8 — PHY 494: Homework assignment (40 points total)

Due **Tuesday** (after Spring Break), March 17, 2020, 11:59pm.

Submission is to your **private GitHub repository**.

Read the following instructions carefully. Ask if anything is unclear.

1. `cd` into your assignment repository (change *YourGitHubUsername* to your GitHub username) and run the update script `./scripts/update.sh` (replace *YourGitHubUsername* with your GitHub username):

   ```
   cd  assignments-2020-YourGitHubUsername
   bash ./scripts/update.sh
   ```

   It should create three subdirectories[1] `assignment_08/Submission`, `assignment_08/Grade`, and `assignment_08/Work`.

2. You can try out code in the `assignment_08/Work` directory but you don't have to use it if you don't want to. Your grade with comments will appear in `assignment_08/Grade`.

3. Create your solution in `assignment_08/Submission`. Use Git to `git add` files and `git commit` changes.

   You can create a PDF, a text file or Jupyter notebook inside the `assignment_08/Submission` directory as well as Python code (if required). **Name your files `hw08.pdf` or `hw08.txt` or `hw08.ipynb`**, depending on how you format your work. Files with code (if requested) should be named exactly as required in the assignment.

4. When you are ready to submit your solution, do a final `git status` to check that you haven't forgotten anything, commit any uncommited changes, and `git push` to your GitHub repository. Check

[1] If the script fails, file an issue in the Issue Tracker for PHY494-assignments-skeleton and just create the directories manually.

on *your* GitHub repository web page[2] that your files were properly submitted.

You can push more updates up until the deadline. Changes after the deadline will not be taken into account for grading.

Homeworks must be legible and intelligible or may otherwise be returned ungraded with 0 points.

If you implement the function as specified you can run the tests in the file `Submission/test_hw08.py` with `pytest`

```
cd Submission
pytest -v test_outerplanets.py
```

and all tests should pass. If you have errors, have a look at the output and try to figure out what is still not working. Having the tests pass is not a guarantee that you will get full points (but it is general a very good sign!).

Collaboration: Up to three students may form a team and solve the homework together. Each student may submit the same solution to their own repo. **Add a text file `COLLABORATION.txt`** to the repository in which you

- list the names of all team members (last name, first name), and
- provide a brief statement ($<$ half a page) as to who in the team contributed what to the solution

If you solved the problem on your own, just put your own name as the single name in `COLLABORATION.txt` and state in the file "I solved and completed the homework by myself.".

8.1 Discovery of Neptune (3-body problem) (40 points)

Historically, the planet Neptune was discovered because of its (small) observable influence on the orbit of Uranus. Neptune attracts Uranus and

[2] `https://github.com/ASU-CompMethodsPhysics-PHY494/assignments-2020-`*`YourGitHubUsername`*

planet	mass (M_{sol})	distance (AU)	orbital period (yr)	angular position (1690) θ
Sun	1	—	—	—
Uranus	4.366244×10^{-5}	19.1914	84.0110	205.640°
Neptune	5.151389×10^{-5}	30.0611	164.7901	288.380°

Table 1: Parameters for Uranus and Neptune

this leads to small changes in Uranus' orbital velocity. In this problem you are studying a simplified situation only containing the Sun, Uranus and Neptune.[3]

Assume that to a first approximation that the orbits of Uranus and Neptune are circular (so that you can easily calculate the initial angular velocity)[4] and co-planar. The initial position is given by an angle θ relative to the x-axis and the distance r from the sun, as listed in Table 1.

Calculate the variation in the angular velocity of Uranus $\Delta\omega_U(t)$ due to the influence of Neptune: Compare the instantaneous angular velocity[5] $\omega_U(t)$ to the one in the absence of Neptune, ω_U^0:

$$\Delta\omega_U(t) = \omega_U(t) - \omega_U^0 \tag{1}$$

by plotting $\Delta\omega_U(t)$ over t and briefly comment on your result.

[3] For more background, see e.g., *Computational Physics* 9.7.

[4] In your simulation, do not enforce circular orbits, though. With the given parameters, orbits should turn out to be nearly circular.

[5] The angular velocity ω for circular motion is

$$v = \frac{2\pi r}{T} = \omega r$$
$$\omega = \frac{2\pi}{T} = \frac{v}{r}$$

where T is the period and v the speed (velocity is normal to the radial vector $\mathbf{r}$, r is the distance from the center).

8.1.1 General considerations and requirements

- Use astronomical units: AU for length (1 AU is the distance of the earth from the sun), year for time, mass in terms of solar masses (i.e. sun's mass $M = 1$) — all as provided in Table 1, gravitational constant (in Eq. 2) in AU: $G = 4\pi^2$
- Use the *Velocity Verlet* algorithm[6] with a time step of 0.1 years to integrate the equations of motions of Uranus and Neptune.
- The sun can be considered stationary ($M \gg m_U \approx m_N$).
- Analyze 160 years (almost one complete orbit of Neptune).
- Skeleton code is provided in `outerplanets.py`; you can use it but you don't have to. However, you must write your code so that
  1. it can be imported as `import outerplanets`
  2. it contains a function `integrate_orbits(dt=0.1, t_max=320, coupled=True)` to do the integration; the function has to return a tuple `time, r, v` (see `outerplanets.py` for details).

  A test case `test_outerplanets.py` is provided; you can run

  ```
  pytest -v test_outerplanets.py
  ```

  and all tests should pass if you have done everything correctly.
- Also plot the orbits of Uranus and Neptune.
- The file `outerplanets.py` must be a separate file. Provide the two figures as individual PDFs[7] (use `ax.figure.savefig("filename.pdf")`). Other descriptions and code can be submitted as a notebook or in other suitable forms.

[6] Hidden Bonus: Try using another integration algorithm such as RK4 and compare the results to the calculation with Velocity Verlet. **[bonus +5*]**

[7] If PDF does not work for whatever reason, use the JPG format instead of PDF.

To calculate $\Delta\omega$ in Eq. 1 you need to run your integration

1. *without* the interaction between Neptune and Uranus (`coupled=False`) and

2. *with* the gravitational interaction.

8.1.2 Angular velocity

You can calculate the instantaneous angular velocity as[8]

$$\omega(t) = \frac{v(t)}{r(t)}, \quad r = |\mathbf{r}|,\ v = |\mathbf{v}|$$

where position $\mathbf{r}(t)$ and velocity $\mathbf{v}(t)$ are determined from the integrator.

8.1.3 Interactions

This is a *three-body* problem. You must compute the forces for Uranus and for Neptune, using the three interactions

1. Uranus — Sun, $\mathbf{F}_{U,S}$

2. Neptune — Sun, $\mathbf{F}_{N,S}$

3. Uranus – Neptune, $\mathbf{F}_{U,N}$

Thus, for example, the force on Uranus is $\mathbf{F}_U = \mathbf{F}_{U,S} + \mathbf{F}_{U,N}$. Newton's force law for gravitation is

$$\mathbf{F} = -\frac{GmM}{r^2}\hat{\mathbf{r}} \tag{2}$$

$$\hat{\mathbf{r}} = \frac{1}{\sqrt{x^2+y^2}}\begin{pmatrix} x \\ y \end{pmatrix}$$

where G is the gravitational constant, m and M are the masses of the two bodies, and $\mathbf{r}$ is the vector between the positions of the two bodies. Remember that Newton's third law states

$$\mathbf{F}_{12} = -\mathbf{F}_{21}. \tag{3}$$

[8] The code contains the function `omega()` to perform the calculation.