

Computer Practical: Solar System Model

Paul Connolly, September 2020

1 Overview

In this computer practical, a solar system model implemented in the compiled language FORTRAN 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, and noise frequencies).

2 Model formulation

The three-dimensional model is implemented in FORTRAN. The FORTRAN code solves for the motion of the planets and writes the output into data lists called 'arrays', which it then writes to a special file format known as NetCDF. This file can then be read in and plotted by Python. It predicts and stores the position and velocity in all three dimensions of all of 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:

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

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

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

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:

$$|\vec{F}_{1,2}| = -G \frac{m_1 m_2}{r_{1,2}^2} \quad (3)$$

where $G = 6.67 \times 10^{-11} \text{ m}^3 \text{ kg}^{-1} \text{ s}^{-2}$ 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:

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

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

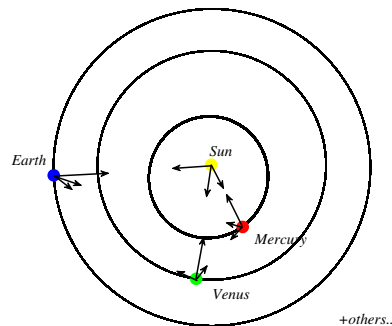


Figure 1: Schematic of the scenario being modelled, where the arrows represent the forces each body experiences due to the interaction with all other bodies.

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

$$(7)$$

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:

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

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

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:

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

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

3 Downloading and compiling the code

Log into the server computer with a Google Chrome SSH window. Also, log into the server computer with a Google Chrome SFTP window. Space the windows out on your screen so that you can easily move between them. If you have not done this already, download the code you will be using today from GitHub by typing the following:

```
git clone https://github.com/EnvModelling/solar-system-model
```

this should download the code to your working directory.

In order to access the files you will need to change the working directory to where the files are. Type the following:

```
cd solar
```

followed by the `tab` key and the command should auto-complete. Then press the `enter` key.

Type `ls` followed by the `enter` key. The screen should list the files in this directory.

Finally, type `make` followed by the `enter` key. This will compile the FORTRAN code into machine code that the computer can run.

4 Viewing and Editing the code

We will use the `nano` text editor to view some files. The file that takes input data is called `namelist.in`. To see what variables we can change, type:

```
nano -l namelist.in
```

the `-l` means to show line numbers. The code can be edited in the text editor and saved by typing `Ctrl-X` at the same time and pressing `Y` to save the file. The important lines are lines 42-49 of this file. The `interact` variable has length 10 and each element is set to either `.true.` or `.false.` `.true.` means that object interacts with *all* other objects; whereas `.false.` means it does not interact.

The exclamation mark is a so-called 'comment' in FORTRAN. This means the code does not read it. Therefore we can see the default is for the the planets to only interact with the sun (as lines 48 and 49 do not have an exclamation mark at the start.

Table 1: Parameters that you will change, which control the behaviour of the model.

Variable	Default value	Description
<code>interact(1:10)</code>	All false except the first one	specify which bodies interacting with each other
<code>tfinal</code>	<code>1.d4</code>	this means $1 \cdot e4$. It is the length of the simulation in years.
<code>gm</code>	values for the solar system	The product of G and the object's mass.

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

Planet	Period in earth years
Mercury	0.25
Venus	0.6
Earth	1.0
Mars	1.9
Jupiter	11.9
Saturn	29.5
Uranus	84.0
Neptune	164.8
Pluto	248.0

5 Running the model

After the code has been compiled (see Section 3), the procedure for running the model is to type:

1. Edit the file `namelist.in` to configure the model.
2. Type `./run.sh` at the command line, followed by `enter` to run the model.

Table 1 describes the parameters in `namelist.in` that can be modified, although obviously you can hack any part of the code if you want to. Not that if you do change any FORTRAN code you will need to type `make` again to compile the code to the machine executable.

6 Experiments

FOR ALL EXPERIMENTS: I STRONGLY SUGGEST YOU OPEN UP A WORD OR POWERPOINT DOCUMENT AND INSERT THE FIGURES AND MAKE SOME NOTES AS YOU GO. YOU COULD BE ASKED QUESTIONS ABOUT THEM IN THE ASSESSMENT.

6.1 3-D Plot of planetary orbits

The first simulation to do is a standard simulation where the planets only interact with the sun. Kepler orbits are accurate over 1000 year time-scales and produce perfectly elliptical orbits (so-called Keplerian orbits after the Philosopher who studied the mathematics of the orbits of the planets). This is the default case, so we just need to type:

```
./run.sh
```

at the command line, followed by `enter`. The code will take about 2-minutes to run and an output file will be created. Check that the file is there by typing `ls /tmp/<username>/` where `<username>` is your username.

Exercise: To plot the orbits of the planets you will need to run a Python script. Type the following, pressing enter after each line

```
cd python/  
python3 plot_solar_system01.py
```

after a short while Python will put a file in the `ls /tmp/<username>/` directory. To download it, go to your SFTP window and type:

```
get /tmp/<username>/orbits.png
```

followed by enter, where <username> is your username. This should be a 3-D plot of the planets orbits.

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

You will now plot the distance between the sun and the planets against time. From the SSH window in the python directory type `python3 plot_solar_system02.py` at the command line, followed by enter. When it is finished go to your SFTP window and type:

```
get /tmp/<username>/milankovitch.png
```

followed by enter, where <username> is your username.

You should see 9 plots showing the distance between the planets and the sun vs time. These show various peaks and troughs, which are distances of perihelion (closest approach) and aphelion (furthest distance). The shape of the oscillations do not change over the 10000 years shown.

Exercise: Look at the plot of the sun-Pluto 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).

6.3 Fourier Transform (and harmonics)

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. In the SSH window, in the python directory type:

```
python3 plot_solar_system03a.py
```

followed by enter. When it is finished go to your SFTP window and type:

```
get /tmp/<username>/fourier.png
```

followed by enter, where <username> is your username.

This is a frequency spectrum of all of the planet-sun distances.

Exercise: How do the peak values in each of the plots compare to the orbital period of each planet (Table 2)?

Exercise: We will look at the plot for Earth in more detail. Edit the `plot_solar_system03a.py` by typing `nano -l plot_solar_system03a.py`. On line 48 you will need to edit the line so that it says:

```
plt.xlim((0.1, 1.2))
```

(be sure to delete the # sign, which is a comment line in Python). After editing, type Ctrl-X, and then save. Then run the python script again:

```
python3 plot_solar_system03a.py
```

Download the `fourier.png` plot again, using SFTP, which will now have a zoomed in x-axis (make sure you have saved and made notes on the previous plot).

You can see there are several peaks of lower power than the main peak. Estimate their value from the plot write down their values.

Take the reciprocal of the x-value of these 4 peaks 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 (the one corresponding to the orbital period). What do you notice?

Why do you think you get this result?

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

6.4 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 Milankovitch cycles and are important to the study of paleo-climate. Here you will run the model including this interaction, albeit only for 10,000 years, so you wont see the full Milankovich cycle (although there is nothing to stop you running the model for longer)¹. The settings to use are as follows. You will need to move out of the python directory up one. Type:

```
cd ..
```

followed by `enter` to do this. Then edit `namelist.in` to:

- delete the exclamation mark (comment) at the start of lines 45 and 46.
- add exclamation marks (a comment) at the start of lines 48 and 49.

Now run the model again by typing `./run.sh` at the command line, followed by `enter`. After 2 minutes the simulation will be finished. After this change back to the python directory by typing:

```
cd python
```

followed by `enter`.

Exercise: Now edit the `plot_solar_system03a.py` by typing `nano -l plot_solar_system03a.py`.

On line 48 you will need to edit the line so that it says:

```
plt.xlim((10, 30))
```

type `Ctrl-X`, and then save. Then run the python script again:

```
python3 plot_solar_system03a.py
```

followed by `enter`.

Download the `fourier.png` plot again, using SFTP, which will now have a zoomed in x-axis on Saturn.

You should see a little bump on the line for Saturn. 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. Read the value of this off the x-axis (approximately).

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

$$\frac{1}{T} = \frac{1}{T_j} - \frac{1}{T_s} \quad (11)$$

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?

¹If you would like to run the model for 100,000 years, change line 39 of `namelist.in` to `tfinal=1.d5,`

6.5 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: from the python directory type `python3 plot_solar_system03b.py` on the command line and press `enter`. Then download the file from the SFTP window by typing:

```
get /tmp/<username>/fourier_sun.png
```

where `<username>` is your actual username, followed by `enter`.

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?