Exercise set 11: Numerical Methods for ODEs

André Massing

Apr 14, 2020

The Python codes for this note are given in `ode.py`.

As always, we start by calling the necessary modules:

Exercise 1: Heun's method

a) Write down the Butcher tableau for Heun's method

b) Use the order conditions for Runge-Kutta methods to show that Heun's method has convergence order 2 but not 3.

Exercise 2: Implementation and Testing of Runge-Kutta methods

In this problem, you are asked to compare two 3-rd order Runge-Kutta methods. As as test case, consider the IVP

$$y'(t) = e^{y(t)}(1+t), \quad y(0) = -\frac{1}{2}, \tag{1}$$

which has the solution

$$y(t) = -\ln\left(e^{1/2} - t - \frac{t^2}{2}\right) \tag{2}$$

a) Implement the three-stage explicit Runge-Kutta known as Heun's third order method defined by the Butcher tableau

$$\begin{array}{c|ccc}
0 & & & \\
\frac{1}{3} & \frac{1}{3} & & \\
\frac{2}{3} & 0 & \frac{2}{3} & \\
\hline
 & \frac{1}{4} & 0 & \frac{3}{4}
\end{array} \tag{3}$$

Use the order conditions for Runge-Kutta methods to theoretically verify that this method is of 3rd order. Then run a convergence study by solving the IVP (1) numerically and calculate the experimentally observed convergence rate. Also write out the error for each time-step size you chose in your convergence rate study.

Hint. Start with recalling the general Runge-Kutta class

```
class Explicit_Runge_Kutta:
    def __init__(self, a, b, c):
        self.a = a
        self.b = b
        self.c = c

    def __call__(self, y0, t0, T, f, Nmax):
        # Extract Butcher table
        a, b, c = self.a, self.b, self.c
```

```
        # Stages
        s = len(b)
        ks = [np.zeros_like(y0, dtype=np.double) for s in range(s)]

        # Start time-stepping
        ys = [y0]
        ts = [t0]
        dt = (T - t0)/Nmax

        while(ts[-1] < T):
            t, y = ts[-1], ys[-1]

            # Compute stages derivatives k_j
            for j in range(s):
                t_j = t + c[j]*dt
                dY_j = np.zeros_like(y, dtype=np.double)
                for l in range(j):
                    dY_j += dt*a[j,l]*ks[l]

                ks[j] = f(t_j, y + dY_j)

            # Compute next time-step
            dy = np.zeros_like(y, dtype=np.double)
            for j in range(s):
                dy += dt*b[j]*ks[j]

            ys.append(y + dy)
            ts.append(t + dt)

        return (np.array(ts), np.array(ys))
```

and

```
def compute_eoc(y0, t0, T, f, Nmax_list, solver, y_ex):
    errs = [ ]
    for Nmax in Nmax_list:
        ts, ys = solver(y0, t0, T, f, Nmax)
        ys_ex = y_ex(ts)
        errs.append(np.abs(ys - ys_ex).max())
        print("For Nmax = {:3}, max ||y(t_i) - y_i||= {:.3e}".format(Nmax,errs[-1]))

    errs = np.array(errs)
    Nmax_list = np.array(Nmax_list)
    dts = (T-t0)/Nmax_list

    eocs = np.log(errs[1:]/errs[:-1])/np.log(dts[1:]/dts[:-1])
    return errs, eocs
```

b) Redo the previous exercise with Kutta's third order method defined by

$$
\begin{array}{c|ccc}
0 & & & \\
\frac{1}{2} & \frac{1}{2} & & \\
1 & -1 & 2 & \\
\hline
 & \frac{1}{6} & \frac{2}{3} & \frac{1}{6}
\end{array}
$$

Which of the two previous methods would you prefer?

c) Finally, implement the classical four-stage explicit Runge-Kutta given by

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array} \tag{4}$$

Show both theoretically and experimentally, that this is a 4-th order method.

Exercise 3: SIR Model

In the exercise you are asked to numerically solve the SIR model

$$S' = -\beta S I \tag{5}$$
$$I' = \beta S I - \gamma I \tag{6}$$
$$R' = \gamma I, \tag{7}$$

using a RKM of your choice. Recall that β denotes the infection rate, and γ the removal rate.

a) Show that any solution of the SIR system is conservative in the following sense:

$$S(t) + I(t) + R(t) = \text{const.} \tag{8}$$

b) Solve the SIR model for disease with $\beta = 10/(40 \cdot 8 \cdot 24)$ and $\gamma = 3/(15 \cdot 24)$. The dimensions for β and γ are 1/hour.

Start with 50 healthy individuals and 1 infected person. Simulate the spread of the disease for 30 days taking a time step of 6 minutes. Plot the final solution, that is, $S(t), I(t)$ and $R(t)$. Also check the conservation property for your chosen RKM by plotting $S(t) + I(t) + R(t)$ as well.