Portfolio Assignment 1: Using solve_ivp to study charged particle motion

David Dickinson d.dickinson@york.ac.uk

Semester 1

1 Guidance

You should submit all of your code, figures and any working or written answers (this can be a scan of hand written work and does not need to be typed up, but should be legible).

It should be possible for the marker to run your code by typing `python <your_file>` (where `<your_file>` is the submitted code file name) and this should produce any relevant plots.

Some sample example results are shown at the end in order to allow you to check your implementation. These are intentionally not presented perfectly and you should make sure that any figures you produce are appropriately labelled and clear.

The code you submit should try to follow best practice including, but not limited to, clear structure, sensible variable names, avoiding duplication, clear structure and appropriate code comments.

Should you make a mistake in, or be unable to write, code that is used in a later part of the assignment the error will be carried forward as appropriate so that you are not penalised multiple times for the same mistake.

Should your results not agree with the sample results then you may use the sample results to inform your response to any interpretation or discussion questions if you wish. You should make it clear which set of results you are referring to.

2 Overview

In this assignment you are going to use Python and the SciPy library to study charged particle motion in the presence of externally imposed electric and magnetic fields. in 3D

The equation of motion of a particle with mass m and charge q in the presence of an electric field $\boldsymbol{E}$ and magnetic field $\boldsymbol{B}$ is given by

$$\frac{m}{q}\frac{d^2\boldsymbol{x}}{dt^2} = \boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B}$$

where $\boldsymbol{x} = \{x, y, z\}$ is the particle position and $\boldsymbol{v} = \{v_x, v_y, v_z\}$ is the particle velocity.

3 Normalisation

The first thing to do is to normalise our equation, to make all quantities dimensionless with magnitudes ~ 1 and to identify the important control parameters (if any). A reasonable starting place is to pick a typical length scale and time (or velocity scale). In this case we will choose the thermal Hydrogen gyro-radius ρ_H and gyro-period, τ_H:

$$\tau_H = \frac{m_H}{q_H B_r} \qquad \rho_H = \frac{m_H v_{th,H}}{q_H B_r} = v_{th,H}\tau_H$$

where m_H is the Hydrogen mass, q_H is the Hydrogen charge and we introduce B_r at the normalising magnetic field such that $\hat{\boldsymbol{B}} = \boldsymbol{B}/B_r$ and $v_{th,H}$ as the normalising Hydrogen thermal velocity such that $\hat{\boldsymbol{v}} = \boldsymbol{v}/v_{th,H}$, where *hats* on variables represent normalised quantities.

Task [2 marks]: Using this choice of normalisations normalise the equation of motion clearly defining all of your normalised terms and writing out the three components of the final normalised vector equation (i.e. the equations of motion for each of $\hat{x}$, $\hat{y}$, $\hat{z}$).

4 Implementation

We will now consider numerically solving for the motion of individual charged particles using the normalised equation. Throughout the remainder of the assignment we will simplify the problem by considering a magnetic field pointing in the z direction, electric fields restricted to the $x - y$ plane and particles with initial $v_z = 0$. Under these restrictions we no longer need to consider the motion of the particle in z.

Task [1 mark]: In python write a function, `B_func`, that returns $\hat{B}_z(\hat{x}, \hat{y})$ given the position $\{\hat{x}, \hat{y}\}$ according to the relation

$$\hat{B}_z(\hat{x}, \hat{y}) = \frac{B_0 x_0}{\hat{x} - x_c}$$

with x_0, x_c and B_0 being additional arguments to the function and use the code below to generate a plot of the resulting magnetic field. Include this plot in your submission.

```python
import matplotlib.pyplot as plt
from numpy import zeros, linspace
nx = 1001 ; ny = 501 ; L_x = 100.0 ; L_y = 50.0
xx = linspace(0, L_x, nx)
yy = linspace(0, L_y, ny)
Bmag = zeros([nx, ny])
x_0 = L_x / 2 ; x_c = -L_x
B_0 = 1.0
for ix, x in enumerate(xx):
    for iy, y in enumerate(yy):
        Bmag[ix, iy] = B_func(x, y, x_0, x_c, B_0)

plt.pcolormesh(xx, yy, Bmag.T, shading = 'auto')
plt.xlabel(r'$\hat{x}$') ; plt.ylabel(r'$\hat{y}$')
plt.title("Example B_func output")
plt.colorbar()
```

Task [5 marks]: Write a python code using solve_ivp to integrate the equations of motion for a single particle assuming an externally applied magnetic field determined by `B_func` and zero electric field. Your code should integrate from $t = 0$ to $t = 500$, recording the state at every integer t in this range, given the initial positions and velocities of a charged particle with normalised charge and mass defined in the code.

5 Application

Task [4 marks]: Apply your code to find the motion of particles with the following initial positions, velocity and properties and plot the trajectories of each of these on top of the magnetic field `pcolormesh` produced earlier: Plot trajectories with B in background - need one x axis of time and other x axis of x position

- $\underline{\hat{x}(t=0)}$ =3L_x/4, $\underline{\hat{y}(t=0)}$ =L_y/2, $\underline{\hat{v}_x(t=0)}$ =1.0, $\underline{\hat{v}_y(t=0)}$ =0.0, $\hat{q} = 1.0$, $\hat{m} = 1.0$
- $\hat{x}(t=0)$ =L_x/2, $\hat{y}(t=0)$ =L_y/2, $\hat{v}_x(t=0)$ =1.0, $\hat{v}_y(t=0)$ =0.0, $\hat{q} = -1.0$, $\hat{m} = 1.0$
- $\hat{x}(t=0)$ =L_x/4, $\hat{y}(t=0)$ =L_y/2, $\hat{v}_x(t=0)$ =1.0, $\hat{v}_y(t=0)$ =0.0, $\hat{q} = 1.0$, $\hat{m} = 0.1$
- $\hat{x}(t=0)$ =L_x/2, $\hat{y}(t=0)$ =L_y/2, $\hat{v}_x(t=0)$ =-1.0, $\hat{v}_y(t=0)$ =0.0, $\hat{q} = 1.0$, $\hat{m} = 0.1$

You should use the `B_0, L_x, x_0, x_c, L_y` values as given in the earlier plotting snippet. You should also try to minimise code duplication.

Hint: You may want to specify the colour of your trajectories by using the `color` argument to the `plot` function.

Task [1 mark]: Describe and explain the motion observed for these particles.

Task [2 marks]: Calculate and plot the kinetic energy of these four particles as a function of time. Record the final kinetic energy divided by the initial kinetic energy and comment on how well the kinetic energy is conserved.

Task [3 marks]: Modify your code to introduce a constant $\hat{E}_y = 0.01$ to the system, apply this to the same four particles as earlier to produce a new plot of the trajectories.

Task [2 marks]: Describe in what way the kinetic energy conservation is modified and why.

6 Example plots

To help you check your implementation the following figure shows an example result. It should be noted that whilst the figures shows “correct” data the presentation can be improved for full marks.

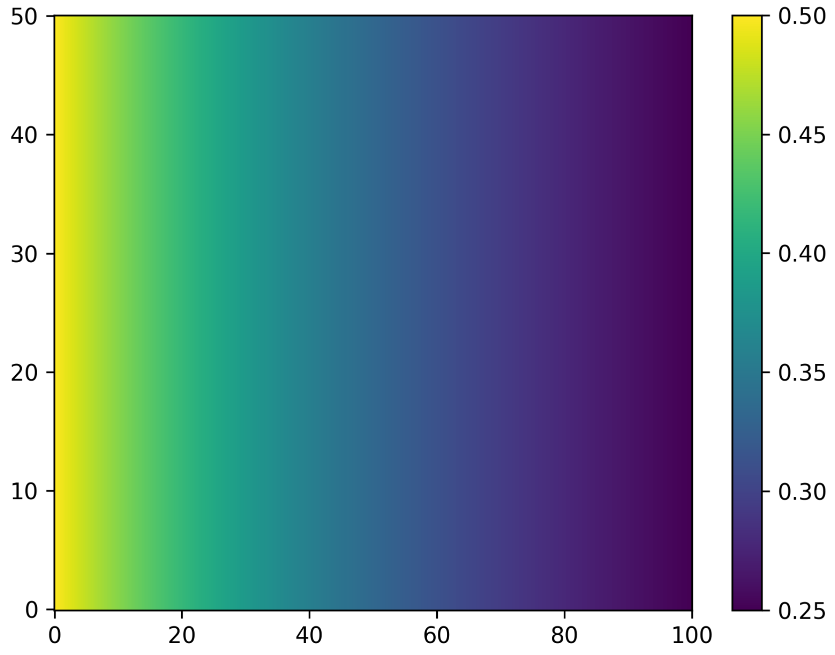

Figure 1: Colour map of $\hat{B}_z$.

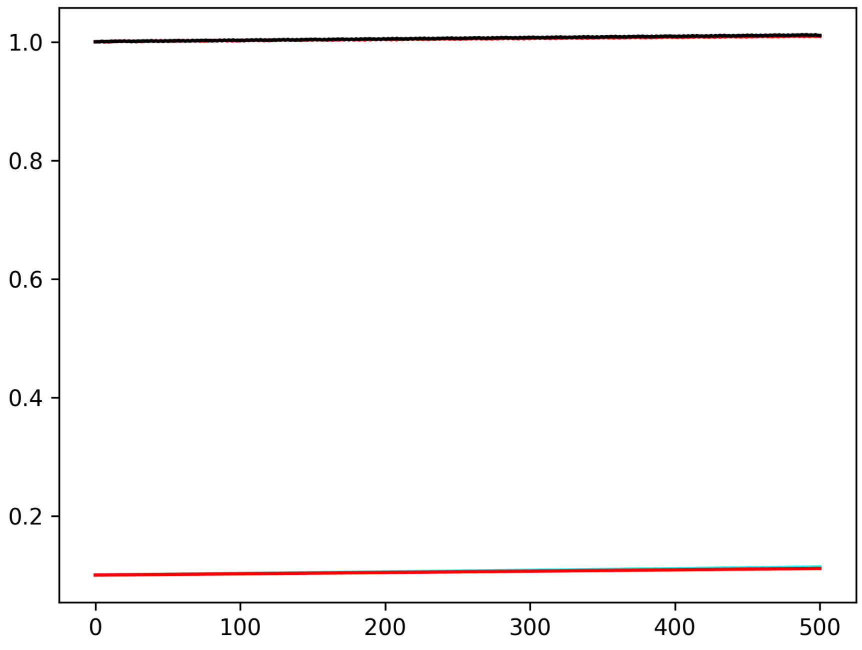

Figure 2: Plot of the kinetic energy vs time for $\hat{E}_y = 0$.

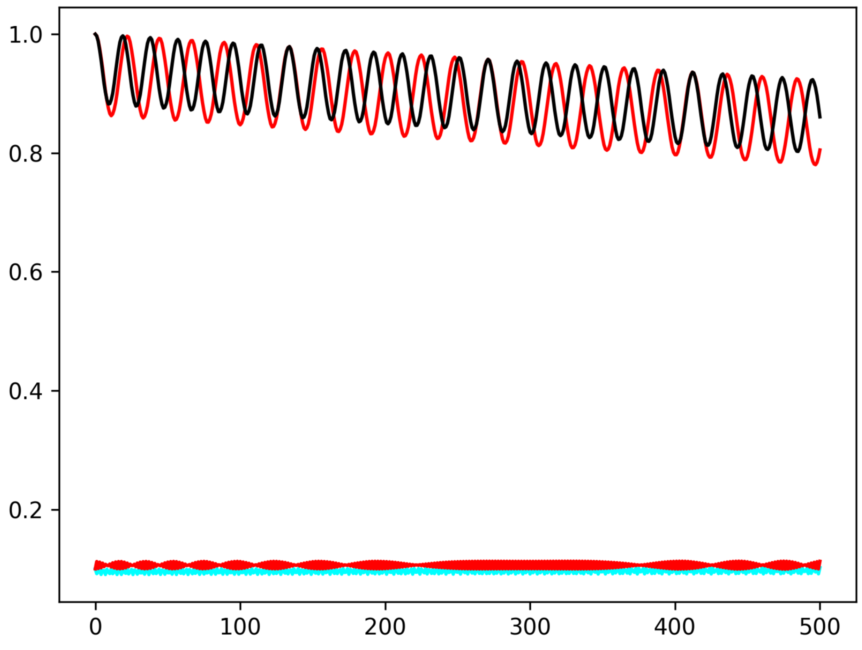

Figure 3: Plot of the kinetic energy vs time for $\hat{E}_y = 0.01$.