Exercises for the lecture

Fundamentals of Simulation Methods

WS 2016/17

Lecturer: Frauke Gräter and Rüdiger Pakmor

Exercise sheet 4 ***(due date: Nov 17, 2016, 11:59pm)***

Adaptive stepsize integration and 1D density particle mesh

1) Integration with an adaptive stepsize (30 points)

You want to test an integration algorithm with adaptive stepsizes by integrating the equation

$$\frac{d}{dt}x = f(t) = t^2 \tag{1}$$

with the initial value $x(t = 0) = 1$ up to $t = 1$. Your algorithm should have a maximum relative error 10^{-5}, estimated by evaluating the difference of the integration with two consecutive half timesteps and with the full timestep. Use the template `adaptive.c` (uploaded on moodle platform) with euler forward method to do so.

a) In the template `adaptive.c` the Euler intergration and an output are missing. Add these functions to the code and plot the result `x(t)`. Also write out the timestep and the relative error in every point.

b) Interpret your output and explain why the errors are unexpectedly large.

c) How can you change this? Explain in detail, which lines to edit and why. Correct the code and compare the new result to a).

d) Compute and add the analytical result $x(t)$ to your plot.

2) Particles in a discrete 1D density field (20 points)

The following two functions are part of a code which adds particles to a discrete 1D density field (first function), solves Poisson's equation in some way to calculate a discretised force field (not shown here) and maps the force field back to the particles (second function).

a) Assuming that all parts of the code which are not shown work perfectly fine, what problem will occur when running the code with the two functions shown below?

b) Modify the second function such that the above problem is solved. It's sufficient if you write pseudo code.

c) Do the functions work for periodic boundary conditions? If not, what would one have to change?

```c
void add_particle_to_density_field(double rho_field[N],/* the density field
    discretised on a grid of length N */
              int N,
              double cellsize, /* size of a single cell of the grid */
              double particle_pos, /* the particle position */
              double particle_mass) /* the particle mass */
{
  double xx = particle_pos / cellsize;
  int i = floor(xx); /* floor(x) truncates all decimal digits of the floating point
      number x, similar to ((int) x) in C (or: it does a strict rounding to the next
      lower integer number; floor(5.9) = 5) */

  double u = xx - i;

  int ii = i + 1;
  if(ii >= N)
    ii = 0;

  rho_field[i] += (1 - u) * particle_mass / cellsize;
  rho_field[ii] += u * particle_mass / cellsize;
}

double interpolate_force_field_to_particle_position(double force_field[N],/* the force
    field discretised on a grid of length N */
                    int N,
                    double cellsize, /* size of a single cell of the grid */
                    double particle_pos, /* the particle position */
                    double particle_mass) /* the particle mass */
{
  double xx = particle_pos / cellsize;

  int i = floor(xx + 0.5);

  double acceleration = force_field[i];

  return acceleration * particle_mass;
}
```

Fundamentals of Simulation methods
Solution Sheet 4

Bhavya Joshi & Florian Jörg

November 23, 2016

Exercise 1:

a)

After modification of the given script, we get the results given in table 1.

Table 1: ??

time	x_{Euler}	Δt	absolute error	x_{ana}
0	1	0.5	0	1
0.5	1.01562	0.25	0.015625	1.04167
0.75	1.0957	0.125	0.0175781	1.14062
0.875	1.17212	0.0625	0.00610352	1.22331
0.9375	1.22171	0.03125	0.0017395	1.27466
0.96875	1.24964	0.015625	0.000461578	1.30305
0.984375	1.26442	0.0078125	0.000118732	1.31795
0.992188	1.27202	0.00390625	3.01003e-05	1.32558
0.996094	1.27587	0.00390625	7.57724e-06	1.32944
1	1.27976	0.00390625	7.60704e-06	1.33333

b)

Interpretation of the outputs!

c)

Solution to the big errors?!

d)

The analytical solution follows from the integration using separation of variables and is given by

$$\begin{aligned} & \frac{d}{dt}x = t^2 \\ \Rightarrow \quad & \int dx = \int t^2 dt \\ \Rightarrow \quad & x(t) = \frac{1}{3}t^3 + c \end{aligned}$$

The integration constant c can be computed by using the condition that the given initial value $x(t = 0) = 1$ must be met. Therefore $c = 1$. And the analytical solution is given by

$$x(t) = \frac{1}{3}t^3 + 1 \tag{1}$$

Figure 1 shows the solution computed using the forward Euler Method together with the analytical solution given in equation 1.

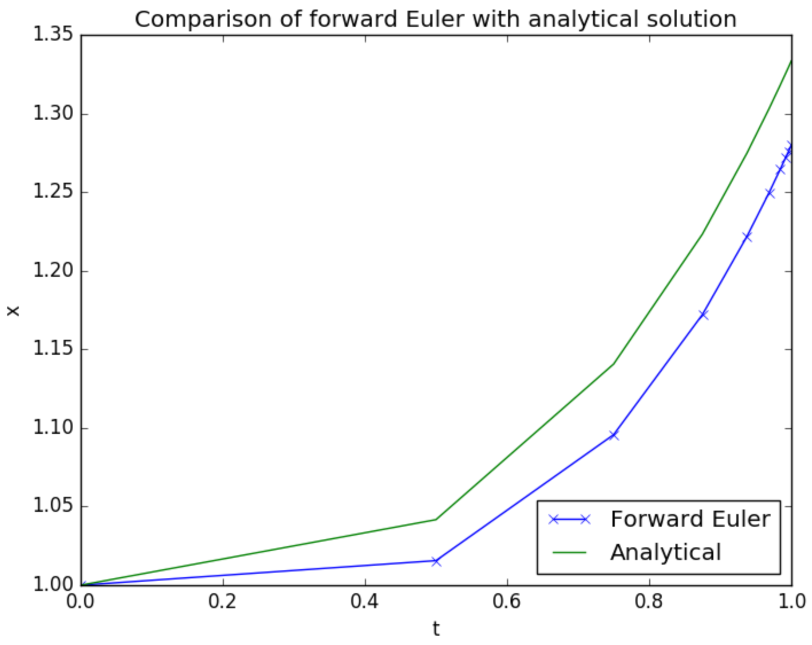

Figure 1: Comparison of the result of forward Euler method and analytical solution

Exercise 2:

a)

First - The first fuction is not returning the any value.
Second - The value of variable **xx** provided here will not fulfill our aim behind

the calculations.
Third - In second fuction accelaration did not find the right way.

b)

Correction is submitted in the first equation program named $ex42.c$

c)

For first fuction it works for periodic boudary condition, but for second fuction it will not work for boundary condition.

1 Solution$_classexercise_2$

we want to assign a mass to the grid point of the 1D grid.
So in given example, for position 5.9, we want to assign 10 percetage of mass to grid 5 and 90 percent of mass to gird 6.