Project 3

Cory Alexander Balaton & Janita Ovidie Sandtrøen Willumsen
https://github.uio.no/FYS3150-G2-2023/Project-3
(Dated: October 15, 2023)

Add an abstract for the project?

I. INTRODUCTION

II. METHODS

Newton's second law of motion is defined as

$$\frac{d^2\mathbf{r}}{dt^2} = \frac{\mathbf{F}\left(t, \frac{d\mathbf{r}}{dt}, \mathbf{r}\right)}{m}. \tag{1}$$

We can then rewrite the second order ODE from equation 1 into a set of coupled first order ODEs. We now redefine Newton's second law of motion as

$$\begin{aligned} \frac{d\mathbf{r}}{dt} &= \mathbf{v} \\ \frac{d\mathbf{v}}{dt} &= \frac{\mathbf{F}(t, \mathbf{v}, \mathbf{r})}{m}. \end{aligned} \tag{2}$$

A. Forward Euler

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

$$\begin{aligned} \mathbf{r}_{i+1} &= \mathbf{r}_i + h \cdot \frac{d\mathbf{r}_i}{dt} = \mathbf{r}_i + h \cdot \mathbf{v}_i \\ \mathbf{v}_{i+1} &= \mathbf{v}_i + h \cdot \frac{\mathbf{v}_i}{dt} = \mathbf{v}_i + h \cdot \frac{\mathbf{F}\left(t_i, \mathbf{v}_i, \mathbf{r}_i\right)}{m}, \end{aligned} \tag{3}$$

where $\mathbf{v}_i$ is the current velocity of the particle, $\mathbf{r}_i$ is the current position of the particle, m is the mass of the particle, and h is a predetermined timestep.

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)
    new_state ← particles                ▷ Create a copy of the particles
    for i = 1, 2, ..., |particles| do
        new_state_i.v ← particles_i.v + dt · F/m
        new_state_i.r ← particles_i.r + dt · particles_i.v
    particles ← new_state
```

B. 4th order Runge-Kutta

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

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

where

$$\begin{aligned} \mathbf{k}_{\mathbf{v},1} &= \frac{\mathbf{F}(t_i, \mathbf{v}_i, \mathbf{r}_i)}{m} \\ \mathbf{k}_{\mathbf{r},1} &= \mathbf{v}_i \\ \mathbf{k}_{\mathbf{v},2} &= \frac{\mathbf{F}(t_i + \frac{h}{2}, \mathbf{v}_i + h \cdot \frac{\mathbf{k}_{\mathbf{v},1}}{2}, \mathbf{r}_i + h \cdot \frac{\mathbf{k}_{\mathbf{r},1}}{2})}{m} \\ \mathbf{k}_{\mathbf{r},2} &= \mathbf{v}_i + h \cdot \frac{\mathbf{k}_{\mathbf{v},1}}{2} \\ \mathbf{k}_{\mathbf{v},3} &= \frac{\mathbf{F}(t_i + \frac{h}{2}, \mathbf{v}_i + h \cdot \frac{\mathbf{k}_{\mathbf{v},2}}{2}, \mathbf{r}_i + h\frac{\cdot \mathbf{k}_{\mathbf{r},2}}{2})}{m} \\ \mathbf{k}_{\mathbf{r},3} &= \mathbf{v}_i + h \cdot \frac{\mathbf{k}_{\mathbf{v},2}}{2} \\ \mathbf{k}_{\mathbf{v},4} &= \frac{\mathbf{F}(t_i + h, \mathbf{v}_i + h \cdot \mathbf{k}_{\mathbf{v},3}, \mathbf{r}_i + h \cdot \mathbf{k}_{\mathbf{r},3})}{m} \\ \mathbf{k}_{\mathbf{r},4} &= \mathbf{v}_i + h \cdot \frac{\mathbf{k}_{\mathbf{v},1}}{2}. \end{aligned} \tag{5}$$

Algorithm 2 RK4 method

procedure EVOLVE RK4(dt)
  $N \leftarrow$ Number of particles inside the Penning trap
  $orig\_p \leftarrow$ Copy of particles ▷ Keep the original particles
  $tmp\_p \leftarrow$ Array of particles of size N
  $\mathbf{k_r} \leftarrow$ 2D array of vectors of size $4 \times N$
  $\mathbf{k_v} \leftarrow$ 2D array of vectors of size $4 \times N$
  for $i = 1, 2, \ldots, N$ **do**
    $\mathbf{k}_{\mathbf{r},1,i} \leftarrow \mathbf{v}$
    $\mathbf{k}_{\mathbf{v},1,i} \leftarrow \frac{\mathbf{F}}{m}$
    $tmp\_p_i.\mathbf{r} \leftarrow orig\_p_i.\mathbf{r} + \frac{dt}{2} \cdot \mathbf{k}_{\mathbf{r},1,i}$
    $tmp\_p_i.\mathbf{v} \leftarrow orig\_p_i.\mathbf{v} + \frac{dt}{2} \cdot \mathbf{k}_{\mathbf{v},1,i}$
  $particles \leftarrow tmp\_p$ ▷ Update particles
  for $i = 1, 2, \ldots, N$ **do**
    $\mathbf{k}_{\mathbf{r},2,i} \leftarrow \mathbf{v}$
    $\mathbf{k}_{\mathbf{v},2,i} \leftarrow \frac{\mathbf{F}}{m}$
    $tmp\_p_i.\mathbf{r} \leftarrow orig\_p_i.\mathbf{r} + \frac{dt}{2} \cdot \mathbf{k}_{\mathbf{r},2,i}$
    $tmp\_p_i.\mathbf{v} \leftarrow orig\_p_i.\mathbf{v} + \frac{dt}{2} \cdot \mathbf{k}_{\mathbf{v},2,i}$
  $particles \leftarrow tmp\_p$ ▷ Update particles
  for $i = 1, 2, \ldots, N$ **do**
    $\mathbf{k}_{\mathbf{r},3,i} \leftarrow \mathbf{v}$
    $\mathbf{k}_{\mathbf{v},3,i} \leftarrow \frac{\mathbf{F}}{m}$
    $tmp\_p_i.\mathbf{r} \leftarrow orig\_p_i.\mathbf{r} + dt \cdot \mathbf{k}_{\mathbf{r},3,i}$
    $tmp\_p_i.\mathbf{v} \leftarrow orig\_p_i.\mathbf{v} + dt \cdot \mathbf{k}_{\mathbf{v},3,i}$
  $particles \leftarrow tmp\_p$ ▷ Update particles
  for $i = 1, 2, \ldots, N$ **do**
    $\mathbf{k}_{\mathbf{r},4,i} \leftarrow \mathbf{v}$
    $\mathbf{k}_{\mathbf{v},4,i} \leftarrow \frac{\mathbf{F}}{m}$
    $tmp\_p_i.\mathbf{r} \leftarrow orig\_p_i.\mathbf{r} + \frac{dt}{6} \cdot (\mathbf{k}_{\mathbf{r},1,i} + \mathbf{k}_{\mathbf{r},2,i} + \mathbf{k}_{\mathbf{r},3,i} + \mathbf{k}_{\mathbf{r},4,i})$
    $tmp\_p_i.\mathbf{v} \leftarrow orig\_p_i.\mathbf{v} + \frac{dt}{6} \cdot (\mathbf{k}_{\mathbf{v},1,i} + \mathbf{k}_{\mathbf{v},2,i} + \mathbf{k}_{\mathbf{v},3,i} + \mathbf{k}_{\mathbf{v},4,i})$
  $particles \leftarrow tmp\_p$ ▷ 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