Project 3

Cory Alexander Balaton & Janita Ovidie Sandtrøen Willumsen
https://github.uio.no/FYS3150-G2-2023/Project-3
(Dated: October 16, 2023)

Add an abstract for the project?

I. INTRODUCTION

II. METHODS

For a multi-particles system, we need to modify $\mathbf{F}$ to account for the force of other particles in the system acting upon each other. To do that, we add another term to $\mathbf{F}$

$$\mathbf{F}_i(t, \mathbf{v}_i, \mathbf{r}_i) = q_i \mathbf{E}(t, \mathbf{r}_i) + q_i \mathbf{v}_i \times \mathbf{B} - \mathbf{E}_p(t, \mathbf{r}_i), \tag{1}$$

where i and j are particle indices and

$$\mathbf{E}_p(t, \mathbf{r}_i) = q_i k_e \sum_{j \neq i} q_j \frac{\mathbf{r_i} - \mathbf{r_j}}{|\mathbf{r_i} - \mathbf{r_j}|^3}. \tag{2}$$

Newton's second law for a particle i is then

$$\frac{d^2 \mathbf{r}_i}{dt^2} = \frac{\mathbf{F}\left(t, \frac{d\mathbf{r}_i}{dt}, \mathbf{r_i}\right)}{m_i}, \tag{3}$$

We can then rewrite the second order ODE from equation 3 into a set of coupled first order ODEs. We now rewrite Newton's second law of motion as

$$\begin{aligned} \frac{d\mathbf{r}_i}{dt} &= \mathbf{v}_i \\ \frac{d\mathbf{v}_i}{dt} &= \frac{\mathbf{F}(t, \mathbf{v}_i, \mathbf{r}_i)}{m_i}. \end{aligned} \tag{4}$$

A. Forward Euler

For a single particle, the forward Euler method for a coupled system is expressed as

$$\begin{aligned} \mathbf{r}_{i+1} &= \mathbf{r}_i + h \cdot \frac{d\mathbf{r}_i}{dt} = \mathbf{r}_i + h \cdot \mathbf{v}_i \\ \mathbf{v}_{i+1} &= \mathbf{v}_i + h \cdot \frac{\mathbf{v}_i}{dt} = \mathbf{v}_i + h \cdot \frac{\mathbf{F}\left(t_i, \mathbf{v}_i, \mathbf{r}_i\right)}{m}, \end{aligned} \tag{5}$$

where i is the current time step of the particle m is the mass of the particle, and h is the time step length.

When dealing with a multi-particle system, we need to ensure that we do not update the position of any particles until every particle has calculated their next step. An easy way of doing this is to create a copy of all the particles, then update the copy, and when all the particles have calculated their next step, simply replace the particles with the copies. Algorithm 1 provides an overview on how that can be achieved.

Algorithm 1 Forward Euler method

> **procedure** EVOLVE FORWARD EULER($particles, dt$)
> $N \leftarrow$ Number of particles in $particles$
> $a \leftarrow$ Calculate $\frac{\mathbf{F_i}}{m_i}$ for each particle in $particles$
> **for** $i = 1, 2, \ldots, N$ **do**
> $particles_i.\mathbf{r} \leftarrow particles_i.\mathbf{r} + dt \cdot particles_i.\mathbf{v}$
> $particles_i.\mathbf{v} \leftarrow particles_i.\mathbf{v} + dt \cdot a_i$

B. 4th order Runge-Kutta

For a single particle, we can express the 4th order Runge-Kutta (RK4) method as

$$\begin{aligned} \mathbf{v}_{i+1} &= \mathbf{v}_i + \frac{h}{6}\left(\mathbf{k}_{\mathbf{v},1} + 2\mathbf{k}_{\mathbf{v},2} + 2\mathbf{k}_{\mathbf{v},3} + \mathbf{k}_{\mathbf{v},4}\right) \\ \mathbf{r}_{i+1} &= \mathbf{r}_i + \frac{h}{6}\left(\mathbf{k}_{\mathbf{r},1} + 2\mathbf{k}_{\mathbf{r},2} + 2\mathbf{k}_{\mathbf{r},3} + \mathbf{k}_{\mathbf{r},4}\right), \end{aligned} \tag{6}$$

where

$$\begin{aligned} \mathbf{k}_{\mathbf{v},1} &= \frac{\mathbf{F}(t_i, \mathbf{v}_i, \mathbf{r}_i)}{m} \\ \mathbf{k}_{\mathbf{r},1} &= \mathbf{v}_i \\ \mathbf{k}_{\mathbf{v},2} &= \frac{\mathbf{F}(t_i + \frac{h}{2}, \mathbf{v}_i + h \cdot \frac{\mathbf{k}_{\mathbf{v},1}}{2}, \mathbf{r}_i + h \cdot \frac{\mathbf{k}_{\mathbf{r},1}}{2})}{m} \\ \mathbf{k}_{\mathbf{r},2} &= \mathbf{v}_i + h \cdot \frac{\mathbf{k}_{\mathbf{v},1}}{2} \\ \mathbf{k}_{\mathbf{v},3} &= \frac{\mathbf{F}(t_i + \frac{h}{2}, \mathbf{v}_i + h \cdot \frac{\mathbf{k}_{\mathbf{v},2}}{2}, \mathbf{r}_i + h \frac{\cdot \mathbf{k}_{\mathbf{r},2}}{2})}{m} \\ \mathbf{k}_{\mathbf{r},3} &= \mathbf{v}_i + h \cdot \frac{\mathbf{k}_{\mathbf{v},2}}{2} \\ \mathbf{k}_{\mathbf{v},4} &= \frac{\mathbf{F}(t_i + h, \mathbf{v}_i + h \cdot \mathbf{k}_{\mathbf{v},3}, \mathbf{r}_i + h \cdot \mathbf{k}_{\mathbf{r},3})}{m} \\ \mathbf{k}_{\mathbf{r},4} &= \mathbf{v}_i + h \cdot \frac{\mathbf{k}_{\mathbf{v},1}}{2}. \end{aligned} \tag{7}$$

Algorithm 2 RK4 method

```
procedure EVOLVE RK4(dt)
    N ← Number of particles inside the Penning trap
    orig_p ← Copy of particles                    ▷ Keep the original particles
    tmp_p ← Array of particles of size N
    k_r ← 2D array of vectors of size 4 × N
    k_v ← 2D array of vectors of size 4 × N
    for i = 1, 2, ..., N do
        k_{r,1,i} ← v
        k_{v,1,i} ← F/m
        tmp_p_i.r ← orig_p_i.r + dt/2 · k_{r,1,i}
        tmp_p_i.v ← orig_p_i.v + dt/2 · k_{v,1,i}
    particles ← tmp_p                             ▷ Update particles
    for i = 1, 2, ..., N do
        k_{r,2,i} ← v
        k_{v,2,i} ← F/m
        tmp_p_i.r ← orig_p_i.r + dt/2 · k_{r,2,i}
        tmp_p_i.v ← orig_p_i.v + dt/2 · k_{v,2,i}
    particles ← tmp_p                             ▷ Update particles
    for i = 1, 2, ..., N do
        k_{r,3,i} ← v
        k_{v,3,i} ← F/m
        tmp_p_i.r ← orig_p_i.r + dt · k_{r,3,i}
        tmp_p_i.v ← orig_p_i.v + dt · k_{v,3,i}
    particles ← tmp_p                             ▷ Update particles
    for i = 1, 2, ..., N do
        k_{r,4,i} ← v
        k_{v,4,i} ← F/m
        tmp_p_i.r ← orig_p_i.r + dt/6 · (k_{r,1,i} + k_{r,2,i} + k_{r,3,i} + k_{r,4,i})
        tmp_p_i.v ← orig_p_i.v + dt/6 · (k_{v,1,i} + k_{v,2,i} + k_{v,3,i} + k_{v,4,i})
    particles ← tmp_p                             ▷ Final update
```

$\mathbf{F}$ in the algorithm does not take any arguments as it uses the velocities and positions of the particles inside the array *particles* to calculate the total force acting on particle i.

III. RESULTS

IV. CONCLUSION