Jamie Grieser 29.10.2019.

Simulation Methods - Problem Set 2

Ex 01 Order of an integration scheme

For a numerical solution y_{num} to be second-order accurate, it is necessary that $\| y - y_{num} \| = \mathcal{O}(h^2)$ where y is the exact solution and h is the step size.

Thus, we calculate $y_{n+1} - y(t_n + \Delta t)$ where Δt is the step size:

We begin with the Taylor expansion of $y(t_n + \Delta t)$ around t_n:

$$\Rightarrow \quad y(t_n + \Delta t) = y(t_n) + \left.\frac{\partial y}{\partial t}\right|_{t=t_n} \Delta t + \mathcal{O}(\Delta t^2) = y_n + \left.\frac{\partial y}{\partial t}\right|_{t_n} + \mathcal{O}(\Delta t^2)$$

Plugging this in yields:

$$y_{n+1} - y_n - \left.\frac{\partial y}{\partial t}\right|_{t_n} + \mathcal{O}(\Delta t^2)$$

Now we use the definition of the Runge-Kutta scheme to rewrite y_{n+1} and simplify:

$$\left[\frac{1}{2}(k_1 + k_2) - \left.\frac{\partial y}{\partial t}\right|_{t_n}\right]\Delta t + \mathcal{O}(\Delta t^2) \quad ; \quad k_1 = f(t_n, y_n) = \left.\frac{\partial y}{\partial t}\right|_{t_n}$$

$$= \left(\frac{k_2}{2} - \frac{1}{2}\left.\frac{\partial y}{\partial t}\right|_{t_n}\right)\Delta t + \mathcal{O}(\Delta t^2)$$

We calculate another Taylor expansion:

$$k_2 = f(y_n + k_1 \Delta t, t_n + \Delta t) = f(y_n, t_n) + \left.\frac{\partial f}{\partial y}\right|_{y=y_n} k_1 \Delta t + \left.\frac{\partial f}{\partial t}\right|_{t=t_n} \Delta t + \mathcal{O}(\Delta t^2)$$

$$\Rightarrow \quad \frac{1}{2}\left(k_2 - \left.\frac{\partial y}{\partial t}\right|_{t_n}\right)\Delta t = \frac{1}{2}\left(k_2 - f(y_n, t_n)\right)\Delta t$$

$$= \left[\left.\frac{\partial f}{\partial y}\right|_{y=y_n} \Delta t + \left.\frac{\partial f}{\partial t}\right|_{t=t_n} + \mathcal{O}(\Delta t^2)\right]\Delta t + \mathcal{O}(\Delta t^2)$$

$$= \mathcal{O}(\Delta t^2)$$

$\Rightarrow$ Heuns method is of 2^{nd} order accuracy.

Ex 02 Double Pendulum

(a) Lagrangian of the double pendulum:

$$\mathcal{L} = \frac{m_1}{2}(l_1\dot{\phi}_1)^2 + \frac{m_2}{2}\left[(l_1\dot{\phi}_1)^2 + (l_2\dot{\phi}_2)^2 + 2l_1l_2\dot{\phi}_1\dot{\phi}_2\cos(\phi_1-\phi_2)\right]$$
$$- m_1gl_1(1-\cos\phi_1) - m_2g\left[l_1(1-\cos\phi_1) + l_2(1-\cos\phi_2)\right]$$

Conjugate momenta:

$$q_1 = \frac{\partial\mathcal{L}}{\partial\dot{\phi}_1} \; ; \; q_2 = \frac{\partial\mathcal{L}}{\partial\dot{\phi}_2}$$

$$\Rightarrow q_1 = m_1l_1^2\dot{\phi}_1 + m_2l_1^2\dot{\phi}_1 + m_2l_1l_2\dot{\phi}_2\cos(\phi_1-\phi_2) \quad (1)$$
$$q_2 = m_2l_2^2\dot{\phi}_2 + m_2l_1l_2\dot{\phi}_1\cos(\phi_1-\phi_2) \quad (2)$$

Derivatives after ϕ_1 and ϕ_2:

$$\frac{\partial\mathcal{L}}{\partial\phi_1} = -m_2l_1l_2\dot{\phi}_1\dot{\phi}_2\sin(\phi_1-\phi_2) - (m_1+m_2)gl_1\sin(\phi_1)$$
$$\frac{\partial\mathcal{L}}{\partial\phi_2} = +m_2l_1l_2\dot{\phi}_1\dot{\phi}_2\sin(\phi_1-\phi_2) - m_2gl_2\sin(\phi_2)$$

Thus we find the equations of motion to be:

$$\frac{d}{dt}q_1 = +m_2l_1l_2\dot{\phi}_1\dot{\phi}_2\sin(\phi_1-\phi_2) - (m_1+m_2)gl_1\sin(\phi_1)$$
$$\frac{d}{dt}q_2 = m_2l_1l_2\dot{\phi}_1\dot{\phi}_2\sin(\phi_1-\phi_2) - m_2gl_2\sin(\phi_2)$$

(b) From equations (1) and (2), we can obtain expressions for $\dot{\phi}_1$ and $\dot{\phi}_2$:

$$q_1 = (m_1+m_2)l_1^2\dot{\phi}_1 + m_2l_1l_2\dot{\phi}_2\cos(\phi_1-\phi_2)$$
$$q_2 = m_2l_2^2\dot{\phi}_2 + m_2l_1l_2\dot{\phi}_1\cos(\phi_1-\phi_2)$$

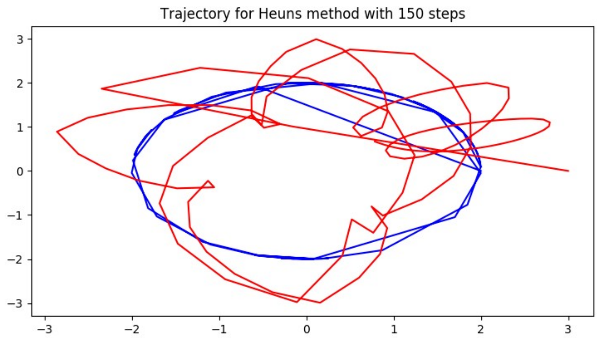
This a 2x2 system of linear equations. The solution reads:

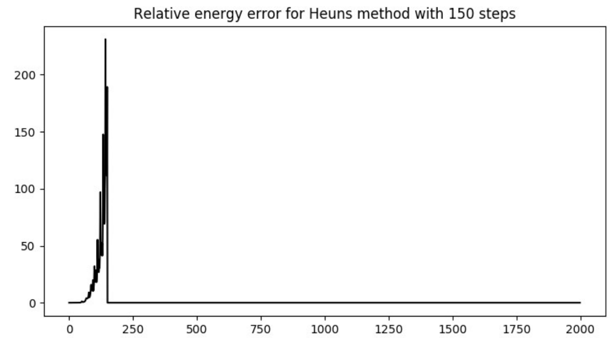
$$\dot{\phi}_1 = \frac{m_2l_2^2q_1 - m_2l_1l_2\cos(\phi_1-\phi_2)q_2}{m_2(m_1+m_2)l_1^2l_2^2 - m_2^2l_1^2l_2^2\cos^2(\phi_1-\phi_2)} =: f_1(q_1,q_2,\phi_1,\phi_2)$$
$$\dot{\phi}_2 = \frac{(m_1+m_2)l_1^2q_2 - m_2l_1l_2\cos(\phi_1-\phi_2)q_1}{m_2(m_1+m_2)l_1^2l_2^2 - m_2^2l_1^2l_2^2\cos^2(\phi_1-\phi_2)} =: f_2(q_1,q_2,\phi_1,\phi_2)$$

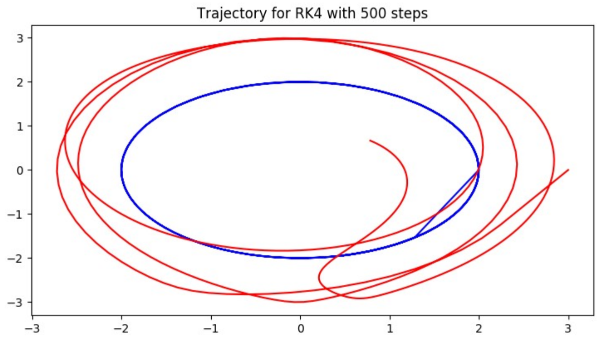
We can then express $\dot{q}_1$ and $\dot{q}_2$:

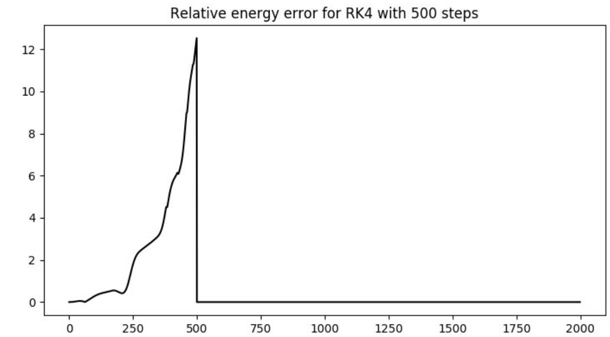
$$\dot{q}_1 = -m_2l_1l_2f_1f_2\sin(\phi_1-\phi_2) - (m_1+m_2)gl_1\sin\phi_1 =: f_3(q_1,q_2,\phi_1,\phi_2)$$
$$\dot{q}_2 = m_2l_1l_2f_1f_2\sin(\phi_1-\phi_2) - m_2gl_2\sin(\phi_2) =: f_4(q_1,q_2,\phi_1,\phi_2)$$

(c) + (d) see source code in .py file









Comment on exercises (c) and (d):

I implemented the schemes as given in the lecture, but each time I ran the simulation it broke down after roughly 150 steps (500 for RK4). The plots above show the behavior of the system until the breakdown. The trajectory looks meaningful, but the relative energy error goes well beyond the expected limits (should be between 0 and 1)
I do not know what exactly the problem is, but I triple-checked my code and calculations and could not find any errors.

Still, the error seems to be much smaller for the RK4 scheme compared to the Heun scheme, which is also what I would expect as Heun's scheme is second order accurate while RK4 is fourth-order accurate.