Excercise to numerical Integration and ODEs I
Practical in Numerical Astronomy SS 2010

Siegfried Eggl

1 Excercises in short

- Define the quality of the approximation $f(t)$ achieved by explicit Euler's numerical integration of the function
$\dot{f}(t) = \frac{df(t)}{dt} = Sin(t)$ depending on the stepzie h and the type of variables being used (*single-*, *double precision*) and compare your results to the ones gained from numerical differentiation!
- Solve the ordinary differential equation of the dampened Harmonic Oscillator *analytically and numerically.* Use Euler's explicit method, Runge-Kutta's method, as well as the Simpson's-Rule and evaluate their performances in terms of conservation of total energy and phase-space properties by comparing them to the analytical solution.

2 Numerical Integration

Once again, let $f : \mathbb{R} \to \mathbb{R}$, where we denote $h = t - t_0$ as being the so-called 'stepsize', $\{t, t_0, h\} \in \mathbb{R}$. As we have seen lately, using truncated Taylor Series will result in viable approximations of function values at small displacements from the origin of developement, given the function's value as well as the function's derivatives at the origin.

$$f(t_0 + h) \approx f(t_0) + \dot{f}(t_0)h + \frac{\ddot{f}(t_0)}{2!}h^2 + \cdots + \frac{f^{(n)}(t_0)}{n!}h^n + O(h^{n+1})$$

Think of the function $f(t)$ now as being the integral of $\dot{f}(t)$. Truncating the Taylor Series after the first two terms will give:

$$f(t_0 + h) = f(t_0) + \dot{f}(t_0) \cdot h + O(h^2)$$

Using this linear approximation we are able to calculate up to terms of $O(h^2)$:

$$\int_{t_0}^{t_0+h} \dot{f}(t)dt = f(t_0 + h) - f(t_0) \simeq \dot{f}(t_0) \cdot h$$

Geometrically speaking, we try to model the real value of the integral in terms of the area of a triangle, spanned by the tangent $\dot{f}$ and the stepsize h. This works perfectly if we want to mimic linear functions, but the approximation will become worse and worse the more f deviates from the linear form. There are at least two ways to escape this dilemma:

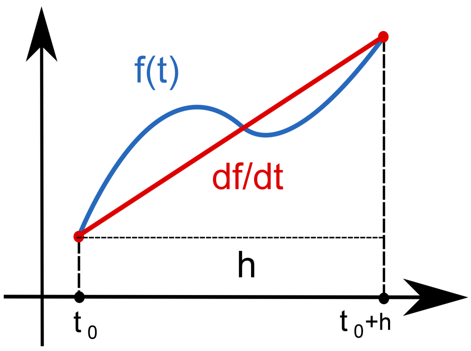

Figure 1: Triangle approximation ($h \cdot \dot{f}$) to the function $f(t)$ representing the value of the integral $\int_{t_0}^{t_0+h} \dot{f}(t)dt$.

1. use polynomials of higher order as approximands
2. make the stepsize h smaller, and consequently do more steps to cover the same integration range

Though, both of these points have their disadvanteges. Higher order polynomials require the knowlege (or creation) of more integral-function-values, e.g. $f(t_0 - h), f(t_0 - 2h), f(t_0 - h/2)$, whereas smaller stepsizes h require more calculation steps in order to arrive at $f(t_0 + h)$, thereby accumulating unavoidable roundoff errors. The "art of numerics" partly consists of finding/guessing the optimal ansatz for a given problem...

3 Higher Order Algorithms? Easier said than done...

Just take more terms of the corresponding Taylor Series and... wait a minute. By now you (should) have found out, that numerical generation of derivatives is not that good an idea. Yet, for propagating the Taylor Series we need to have higher functional derivatives?! Not necessarily! Applying clever combinations of Taylor Series in different directions (e.g. $f(t_0 + h) \simeq f(t_0) + h\dot{f}(t_0)$ and $f(t_0 - h) \simeq f(t_0) - h\dot{f}(t_0)$) one can eliminate terms of certain order (in this case of $O(h^2)$) from the combined series, and therefore produce an algorithm of *second order* (exact up to $O(h^3)$).

German mathematicians C.Runge and M.W.Kutta found their famous integration scheme by comparing the Taylor expansion up to $O(h^5)$ with an ansatz $f(t + h) = f(t) + h(a_1k_1 + a_2k_2 + a_3k_3 + a_4k_4)$.

For a given differential equation of the form

$$f' = g(t, f), \quad f(t_0) = f_0$$

representing an initial value problem, the RK4 algorithm amounts to

$$\begin{aligned} f_{n+1} &= f_n + h\left(\tfrac{1}{6}k_1 + \tfrac{1}{3}k_2 + \tfrac{1}{3}k_3 + \tfrac{1}{6}k_4\right) \\ t_{n+1} &= t_n + h \end{aligned}$$

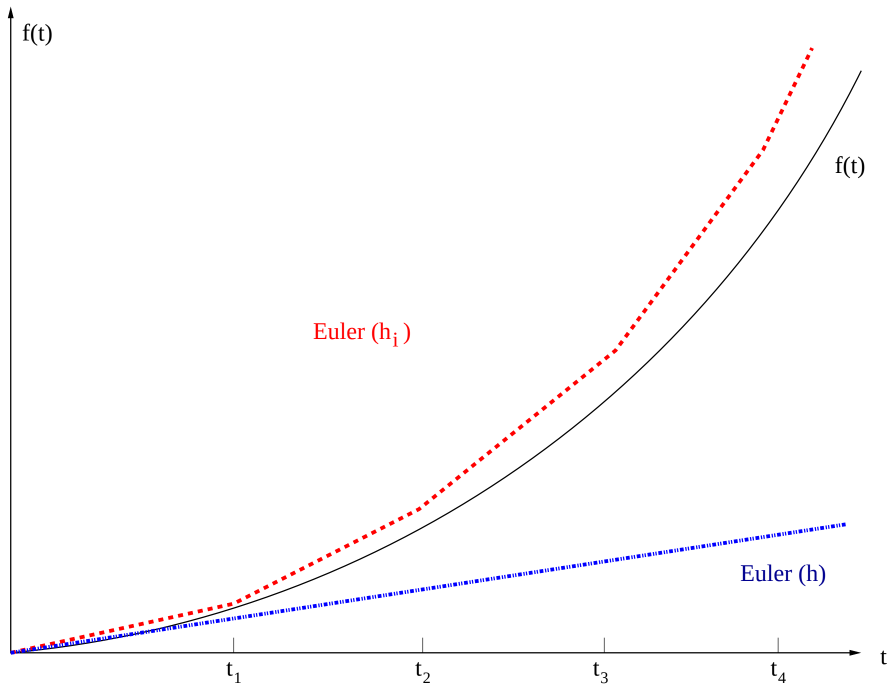

Figure 2: Smaller stepsizes (h) will reduce the truncation error, but increase the number of required calculations.

with the following coefficients.

$$\begin{aligned} k_1 &= g(t_n, f_n) \\ k_2 &= g(t_n + \tfrac{1}{2}h, f_n + \tfrac{1}{2}hk_1) \\ k_3 &= g(t_n + \tfrac{1}{2}h, f_n + \tfrac{1}{2}hk_2) \\ k_4 &= g(t_n + h, f_n + hk_3) \end{aligned}$$

By (tiresome) expansion of these equations and comparison to the actual Taylor Series, one can see, that their resulting coefficients made it possible to actually chop all the first terms in the Taylor Series up to $O(h^5)$!

Another way to achieve higher order integration approximants has been discovered by I. Newton and R. Cotes. Their approach was to fit the function that is to be integrated $g(t)$ (our former $\dot{f}$) with polynomials, and integrate those interpolating polynomials instead of $g(t)$.

Let $L(t)$ be one of those interpolating polynomials - they can be split into 'Lagrange basis polynomials' $l_i(t)$, an analogon to unit vectors in algebra, that may be multiplied with the function values at given points $g(t_i) = g(t_0 + i \cdot h),\ i \in \mathbb{N}$ in order to fit them to our desired function $g(t)$.

$$\int_{t_0}^{t_0+h} g(t)\,dt \approx \int_{t_0}^{t_0+h} L(t)\,dt = \int_{t_0}^{t_0+h} \big(\sum_{i=0}^{n} g(t_i)\, l_i(t)\big)\,dt = \sum_{i=0}^{n} g(t_i) \underbrace{\int_{t_0}^{t_0+h} l_i(t)\,dt}_{w_i}$$

The benefit of this is, that the integrals over these basis functions $l_i(t)$ *are known* *in advance*! So they may be rewritten in terms of 'weighting coefficiens' w_i that do no longer depend on t. Therefore the integral becomes:

$$\int_{t_0}^{t_0+h} g(t)\,dt \approx \sum_{i=0}^{n} w_i\, g(t_i)$$

A whole set of numerical integration algorithms of different orders can thus be derived.

degree n	name	weights w_i	error
0	Rectangle Rule	0 1	$\frac{h^2}{2}\ f'(\xi)$
1	Trapezoidal Rule	$\frac{1}{2}\quad\frac{1}{2}$	$\frac{h^3}{12} f''(\xi)$
2	Simpson's Rule	$\frac{1}{6}\quad\frac{4}{6}\quad\frac{1}{6}$	$\frac{(\frac{1}{2}h)^5}{90} f^{(4)}(\xi)$
3	3/8-Rule	$\frac{1}{8}\quad\frac{3}{8}\quad\frac{3}{8}\quad\frac{1}{8}$	$\frac{3(\frac{1}{3}h)^5}{80} f^{(4)}(\xi)$
4	Milne-Rule	$\frac{7}{90}\quad\frac{32}{90}\quad\frac{12}{90}\quad\frac{32}{90}\quad\frac{7}{90}$	$\frac{8(\frac{1}{4}h)^7}{945} f^{(6)}(\xi)$
5	6-Point-Rule	$\frac{19}{288}\quad\frac{75}{288}\quad\frac{50}{288}\quad\frac{50}{288}\quad\frac{75}{288}\quad\frac{19}{288}$	$\frac{275(\frac{1}{5}h)^7}{12096} f^{(6)}(\xi)$

where $t_0 \le \xi \le t_0 + h$

Simpson's Rule will be for example:

$$\int_{t_0}^{t_0+h} g(t)\,dt \approx \frac{h}{6}(g(t_0) + 4g(t_1) + g(t_2))$$

where $t_i = t_0 + h/n \cdot i$.

4 Exercises in detail

4.1 Quality control

Show the dependency of the numerical approximations to an analytical integral on the stepsize h using the function $\dot{f}(t) = Sin(t)$ and explicit Euler's method. In order to achieve that, compare $f(t_0)_{analytical}$ to $f(t_0)_{numerical}$ and calculate the relative deviation $R(h)$:

$$R(h) = log_{10} \frac{f(t_0)_{analytical} - f(t_0)_{numerical}}{f(t_0)_{analytical}} \tag{1}$$

at $t_0 = 5$ for at least 100 different stepsizes h between 1 and 10^{-6} for single precision (using Fortran90: *real* variables) and h between 1 and 10^{-15} for double precision (using Fortran90: *real*8* variables). Plot the results and compare them to those of your numerical differentiation!

4.2 The (dampened) Harmonic Oscillator

Being the most basic example for an equation of motion, the Harmonic Oscillator provides a nice playground in order to get acquainted with numerical integration. Its mathematical description can be stated in an ordinary differential equation of second order

$$\frac{d^2x}{dt^2} + r\frac{dx}{dt} + kx = 0$$

with $x = x(t)$, r being a friction coefficient and k the spring constant. Solve this equation analytically, and find its aperiodic limit (transition point between oscillations and exponential decay) for $k = 1$! In order to treat this equation numerically, it is convenient to split it into two ODE's of first order by introducing the velocity v as a new variable.

$$\dot{x}(t) = v(t)$$
$$\dot{v}(t) = -kx(t) - rv(t)$$

These equations can be simultaneously propagated.

Now use Euler's method, Runge-Kutta's method of 4th order, and Simpson's-Rule to find the same limit. You will have to experiment with different parameter sets (k, r). A total integration time of 100 timeunits should do. Plot your results (I suggest you use a logarithmic scale in order to be able to find the limit case)!

Generate a phase-space diagram (x vs $\dot{x}$) of the *frictionless* Harmonic Oszillator ($r = 0$) and compare the three methods for different stepsizes ($h = 0.001, 0.1, 0.5, 1, 10$) with the analytic solution!

Also compute and plot the total energy $E = T + U$ of the system against integration time t! T denotes the kinetic and U the potential energy at every moment t. What happens to E when you apply different stepsizes and different methods?

5 Requirements

One protokoll containing

- an introduction to the problems posed.
- answers to *each and every* question.
- *no source code!* Latter shall be sent to *siegfried.eggl@univie.ac.at* in form of a *compilable* textfile.

Please send the protocol as a PDF file to *siegfried.eggl@univie.ac.at*

References

Cash, J. R., Karp, A. H.: *A Variable Order Runge-Kutta Method for Initial Value Problems with Rapidly Varying Right-Hand Sides* ACM Transactions on Mathematical Software, Vol. 16, No. 3, *p. 201-222* (1990)

Vesely, F.: *Computational Physics - An Introduction* Springer US, *p.105 et sqq.* (2001)

Eggl,S., Dvorak,R. *An Introduction to Common Numerical Integration Codes Used in Dynamical Astronomy* , Lecture Notes in Physics Vol 790, Springer, *p. 431-477* (2010)