# Numerical Solutions of Differential Equations - Project #2

**due 2021 JAN 19, 2:55 p.m.**

## 1 The context

The restricted three-body system is a model for studying the motion of an earth-moon satellite or of an asteroid close enough to the earth to be strongly influenced by the earth and the sun.

In this system, the two heavy bodies are regarded as revolving in fixed orbits about their common centre of mass while the small body is attracted by the heavy bodies but never affecting their motion. To further simplify the problem, we approximate orbits of the heavy bodies as circles so that we can work in a frame of reference that rotates with the two heavy bodies. In this frame, the two heavy bodies are considered as fixed in space with their rotation translated into a modification of the equations of gravitational motion. For simplicity, we scale the units to reduce a number of constants to one. We also scale the masses of the two heavy bodies to $1-\mu$ and $\mu$ and their positions relative to the moving reference frame by the vectors $\mu e_1$ and $(\mu-1)e_1$ respectively so that their center of mass is at the origin of coordinates.

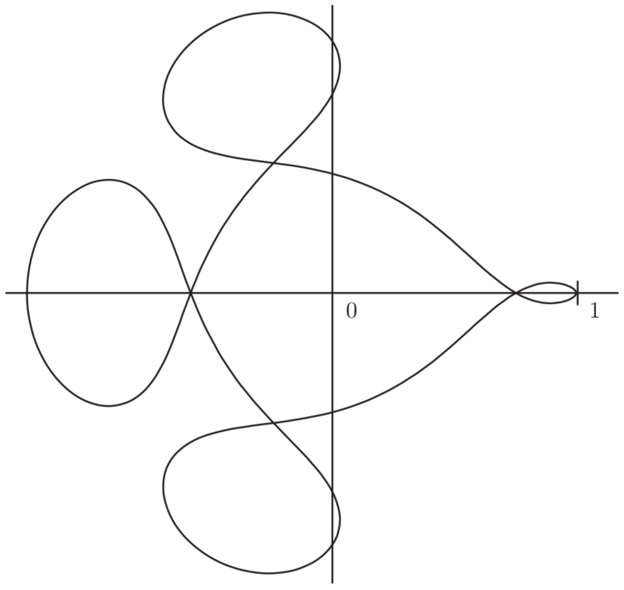
Denote by $u_1$, $u_2$ and $u_3$ the scalar variables representing the position coordinates of the small body and by $u_4$, $u_5$ and $u_6$ the corresponding velocity coordinates. The equations of motion are

$$\left\{\begin{array}{rl} u_1' = & u_4, \\ u_2' = & u_5, \\ u_3' = & u_6, \\ u_4' = & 2u_5 + u_1 - \frac{\mu(u_1+\mu-1)}{(u_2^2+u_3^2+(u_1+\mu-1)^2)^{3/2}} \\ & - \frac{(1-\mu)(u_1+\mu)}{(u_2^2+u_3^2+(u_1+\mu)^2)^{3/2}}, \\ u_5' = & -2u_4 + u_2 - \frac{\mu u_2}{(u_2^2+u_3^2+(u_1+\mu-1)^2)^{3/2}} \\ & - \frac{(1-\mu)u_2}{(u_2^2+u_3^2+(u_1+\mu)^2)^{3/2}}, \\ u_6' = & - \frac{\mu u_3}{(u_2^2+u_3^2+(u_1+\mu-1)^2)^{3/2}} \\ & - \frac{(1-\mu)u_3}{(u_2^2+u_3^2+(u_1+\mu)^2)^{3/2}}. \end{array}\right. \tag{1}$$

Planar motion is possible; that is, solutions which satisfy $u_3 = u_6 = 0$ at all times. One of these is shown below, with the values of $(u_1, u_2)$ plotted as the orbit evolves. The heavier mass is at the point $(-\mu, 0)$ and the lighter mass is at $(1-\mu, 0)$, where $(0, 0)$ is marked 0 and $(1, 0)$ is marked 1. For this calculation the value of $\mu = 1/81.45$ was selected, corresponding to the earth-moon system. The initial values for this computation were

$$\begin{array}{rl} & (u_1, u_2, u_3, u_4, u_5, u_6) \\ = & (0.994, 0, 0, 0, -2.0015851063790825224, 0) \end{array} \tag{2}$$

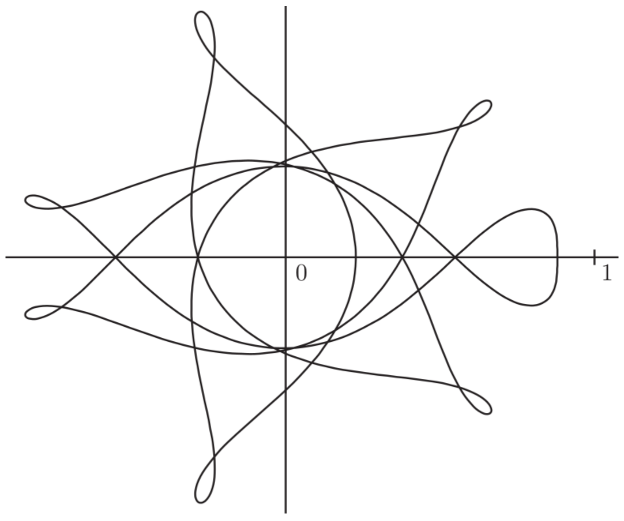
and the period was $T_1$ =17.06521656015796.



The plot below illustrates another solution of (1) with a different initial condition

$$\begin{array}{rl} & (u_1, u_2, u_3, u_4, u_5, u_6) \\ = & (0.87978, 0, 0, 0, -0.3797, 0) \end{array} \tag{3}$$

and a different period $T_2$ =19.14045706162071.



## 2 The assignment

Write a `C++` package to implement the following methods,

- Adams-Bashforth methods ($p = 1, 2, 3, 4$);
- Adams-Moulton methods ($p = 2, 3, 4, 5$);
- BDFs ($p = 1, 2, 3, 4$);
- the classical Runge-Kutta method.

You must use a *factory* pattern such as those discussed in the book by A. Alexandrescu, *Modern C++ design*, Addison-Wesley, 2001.

You must test each method against the restricted three-body system with (2) and (3) in Section 1. Since the solutions are periodic, you should compute the solution to $T_i$, $i = 1, 2$, with an appropriately chosen time-step size and with the initial condition as the exact solution at $T_i$ for calculating the solution errors. For each method and for each test case, you are supposed to

(a) write an input file where the name and order of the method are parsed in the `main` program to generate an object of (a derived class of) the class `TimeIntegrator` by the singleton object `TimeIntegratorFactory`,

(b) perform a sequence of grid refinement test where solution errors, convergence rates, and CPU time are reported to demonstrate convergence with a correct order of accuracy (this is the criterion of whether your test passes or fails!),

(c) report a time-step size with which the plot of the solutions is visually indistinguishable from the corresponding plot in Section 1.

You should organize your results in a coherent story that relates your numerical results to the math theory of IVP problems. Your story should contain two plots of your solution to (2): the Euler's method with 24000 steps and the classical Runge-Kutta methods with 6000 steps. In addition, to achieve an error of $10^{-3}$ based on the max-norm of the solution error, which method is the winner in terms of total CPU time?

**Extra credits**: nontrivial adaptive time stepping with theory and results.

## 3 How to submit

Your submission must contain

(a) the LaTeX source code and its Makefile so that the command "`make story`" generates a document that contains the story required in Section 2,

(b) a `C++` package so that the command "`make run`" would trigger the compilation of your source code, the production of the executable, the running of your tests, the display of test results, and even the generation of the elements in your story.

You should archive your source code in a single gzipped tar ball (**format**: `YourName_project2.tar.gz`) and send it to the TA's email. A number of tips are given as follows.

(i) You can use either `GNU Make` or `cmake` or a mixture of them.

(ii) You may use either `GNU plot` or `matlab` to plot your results.

(iii) You can use Chinese or English for the story document.

(iv) Your gzipped tar ball should neither contain anything that can be generated from your Makefile, nor contain anything irrelevant to this homework. In other words, your answers to this homework should be both *sufficient* and *necessary.*

(v) You are encouraged to use a unit test framework such as `CppUnit`; of course you can choose your own unit-test framework as you see fit.