Worksheet 1: Integrators

Cameron N. Stewart

October 25, 2017

Institute for Computational Physics, University of Stuttgart

Contents

1 Introduction

In this worksheet we use will be using molecular dynamics simulation to look at the trajectory of a cannonball under the influence of gravity, friction, and wind and at a 2D representation of the solar system. We will be studying the behavior of a few different integrators in the latter simulation.

2 Cannonball

2.1 Simulating a cannonball

In this first exercise we simulate a cannonball in 2D under gravity in the absence of friction. The cannonball has a mass $m = 2.0\,\mathrm{kg}$,we take gravitational acceleration to be $g = 9.81\,\frac{\mathrm{m}}{\mathrm{s}^2}$, the cannonball has initial position $\boldsymbol{x}(0) = \boldsymbol{0}$, and initial velocity $\boldsymbol{v}(0) = \begin{pmatrix}50\\50\end{pmatrix}\frac{\mathrm{m}}{\mathrm{s}}$. We will use the simple Euler scheme to integrate or system. This is given by:

$$\boldsymbol{x}(t+\Delta t) = \boldsymbol{x}(t) + \boldsymbol{v}(t)\Delta t \tag{1}$$

$$\boldsymbol{v}(t+\Delta t) = \boldsymbol{v}(t) + \frac{\boldsymbol{F}(t)}{m}\Delta t \tag{2}$$

This is essentially just the Taylor expansion of position and velocity cut off below second order. We implement the simple Euler algorithm in python as follows:

```
def step_euler(x, v, dt):
    f = compute_forces(x)
    x += v*dt
    v += f*dt/m
    return x, v
```

The forces are computed simply with

```
def compute_forces(x):
    f = np.array([0.0, -m*g])
    return f
```

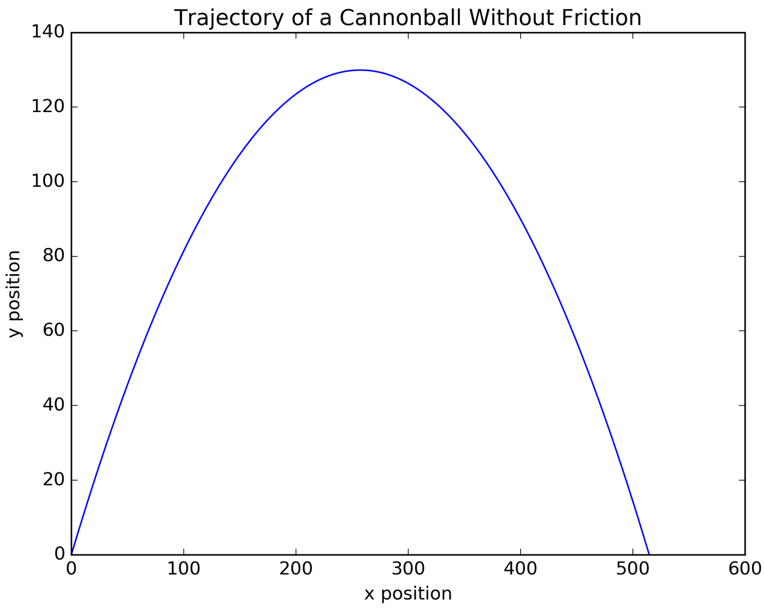

Figure 1: Trajectory of a cannonball with no friction using the simple Euler scheme

We integrate until the cannonball reaches the ground with a timestep $\Delta t = 01\,\mathrm{s}$. The trajectory can be seen in fig. 1 and the source code can be found at `src/cannonball.py`. As expected, the trajectory looks parabolic.

2.2 Influence of friction and wind

Next we will include a velocity dependant friction term in the force given by

$$F_{\text{fric}}(\boldsymbol{v}) = -\gamma(\boldsymbol{v} - \boldsymbol{v}_0) \tag{3}$$

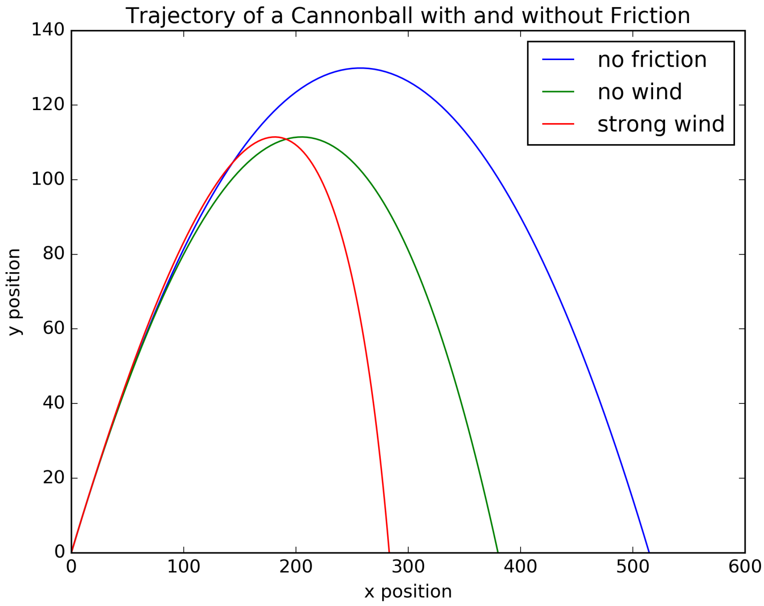

Figure 2: Trajectory of a cannonball with and without friction and for two different wind speeds

where $\boldsymbol{v}_0 = \begin{pmatrix} v_{\mathrm{w}} \\ 0 \end{pmatrix} \frac{\mathrm{m}}{\mathrm{s}}$ is the wind speed. The compute forces function was modified into the following form:

```
def compute_forces(x, v, y, vw):
    f_fric = -y*(v - np.array([vw, 0.0]))
    f = np.array([0.0, -m*g])+f_fric
    return f
```

We used a value of $\gamma = 0.1$ for the friction coefficient and once again used a time step of $\Delta t = 0.1$.

Figure 2 shows this simulation in three different cases. The original parabola is shown with no friction along with the cases $v_{\mathrm{w}} = 0$ and $v_{\mathrm{w}} = -50$. Adding friction lowers the maximum height and distance where as adding wind only changes the distance. The code can be found in `src/cannonball_fric.py`.

Finally, we ran the simulation for various wind speeds until the cannonball landed near its original launching point as seen in fig. 3. This occurred at wind speed near $v_{\mathrm{w}} = -200$.

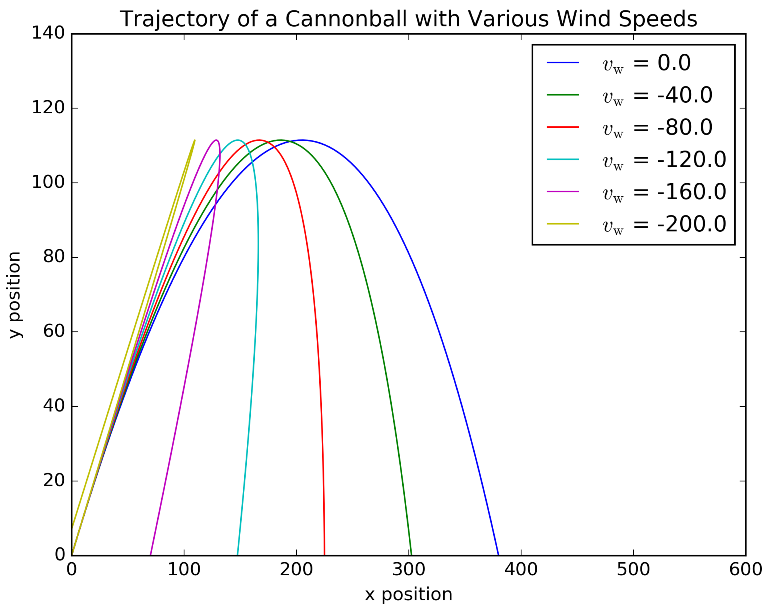

Figure 3: Trajectory of a cannonball with friction at various wind speeds