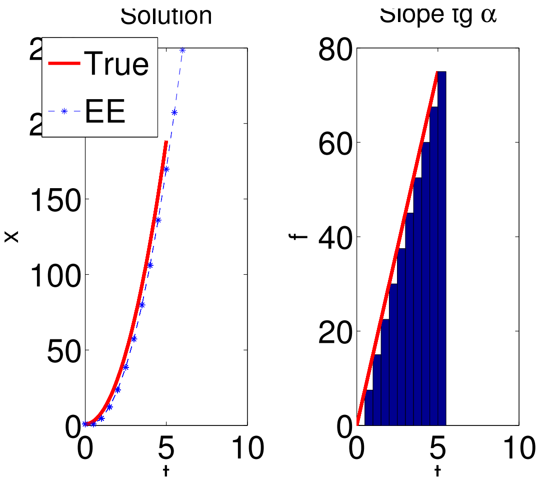
Cons:

- to get accurate result one requires quite small step size
- costly method
- sometimes does not converge to the solution or requires step size restriction (ODE 2)

Explicit Euler method $h = 1$



Explicit Euler method $h = 0.5$



Implicit (backward) Euler method

The method is also based on the piecewise constant interpolation $(P_0(t))$ but this time the constant a_0 is obtained from the interpolating condition

$$a_0 = f(x_{n+1}, t_{n+1})$$

leading to

$$x_{n+1} = x_n + f(x_{n+1}, t_{n+1})h, \quad n = 0, 1, ..$$

Note that this is implicit scheme as the value of x_{n+1} is unknown.

Mathematical formulation

The Taylor expansion reads

$$x(t) \approx x(t+h) + \dot{x}|_{t+h}(t-(t+h)) + h.o.t$$

i.e.

$$x(t) \approx x(t+h) - hf(x(t+h), t+h)$$

Hence,

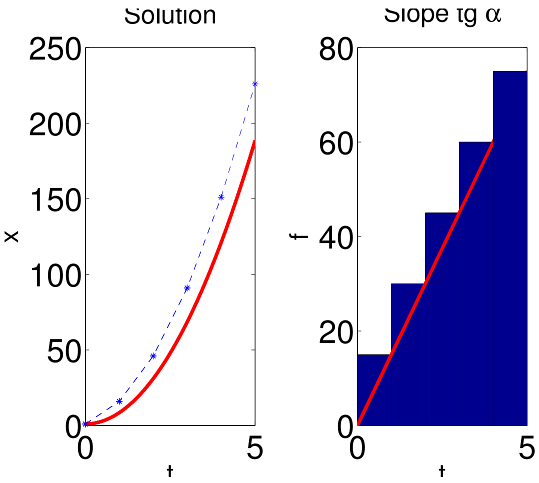
$$x_{n+1} = x_n + f(x_{n+1}, t_{n+1})h, \quad n = 0, 1, ..$$

and

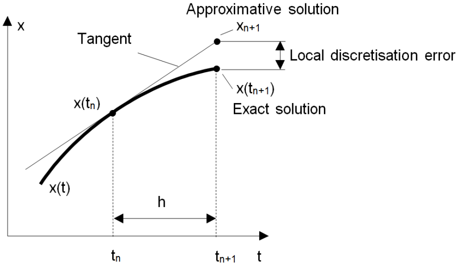
$$\dot{x}|_{t+h} \approx \frac{x(t) - x(t+h)}{t-(t+h)}$$

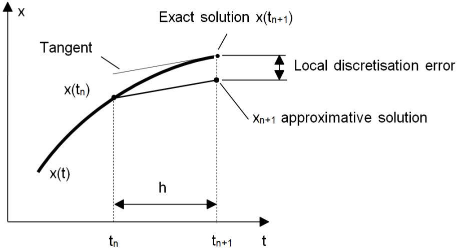
which is backward difference. Hence, the method is also known as **backward Euler method**.

Implicit Euler method $h = 1$



Explicit vs Implicit Euler method





x
Exact solution $x(t_{n+1})$
Tangent
Local discretisation error
$x(t_n)$
x_{n+1} approximative solution
$x(t)$
h
t_n
t_{n+1}
t

Pros & Cons

Pros:

- simple numerical scheme
- easy implementation

Cons:

- one requires linear and nonlinear solvers
- costly method
- not enough accurate

Trapezodial rule

Belongs also to the one step family, however this time is used linear interpolation by $P_1(t) = a_0 + a_1 t$. The coefficients a_0 are obtained by solving

$$a_0 + a_1 t_n = f(x_n, t_n), \quad a_0 + a_1 t_{n+1} = f(x_{n+1}, t_{n+1})$$

which results in the area of trapezoid

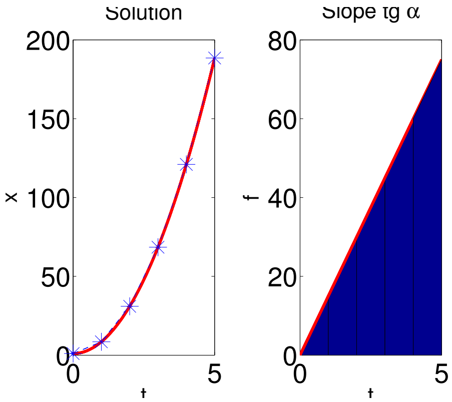
$$\int_0^T f(t)dt \approx \sum_{n=1}^{N-1} \frac{h}{2} f(t_n) + f(t_{n+1}))$$

Hence, the trapezoidal scheme for solving ODE reads

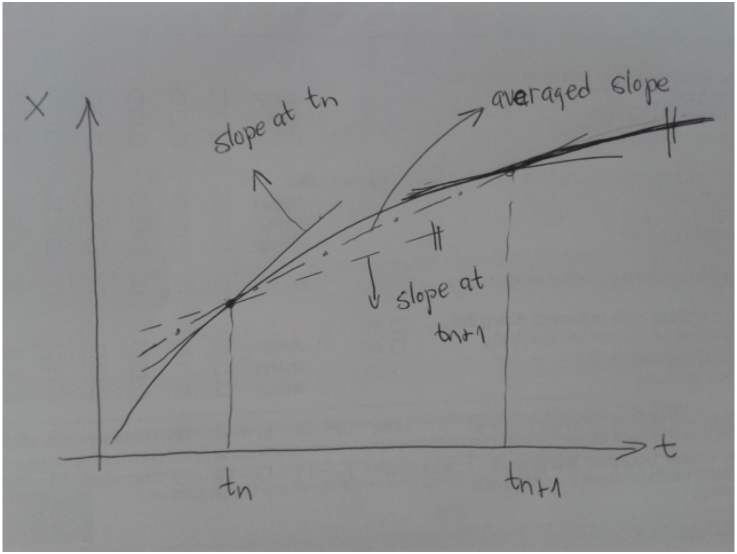
$$x_{n+1} = x_n + \frac{h}{2}(f(t_n, x_n) + f(t_{n+1}, x_{n+1}))$$

and as x_{n+1} appears on both left and righ hand side, the method is known to be **implicit**.

Trapezoidal rule



Trapezoidal rule



Midpoint rule

The method is based on the piecewise constant interpolation in which $a_0 = f(t_n + 0.5, x(t_n + 0.5h))$ such that

$$x_{n+1} = x_n + h(f(t_n + \frac{h}{2}, x(t_n + \frac{h}{2}))$$

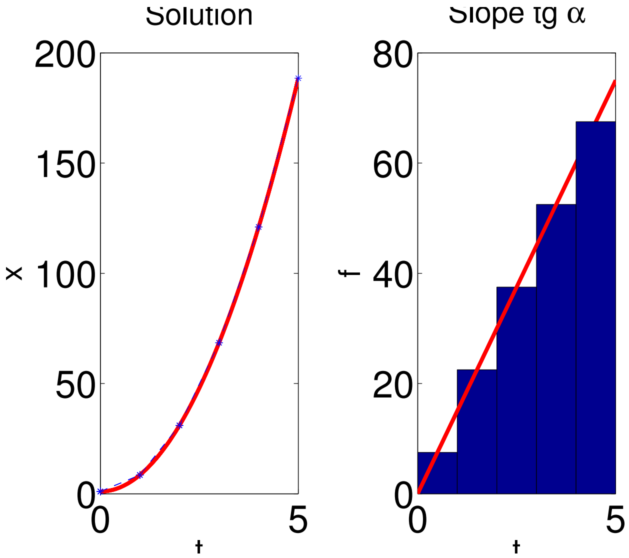
holds. However, we cannot use this equation as $x(t_n + \frac{h}{2})$ is not known. To compute this value one may use explicit Euler method (predictor method):

$$x(t_n + \frac{h}{2}) = x(t_n) + \frac{h}{2} f(t_n, x_n)$$

which leads us to the explicit formula

$$x_{n+1} = x_n + hf(t_n + \frac{h}{2}, x(t_n) + \frac{h}{2} f(t_n, x_n))$$

Explicit midpoint rule



Implicit midpoint rule

The implicit version can be obtained by approximating the value

$$x(t_n + \frac{h}{2}) = \frac{1}{2}(x_n + x_{n+1})$$

as a midpoint of line segment. This then leads to

$$x_{n+1} = x_n + hf(t_n + \frac{h}{2}, \frac{1}{2}(x_n + x_{n+1}))$$

Overview

Gathering previous numerical methods one has

- explicit Euler method $x_{n+1} = x_n + hf(t_n, x_n)$
- implicit Euler method $x_{n+1} = x_n + hf(t_{n+1}, x_{n+1})$
- trapezoidal rule $x_{n+1} = x_n + \frac{h}{2}(f(t_n, x_n) + f(t_{n+1}, x_{n+1}))$
- explicit midpoint rule $x_{n+1} = x_n + hf(t_n + \frac{h}{2}, x(t_n) + \frac{h}{2} f(t_n, x_n))$
- implicit midpoint rule $x_{n+1} = x_n + hf(t_n + \frac{h}{2}, \frac{1}{2}(x_n + x_{n+1}))$

Multistep methods

They are similar to one step methods. The only difference is that the interpolation is performed over several steps, and thus the name "multistep method". In other words,

$$x(t_{n+1}) = x(t_{n-j}) + \int_{t_{n-j}}^{t_{n+1}} f(t, x(t))dt \approx x(t_{n-j}) \int_{t_{n-j}}^{t_n} P_m(t)dt$$

and f is integrated on the time interval from t_{n-j} up to t_{n+1} where $j > 0$ and $j \leq k$, and polynomial P_m is interpolated given set of interpolation points $(x(t_i), t_i), i = 1, ..., m+1$.

Adams-Bashforth formulas

They are simple extension of explicit one step formulas. Similarly to the explicit one step method, the integration starts from the last known solution. However, the interpolation is done over several existing steps. As an example let us observe the method obtained by linear interpolation $P_1(t) = a_0 + a_1 t$ in which coefficients are obtained from the interpolating conditions (solutions from the last two steps)

$$a_1 t_n + a_0 = f_n$$

and

$$a_1 t_{n-1} + a_0 = f_{n-1}.$$

After solving the previous system and integrating polynomial, one obtains

$$x_{n+1} = x_n + \frac{3}{2} h f_n - \frac{1}{2} h f_{n-1}$$

which represents the difference equation of second order.

Adams-Bashforth formulas

Taking different number of interpolation points and $j = 0$ in general formula we get the Adams-Bashforth formulas (1883):

$$\begin{aligned}
k &= 1: & x_{n+1} &= x_n + hf_n, \\
k &= 2: & x_{n+1} &= x_n + h\left(\tfrac{3}{2}f_n - \tfrac{1}{2}f_{n-1}\right), \\
k &= 3: & x_{n+1} &= x_n + h\left(\tfrac{23}{12}f_n - \tfrac{16}{12}f_{n-1} + \tfrac{5}{12}f_{n-2}\right), \\
k &= 4: & x_{n+1} &= x_n + h\left(\tfrac{55}{24}f_n - \tfrac{59}{24}f_{n-1} + \tfrac{37}{24}f_{n-2} - \tfrac{9}{24}f_{n-3}\right)
\end{aligned}$$

Note: The first method is explicit Euler rule (one step method obtained by piecewise interpolation). Other methods are multistep as they need more than one starting value. In case when these values are not given, they can be computed with a one-step method.

Nyström formulas

Generalasing AB methods, one may obtain Nyström formulas by starting integration from x_{n-1} and interpolating over the last k known steps

$$\begin{aligned} k &= 2: & x_{n+1} &= x_{n-1} + 2hf_n, \\ k &= 3: & x_{n+1} &= x_{n-1} + h\left(\tfrac{7}{3}f_n - \tfrac{2}{3}f_{n-1} + \tfrac{1}{3}f_{n-2}\right), \\ k &= 4: & x_{n+1} &= x_n + h\left(\tfrac{8}{3}f_n - \tfrac{5}{3}f_{n-1} + \tfrac{4}{3}f_{n-2} - \tfrac{1}{3}f_{n-3}\right) \end{aligned}$$

Note: For $k = 2$ we get the explicit midpoint rule

Implicit multistep methods

If interpolation in the multistep method is done such that the interpolating conditions also include the unknown point $t_{n+1}, f(t_{n+1})$ then such methods are known as the implicit multistep methods. Typical example are Adams+Moulton methods

$$
\begin{aligned}
k &= 1: & x_{n+1} &= x_n + \frac{h}{2}(f_{n+1} + f_n),\\
k &= 2: & x_{n+1} &= x_n + h\left(\tfrac{5}{12} f_{n+1} + \tfrac{8}{12} f_n - \tfrac{1}{12} f_{n-1}\right),\\
k &= 3: & x_{n+1} &= x_n + \frac{h}{24}\left(9 f_{n+1} + 19 f_n - 5 f_{n-1} + f_{n-2}\right)
\end{aligned}
$$

General form

Methods we have just introduced are linear multistep methods of the form (only linear combination of x's and f's)

$$\sum_{l=0}^{k} a_l x_{n+l} = h \sum_{l=0}^{k} b_l f(t_{n+l}, x_{n+l})$$

i.e. the linear difference or recursive equations. Observing the numerical scheme as a linear difference equation one may study Lyapunov stability. On the other hand observing the numerical scheme as recursive equation one may study the accuracy of the scheme and its convergence (Banach fixed point theorem).

Accuracy

To measure accuracy of chosen numerical scheme, one may define local or global error. The local error is defined as

$$\epsilon_{loc} = x_a(t_{n+k}) - x(t_{n+k})$$

in which the value $x_a(t_{n+k})$ is the exact (e.g. algebraic) solution and $x(t_{n+k})$ is the value obtained by LMM numerical scheme starting from the exact value (i.e. the interpolating points are exact). For example, the local error of the first step would be the difference between the exact solution obtained from the exact initial value and numerical solution obtained from the exact initial value too. The numerical solution in one step obtained via LMM starting from the exact values reads:

$$a_k x(t_{n+k}) + \sum_{j=0}^{k-1} a_j x_a(t_{n+j}) = h \sum_{j=0}^{k} b_j f(t_{n+j}, x_a(t_{n+j}))$$

Note: the exact solution appears in all the terms besides the first one. This means that we predict next time step by assuming that all the initial conditions and all the tangents are exact.

Accuracy

Hence, the **local error** reads

$$\epsilon_{loc} = x_a(t_{n+k}) - x(t_{n+k})$$

$$\epsilon_{loc} = x_a(t_{n+k}) + a_k^{-1}\left(\sum_{j=0}^{k-1} a_j x_a(t_{n+j}) - h\sum_{j=0}^{k} b_j f(t_{n+j}, x_a(t_{n+j}))\right)$$

Multiply both sides by a_k to obtain

$$a_k\epsilon_{loc} = a_k x_a(t_{n+k}) + \sum_{j=0}^{k-1} a_j x_a(t_{n+j}) - h\sum_{j=0}^{k} b_j f(t_{n+j}, x_a(t_{n+j}))$$

which is then

$$a_k\epsilon_{loc} = \sum_{j=0}^{k} a_j x_a(t_{n+j}) - h\sum_{j=0}^{k} b_j f(t_{n+j}, x_a(t_{n+j}))$$

Accuracy

The value $x_a(t_{n+j})$ can be obtained from the Taylor expansion around the point in time t_n such that

$$x_a(t_{n+j}) = x_a(t_n) + x_a'(t_n)(t_{n+j} - t_n) + \frac{x_a''(t_n)}{2}(t_{n+j} - t_n)^2 + h.o.t.$$

and

$$x_a'(t_{n+j}) = x_a'(t_n) + x_a''(t_n)(t_{n+j} - t_n) + \frac{x_a'''(t_n)}{2}(t_{n+j} - t_n)^2 + h.o.t.$$

Note that the difference

$$t_{n+j} - t_n = (n + j)h - nh = jh$$

Accuracy

Hence,

$$\begin{aligned}
a_k \epsilon_{loc} &= \sum_{j=0}^{k} a_j x_a(t_{n+j}) - h \sum_{j=0}^{k} b_j f(t_{n+j}, x_a(t_{n+j})) \\
&= \sum_{j=0}^{k} a_j x_a(t_{n+j}) - h \sum_{j=0}^{k} b_j x_a'(t_{n+j}) \\
&= \sum_{j=0}^{k} a_j \left[x_a(t_n) + x_a'(t_n)(jh) + \frac{x_a''(t_n)}{2}(jh)^2 + h.o.t. \right] \\
&\quad - h \sum_{j=0}^{k} b_j \left[x_a'(t_n) + x_a''(t_n)(jh) + \frac{x_a'''(t_n)}{2}(jh)^2 + h.o.t. \right]
\end{aligned}$$

Accuracy

$$
\begin{aligned}
a_k \epsilon_{loc} &= \left[\sum_{j=0}^{k} a_j\right] x_a(t_n) + h \left[\sum_{j=0}^{k} (j a_j - b_j)\right] x_a'(t_n) \\
&+ h^2 \left[\sum_{j=0}^{k} (\frac{j^2}{2} a_j - j b_j)\right] x_a''(t_n) + \cdots + \\
&+ \cdots + h^{q+1} \left[[\sum_{j=0}^{k} (\frac{j^{q+1}}{(q+1)!} a_j - \frac{j^q}{q!} b_j)\right] x_a^{(q+1)}(t_n) + \mathcal{O}(h^{q+2})
\end{aligned}
$$

Consistency

We would like that the local error reduces with the time step size h. This condition is known as **consistency**. The linear multistep is consistent if

$$\lim_{h \to 0} \| \frac{\epsilon_{loc}}{h} \| = 0$$

The scheme is **consistent of order** p if

$$\max \| \frac{\epsilon_{loc}}{h} \| \leq Ch^p$$

Consistency

For linear multistep method this means that the term

$$\frac{\epsilon_{loc}}{h} = \frac{1}{h}\left[\sum_{j=0}^{k} a_j\right] x_a(t_n) + \left[\sum_{j=0}^{k}(ja_j - b_j)\right] x_a'(t_n)$$
$$+h\left[\sum_{j=0}^{k}(\frac{j^2}{2}a_j - jb_j)\right] x_a''(t_n) + \cdots +$$
$$+\cdots + h^q\left[[\sum_{j=0}^{k}(\frac{j^{q+1}}{(q+1)!}a_j - \frac{j^q}{q!}b_j)\right] x_a^{(q+1)}(t_n) + \mathcal{O}(h^{q+1})$$

should go to zero when $h \to 0$

Consistency

Hence, the method is consistent if

$$\sum_{j=0}^{k} a_j = 0, \quad \sum_{j=0}^{k}(j a_j - b_j) = 0$$

and

$$\sum_{j=0}^{k}\left(\frac{j^q}{(q)!} a_j - \frac{j^{q-1}}{(q-1)!} b_j\right) = 0, \quad q = 2, \cdots p$$

Consistency of Adams-Bashforth method

Having

$$x_{n+2} = x_{n+1} + \frac{h}{2}(-f(t_n, x_n) + 3f(t_{n+1}, x_{n+1}))$$

where

$$a_0 = 0, \quad a_1 = -1, \quad a_2 = 1$$

and

$$b_0 = -0.5, \quad b_1 = 1.5, \quad b_2 = 0$$

one may check if the following conditions hold

$$\sum_{j=0}^{k} a_j = 0, \quad \sum_{j=0}^{k} (j a_j - b_j) = 0$$

Consistency of Adams-Bashforth method

$$\sum_{j=0}^{k} a_j = 0 - 1 + 1 \equiv 0$$

$$\sum_{j=0}^{k} (j a_j - b_j) = 0 \cdot a_0 - b_0 + a_1 - b_1 + 2a_2 - b_2$$

$$\sum_{j=0}^{k} (j a_j - b_j) = 0.5 - 1 - 1.5 + 2 - 0 \equiv 0$$

Hence, the method is consistent.

Consistency order of Adams-Bashforth method

Let us check now

$$\sum_{j=0}^{k}\left(\frac{j^q}{(q)!}a_j - \frac{j^{q-1}}{(q-1)!}b_j\right) = 0, \quad q = 2, \cdots p$$

$$\sum_{j=0}^{k}\left(\frac{j^2}{(2)!}a_j - \frac{j^{2-1}}{(2-1)!}b_j\right) = -0.5 - 1.5 + 2 - 0 \equiv 0$$

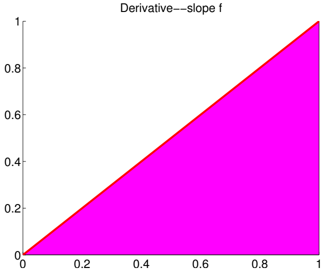
$$\sum_{j=0}^{k}\left(\frac{j^3}{(3)!}a_j - \frac{j^{3-1}}{(3-1)!}b_j\right) = 9/12 \neq 0$$

Hence, the order is $p = 2$.

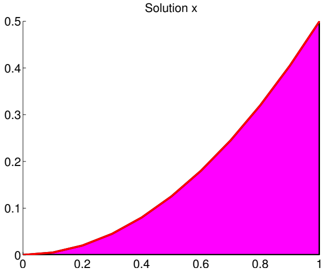
Consistency of explicit Euler

Let us integrate the ODE $\dot{x} = t$ in time interval $[0, 1]$ by taking whole interval as one step. Thus, one has

$$x_1 = x_0 + f_0 h = 0 + 0 \cdot 1 = 0, \quad x_a(t) = 0.5t^2$$



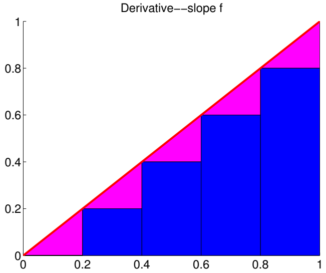
a) error in slope



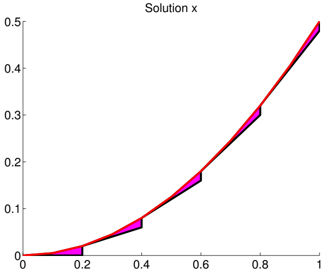
b) error in solution (local error) $= 0.5$

Consistency of explicit Euler

or let us cut the time interval into several substeps and in each step always start from the exact solution



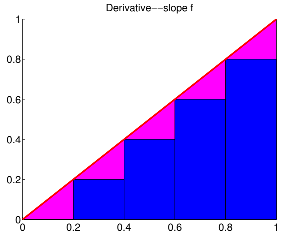
a) error in slope



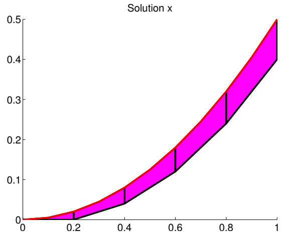
b) error in solution (local error)≈ 0.02

Is consistency enough?

In order to see this, let us compute numerically the solution x_1 in one step starting from x_0 (which is analytical (**exact**)). Then, let us compute the next step starting from the **last** numerical solution x_1 to obtain x_2 etc. Finally, let us compute the total error between **full** numerical solution and analytical one over some time interval.



a) error in slope [a]

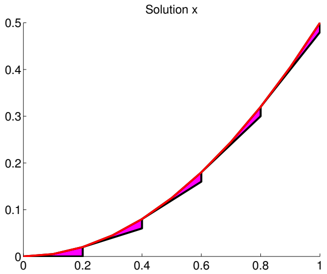


b) error in solution (global error)≈ 0.1

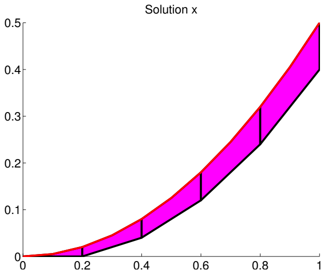
[a] same as before as f does not depend on x

Is consistency enough?

By comparing both of figures, we may see that the full numerical solution gives us bigger error (This is to expect as local error gets integrated over time. Also, the left picture is not realistic as usually one does not know the exact solution).



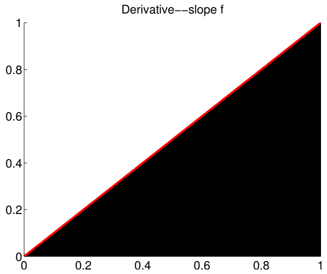
a) local error ≈ 0.02



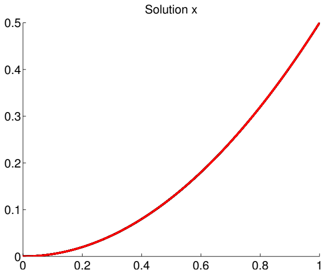
b) global error ≈ 0.1

Is consistency enough?

To reduce the error, one has to reduce the time step size. Hence, let us make the step size 100 times smaller, then we get



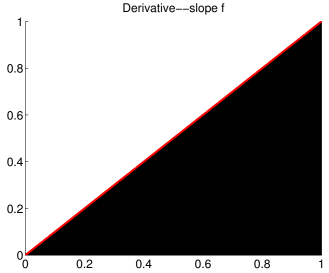
a) error in slope very small



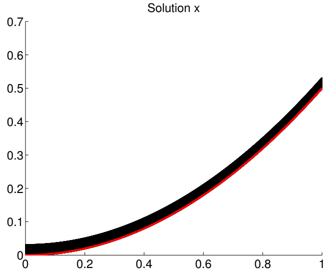
b) error in solution very small

But, the initial condition is...

maybe not exact (think about multistep method and only one starting point, or unknown initial condition)...Let us make inexact initial condition by perturbing it a bit (for a value 0.03)



a) error in slope very small [a]



b) error in solution not very small!!

[a]because the slope does not depend on x in this example, i.e. $f = t$

Thus,

in order that the numerical scheme gives us the exact solution (i.e. convergent solution), following requirements have to satisfied

- the method has to be consistent,
- as well as stable on perturbation of initial conditions (i.e. zero stability)

$$\text{convergence}=\text{consistency}+\text{zero stability}$$

Convergence

Linear multistep method is known to be convergent if

$$\max_{t_n \in T} \|\epsilon_{glob}(t_n, h)\| \to 0, \quad h \to 0$$

The order of convergence is q if

$$\max_{t_n \in T} \|\epsilon_{glob}(t_n, h)\| \leq Ch^q$$

where

- the global error is
$$\epsilon_{glob} = x_a(t) - x(t)$$
- $x_a(t)$ is the exact solution of the differential equation $\dot{x} = f(t, x)$
- $x(t)$ is the approximate solution
- C is a constant independent of h

Convergence

Convergence implies consistency. Consistency does not imply convergence.

Theorem

An integration scheme is convergent if and only if it is consistent and zero stable, and in case it is convergent, the order of consistency and the order of convergence are equal.

Zero-stability

Linear multistep method

$$\sum_{j=0}^{k} a_j x_{n+j} = h \sum_{j=0}^{k} b_j f(t_{n+j}, x_{n+j}), \quad n = 0, \dots, N-k$$

applied on ODE $\dot{x} = f(t, x)$ gives the stability condition

$$\dot{x} = 0 \rightarrow \sum_{j=0}^{k} a_j x_{n+j} = 0$$

which is **linear homogeneous difference equation**.

Linear homogeneous difference equation

The difference equation

$$\sum_{j=0}^{k} a_j x_{n+j} = 0$$

has for a general solution:

$$u_n = \sum_{i=1}^{r} p_i \xi_i^n,$$

where ξ_i are the roots of first characteristic polynomial

$$\rho(\xi) = \sum_{i=0}^{k} a_i \xi^i.$$

Hence, we need to study stability of difference equiation with respect to the zero (initial) conditions.

Stability of difference equation

Consider: general difference equation of order 1 and dimension d

$$\mathbf{x}_{n+1} = F(\mathbf{x}_n), \qquad \mathbf{x}_n \in \mathbb{R}^d,\ n \in \mathbb{N}.$$

Definition: An equilibrium point of the dynamical system

$$\mathbf{x}_{n+1} = F(\mathbf{x}_n), \qquad \mathbf{x}_n \in \mathbb{R}^d,\ n \in \mathbb{N}$$

is a state–vector $\mathbf{x}_* \in \mathbb{R}^d$ such that

$$F(\mathbf{x}_*) = \mathbf{x}_*$$

holds.

Stability of difference equation

Stability criteria: of $x_{n+1} = Ax_n$

- If all λ_i of A have absolute value smaller than one: $(\forall i = 1, \dots, d : |\lambda_i| < 1)$, then for every $x_0 \in \mathbb{R}^d$ the sequence $x_n \overset{n\to\infty}{\longrightarrow} 0$, and x_* is asymptotically stable.
- If any λ_i of A has absolute value greater than one: $(\exists i : |\lambda_i| > 1)$ then there exist $x_0 \in \mathbb{R}^d$ such that the sequence $x_n \overset{n\to\infty}{\longrightarrow} \infty$, and x_* is unstable.
- If all λ_i of A have absolute value smaller or equal than one: $(\forall i = 1, \dots, d : |\lambda_i| \leq 1)$ and if there are λ_j's with $|\lambda_j| = 1$, then we cannot decide whether or not x_* is stable.
- if the system is nonlinear then compute Jacobian and apply previous rules

Zero stability of approximated ODE

Another way of investigating stability of

$$\sum_{l=0}^{k} a_l x_{m+l} = 0$$

is to look at general solution:

$$u_m = \sum_{i=1}^{r} p_i \xi_i^m,$$

where ξ_i are the roots of ρ.

Zero stability of approximated ODE

A scheme satisfies the root condition (is zero stable) if every root ξ_i of the first characteristic polynomial ρ

$$\rho(\xi) = \sum_{i=0}^{k} a_i \xi^i.$$

has magnitude smaller than one, $|\xi_i| \leq 1$, and if every root ξ_i with $|\xi_i| = 1$ is a simple root of ρ.

Definition

Definition

In order for a consistent scheme to be convergent, a stability property has to be fulfilled:

A scheme satisfies the root condition (is zero stable) if every root ξ_i of the first characteristic polynomial ρ has magnitude smaller than one, $|\xi_i| \leq 1$, and if every root ξ_i with $|\xi_i| = 1$ is a simple root of ρ.

Exercise

The method

$$x_{n+1} = x_n + h\left(\frac{5}{12}f_{n+1} + \frac{8}{12}f_n - \frac{1}{12}f_{n-1}\right)$$

applied on

$$\dot{x} = f(t, x)$$

gives the stability condition

$$x_{n+1} - x_n = 0$$

i.e. characteristic equation

$$\xi - 1 = 0$$

The single root is equal to 1. Hence, the method is zero stable but not attractive.

Is zero-stability enough?

The next question to ask is if zero-stability is enough to obtain convergence and accuracy (**namely, the method can converge, but to the wrong solution!!**). Let us observe the numerical integration of the following two systems by explicit Euler method:

$$\dot{x} = -2000x, \quad x(0) = 1$$

and

$$\dot{x} = -x, \quad x(0) = 1$$

with the time step size $h = 10^{-3}$ in the time interval $[0, 2]$.

Convergence

Before integration let us check if the method is

1. consistent:

$$\rho(\xi) = \xi - 1 \Rightarrow \rho(1) = 0, \rho'(1) = 1$$

,

$$F = f_n \Rightarrow f = \frac{F}{\rho'(1)}$$

2. zero stable

$$\rho(\xi) = \xi - 1 = 0 \Rightarrow \xi = 1$$

Hence, the method is consistent and zero stable, and thus the method is convergent when $h \to 0$.

Convergence

By comparing numerical results one obtains

Case	Truth	Numerical	Relative error
Case I	0.1353	0.1352	0.001
Case II	1.383e-87	3.055e-92	0.999

This table shows that even the step size was taken to be almost equal to zero, the relative error can be still huge (99% in second case). But the method is convergent?

Convergence

The method is convergent as can be seen in the following table because the error goes to zero when the time step size decreases.

Case	Case I	Case II
h=1	1	7.08e+90
h=0.1	0.1017	8.78e+105
h=1e-3	0.001	0.9999
h=1e-6	≈1e-6	0.00995

Since one cannot adopt too small h, the question is now:

When to stop? Which step size is small enough to give us desried accuracy?

Global convergence

To answer this one, first the global error has to be studied:

$$\epsilon_{glob}^{n+1} = x_a(t_{n+1}) - x(t_{n+1})$$

in which x_a is analytical and x full numerical solution. Having in mind that

$$x(t_{n+1}) = x(t_n) + hf(t_n, x_n) = x(t_n) + h\lambda x_n, \quad n = 0, 1, 2, ...$$

one may write

$$\epsilon_{glob}^{n+1} = x_a(t_{n+1}) - x(t_n) - h\lambda x_n$$

Global convergence

Expanding x_a into Taylor serie

$$x_a(t_{n+1}) = x_a(t_n) + h\dot{x}_a(t_n) + \frac{h^2}{2}\ddot{x}_a + h.o.t.$$

one obtains

$$\epsilon_{glob}^{n+1} = x_a(t_n) + h\dot{x}_a(t_n) + \frac{h^2}{2}\ddot{x}_a - x_n - h\lambda x_n$$

Here, the first derivative is known from given ODE

$$\dot{x}_a(t_n) = \lambda x_a(t_n)$$

Global convergence

This finally gives expression for the global error

$$\epsilon_{glob}^{n+1} = x_a(t_n) + h\lambda x_a(t_n) + \frac{h^2}{2}\ddot{x}_a - x_n - h\lambda x_n + h.o.t.$$

Collecting terms together one obtains

$$\epsilon_{glob}^{n+1} = (1 + h\lambda)(x_a(t_n) - x_n) + \frac{h^2}{2}\ddot{x}_a + h.o.t.$$

$$\epsilon_{glob}^{n+1} = (1 + h\lambda)\epsilon_{glob}^{n} + \frac{h^2}{2}\ddot{x}_a$$

Note that in last relation the term $\frac{h^2}{2}\ddot{x}_a$ represents the local error

$$\epsilon_{loc} = \frac{h^2}{2}\ddot{x}_a$$

Global convergence

Looking at

$$\epsilon_{glob}^{n+1} = (1 + h\lambda)\epsilon_{glob}^{n} + \epsilon_{loc}$$

one may conclude that the global error in $n + 1$ step does not only dependend on the currecnt local error ϵ_{loc} but also on the propagated error from the previous step $(1 + h\lambda)\epsilon_{glob}^{n}$. This error gets multiplied by $(1 + h\lambda)$, and hence the behaviour of ϵ_{glob} in time will be driven by λ (given model we cannot change) and the step size h (one can modify). After n steps the global error is proportional to

$$(1 + h\lambda)^n$$

and is driven by power law of n.

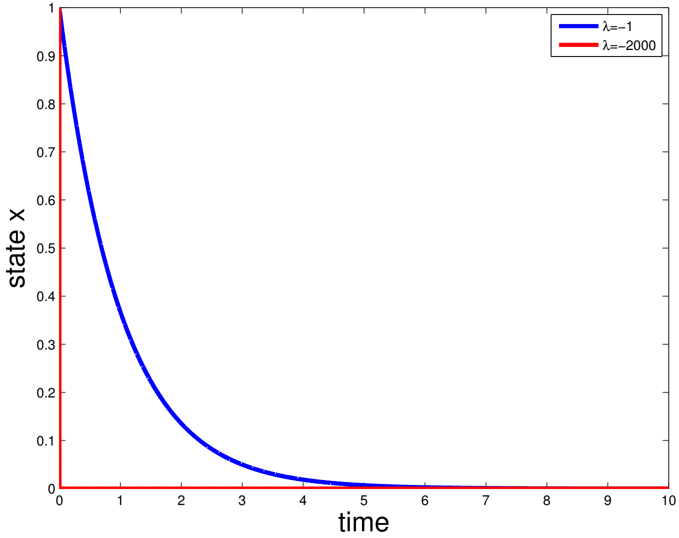
Global convergence

In our examples this means

$$(1 + h\lambda)^n \Rightarrow (1 - h)^n \quad \text{or} \quad (1 - 2000h)^n$$

However, as both $1 - h$ and $1 - 2000h$ are smaller than 1 by absolute value (for $h = 10^{-6}$), the global error will converge to zero (this corresponds to the exact solution) with $n \to \infty$. However, for $h = 10^{-3}$, the first term will converge to zero whereas the other not. Reason is higher value of λ in second ODE. One may show that the higher the value of λ is, the more difficult is to integrate. In other words ODE becomes stiffer. Unfortunately, most of practical examples are considered to be stiff. Due to this reason, this will be the main subject of ODE2!!

Non-stiff vs stiff - response



Global convergence

After previous observations have been made, one may give the following conclusion:

In order to get convergent and accurate method for some system ($\dot{x} = \lambda x$), one has to study its stability with respect to the time step size (i.e. stability of numerical method=difference equation)

Absolute Stability

To judge if the numerical method with the desired $h > 0$ is accurate enough, the term of absolute stability is introduced. This stability is tested on the Dahlquist problem:

$$\dot{x} = \lambda x, \quad x(0) = 1$$

whose exact solution is

$$x = \exp(\lambda t)$$

$$\lim_{t \to \infty} |x(t)| = \begin{cases} 0, & \text{if } \lambda < 0 \\ 1, & \text{if } \lambda = 0 \\ \infty, & \text{if } \lambda > 0 \end{cases}$$

We are interested in the case $\lambda < 0$. In this case the ODE is assymptotically stable.

Absolute Stability

Why we study Dahlquist problem?

Because any system can be represented as a linear system of ODEs. In case that you study nonlinear ODE then linearise and observe Jacobian.

Absolute Stability

If the previous ODE is solved by explicit Euler method, then one obtains numerical solution in a form

$$x_{n+1} = x_n + h\lambda x_n = (1 + h\lambda)x_n \Rightarrow x_n = (1 + h\lambda)^n x_0$$

This difference equation is stable when

$$|1 + h\lambda| \leq 1$$

Hence, one obtains restirction on the step size

$$-2 \leq h\lambda \leq 0$$

Absolute Stability

It is common practice to speak about absolute stability in the region of complex z plane

$$z = h\lambda$$

instead in terms of the step size. This allows λ to be complex.

Allowing λ to be complex comes from the fact that in practice we are usually solving a system of ordinary differential equations (ODEs). In the linear case it is the eigenvalues of the coefficient matrix that are important in determining stability. In the nonlinear case we typically linearize and consider the eigenvalues of the Jacobian matrix. Hence λ represents a typical eigenvalue and these may be complex even if the matrix is real.

Absolute Stability

Thus, to get absolutely stable method it must be satisfied

$$|R(z)| \leq 1$$

in which

$$R(z) = 1 + z$$

What if we have general linear multistep method?

Absolute Stability

When general linear multistep method

$$\sum_{j=0}^{k} a_j x_{n+j} = h \sum_{j=0}^{k} b_j f(t_{n+j}, x_{n+j})$$

is applied on

$$\dot{x} = \lambda x$$

one obtains

$$\sum_{j=0}^{k} a_j x_{n+j} = h \sum_{j=0}^{k} b_j \lambda x_{n+j}$$

Absolute Stability

i.e.

$$\sum_{j=0}^{k}(a_j - h\lambda b_j)x_{n+j} = 0$$

which is difference equation. Its stability is given by the roots of characteristic polynomial

$$\rho(\xi) - z\sigma(\xi) = 0$$

in which

$$\rho(\xi) = \sum_{j=0}^{k} a_j \xi^j, \quad \sigma(\xi) = \sum_{j=0}^{k} b_j \xi^j$$

Absolute Stability

The difference equation is stable if roots

$$\rho(\xi) - z\sigma(\xi) = 0$$

are smaller by amplitude than 1, or eventually only one of them is equal to 1. With respect to this one defines

Definition

The region of absolute stability for the LMM is the set of points z in the complex plane for which the polynomial $\rho(\xi) - z\sigma(\xi) = 0$ satisfies the root condition.

$$G_s := \{z \in \mathbb{C} \;:\; |\xi(z)| \leq 1\}$$

Example

For Euler method

$$x_{n+1} = x_n + h\lambda x_n$$

one has

$$x_{n+1} - x_n \Rightarrow \rho(\xi) = \xi - 1$$

and

$$h\lambda x_n \Rightarrow z\sigma(\xi) = z$$

Thus

$$\rho(\xi) - z\sigma(\xi) = 0$$

$$\xi - 1 - z = 0 \Rightarrow \xi_1 = 1 + z = R(z)$$

From this follows stability region

$$G_s := \{z \in \mathbb{C} \ : \ |1 + z| \leq 1\}$$

Example

To plot the previous region, one has to notice the following: z on boundary of the stability region gives the root ξ of absolute value equal to 1. In polar coordinates this means that

$$\xi = e^{i\theta}, \quad \theta \in [0, 2\pi]$$

Since this is the root of

$$\rho(\xi) - z\sigma(\xi) = 0$$

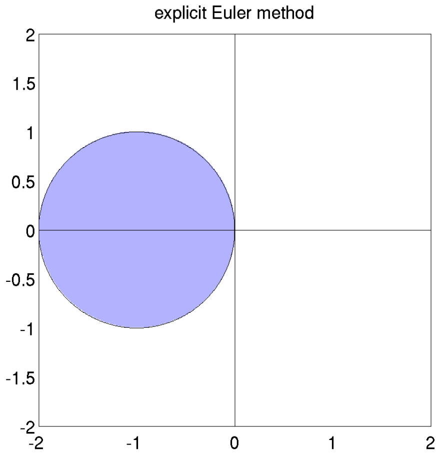
one has

$$\rho(e^{i\theta}) - z\sigma(e^{i\theta}) = 0$$

and hence

$$z = \frac{\rho(e^{i\theta})}{\sigma(e^{i\theta})}$$

Example



Example

For implicit Euler method

$$x_{n+1} = x_n + h\lambda x_{n+1}$$

one has

$$x_{n+1} - x_n \Rightarrow \rho(\xi) = \xi - 1$$

and

$$h\lambda x_{n+1} \Rightarrow z\sigma(\xi) = z\xi$$

Thus

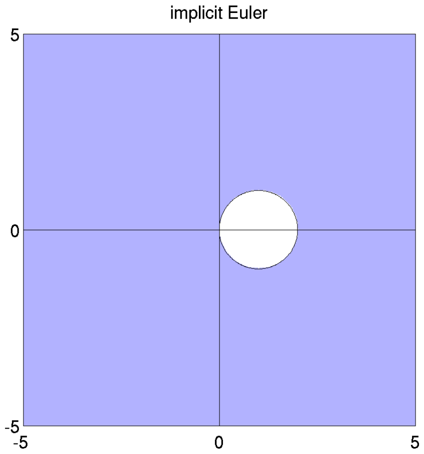
$$\rho(\xi) - z\sigma(\xi) = 0$$

$$\xi - 1 - z\xi = 0 \Rightarrow \xi_1 = \frac{1}{1-z} = R(z)$$

From this follows stability region

$$G_s := \{z \in \mathbb{C} \ : \ |\frac{1}{1-z}| \leq 1\}$$

Example


implicit Euler