Computational Tools

Delivery Date: 31/01/2022

Rules: One of the projects should be in Mathematica and the other one in Python.

1 Project 1: Solving Kepler's Equation

Kepler's equation

$$M = E - e\sin E \tag{1}$$

is transcendental in E, i.e. given M a solution in E cannot be expressed in closed form. However, during the years there have been developed a variety of numerical and analytical methods to compute the solutions.

• **TASK 1:** Write a root finding algorithm for Kepler's equation using the Newton-Raphson method. Compute the values of $E(M)$ with $M \in [0, 2\pi]$ for eccentricities $e = 0.1, 0.3, 0.5, 0.7, 0.9$ and make a plot to present your results.

In 1770, Lagrange's work on Kepler's equation led to a generally useful expansion theorem. Given an equation of the form $y = x + \alpha\phi(y)$, its solution is approximated by the series expansion:

$$y = x + \sum_{n=1}^{\infty} \frac{\alpha^n}{n!} \frac{d^{n-1}}{dx^{n-1}} \phi(x)^n \tag{2}$$

• **TASK 2:** Apply Langrage's theorem to Kepler's equation and write a code that computes the series solution. Use your program to compute the series expression $E(M)$ by keeping terms up to order $n = 3$ and $n = 10$. Do a trigonometric reduction of the outcome, such that the series takes the form $E = M + \sum_n \Pi_n(e)\sin(nM)$, with $\Pi_n(e)$ polynomials of e. Evaluate the order three ($n = 3$) and order ten ($n = 10$) series for $e = 0.3$ and compare it with the numerical results of TASK 1. Do the same for $e = 0.9$. Discuss your results.

Another representation of the solution in E is given by re-arranging the terms as a Fourier-Bessel series expansion:

$$E = M + \sum_{j=1}^{\infty} \frac{2}{j} J_j(je) \sin jM \tag{3}$$

where the coefficients J_n are Bessel functions of the first kind, defined by the infinite series

$$J_n(x) = \sum_{k=0}^{\infty} \frac{(-1)^k}{k!(n+k)!} \left(\frac{x}{2}\right)^{n+2k} \tag{4}$$

• **TASK 3:** Write a program that computes the Bessel functions $J_n(x)$ and make a plot of the functions $J_1, J_2, \ldots J_5$ for $x \in [0, 15]$. Use your Bessel functions to implement the Fourier-Bessel expansion of E and compute the series keeping terms up to $j = 3$ and $j = 10$. For $e = 0.3$ evaluate $E(M)$ from the order three ($j = 3$) and order ten ($j = 10$) series and compare with the numerical results of TASK 1. Do the same for $e = 0.9$. What do you observe? Compare with the results of the other tasks and discuss.

2 Project 2: Rigid-body dynamics

The free rotation of a rigid body is governed by a set of three first order differential equations (Euler's equations):

$$\begin{aligned} I_1\dot{\omega_1}(t) &= (I_2 - I_3)\omega_2(t)\omega_3(t) \\ I_2\dot{\omega_2}(t) &= (I_3 - I_1)\omega_3(t)\omega_1(t) \\ I_3\dot{\omega_3}(t) &= (I_1 - I_2)\omega_1(t)\omega_2(t) \end{aligned}$$

Following the laws of conservation for the energy and the angular momentum, two integrals of Euler's equations exist:

$$I_1\omega_1^2 + I_2\omega_2^2 + I_3\omega_3^2 = 2E$$
$$I_1^2\omega_1^2 + I_2^2\omega_2^2 + I_3^2\omega_3^2 = M^2$$

where E is the rotational kinetic energy, and M the magnitude of the angular momentum. Using the two integrals, Jacobi managed to solve Euler's equations. The solution is given in terms of the Jacobi elliptic functions:

$$\omega_1(t) = \sqrt{\frac{2EI_3 - M^2}{I_1(I_3 - I_1)}}cn(\tau, k^2)$$
$$\omega_2(t) = \sqrt{\frac{2EI_3 - M^2}{I_2(I_3 - I_2)}}sn(\tau, k^2)$$
$$\omega_3(t) = \sqrt{\frac{M^2 - 2EI_1}{I_3(I_3 - I_1)}}dn(\tau, k^2)$$

where

$$\tau = t\sqrt{\frac{(I_3 - I_2)(M^2 - 2EI_1)}{I_1 I_2 I_3}}, \quad k^2 = \frac{(I_2 - I_1)(2EI_3 - M^2)}{(I_3 - I_2)(M^2 - 2EI_1)}$$

• **TASK 1:** Write a program that computes the analytical solution for the free rotation. Draw the solution $\omega_1(t), \omega_2(t), \omega_3(t)$ for a body with $I_1 = 0.8, I_2 = 0.9, I_3 = 1.0$ and initial conditions $\omega_1(0) = 1.0, \omega_2(0) = 0.0, \omega_3(0) = 2.0$ up to $t_{max} = 100$. Compute the relative error in the energy $log(|(E(t) - E(0))/E(0)|)$ for the analytic solution.

Another way to compute the solutions of Euler's equation is via numerical integration of the first order differential equations. One commonly used method is the Runge-Kutta of 4th order.

• **TASK 2:** Write a program that solves Euler's equations with the Runge-Kutta method of 4th order. Draw the solution $\omega_1(t), \omega_2(t), \omega_3(t)$ for the same parameters and initial condition as in TASK 1, up to $t_{max} = 100$ with a time-step of $dt = 0.1$. Compute the relative error $log(|(E(t) - E(0))/E(0)|)$ for the RK4 numerical solution. What do you observe?

Finally, there exist splitting methods for the solution of the free rigid body motion. In these methods the total problem can be split into components that can be solved exactly. A simple splitting for the Euler's problem is the following:

$$H = \frac{1}{2}\left(\frac{M_1^2}{I_1} + \frac{M_2^2}{I_2} + \frac{M_3^2}{I_3}\right) = H_{M_1} + H_{M_2} + H_{M_3}, \quad \text{with} \quad H_{M_1} = \frac{M_1^2}{2I_1}, H_{M_2} = \frac{M_2^2}{2I_2}, H_{M_3} = \frac{M_3^2}{2I_3}$$

where $\mathbf{M} = (M_1, M_2, M_3) = (I_1\omega_1, I_2\omega_2, I_3\omega_3)$ is the angular momentum vector. Each of H_{M_i} can be solved exactly and the solutions are given from:

$$\Phi^{H_{M_1}} : \mathbf{M}(t) = R_x(M_1(0)t/I_1)\mathbf{M}(0)$$
$$\Phi^{H_{M_2}} : \mathbf{M}(t) = R_y(M_2(0)t/I_2)\mathbf{M}(0)$$
$$\Phi^{H_{M_3}} : \mathbf{M}(t) = R_z(M_3(0)t/I_3)\mathbf{M}(0)$$

where R_x, R_y, R_z are the clockwise rotation matrices about the x, y and z-axis respectively. A step dt of the splitting method consists of combining the exact solution for each H_{M_i} with the following order:

$$\Phi^H(dt) = \Phi^{H_{M_1}}(dt/2) \circ \Phi^{H_{M_2}}(dt/2) \circ \Phi^{H_{M_3}}(dt) \circ \Phi^{H_{M_2}}(dt/2) \circ \Phi^{H_{M_1}}(dt/2) + \mathcal{O}(dt^3).$$

• **TASK 3:** Write a program that solves Euler's equations with the splitting method. Draw the solution $\omega_1(t), \omega_2(t), \omega_3(t)$

for the same parameters and initial condition as in TASK 1, up to $t_{max} = 100$ with a time-step of $dt = 0.02$. Compute the relative error $log(|(E(t) - E(0))/E(0)|)$ for the splitting method solution. What do you observe? Compare with the results of the other tasks and discuss.

Hints

- Kepler's equation is a particular case of the Lagrange's theorem where $y = E$, $x = M$, $\alpha = e$ and $\phi = sin$.
- The order (i.e. n,j) of a series is the number of terms (i.e. n_{max},j_{max}) we add in the sum.
- In the implementation of the Bessel functions k_{max} cannot be infinite in a computer, but it can be either a large number (be aware of overflows in the calculation of the factorials), or optimally one controls the size of the terms being added and truncates at a sufficient order.
- For the Bessel functions of first kind, Mathematica has the BesselJ[] and python has scipy.special.jv(), so you can test your implementations.
- The Jacobi elliptic functions in Mathematica are calculated from JacobiCN[], JacobiSN[] and JacobiDN[]. In python all three of them are given from scipy.special.ellipj().
- The symbol $\circ$ denotes the composition of two functions/operators, i.e. $f \circ g = f(g(x))$. In the splitting method, the output of the exact solution for a partial step $\Phi_{H_{M_i}}$ is used as an initial condition for the next partial step and so on.
- The matrices $R_x(\theta)$, $R_y(\theta)$, $R_z(\theta)$ are:

$$R_x(\theta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & \sin(\theta) \\ 0 & -\sin(\theta) & \cos(\theta) \end{pmatrix}, R_y(\theta) = \begin{pmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{pmatrix}, R_z(\theta) = \begin{pmatrix} \cos(\theta) & \sin(\theta) & 0) \\ -\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$