Computer Practical: Solar System Model

Paul Connolly, September 2015

1 Overview

In this computer practical, a solar system model implemented in MATLAB is used to demonstrate analysis of time-series data using Fourier methods and understand interactions between planets and stars. Fourier methods are often used to analyse time-series that are periodic (e.g. seismology).

2 Model formulation

The three-dimensional model is implemented in MATLAB. MATLAB solves for the motion of the planets and writes the output into data lists called 'arrays'. It predicts and stores the position and velocity in all three dimensions of all the planets and the sun. The model applies both Newton's Second Law of Motion ($\vec{F} = m\vec{a}$) and Newton's Law of Universal Gravitation:

$$\vec{F} = -G\frac{m_1 m_2}{r^3}\vec{r} \tag{1}$$

Consider a mass, m_1, and mass, m_2, separated in space by distance:

$$r_{1,2} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2 + (z_1 - z_2)^2} \tag{2}$$

Note that this is just the distance between two points at (x_1, y_1, z_1) and (x_2, y_2, z_2) respectively. Newton's law of Gravitation says the magnitude of the force, $|\vec{F_{1,2}}|$ between these two masses is:

$$|\vec{F_{1,2}}| = -G\frac{m_1 m_2}{r_{1,2}^2} \tag{3}$$

where $G = 6.67 \times 10^{-11}$ m^3 kg^{-1} s^{-2} is the gravitational constant. We need to find the force acting along the x, y and z axes in three dimensions, so we apply Equation 1 to get:

$$F_{x,1,2} = -G\frac{m_1 m_2}{r_{1,2}^3} \times (x_2 - x_1) \tag{4}$$

$$F_{y,1,2} = -G\frac{m_1 m_2}{r_{1,2}^3} \times (y_2 - y_1) \tag{5}$$

$$F_{z,1,1} = -G\frac{m_1 m_2}{r_{1,2}^3} \times (z_2 - z_1) \tag{6}$$

$$\tag{7}$$

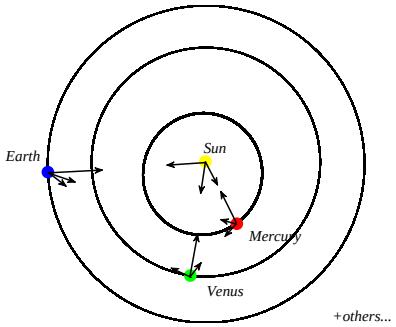

Figure 1: Schematic of the scenario being modelled, where the arrows represent the forces each body experiences due to the interaction with all other bodies.

Table 1: Parameters controlling the behaviour of the model.

Variable	Default value	Description
`which_bodies`	`ALL_BODIES`	which planets (and sun) to consider in the computations
`which_interactions`	`INTERACT_WITH_SUN_ONLY`	specify which bodies interacting with each other
`which_star`	`MOVING_SUN`	Whether to allow sun to wobble or be fixed
`which_output`	`FIRST`	Save the variables to different names
`run_time`	`1000`	the length of the simulation in earth years.
`plot_figures`	`false`	plot the figures as png files when doing animation
`plot_stride`	`10`	the number of time-levels to skip when plotting.

Table 2: Approximate orbital periods of each of the planets.

Planet	Period in earth years
Mercury	0.2
Venus	0.6
Earth	1.0
Mars	1.9
Jupiter	11.9
Saturn	29.5
Uranus	84.0
Neptune	164.8
Pluto	248.0

all we have done here is take the magnitude of the force and multiplied by the ratio of the distance along each respective axis and the distance between the masses. This is just two interacting masses, but for multiple we can just add up the contributions from each mass, e.g. for the x-axis:

$$F_{x,1} = -G\frac{m_1 m_2}{r_{1,2}^3} \times (x_2 - x_1) - G\frac{m_1 m_3}{r_{1,3}^3} \times (x_3 - x_1) - \cdots \tag{8}$$

$$F_{x,1} = -\sum_{i \neq 1} G\frac{m_1 m_i}{r_{1,i}^3} \times (x_i - x_1) \tag{9}$$

This is just for the *net force* on one mass, but we need to consider the forces on all masses. Once we know these forces, which change with time, we then set equal to the product of mass and acceleration along each axis. So for the motion along the x-axis we have:

$$F_{x,1} = m_1 \frac{d^2 x}{dt^2} \tag{10}$$

These are the equations that the MATLAB model can solve for us.

3 Running the model

The MATLAB files required are `run_solar_system.m`, a script that runs the model, `solve_solar_system.m`, a function which solves equations like Equation 10 for each body using a numerical scheme, `animate_solar_system.m`, a script that animates the positions of the planets, and some plotting functions: `plot_solar_system_one.m`, `fourier_solar_system_one.m`, `fourier_solar_system_two.m` and `fourier_solar_system_suns_motion.m`. They should be put in the current working directory. The procedure for running the model is

1. Edit Section 1 of `run_solar_system.m` to configure the model.
2. Type `run_solar_system` at the MATLAB prompt to run the model.
3. Type either:
   - `animate_solar_system`
   - `plot_solar_system_one`
   - `fourier_solar_system_one`
   - `fourier_solar_system_two`
   - `fourier_solar_system_suns_motion`

   at the MATLAB prompt to view the results.

Table 1 describes the parameters in Section 1 of `run_solar_system.m` that can be modified, although obviously you can hack any part of the code if you want to.

4 Experiments

4.1 Animation of the planets

The first simulation to do is a standard simulation where the planets only interact with the sun. This is accurate over 1000 year time-scales and produces perfectly elliptical orbits (so-called Keplerian orbits after the Philosopher who studied the mathematics of the orbits of the planets). The settings to use are as follows:

- `which_bodies = ALL_BODIES`
- `which_interactions = INTERACT_WITH_SUN_ONLY`
- `which_star = MOVING_SUN`
- `which_output = FIRST`
- `run_time = 1000.`

so set these in `run_solar_system.m`, save and type `run_solar_system` at the command prompt. The model will take a few minutes to run after which the output will be in the MATLAB base memory (you should be able to see variables in the variable window and / or by typing `whos` on the command line).

Exercise: Animate the orbits of the planets by typing `animate_solar_system` at the MATLAB command window and pressing enter. You should see the planets orbiting around the sun. Use the zoom function on the figure window to zoom in and see the detail. After you've had enough, close the window and the program will stop.

4.2 Time series plot of sun-planet distance: perihelion and aphelion)

Now look at the distance between the sun and the planets against time. Do this by typing `plot_solar_system_one` at the MATLAB command line. You should see two plots: one of the distance between the sun and the inner most 4 planets versus time and another for the outer planets (and pluto!). These show various peaks and troughs, which are distances of perihelion (closest approach) and aphelion (furthest distance).

Exercise: Look at the plot of the sun-earth distance versus time. From this plot how can you get the orbital period of earth? (i.e. the time taken to go once around the sun).

4.3 Fourier Transform (and a-harmonic waves)

Now you will look at the periodic oscillations in your data a different way by performing a *Fourier transform*, which is a plot of power against frequency (or time-period) of a wave. Type `fourier_solar_system_one` on the MATLAB command line, which will perform Fourier transforms of each sun-planet distance versus time and plot them out.

Exercise: Using the zoom function to find the time period that corresponds to the peak in power and write them down for each of the planets. How do these values compare to the orbital period of each planet (Table 2)?

Better still type `[xx,yy]=ginput` on the MATLAB command line and using the mouse, click on each of the peaks. Once you've clicked on all 9, press enter and the values will be saved in the `xx` array.

Exercise: Now look at the plot for Mercury. You can see there are several peaks of lower power than the main peak. Using the zoom function (or `ginput`) find the period that the 4 largest peaks correspond to and write down their values.

Take the reciprocal of these 4 values to get the orbital frequency (i.e. one divided by the time period) and then divide each result by the smallest of the 4 frequencies. What do you notice?

Why do you think you get this result?

You should notice that each of the frequencies is a multiple of the lowest frequency (the fundamental frequency). This arises because the orbit of the planets are not circular, but elliptical.

If they were circular we could describe the orbit by a single sine wave that has the fundamental frequency. The additional 'whole number multiples' of the fundamental frequency are needed to describe the 'flattening out' of the sun-planet distance in the peaks and troughs.

So if you spot any waves in the Fourier plot that have a whole number multiple of the main peak's frequency it is likely due to this effect.

4.4 Interaction between planets / conjunction

Jupiter, with its orbital period of around 11.8 years interacts with the planets to perturb their orbits from perfect ellipses over 10,000 and 100,000 year time-scales. These are known as Milankovich cycles and are important to the study of paleoclimate. Here you will run the model including this interaction, albeit only for 1000 years, so you wont see the Milankovich cycle (although there is nothing to stop you running the model for longer)[1]
The settings to use are as follows:

- `which_bodies = ALL_BODIES`
- `which_interactions = INTERACT_WITH_ALL_BODIES`
- `which_star = MOVING_SUN`
- `which_output = SECOND`
- `run_time = 1000.`

Now run the `run_solar_system` file again by entering it in the MATLAB command prompt.

Exercise: Now look at the Fourier transform of both simulations by typing `fourier_solar_system_two`. You should see some small differences between the two simulations.

Click on the Saturn plot and go back to the MATLAB command window and type `xlim([10 30])`.

You should see a little bump on the green line, not visible on the red. This is the period corresponding to the conjunction of Saturn and Jupiter: the so called 'Great conjunction'. This is when Jupiter and Saturn are closest together. Find the time period of this conjunction by using zoom or `[xx,yy]=ginput`.

[1] If you would like to run the model for 100,000 years, replace line 71 of `solve_solar_system.m` to `tsol=[0:0.1:varargin{5}];` to avoid using too much memory

Exercise: We may calculate the period of the great conjunction, T, using the formula:

$$\frac{1}{T} = \frac{1}{T_j} - \frac{1}{T_s} \tag{11}$$

where $T_j \cong 11.8$ years is the orbital period of Jupiter and $T_s \cong 29.5$ years is the orbital period of Saturn. How does the theoretical value compare to the value from your graph?

4.5 Sun's wobble / Exo-planets

Gas giants in other solar systems are often found by looking at the motion of distant stars. If a 'wobble' is detected then there must be a large object orbiting the parent star. Here we look at the sun's wobble due to planets in our solar system.

Exercise: type `fourier_solar_system_suns_motion` on the MATLAB command line and press enter. You should see a plot of the Fourier transform of the sun's motion.

Can you see the influences of all the planets in the solar system? Which planet(s) have a large influence on the motion / wobble of the sun?