6. — PHY 494: Homework assignment (20 points total)

Due Friday, March 24, 2017, 23:59.

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-2017-YourGitHubUsername
   bash ./scripts/update.sh
   ```

   It should create three subdirectories[1] `assignment_06/Submission`, `assignment_06/Grade`, and `assignment_06/Work`.

2. You can try out code in the `assignment_06/Work` directory but you don't have to use it if you don't want to. Your grade with comments will appear in `assignment_06/Grade`.

3. Create your solution in `assignment_06/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_06/Submission` directory as well as Python code (if required). **Name your files `hw06.pdf` or `hw06.txt` or `hw06.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 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_hw04.py` with `py.test`

```
cd Submission
py.test test_outerplanets.py
```

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

[2] `https://github.com/ASU-CompMethodsPhysics-PHY494/assignments-2017-`*`YourGitHubUsername`*

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

6.1. Baseball physics: curve ball (20 points)

We want to simulate the trajectory of a *curve ball* in the game of baseball. In this problem we simplify the problem somewhat to include the essential physical effects:

- Only consider quadratic terms in the air resistance (ignore linear terms) and assume that the drag coefficient b_2 is independent of velocity. (See Problem 6.5 below for a more realistic approach where $b_2(v)$).
- Consider the *Magnus effect* due to spin but assume that the ball spins at constant angular velocity. (See Problem 6.5 below for a more realistic approach where the angular velocity decays with time.)

Quadratic air resistance An approximately quadratic dependence of the drag force on the velocity occurs at high Reynolds numbers, i.e., turbulent flow (approximately when $\mathrm{Re} > 2300$). An approximate expression is[3]

$$\mathbf{F}_2 = -\frac{1}{2} C_D \rho A v^2 \frac{\mathbf{v}}{v} \tag{1}$$

We are considering the quadratic drag coefficient b_2 to be constant in this problem.

Magnus effect The airflow is changed around a spinning object. The Magnus force is

$$\mathbf{F}_M = \alpha \boldsymbol{\omega} \times \mathbf{v} \tag{2}$$

where $\boldsymbol{\omega}$ is the ball's angular velocity in rad/s (e.g., 200/s for a baseball).

For a sphere the proportionality constant α can be written

$$\mathbf{F}_M = \frac{1}{2} C_L \rho A \frac{v}{\omega} \boldsymbol{\omega} \times \mathbf{v} \tag{3}$$

where C_L is the lift coefficient, ρ the air density, A the ball's cross section. (Advantage of defining C_L this way: when spin and velocity are perpendicular, the Magnus force is simply $F_M = \frac{1}{2} C_L \rho A v^2$.)

C_L is mainly a function of the *spin parameter*[4]

$$S = \frac{r\omega}{v} \tag{4}$$

[3] In the Lecture notes we just denoted quadratic drag with $\mathbf{F}_2 = -b_2 v \mathbf{v}$ and lumped everything into the quadratic drag coefficient b_2. Eq. 1 gives more physical motivation to $b_2 = \frac{1}{2} C_D \rho A$

[4] For a more detailed discussion that also considers an additional v-dependence of C_L through its dependence on $C_D(v)$ see Nathan (2008a).

with the radius r of the ball. $S = v_{\mathrm{spin}}/v$ is the ratio of the speed of a point on the ball's surface to the translational speed of the ball. In general we write

$$\mathbf{F}_M = \frac{1}{2} C_L \frac{\rho A r}{S} \boldsymbol{\omega} \times \mathbf{v} \tag{5}$$

For a baseball, experimental data show approximately a power law dependence of C_L on S

$$C_L = 0.62 \times S^{0.7}. \tag{6}$$

6.2. Baseball equations

In order to simulate the trajectory $\mathbf{r}(t)$ of a baseball, the following equations must be solved:

$$\frac{d\mathbf{r}}{dt} = \mathbf{v} \tag{7}$$

$$\frac{d\mathbf{v}}{dt} = -g\hat{\mathbf{e}}_y - \frac{b_2}{m} v\mathbf{v} + \alpha\ \boldsymbol{\omega} \times \mathbf{v} \tag{8}$$

with

$$b_2 = \frac{1}{2} C_D \rho A. \tag{9}$$

The dependence of the dynamical parameters on spin and velocity is

$$\mathbf{F}_M = \alpha\ \boldsymbol{\omega} \times \mathbf{v} \tag{10}$$

$$v = \sqrt{\mathbf{v}} \tag{11}$$

$$S = \frac{r\omega}{v} \tag{12}$$

$$C_L = 0.62 \times S^{0.7} \tag{13}$$

$$\alpha = \frac{1}{2} C_L \frac{\rho A r}{S} \tag{14}$$

6.3. Parameters

Use a ball diameter of 7.468 cm, mass of 148.83 g, and a distance of the pitcher from the home plate $R_{\mathrm{homeplate}} = 18.4\,\mathrm{m}$. Use a constant quadratic drag coefficient $C_D = 0.40$, acceleration due to gravity $g = 9.81\ \mathrm{m \cdot s^{-2}}$, and density of air $\rho = 1.225\,\mathrm{kg \cdot m^{-3}}$.

6.4. Simulation

Integrate the equations of motions with the RK4 algorithm (`ode.rk4()`). Stop the integration when $x \geqslant R_{\mathrm{homeplate}}$ or $y < 0.2\,\mathrm{m}$ (i.e., cannot be batted). The integration should be performed inside a function

```python
def simulate_baseball(v0, omega, r0,
                      h=0.01, C_D=0.40, g=9.81, rho=1.225,
                      r=0.07468/2, m=0.14883, R_homeplate=18.4):
```

as provided in the skeleton code file `hw06.py`. As input it should take the initial velocity vector $\mathbf{v}_0$ (as a 2D array (v_x, v_y)), the ball's rotational velocity vector $\boldsymbol{\omega}$ (as a 3D array $(\omega_x, \omega_y, \omega_z)$), and the initial position when leaving the pitcher's hand $\mathbf{r}_0 = (x_0, y_0)$ (for simplicity, set it to $x_0 = 0$ and $y_0 = 2\,\mathrm{m}$).

The function should return the ball's trajectory as an $N \times 4$ array for N time steps along the first axis and $[t, x(t), y(t), z(t)]$ along the second axis (where t is the time and the other three entries are the cartesian components of $\mathbf{r}(t)$).

(a) Simulate a horizontal baseball throw for initial velocity $\mathbf{v} = (30\,\mathrm{m/s}, 0)$. Try out different spins; a good starting value is $\boldsymbol{\omega} = 200\,\mathrm{rad/s} \times (0, 1, 1)$. In particular, simulate the baseball throw with

  (i) almost no spin: $\omega = 0.001 \times (0, 0, 1)$ (our code does not handle $\omega = 0$ gracefully...)

  (ii) $\omega = 200 \times (0, 0, 1)$

  (iii) $\omega = 200 \times (0, 1, 1)$

  [11 points]

(b) Plot the three scenarios in 2D planes: x-y (side view) and x-z (top view). Plot all throws together and add a legend. Briefly describe and discuss the trajectories. **[9 points]**

(c) Bonus: Plot in 3D (see Appendix A). **[bonus +3*]**

6.5. BONUS: Advanced Baseball physics (10* points)

Make your solution for Problem 6.1 more realistic. Include the following improvements and show and discuss in how far they change the results that used a simpler model.

1. The quadratic drag coefficient C_D depends on the velocity. In particular, it exhibits a "drag crisis" whereby its aerodynamic drag sharply *decreases* at a critical velocity v_c (Frohlich, 1984) as shown in Figure 1. Wang (2015) parametrizes the dimensionless drag coefficient $C_D(v)$ as

$$C_d = a + \frac{b}{1 + \exp(\chi)} - c \times \begin{cases} \exp(-\chi^2), & \chi < 0, \\ \exp(-\chi^2/4), & \chi \geqslant 0, \end{cases} \tag{15}$$

$$\chi = \frac{1}{4\,\mathrm{m/s}}(v - v_c)$$

$$v_c = 34\,\mathrm{m/s}$$

$$a = 0.36, \qquad b = 0.14, \qquad c = 0.27.$$

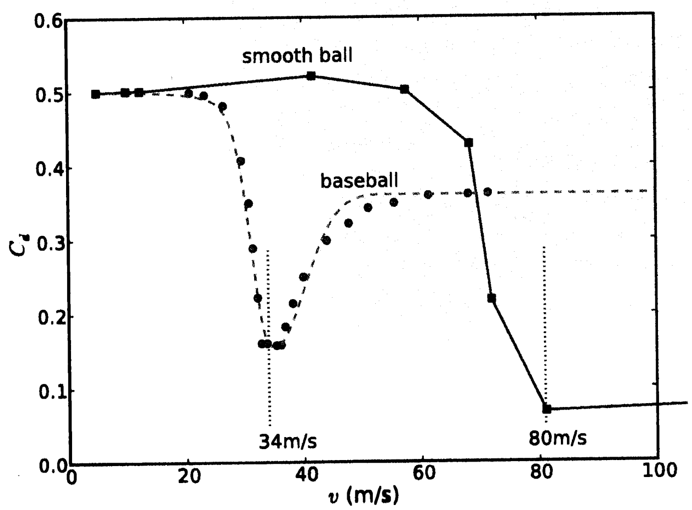

Figure 1: Drag coefficient C_D as a function of velocity for smooth balls (straight line) and baseballs (dashed line). From Wang (2015). Image Copyright ©2015 J. Wang.

Baseballs are pitched in a speed range between 50 mph (for knuckleballs) to 90 mph (fastballs), corresponding to 22 m/s to 41 m/s, which is around the critical velocity $v_c = 34$ m/s for a typical baseball. Therefore, more detailed modelling of C_D might be important to better understand the specific behavior of different pitch types (Frohlich, 1984) (as well as optimum batting parameters (Sawicki et al., 2003)).

2. Due to friction, the ball will not keep spinning at the initial velocity. Instead we can model its slow-down approximately with an exponential decay (Nathan, 2008b) (following Adair (2002))

$$\omega(t) = \omega_0 \exp(-t/\tau). \tag{16}$$

In principle, τ is velocity dependent but here you can make the simplifying assumption $\tau \approx 5\,\mathrm{s}$ (constant). For more details see Nathan (2008b).

A. 3D Plotting with matplotlib

For the 3D-plotting you can use `matplotlib` as described in the mplot3d Tutorial. In the *jupyter* notebook you can enable interactive view where you can rotate the 3D plot

```
%matplotlib notebook
```

For using `matplotlib` for 3D graphs you need to import `Axes3D` as shown below and the use the `projection='3d'` keyword argument for `add_subplot()`.

```python
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure()
ax = fig.add_subplot(1,1,1, projection='3d')

# add plots for multiple throws: note the order
# of the coordinates
ax.plot(X, Z, Y, 'o', label="no spin")

ax.set_xlabel("$x$ (m)")
ax.set_ylabel("$z$ (m)")
ax.set_zlabel("$y$ (m)")
ax.legend(loc="upper left", numpoints=1)
ax.figure.tight_layout()
```

References

Nathan A M, 2008 The effect of spin on the flight of a baseball. *American Journal of Physics* **76** 119–124, http://dx.doi.org/10.1119/1.2805242.

Wang J, 2015 *Computational Modelling and Visualization of Physical Systems with Python* (John Wiley & Sons, Hoboken, NJ).

Frohlich C, 1984 Aerodynamic drag crisis and its possible effect on the flight of baseballs. *American Journal of Physics* **52** 325–334, http://dx.doi.org/10.1119/1.13883.

Sawicki G S, Hubbard M and Stronge W J, 2003 How to hit home runs: Optimum baseball bat swing parameters for maximum range trajectories. *American Journal of Physics* **71** 1152–1162, http://dx.doi.org/10.1119/1.1604384.

Nathan A M, 2008 The effect of spin-down on the flight of a baseball, Tech. rep., University of Illinois, Urbana-Champaign, Urbana, IL, http://baseball.physics.illinois.edu/spindown.pdf.

Adair R K, 2002 *The Physics of Baseball*, 3rd edn. (Harper & Row, Publishers Inc., New York).