Introduction to Scientific Computing Lab 8

MATLAB III

2017

1 Objectives

After completing these exercises you will be able to:

- Write a more in-depth problem using MATLAB

2 Notes

Work individually.

You have two lab sessions to complete this exercise. If you do not manage to fully complete it before the end of the second session, ensure an invigilator comes to check your work so you can get some of the available marks.

When you have completed all exercises, ask a demonstrator to assess your work. They will test your code and ensure it is formatted well with good commenting, structure and variable names.

3 Simulating Motion of a Satellite

Communication has been lost with a satellite. Radar has, however, detected a number of past locations of the satellite. You task is to estimate the path of the satellite using two techniques, and estimate how close it will come to the Earth's surface. The two techniques you will use are based on the two methods that you have seen in the previous two MATLAB examples. These are

- fitting an ellipse (typical orbit shape) to the radar data
- simulating the motion using integration of physical laws

Download `radardata.mat` and `ellipsepoints.m` from blackboard. You will need to use these accompanying files.

When loaded into blackboard, `radardata.mat` contains the x and y coordinates of the radar contact points within the `xs` and `ys` vectors. It also contains a third vector `v` which is the x and y velocity components of the satellite at the first radar contact.

Stage 1: Plot the Locations of the Radar Points

Write a MATLAB script that:

- Plots the radar contact points as blue dots
- Plots the surface of the Earth as a green circle centred at (0,0)
- Ensure your graph displays the Earth as a circle i.e. the axes must be aspect ratio one

NOTE: the Earth has a radius of approximately 6.38×10^6m.

Stage 2: Fit an Ellipse to the Contact Points

The first method you will use to approximate the path of the satellite is to fit an ellipse to the measured data points. An ellipse satisfies the following equation:

$$ax^2 + by^2 + cxy + dx + ey = 1 \tag{1}$$

The file `ellipsepoints.m` contains a function that generates points on an ellipse given the parameters a, b, c, d and e (use '`help ellipsepoints`' for information).

Extend your script to:

- determine the value of a, b, c, d and e to fit the radar locations
- plot the ellipse as a red line over the points plotted in stage 1

Stage 3: Simulate the Motion of the Satellite

The second method you will use is finite difference integration of fundamental physical laws. The acceleration vector acting on body i (the satellite) due to body j (the Earth) is given by:

$$\mathbf{a} = \frac{Gm_j(\mathbf{x}_j - \mathbf{x}_i)}{\|r_{ij}\|^3} \tag{2}$$

where $G = 6.67 \times 10^{-11}\text{m}^3/\text{kgs}^2$ is the gravitational constant, $m_j = 5.97 \times 10^{24}$kg is the mass of body j, $\mathbf{x}_j$ and $\mathbf{x}_i$ are the location vectors of the two bodies, and $\|r_{ij}\|$ is the Euclidean distance between the centre of mass of the two bodies.

Extend your script to:

- Simulate the motion of the satellite using integration of the physical law outlined above starting from the first contact location (remember the velocity of this point is contained within `radardata.mat`). Use the forward difference method, and simulate for 210,000s in 100s intervals.
- Plot the resulting path as a magenta line over the first two stages

Comments, Formatting and Figure

Ensure your code is well commented and formatted. Also ensure your graph has axes labels, a legend, sensible limits and gridlines. E.g. (NOTE: this includes the optional extension; see below)

Optional: Find the Point of Closest Approach

Extend your script to:

- Find the overall minimum distance that is predicted (remember you have two prediction methods) for how close the satellite will pass to the surface of the Earth.
- Display this as a black star and display the final number to the user either on the graph or in the command window