Updated on March 29, 2021

Math 3607: Exam 3 (Written)

Due: 11:59PM, Friday, April 2, 2021

- You may use any function built into core MATLAB (not any toolboxes), any of the book functions (quote), any of functions provided in lectures, and any of your own helper functions. As stated in the syllabus, you must be able to explain how they are supposed to work.
- If a certain part of your script does not run successfully, you may receive no credit on that part of the exam. If any code is found to be plagiarized from the internet or another person, you may receive a zero on the *entire* exam (both parts).
- Vectorize as much as possible and avoid using loops if not needed.

1 Asteroids [25 points]

The Kepler laws characterize the motion of celestial bodies such as asteroids in the solar system in terms of several parameters. Considering the motion within the plane of the orbit, they are related by the equations

$$\tan\frac{\nu}{2} = \sqrt{\frac{1+\epsilon}{1-\epsilon}}\,\tan\frac{\psi}{2}, \tag{1}$$

$$M = \psi - \epsilon\sin\psi, \tag{2}$$

$$r = \frac{a(1-\epsilon^2)}{1+\epsilon\cos\nu} \tag{3}$$

$$a^3 = \mu\left(\frac{\tau}{2\pi}\right)^2, \tag{4}$$

where

- τ (`tau`): the period of the orbit
- ϵ (`epsilon`): the eccentricity of the ellipse
- ψ (`psi`): the eccentric anomaly
- ν (`nu`): the true anomaly, that is, the angle between the body and its *perihelion*[1]
- M: the mean anomaly; proportional to time and $M \in [0, 2\pi]$ over one complete cycle
- r: the distance from the body to the Sun
- a: half of the (maximum) diameter of the ellipse
- μ (`mu`): gravitational parameter; for the Sun, it is 39.47524 AU^3/yr^2.

Two of the parameters, τ and ϵ are readily observed. Equation (2) implicitly defines ψ as a function of M (and therefore time), but it cannot be solved in closed form. Instead, given a value of M, rootfinding must be used to find ψ. The rest of the parameters in the equations can be calculated explicitly from ψ.

In this problem, the eccentric anomaly ψ is computed given the orbital period and eccentricity of an asteroid using `fzero` in MATLAB. From this you can computer other quantities of interest.

[1] The closest approach to the Sun.

(a) Let `M` be a vector of 800 evenly spaced values from 0 to 2π. Set $\epsilon = 0.1$ and solve (2) using `fzero` to compute a value of ψ for each entry of M. Make a labeled plot of $\psi(M)$.

(b) Step through $\epsilon = 0.2, 0.3, \ldots, 0.9$ and plot the resulting $\psi(M)$ each time. Superimpose all plots to the plot from part (a).

(c) The asteroid 324 Bamberga is both one of the largest and one of the most eccentric in the asteroid belt. It has $\epsilon = 0.338$ and $\tau = 4.40$ yr. Plot $\nu(M)$ over $0 \leqslant M \leqslant 2\pi$ for this asteroid. (You will probably find that ν jumps suddenly from π to $-\pi$. As mentioned in a problem solving session, it is not incorrect, but if this bothers you, use `unwrap(nu)` to get a smooth equivalent in $[0, 2\pi]$.)

(d) Halley's Comet has $\epsilon = 0.967$ and $\tau = 75.3$ yr. Plot $r(M)$ for this comet. What are its maximum (*aphelion*) and minimum (*perihelion*) values[2]? (You can check these figures against ones you find on the internet.)

2 Numerical Integration Using Cubic Spline [25 points]

The goal of this problem is to write a program to estimate $\int_a^b f(x)\,dx$, assuming that the values of f at only certain points $a = x_1 < x_2 < \cdots < x_n = b$ are known. The idea of approximation is simple:

- Approximate f by an interpolating *not-a-knot* cubic spline pp.
- Calculate the integral of pp exactly.

That is,

$$\int_a^b f(x)\,dx \approx \int_a^b \mathit{pp}(x)\,dx.$$

All notation on pp used below is consistent with the one used in lectures.

(a) [By hand] Since pp is a piecewise cubic polynomial, one can integrate it explicitly by hand. Write down the formula for

$$\int_a^b \mathit{pp}(x)\,dx$$

in terms of $c_{i,j}$ and $\Delta x_i = x_{i+1} - x_i$. Justify all steps and simplify the final result as cleanly as possible. (Can you interpret the result in terms of polynomial evaluation? This will be extremely useful for the next part.)

(b) [Computer] The coefficients $c_{i,j}$ can be calculated and stored as an $(n-1) \times 4$ matrix

$$C = \begin{bmatrix} c_{1,1} & c_{1,2} & c_{1,3} & c_{1,4} \\ c_{2,1} & c_{2,2} & c_{2,3} & c_{2,4} \\ \vdots & \vdots & \vdots & \vdots \\ c_{n-1,1} & c_{n-1,2} & c_{n-1,3} & c_{n-1,4} \end{bmatrix} \tag{5}$$

simply by

```
pp = spline(xdp, ydp);
C = fliplr( pp.coefs );
```

Use this to write a function `csinteg` which approximates the integral $\int_a^b f(x)\,dx$ by the formula found in the previous part.

[2]The maximum r-value is called the *aphelion* and the minimum r-value the *perihelion*

```
function I = csinteg(xdp, ydp)
% CSINTEG I = csinteg(xdp, ydp)
% Estimates the integral of  f  using a not-a-knot cubic spline interpolant.
% Inputs:
%   xdp   knots; the first and the last determine the interval of integration
%   ydp   function evaluation at xdp



end
```

Note. The key segment of the code is the implementation of the formula derived in part (a). This also happens to be the most expensive part of this code. So do this as efficiently as possible and **vectorize** as much as you can. In particular, AVOID using nested loops. (*Hint.* Consider using `polyval`.)

(c) Test your program by calculating

$$\int_0^2 e^{\sin 7x}\, dx = 2.663219782761539\ldots$$

using n equispaced nodes on $[0, 2]$ for $n = 5, 9, 17, 33$. Plot the log-log graph of the errors against n. Based on the experiment, what is the order of accuracy of the method?