Project 3 - FYS4150

Nanna Bryne, Johan Mylius Kroken, Vetle A. Vikenes
(Dated: October 18, 2022)

We provide an overview of how to structure a scientific report. For concreteness, we consider the example of writing a report about an implementation of the midpoint rule of integration. For each section of the report we briefly discuss what the purpose of the given section is. We also provide examples of how to properly include equations, tables, algorithms, figures and references.

I. INTRODUCTION

Write scientific introduction here

Equations (temporary title)

We will consider an ideal Penning trap, where the electric field is defined by the electric potential

$$V(x,y,z) = \frac{V_0}{2d^2}(2z^2 - x^2 - y^2), \tag{1}$$

where V_0 is the potential applied to the electrodes, d is the characteristic dimension. And ... Define equation more appropriately.

For a single particle, the electric field is given by the negative gradient of the potential in equation (1)

$$\mathbf{E} = -\nabla V = \frac{V_0}{d^2}(x,\, y,\, -2z) \tag{2}$$

A homogeneous magnetic field is imposed in the z-direction

$$\mathbf{B} = B_0\hat{e}_z = (0, 0, B_0). \tag{3}$$

The force $\mathbf{F}$ on a single particle with charge q, is given by the Lorentz Force

$$\mathbf{F} = q\mathbf{E} + q\mathbf{v}\times\mathbf{B}. \tag{4}$$

Inserting the expressions for $\mathbf{E}$ and computing the cross product $\mathbf{v}\times\mathbf{B}$ the resulting Lorentz force on the particle is

$$\begin{aligned}\mathbf{F} &= \frac{qV_0}{d^2}(x,\, y,\, -2z) + qB_0(\dot{y},\, -\dot{x},\, 0)\\ &= \frac{m}{2}\omega_z^2(x,\, y,\, -2z) + m\omega_0(\dot{y},\, -\dot{x},\, 0),\end{aligned} \tag{5}$$

where we defined $\omega_0 \equiv \frac{qB_0}{m}$ and $\omega_z^2 \equiv \frac{2qV_0}{md^2}$. From Newton's second law, we have $\ddot{\mathbf{r}} = \mathbf{F}/m$. Dividing equation (5) by m, we get three equations of motion, one for each spatial component. These differential equations are

$$\ddot{x} - \omega_0\dot{y} - \frac{1}{2}\omega_z^2 x = 0, \tag{6}$$

$$\ddot{y} + \omega_0\dot{x} - \frac{1}{2}\omega_z^2 y = 0, \tag{7}$$

$$\ddot{z} + \omega_z^2 z = 0. \tag{8}$$

Equation (8) has the general solution

$$z(t) = c_1\cos(\omega_z t) + c_2\sin(\omega_z t). \tag{9}$$

For this project, we will assume $q > 0$. (Move this?)

Equations (6) and (7) are coupled. We therefore introduce a complex function $f(t) = x(t) + iy(t)$ whose time derivatives are:

$$\dot{f} = \dot{x} + i\dot{y} \tag{10}$$

$$\ddot{f} = \ddot{x} + i\ddot{y}. \tag{11}$$

We then multiply equation (7) with i and add to equation (6) to obtain:

$$\begin{aligned}\ddot{x} + i\ddot{y} + \omega_0(i\dot{x} - \dot{y}) - \frac{1}{2}\omega_z^2(x+iy) = 0\\ (\ddot{x} + i\ddot{y}) + i\omega_0(\dot{x} + i\dot{y}) - \frac{1}{2}\omega_z^2(x+iy) = 0.\end{aligned} \tag{12}$$

The single differential equation in terms of the complex function $f(t)$ is therefore:

$$\ddot{f} + i\omega_0\dot{f} - \frac{1}{2}\omega_z^2 f = 0. \tag{13}$$

The general solution to equation (13) is

$$f(t) = A_+e^{-i(\omega_+ t + \phi_+)} + A_-e^{-i(\omega_- t + \phi_-)}, \tag{14}$$

where ϕ_+ and ϕ_- are constant phases, A_+ and A_- are positive amplitudes and

$$\omega_\pm = \frac{\omega_0 \pm \sqrt{\omega_0^2 - 2\omega_z^2}}{2}. \tag{15}$$

We see from equation (14) that if $\omega_\pm$ is real, then $|f(t)| < \infty$ for $t\to\infty$ and we have a bounded solution in the xy-plane. For $\omega_\pm$ to be real we see from equation (15) that the following must hold:

$$\omega_0^2 \geq 2\omega_z^2 \tag{16}$$

$$\implies \frac{q}{m} \geq \frac{4V_0}{(B_0 d)^2}. \tag{17}$$

We may regard equation (14) as a sum of vectors $\mathbf{r}_\pm$ with amplitudes $r_\pm = A_\pm$ and directions given by the angles $\alpha_\pm = \omega_\pm t + \phi_\pm$ in the complex plane (representing the xy-plane). The magnitude $|f(t)| = |f|$ for this sum of vectors is then given by:

$$|f| = \sqrt{|\mathbf{r}_+|^2 + |\mathbf{r}_-|^2 + 2|\mathbf{r}_+||\mathbf{r}_-|\cos\left(|\alpha_+ - \alpha_-|\right)}. \tag{18}$$

The maximum distance from the origin, R_+, occurs when the two vectors are pointing in the same direction, i.e. $|\alpha_+ - \alpha_-| = 0 \implies \cos 0 = 1$, which yields:

$$\begin{aligned} R_+ &= \sqrt{A_+^2 + A_-^2 + 2A_+A_-} = \sqrt{(A_+ + A_-)^2} \\ &= A_+ + A_- \end{aligned} \tag{19}$$

Similarily, the minimum distance, R_-, from the origin occurs when the vectors are pointing in opposite directions, i.e. $|\alpha_+ - \alpha_-| = \pi \implies \cos\pi = -1$, which yields:

$$\begin{aligned} R_- &= \sqrt{A_+^2 + A_-^2 - 2A_+A_-} = \sqrt{(A_+ - A_-)^2} \\ &= |A_+ - A_-| \end{aligned} \tag{20}$$

$$\ddot{x}_i - \omega_{0,i}\dot{y}_i - \frac{1}{2}\omega_{z,i}^2 x_i - k_e \frac{q_i}{m_i}\sum_{j\neq i} q_j \frac{x_i - x_j}{|\mathbf{r}_i - \mathbf{r}_j|^3} = 0, \tag{21}$$

$$\ddot{y}_i + \omega_{0,i}\dot{x}_i - \frac{1}{2}\omega_{z,i}^2 y_i - k_e \frac{q_i}{m_i}\sum_{j\neq i} q_j \frac{y_i - y_j}{|\mathbf{r}_i - \mathbf{r}_j|^3} = 0, \tag{22}$$

$$\ddot{z}_i + \omega_{z,i}^2 z_i - k_e \frac{q_i}{m_i}\sum_{j\neq i} q_j \frac{z_i - z_j}{|\mathbf{r}_i - \mathbf{r}_j|^3} = 0, \tag{23}$$

II. METHODS

We aim to create a program in C++ that simulates a set of N_p particles inside a Penning trap. An object-oriented code is befitting this task and we present in the following descriptions of the classes `Particle` and `PenningTrap`.

The purpose of `Particle` is to hold the parameters, such as the position, of a particle. We let an object of this class be initialised with a charge q, a mass m, a position $\mathbf{r} = (x, y, z)$ and a velocity $\mathbf{v} = (v_x, v_y, v_z)$. We add functions to update the latter two.

The `PenningTrap`-class imitates the physical system that is the Penning trap of magnetic field $\mathbf{B} = B_0\hat{e}_z$, electric potential $V = \frac{V_0}{2d^2}(2z^2 - x^2 - y^2)$ and characteristic dimension d. It is friend with the `Particle`-class and, in order to resemble the physical situation as much as possible, accepts only input particles of this type. When filled with $N_\text{p} \geq 1$, an object of `PenningTrap` is ready to simulate the evolution of the `Particle` object(s) for a given period of time and time step with either a Forward-Euler or a 4th order Runge-Kutta numerical scheme.

...

`PenningTrap` offers several additional functionalites. Amongst other things, the class

- offers the choice between 4th order Runge-Kutta or Forward-Euler as integration method
- provides the option to include or exclude Coloumb interactions in the simulation, as they are computationally greedy
- includes a method for adding a time-dependent perturbation to the external electric potential, $V_0 \to V_0(1 + f(1 + \cos(\omega_V t)))$
- has the ability to count the number of particles still trapped, that is the number particles whose position $\mathbf{r}$ is such that $|\mathbf{r}| \leq d$
- can generate a set of identical particles with positions and velocities that are normally distrubuted within the trap's dimensions

Algorithm 1 Forward Euler (FE)

```
U_0 ← ...                                   ▷ Initialise velocities in cube
R_0 ← ...                                   ▷ Initialise positions in cube
h ← (t_n − t_0)/n                           ▷ Compute the time step length
for i = 0, 1, ..., n − 1 do
    for each particle do
        v ← U_{i,particle}
        r ← R_{i,particle}
        dU_{i,particle} ← hF(r, v)/m
        dR_{i,particle} ← h(v + dU_{i,particle})
    U_{i+1} ← U_i + dU_i                    ▷ Update velocities
    R_{i+1} ← R_i + dR_i                    ▷ Update positions
```

Algorithm 2 4th order Runge-Kutta (RK4)

```
U_0 ← ...                                   ▷ Initialise velocities in cube
R_0 ← ...                                   ▷ Initialise positions in cube
h ← (t_n − t_0)/n                           ▷ Compute the time step length
for i = 0, 1, ..., n − 1 do
    for each particle do
        v ← U_{i,particle}
        r ← R_{i,particle}
        k_{v,1} ← hF(r, v)/m                                         ▷ ...
        k_{r,1} ← hv
        k_{v,2} ← hF(r + 1/2k_{r,1}, v + 1/2k_{v,1})/m               ▷ ...
        k_{r,2} ← h(v + 1/2k_{v,1})
        k_{v,3} ← hF(r + 1/2k_{r,2}, v + 1/2k_{v,2})/m               ▷ ...
        k_{r,3} ← h(v + 1/2k_{v,2})
        k_{v,4} ← hF(r + k_{r,3}, v + k_{v,3})/m                     ▷ ...
        k_{r,4} ← h(v + k_{v,3})
        dU_{i,particle} ← (k_{v,1} + 2k_{v,2} + 2k_{v,3} + k_{v,4})/6
        dR_{i,particle} ← (k_{r,1} + 2k_{r,2} + 2k_{r,3} + k_{r,4})/6
    U_{i+1} ← U_i + dU_i                    ▷ Update velocities
    R_{i+1} ← R_i + dR_i                    ▷ Update positions
...                                                                  ▷ ...
```

The algorithm

III. RESULTS

IV. DISCUSSION

V. CONCLUSION

In this section we state three things in a concise manner: what we have done, what we have found, and what should or could be done in the future.