Solar Manual

Scientific Computing II

Final project 7

Emil Levo

UNIVERSITY OF HELSINKI
DEPARTMENT OF PHYSICS

Contents

1 Introduction

Our solar system formed approximately 4.568 million years ago consisting mass-wise mainly of our star, the sun, and eight planets revolving around it. The system will stay roughly like we know it to be today, until all of the hydrogen in the core of our star will have transformed into helium through the fusion process.

The manner in which the celestial bodies of our system move in relation to each other has been described during the ages in very different ways. The geocentric model of the solar system was originally the most accepted one, before Copernicus desribed the system as heliocentric in the 16th century. This picture was additionally altered when Kepler realised that the orbits of the planets are not circular, but rather elliptic and the sun is not located excactly in the center of all orbits, but at some focus point. From this realisation Kepler went on to describe the motion of the planets with the what is now known as Kepler's three laws of planetary motion.

Later on Newton showed Kepler's laws were derivable from his very own theory of gravitation, which describes the planetary motion to this day in a most satisfactory manner, even though a more accurate theory exists. Einstein provided mankind with the most accurate accepted theory to describe the orbits of celestial bodies, his general theory of relativity.

The importance of describing orbits of planets in our system is quite important even from just a very fundamental and existensial point of view for us humans, describing time. The motion of our own planet around our sun has dictated the formation and evolution of life and civilization as we know it on this earth. One year is defined by the time it takes for our planet to accomplish one whole revolution around our sun. Additionally understanding the motion of all objects in our solar system, impacts all human endevours to reach out to space.

Planetary simulations are a great way of describing our planets orbits, to some extent. One has to remember that even a small change in any parameter can hugely impact a system over a great duration of time. Nevertheless, one can very conveniently approximate the orbits of the planets in our system by computational means, which can be seen as an easier option compared to experiment and observation. Even though Newton's theory of gravitation does not yield the most accurate results when describing planetary motion, it still does it so extremely well that it can be effectively used in computer simulations to yield some meaningful results for planetary motion in our solar system. In this work we have used Newton's theory of gravitation to describe the orbits in our solar system with computer simulations.

2 Methods

Newton presented his law of gravitation in his work, the *Principia*, that was published in 1687. The law itself can be expressed in a simple vector form:

$$\mathbf{F_{21}} = -G\frac{m_1 m_2}{|\mathbf{r_{12}}|^2}\hat{\mathbf{r}}_{\mathbf{12}} \tag{1}$$

Equation 1 states that the gravitational force exerted by object 1 upon object 2. G is the empirically derived gravitational constant ($\approx 6.674 \cdot 10^{-11}\ \frac{\mathrm{Nm}^2}{\mathrm{kg}^2}$), m_1 the mass of object 1, m_2 the mass of object 2 and $|\mathbf{r_{12}}|$ the distance between the two objects. The masses handled in this equation are point masses, and the force between them points along the line intersecting both masses. To describe the movement of an object in a many body system, like our Solar System, one needs to take into account the sum of all the forces applied on the object by all other objects in the system. This means we write our force of gravity in the form:

$$\mathbf{F} = -G\sum_{i\neq j}\frac{m_i m_j}{|\mathbf{r_{i,j}}|^2}\hat{\mathbf{r}}_{\mathbf{i,j}} \tag{2}$$

The inequality in the subindex of the sum assures that the object will not exert a force on itself. We can still write this equation in another form that describes the gravitational field $\mathbf{g}(\mathbf{r})$ (acceleration of an object in the point $\mathbf{r}$ of the gravitational field):

$$\mathbf{g}(\mathbf{r}) = -G\sum_{i\neq j}\frac{m_i}{|\mathbf{r_{i,j}}|^2}\hat{\mathbf{r}}_{\mathbf{i,j}} \tag{3}$$

The form presented in equation 3 is the form we use to calculate the accelerations that each object inherits from all others in the simulated system.

The program used for our simulations (named "Solar" for simplicity), is written in Fortran 90 and can describe the motion of objects in an N-body system like our Solar System by calculating the accelerations of all planets (and the sun) and predicting their trajectories with the Velocity Verlet algorithm as presented in the instructions for the final project.

The source code is divided into five different files:

1. **calcforces.f90**
2. **calcenergy.f90**
3. **ingredients.f90**
4. **main.f90**
5. **start.f90**
6. **velverlet.f90**

The file main.f90 is the main file of the program, and the rest are modules created for modularity. 1) **calcforces.f90** calculates the accelerations of all objects that are simulated

as described with equation 3. 2) **calcforces.f90** contains functions to calculate the kinetic and potential energy of each object and the total energy of the system. 3) **ingredients.f90** reads the input file with the readin() subroutine and defines all data structures and constants used by the program. It also writes a possible error message. 4) **main.f90** calls the subroutines from the other modules to run the simulation and prints to the screen the initial conditions of the system and the predictions for the first step. 5) **start.f90** creates the run() subroutine that runs the simulation and writes all data to the screen and output file. 6) **velverlet.f90** contains the functions that predict the next positions and velocities by using the Velocity Verlet algorithm.

The positions, velocities and accelerations are described as vectors in a cartesian coordinate system. The three properties are stored in arrays, that have a size defined by the input file. The positions are in units of km, velocities km/s and accelerations km/s^2. Also the names, indexes and masses of the simulated objects are stored in arrays as read from the input file. The masses of the objects are to be given in kg. The gravitational constant is defined as $6.67408 \cdot 10^{-20}\ \frac{\text{Nkm}^2}{\text{kg}^2}$ in our simulations.

For succesful compilation all files described above needs to be included in the src/ directory. One can compile the code by using the provided compile.sh bash script or by using the commands:

gfortran -c ingredients.f90 velverlet.f90 calcforces.f90 calcenergy.f90 start.f90 main.f90
gfortran -o solar ingredients.o velverlet.o calcforces.o calcenergy.o start.o main.o

mv solar ../run/

The above description can also be found in the src/ directories **README** file. The created executable, named **solar** above, needs to be moved to the run/ directory.

The program can then by run in the run/ directory which should contain the executable **solar** and the input file **input.dat**. The **input.dat** file needs to contain the number of objects, timestep, number of steps, every n:th step to be printed to screen, every m:th step to be written to the file **output.dat** and then all objects to be simulated with their indices, masses, initial positions and velocities. For a succesful read of the file, it needs to be formatted correctly and the parameters need to be in the right order. An example **input.dat** is provided and is recommended to be used for succesful simulation runs. If the simulation should fail, or the data seems off, be sure to check the **errors.out** file for possible error messages. The format of the input and output files are further described in the **README** file in the run directory.

The units for the quantities are described here:

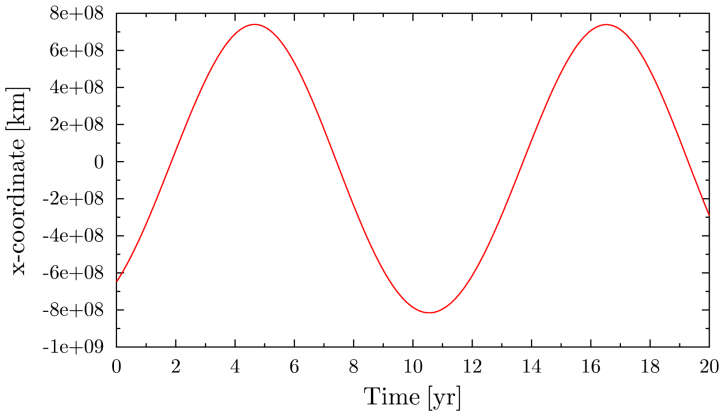
1. Time in s
2. Positions in km
3. Velocities in km/s
4. Accelerations in km/s^2
5. Masses in kg
6. Energy in kgm^2/s^2

3 Results

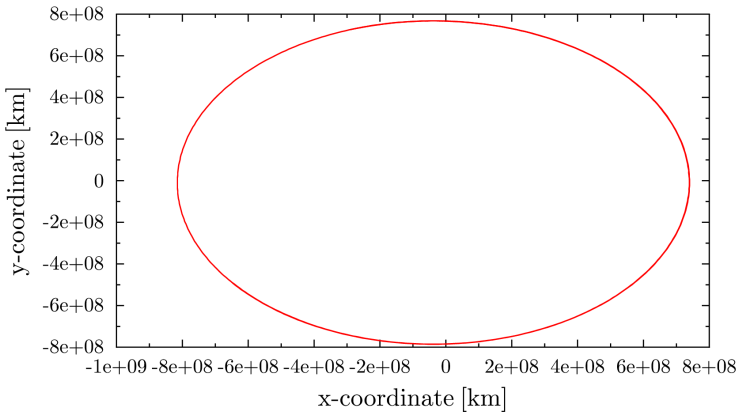
The simulations presented in this section were accomplished with initial data for all planets and the sun obtained from the NASA HORIZONS Web-Interface. All positions and velocities are from the date 15.12.2017. Note that the initial positions are barycentric, i.e. the sun is not in the middle, but the center of mass. It is recommended to use the same **input.dat** files for each part as is presented below, in order to reproduce the results presented here, but feel free to change the initial positions and velocities (as long as they are somewhat consistent with reality) to check that the code works.

1. The Jupiter simulations were accomplished with the **input.dat** file named **JUPITER-input.dat** in the inputs directory. Only the timestep, number of steps and intervals to write and print the data were changed for the simulations. Remember to rename the input-file to **input.dat** in the run/ directory!!

   (a) The period of Jupiter is plotted in Fig. 1. The timestep used in this simulation was $\Delta t = 1000.0s$.



(a) The x-coordinate plotted against time elapsed



(b) The x-coordinate plotted against the y-coordinate

Figure 1: The period of Jupiter

Table 1 shows the different timesteps Δt used for the Jupiter simulations. As you can see from the table, the timestep of $\Delta t = 1000.0$ yields the period that

is closest to the accepted value of $T_{jupiter} = 11.862$ yr. Note that both a too small and too big timestep gives a value that is differs from the accepted value on the first decimal.

Δt [s]	T [yr]
10.0	11.405
100.0	11.871
1000.0	11.8647
10000.0	11.876
1000000.0	11.900

Table 1: Table of the different timesteps used and the corresponding periods for Jupiter

(b) In part b) the system was simulated for one period for Jupiter. To reproduce the same results, set $\Delta t = 1000.0s$ and the number of steps to 374343. The resulting distance traveled in the x-direction can be seen in Fig. 2, where one can upon more detailed inspection see that the final position of Jupiter is a million km less than the original x-coordinate. In other words Jupiter has moved under one full period of motion.

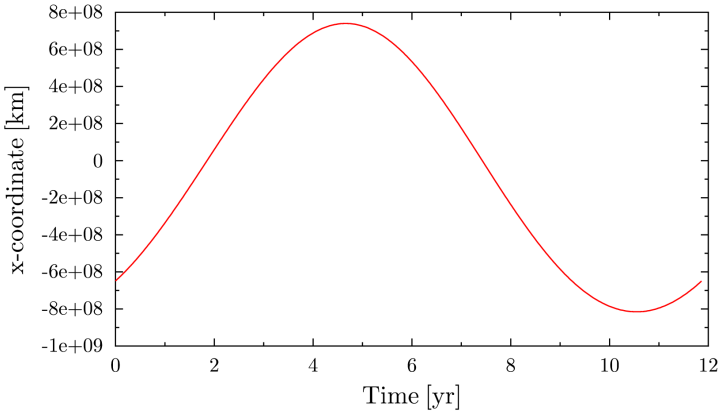
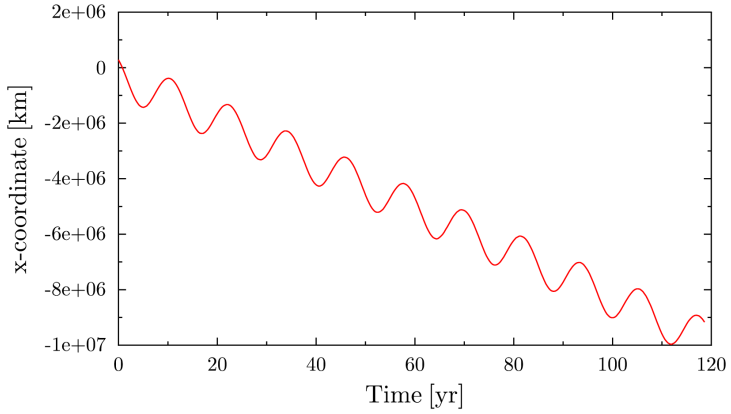
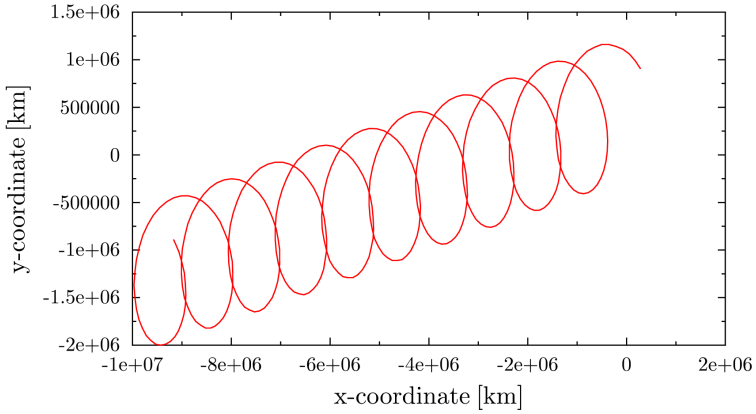


Figure 2: Simulating one period of Jupiter.

(c) The period of the Sun has been visualized in Fig. 3. The timestep $\Delta t = 1000.0s$ was used and the system was simulated for 120 years. One period of motion is also roughly 11 years, and the radius of the motion approximately a million km.



(a) The x-coordinate plotted against time elapsed



(b) The x-coordinate plotted against the y-coordinate

Figure 3: The period of the Sun

2. (a) For approximating one year, we simulated one period of motion of the Earth, (the idea being that the earth should complete one period in one year). The input file is named **TELLUS-input.dat** in the inputs/ directory. The timestep that yielded the most accurate result of $T = 1.000$ yr was $\Delta t = 50.0s$, as can be seen in Table 2.

Δt [s]	T [yr]
10.0	0.9996
50.0	1.0000
100.0	0.9996

Table 2: Table of the different timesteps used and the corresponding periods for the Earth

(b) For approximating one month, we simulated one period of motion of the Moon around the Earth (the idea being that the Moon should complete one period in one year). The input file is named **LUNA-input.dat** in the inputs/ directory. In this simulation the Earth is actually set exactly in the middle of the system, and the moon at the accepted average distance from the earth with its average

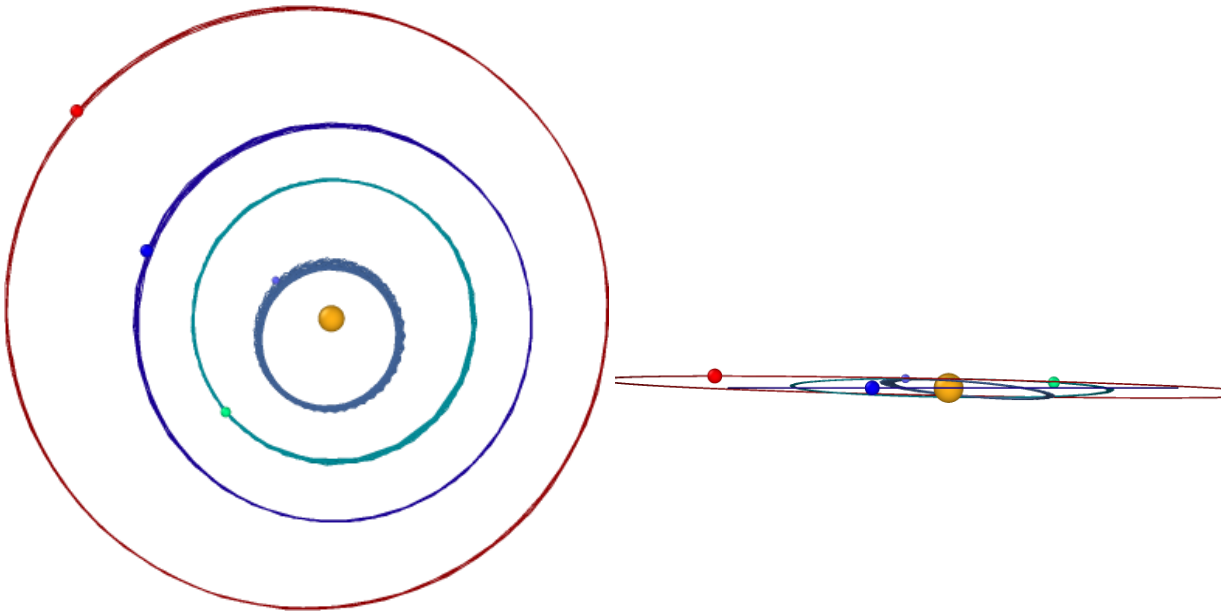
orbital velocity. Table 3 shows that three timesteps, all of which gave a very nice value for the orbital period of the moon, 27.01 days, when the accepted value is 27.3.

Δt [s]	T [days]
1.0	27.01
10.0	27.01
100.0	27.01

Table 3: Table of the different timesteps used and the corresponding periods for the Moon

3. For the final simulations we used the input **SYSTEM-input.dat** in the inputs/ directory.

   (a) First the timestep $\Delta t = 1000.0$ s was used and the system was simulated for 200 years. The orbits of the inner planets can be seen in Fig. 4 and the periods can be seen in Table 4. The approximations for the periods are good to the first decimal, after which they start to differ from the observational values for some of the planets.



(a) (b)

Figure 4: The orbits of the inner planets, visualized with OVITO [1]

Planet	T_{sim} [yr]	T_{obs} [yr]
Mars	1.8907	1.8808
Earth	1.0004	1.0000
Venus	0.61283	0.61519
Mercury	0.24107	0.240846

Table 4: Table of periods of the inner planets

   (b) Shortening the timestep to 10.0 s gives us a more accurate value for the period of Mercury, $T = 0.2410$. Shortening it further to 1.0 s, gave a less accurate value of $T = 0.2420$.

4 Conclusions

To summarize, this work presents a functioning computational method that applies Newton's laws of gravitation to a many-body system. It has to be taken into account that the initial values chosen for all problems presented above, are what they are because of how the planets in our solar system have interacted for a very long period of time. So the results we present for the binary systems, might differ from results when simulating the whole system. For example, in part 3 the period of Jupiter was $T = 11.863$ which is closer to the real value than the period we simulated in part 1 with the same timestep of $\Delta t = 1000.0$.

So the initial parameters are for this program, the most decicive factor for our results to coincide with reality. One other thing that influences our results in a bad way, is the small perturbations and accuracy of the reals that the vectors in this program use. The small perturbations can influence the system greatly given great periods of time.

A final note on the accuracy of these simulations can be given to everything that is missing in the program. The planets are point masses that do not rotate around their own axes, and there is a lot of other physical stuff missing from this solar system like asteroid belts and well, poor Pluto. But still, given that there is a lot missing, the simple Newtonian approach with point masses give us nice results for the periods of the planets, and we were able to predict some of our most used measures of time.

References

[1] Alexander Stukowski. "Visualization and analysis of atomistic simulation data with OVITO–the Open Visualization Tool". *Mod. Sim. Mat. Sci. Eng.* 18.1 (2010), p. 015012.