Lab 4 – Ordinary differential equations

Scientific Programming in Python with Applications in Physics

UPPSALA UNIVERSITET

Learning outcomes

After this lab, you should have learned how to use `scipy.solve_ivp` to solve ordinary differential equations.

Ordinary differential equations occur often in physics. Here you will use `scipy.integrate.solve_ivp()` (ivp – initial value problem) `https://docs.scipy.org/doc/scipy/reference/generated/scipy.integrate.solve_ivp.html` to solve the equations of motion of a projectile. We will simulate a ball being thrown upwards. We will study situations with different kinds of drag. We start by writing a function that calculates the change in the ball's velocity, i.e. its acceleration. The solver will then use this function.

Question 1

Define a function which accepts an array containing the initial conditions (called `y` in the documentation), position and velocity: (x, y, z, v_x, v_y, v_z) of the particle and the time (array with time points) and which returns the derivative

$$\frac{dy}{dt} = (v_x, v_y, v_z, 0, 0, -g).$$

The value of g can be found in `scipy.constants`. The function will be of the form

```
def func(t, y):   # t is the independent variable (time).
    ...           # Calculate the derivative of y,
    returned dydt # return it
```

Don't forget to give your function a meaningful name.

Question 2

Define a vector representing the starting conditions for your particle, position, and velocity. Choose simple ones, like starting at the origin and setting one of the velocity components to zero. Also, define the time range for your calculations. This might look like

```
y0 = [0, 0, 0, 1.0, 0.0, 20.0]
# 1 m/s to the right, 20 m/s up
times = (0, 50)
# Start at t=0, stop at t=50
```

Question 3

Solve the equations by using

```
from scipy.integrate import solve_ivp
solution = solve_ivp(func, times, start_point, t_eval=xxx)
```

`xxx` should be an array of time points for which you want (need) the positions and velocities, e.g. an `np.linspace()`. `solution` will contain `solution.t` and `solution.y`. Make sure you understand what they are.

Question 4

Plot your solution. Do not forget x- and y-labels. Also, plot the analytical solution, and compare the two.

Question 5

Create a new function that accounts for drag. Add in a drag term proportional to the velocity of the projectile. Plot the heights (with and without drag) as a function of the x-position and as a function of time.

Question 6

What happens if you assume the drag to be proportional to the square of the speed?